Naval Health Research Center

AD-A261 364

(2)

COMPUTER ASSISTED MEDICAL DIAGNOSIS (CAMD) SYSTEM

VERSION 1.0

TECHNICAL MANUAL

H. L. Ly

R. W. Dobbins

D. M. Pearsall

DTIC
ELECTE
MAR 09 1993
S E D

Report No. 92-25

93-04977

Approved for public release: distribution unlimited.

NAVAL HEALTH RESEARCH CENTER
P.O. BOX 85122
SAN DIEGO, CALIFORNIA 92186-5122

NAVAL MEDICAL RESEARCH AND DEVELOPMENT COMMAND
BETHESDA, MARYLAND

Computer Assisted Medical Diagnosis (CAMD) System
Version 1.0
Technical Manual

Prepared by:

Hoa L. Ly
Roy W. Dobbins
Dianna M. Pearsall

Naval Health Research Center
Medical Decisions Support Department
P.O. Box 85122
San Diego, CA 92186-5122

Accesion For	
NTIS CRA&I	X
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	

Report No. TR 92-25 was supported by the Naval Medical Research and Development Command, Bethesda, MD, Department of the Navy, under Work Unit No. 63706N M0095.005-6103. The views expressed in this article are those of the authors and do not reflect the official policy or position of the Department of the Navy, Department of Defense, nor the U.S. Government.

SUMMARY

This technical manual contains the information on the program code, data elements, file structures needed to maintain the Computer Assisted Medical Diagnostic System. This documentation was created using the FOXDOC Version 2.0 program.

Table of Contents

Introduction 1

Section I. System Summary. 1

Section II. Menu Summary. 1

Section III. Screen Summary. 4

section IV. Report Form File Summary 21

Section V. Data Dictionary 22

Database Structure Summary 22

Database Field Summary 28

Index Parameter Summary 30

Section VI. Tree Diagram. 34

Section VII. Procedure and Function Summary. 51

Section VIII. Program Source Code 75

Introduction. *This document was created using the FOXDOC Version 2.0 program to generate the technical documentation for the Computer Assisted Medical Diagnosis System (CAMD) system. The documentation is separated into eight sections (1) System Summary, (2) Menu Summary, (3) Screen Summary, (4) Report Form File Summary, (5) Data Dictionary, (6) Tree Diagram, (7) Procedure Summary, and (8) Program Source Code.*

Section I. System Summary. *See the tree diagram for programs, procedures, functions and format files.*

This system has:

```
 17695 lines of code
     1 program file
    59 procedure files
   301 procedures and functions
    23 databases
     1 structural index file
    31 index files
     1 report form
     2 menu files
    29 screen files
     4 other files
     2 cross-referenced tokens
```

Section II. Menu Summary. *The following description lists the menu options used to drive the system. Two menu files in CAMD system are CAMD.MNX and CAMDTR.MNX.*

CAMD.MNX is main menu to driven the system. Last updated: 10/07/92 at 11:51

OPTION NAME	KEYS\FUNCTION CALLED
System	ALT+S (Submenu _CSYSTEM)
Help...	F1 _MST_HELP

Change Password	do pwdedt
Backup	do _mhit with "backup"
Restore	do _mhit with "restore"

Filer	_MST_FILER
Calculator	_MST_CALCU
Calendar/Diary	_MST_DIARY
Puzzle	_MST_PUZZL
File	ALT+F (Submenu _CFILE)
Reindex	DO reindex

Quit	DO _quit
Edit	ALT+E (Submenu _CEDIT)
Undo	CTRL+U _MED_UNDO
Redo	CTRL+R _MED_REDO
----------	_MED_SP100
Cut	CTRL+X _MED_CUT
Copy	CTRL+C _MED_COPY
Paste	CTRL+V _MED_PASTE
Clear	_MED_CLEAR

Select All	CTRL+A _MED_SLCTA

Goto Line...	_MED_GOTO
Find...	CTRL+F _MED_FIND
Find Again	CTRL+G _MED_FINDA

```
   Replace All                    _MED_REPLA
   ------------
   Preferences...                 _MED_PREF
Diagnosis                         ALT+D    (Submenu DIAGNOSIS)
   Register Patient...            DO _mhit WITH "m_pat"
   Encounter...                   DO _mhit WITH "m_newenc"
   SF600 Report                   DO _mhit WITH "m_sf600"
   ------------
   Disease/Treatment Lookup       DO _mhit WITH "m_lookup"
   ------------
   Encounter Training Mode        (procedure)
Window                            ALT+W    (Submenu _CWINDOW)
   Move                           CTRL+F7  _MWI_MOVE
   Size                           CTRL+F8  _MWI_SIZE
   Zoom ^X                        CTRL+F10 _MWI_ZOOM
   Zoom ^Y                        CTRL+F9  _MWI_MIN
   Cycle                          CTRL+F1  _MWI_ROTAT
   Color...                       do colorset.spr
   ------------
   Command                        CTRL+F2  _MWI_CMD
   Debug                          _MWI_DEBUG
   Trace                          _MWI_TRACE
   View                           _MWI_VIEW
```

CAMDTR.MNX is a menu to use for training. Last updated: 09/28/92 at 14:19

```
OPTION NAME                       KEYS\FUNCTION CALLED
System                            ALT+S    (Submenu _CSYSTEM)
   Help...                        F1       _MST_HELP
   ------------
   Change Password                do pwdedt
   Backup                         do _mhit with "backup"
   Restore                        do _mhit with "restore"
   ------------
   Filer                          _MST_FILER
   Calculator                     _MST_CALCU
   Calendar/Diary                 _MST_DIARY
   Puzzle                         _MST_PUZZL
File                              ALT+F    (Submenu _CFILE)
   Reindex                        DO reindex
   ------------
   Quit                           DO _quit
Edit                              ALT+E    (Submenu _CEDIT)
   Undo                           CTRL+U   _MED_UNDO
   Redo                           CTRL+R   _MED_REDO
   ------------
   Cut                            CTRL+X   _MED_CUT
   Copy                           CTRL+C   _MED_COPY
   Paste                          CTRL+V   _MED_PASTE
   Clear                          _MED_CLEAR
   ------------                   _MED_SP200
   Select All                     CTRL+A   _MED_SLCTA
   ------------
   Goto Line...                   _MED_GOTO
   Find...                        CTRL+F   _MED_FIND
   Find Again                     CTRL+G   _MED_FINDA
   Replace And Find Again         CTRL+E   _MED_REPL
   Replace All                    _MED_REPLA
   ------------
   Preferences...                 _MED_PREF
Diagnosis                         ALT+D    (Submenu DIAGNOSIS)
   Register Patient...            DO _mhit WITH "m_pat"
```

```
   Encounter...                    DO _mhit WITH "m_newenc"
   SF600 Report                    DO _mhit WITH "m_sf600"
   ------------
   Disease/Treatment Lookup        DO _mhit WITH "m_lookup"
   ------------
   Encounter Training Mode         DO _mhit WITH "m_training"
Window                             ALT+W    (Submenu _CWINDOW)
   Move                            CTRL+F7  _MWI_MOVE
   Size                            CTRL+F8  _MWI_SIZE
   Zoom ^X                         CTRL+F10 _MWI_ZOOM
   Zoom ^Y                         CTRL+F9  _MWI_MIN
   Cycle                           CTRL+F1  _MWI_ROTAT
   Color...                        _MWI_COLOR
   ------------
   Command                         CTRL+F2  _MWI_CMD
   Debug                           _MWI_DEBUG
   Trace                           _MWI_TRACE
   View                            _MWI_VIEW
TRAINING MODE
```

Section III. Screen Summary. *Twenty eigth (28) screen files were used as input templates in the system; W14.SCX, W10.SCX, W7.SCX, W6.SCX, W6C.SCX, W6B.SCX, W11.SCX, W12.SCX, W2.SCX, W9.SCX, W5.SCX, W3.SCX, W13.SCX, WP3.SCX, WP4.SCX, WP5.SCX, WP6.SCX, WP7.SCX, WP8.SCX, WP9.SCX, W12A.SCX, W12B.SCX, W12C.SCX, W12E.SCX, BACKUP.SCX, RESTORE.SCX, W7A.SCX, COLORSET.SCX.*

W14.SCX Last updated: 02/12/92 at 10:50

```
 0
 1
 2  1: mchoice.......................................▲
 3  ................................................♦
 4  ................................................▐
 5  ................................................▐
 6  ................................................▐
 7  ................................................▐
 8  ................................................▼
 9
10
11                    <  OK  > <Cancel>
```

Window name: W_list
Coordinates: FROM 0,0 TO 0,51
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mchoice	List	"@&T"
2:	Mbuttons	Push button	"@*HT OK;\?Cancel"

W10.SCX Last updated: 06/03/92 at 13:39

```
                    Confirm
 0
 1  1: ............................
 2  ...............................
 3  ...............................
 4  ...............................
 5
 6        <  OK  > <Cancel>
```

	Name	Type	Picture
1:	m.prompt	Field	
2:	mbuttons	Push button	"@*HT OK;Cancel"

W7.SCX Last updated: 05/28/92 at 8:41

Diseases on File

```
 0
 1  1: mchoice.......................▲
 2  ................................♦
 3  ................................
 4  ................................
 5  ................................
 6  ................................
 7  ................................
 8  ................................
 9  ................................▼
10
11
12  <Select>    < View >    <Cancel>
```

Window name: W_disease
Coordinates: FROM 0,0 TO 0,38
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mchoice	List	"@&T"
2: mbuttons	Push button	"@*HT Select;View;\?Cancel"

W6.SCX Last updated: 04/02/92 at 17:27

Numeric Data Entry

```
 0
 1  2: ..............................................
 2  ................................................
 3  ................................................
 4
 5  Range 5: mlo....  to  3: mhi....
 6
 7
 8     1: mval...              4: munits.....
 9
10
11
12                < OK >     <Cancel>
```

Window name: W_num
Coordinates: FROM 0,0 TO 0,54
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mval	Field	"@KZ"
2: mprompt	Field	"@T"
3: mhi	Field	"@BZ"
4: munits	Field	"@T"
5: mlo	Field	"@BZ"
6: mbuttons	Push button	"@*HT OK;\?Cancel"

W6C.SCX Last updated: 04/02/92 at 17:29

Logical Data Entry

```
 0
 1  2: ..............................................
 2  ................................................
 3  ................................................
 4
 5
 6                 (•) Yes
 7                 ( ) No
 8
 9
10
11                 <  OK  >     <Cancel>
```

Window name: W_check
Coordinates: FROM 0,0 TO 5,53
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mbuttons	Push button	"@*HT OK;\?Cancel"
2: mprompt	Field	"@T"
3: mchoice	Radio button	"@*RVN Yes;No"

W6B.SCX Last updated: 04/02/92 at 17:28

Single Choice Data Entry

```
 0
 1  2: ...................................................
 2  ......................................................
 3  ......................................................
 4
 5
 6  3: mchoice...........................................▲
 7  ......................................................♦
 8  ......................................................▐
 9  ......................................................▐
10  ......................................................▐
11  ......................................................▼
12
13
14                    <  OK  >     <Cancel>
```

Window name: W_check
Coordinates: FROM 0,0 TO 7,62
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mbuttons	Push button	"@*HT OK;\?Cancel"
2: mprompt	Field	"@T"
3: mchoice	List	"@&T"

W11.SCX Last updated: 06/09/92 at 15:20

Pending Encounters

```
 0
 1  Patient  3: mkey...
 2
 3  Date     4: mdate..
 4
 5
 6
 7  1: mchoice.............................................▲
 8  ......................................................♦
 9  ......................................................▐
10  ......................................................▐
11  ......................................................▐
12  ......................................................▐
13  ......................................................▼
14
15
16            <Resume>     < New  >     <Cancel>
```

Window name: W_pend
Coordinates: FROM 0,0 TO 0,58
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mchoice	List	"@&N"
2: mbuttons	Push button	"@*HT Resume;New;\?Cancel"
3: mkey	Field	
4: mdate	Field	

W12.SCX Last updated: 10/13/92 at 14:16

Patient Encounter

```
 0 Date 1: me_da  Time 2: me  Status 3: m  Provider 4: me_providr...
 1 Patient Name 5: me_lastnam............  First 6: me_firstna......
 2 SSN 7: me_ssn..            Sex 8: me_   Age 9:
 3
 4 <Complaint > 11: me_area.......   <Diagnosis> 15: me_diagnos.....
 5 ┌──────────<Symptom>──────────22┐ ┌──────────<Assist>──────────22┐
 6 │                               │ │                              │
 7 │ 13: mcondL................... ▲│ │ 17: mdiagL................... ▲│
 8 │...............................│ │..............................│
 9 │...............................│ │..............................│
10 │...............................│ │..............................│
11 │...............................▼│ │..............................▼│
12 │                               │ │                              │
13 └───────────────────────────────┘ └──────────────────────────────┘
14 └──────────────────────<Treatment >──────────────────────────────21
15 20: me_treat...........................................................▲
16 ..........................................................................
17 ..........................................................................▼
18
19                    <Print >    «Save»    <Cancel>
```

Window name: W_encount
Coordinates: FROM 0,0 TO 0,79
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	me_dateenc	Field	
2:	me_timeenc	Field	
3:	me_state	Field	
4:	me_providr	Field	
5:	me_lastnam	Field	
6:	me_firstna	Field	"@!"
7:	me_ssn	Field	"@R 999-99-9999"
8:	me_sex	Field	"@!"
9:	me_age	Field	
10:	mckcompl	Push button	"@*HN \<Complaint"
11:	me_area	Field	
12:	mckhandp	Push button	"@*HN \<Symptom"
13:	mcondL	List	"@&N"
14:	mckdiag	Push button	"@*HN \<Diagnosis"
15:	me_diagnos	Field	
16:	mckassist	Push button	"@*HN \<Assist"
17:	mdiagL	List	"@&N"
18:	mcktreat	Push button	"@*HN \<Treatment"
19:	chr(30)	Field	
20:	me_treat	Field	
21:	mbuttons	Push button	"@*HT \<Print;\!\?Sa\<ve;Ca\<ncel"
22:	chr(31)	Field	
23:	chr(31)	Field	
24:	chr(30)	Field	
25:	chr(31)	Field	
26:	chr(30)	Field	

W2.SCX Last updated: 06/15/92 at 15:01

Print Encounter

```
 0
 1  Print Encounter to:     (*) File
 2                          ( ) Printer
 3
 4
 5
 6  Filename: 1: mprintfile................
 7
 8
 9
10          <  OK  >     <Cancel>
```

Window name: W_prn
Coordinates: FROM 0,0 TO 0,42
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mprintfile	Field	
2: mbuttons	Push button	"@*HT OK;\?Cancel"
3: mchoice	Radio button	"@*RVN File;Printer"

W9.SCX Last updated: 04/02/92 at 17:33

Warning

```
 0
 1  1: ............................
 2  ..............................
 3  ..............................
 4
 5                <OK>
```

Window name: W_popmsg
Coordinates: FROM 0,0 TO 0,35
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: errstr	Field	
2: mbuttons	Push button	"@*HT OK"

W5.SCX Last updated: 05/28/92 at 8:17

Complaints

```
 0
 1   2: mchoice................▲
 2   ..........................♦   <Select>
 3   ..........................
 4   ..........................     <Cancel>
 5   ..........................
 6   ..........................
 7   ..........................▼
 8
```

Window name: W_area
Coordinates: FROM 0,0 TO 8,45
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mbuttons	Push button	"@*VT Select;\?Cancel"
2: mchoice	List	"@&T"

W3.SCX Last updated: 05/28/92 at 8:15

Patients on File

```
 0
 1   1: mchoice......................▲
 2   ...............................♦
 3   ...............................
 4   ...............................
 5   ...............................
 6   ...............................
 7   ...............................
 8   ...............................
 9   ...............................▼
10
11
12        <Select>     <Cancel>
```

Window name: W_patlist
Coordinates: FROM 0,0 TO 0,36
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mchoice	List	"@&T"
2: mbuttons	Push button	"@*HT Select;\?Cancel"

W13.SCX Last updated: 02/12/92 at 10:21

Undo Options

```
 0
 1
 2    1: mchoice.............▲
 3    ......................♦
 4    ......................▐
 5    ......................▐
 6    ......................▼
 7
 8
 9       <  OK  > <Cancel>
```

Window name: W_undo
Coordinates: FROM 0,0 TO 0,31
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mchoice	List	"@&T"
2: Mbuttons	Push button	"@*HT OK;\?Cancel"

WP3.SCX Last updated: 08/03/92 at 16:03

Patient in File

```
 0
 1    1: mchoice....................▲
 2    ..............................♦
 3    ..............................▐
 4    ..............................▐
 5    ..............................▐
 6    ..............................▐
 7    ..............................▐
 8    ..............................▐
 9    ..............................▐
10    ..............................▼
11
12   <Select>     < Edit >     <Cancel>
```

Window name: Wp3
Coordinates: FROM 0,0 TO 0,40
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mchoice	List	"@&T"
2: mbuttons	Push button	"@*HT \<Select;\<Edit;\?\<Cancel"

```
WP4.SCX                Last updated:  06/14/92 at 17:37

                                Encounter to Report

 0
 1    Social Security: 1: mssn.....
 2
 3    Last Name:       2: mlast......................
 4
 5    First Name:.     3: mfirst.....................
 6
 7    Encounter date:  4: mdateen
 8
 9            « OK »   <Browse>   <Cancel>
```

```
   Window name: Wp4
   Coordinates: FROM 0,0 TO 0,55
 Window options: FLOAT CLOSE MINIMIZE SHADOW
```

Name	Type	Picture
1: mssn	Field	"@R 999-99-9999"
2: mlast	Field	
3: mfirst	Field	
4: mdateenc	Field	
5: mbuttons	Push button	"@*HT \!\<OK;\<Browse;\<Cancel"

```
WP5.SCX                Last updated:  06/14/92 at 18:30

 0 Patient  3: mkey.....................................
 1
 2 Date     4: mdate..
 3
 4
 5  1: mchoice.........................................▲
 6  ..................................................♦
 7  ..................................................
 8  ..................................................
 9  ..................................................
10  ..................................................
11  ..................................................▼
12
13
14                   <  OK  >     <Cancel>
```

```
   Window name: Wp4
   Coordinates: FROM 0,0 TO 0,58
 Window options: FLOAT CLOSE MINIMIZE SHADOW
```

Name	Type	Picture
1: mchoice	List	"@&N"
2: mbuttons	Push button	"@*HT \<OK;\?Cancel"
3: mkey	Field	
4: mdate	Field	

WP6.SCX Last updated: 06/16/92 at 13:04

Report Editing

```
 0
 1  1: mstr.......................................................
 2  .............................................................
 3  .............................................................
 4  .............................................................
 5  .............................................................
 6  .............................................................
 7  .............................................................
 8  .............................................................
 9  .............................................................
10  .............................................................
11  .............................................................
12  .............................................................
13  .............................................................
14  .............................................................
15  .............................................................
16
17                    <Print >    < Save >    <Cancel>
```

Window name: Wp6
Coordinates: FROM 0,0 TO 0,76
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mstr	Field	
2: mbuttons	Push button	"@*HT \<Print;\<Save;\<Cancel"

WP7.SCX Last updated: 06/16/92 at 13:00

Print Option

```
 0
 1  Print Encounter to:     (•) File
 2                          ( ) Printer
 3
 4
 5
 6  Filename: 1: mprintfile................
 7
 8
 9
10          <  OK  >     <Cancel>
```

Window name: Wp7
Coordinates: FROM 0,0 TO 0,42
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mprintfile	Field	
2: mbuttons	Push button	"@*HT OK;\?Cancel"
3: mchoice	Radio button	"@*RVN File;Printer"

WP8.SCX Last updated: 06/16/92 at 9:32

Patient in File

```
 0
 1    1: mchoice.......................▲
 2    ................................◆
 3    ................................▒
 4    ................................▒
 5    ................................▒
 6    ................................▒
 7    ................................▒
 8    ................................▒
 9    ................................▒
10    ................................▼
11
12          <Select>     <Cancel>
```

Window name: Wp3
Coordinates: FROM 0,0 TO 0,40
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mchoice	List	"@&T"
2: mbuttons	Push button	"@*HT \<Select;\?\<Cancel"

WP9.SCX Last updated: 06/24/92 at 10:32

```
 0  1: ............................................................
 1  ..............................................................
 2  ..............................................................
 3  ..............................................................
```

Window name: Wp9
Coordinates: FROM 1,0 TO 4,76
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mtext	Field	

W12A.SCX Last updated: 07/06/92 at 10:38

```
 0  ┌──────────────────────────────<Symptom>──────────────────────────────34┐
 1  │
 2  │  ┌──────────────────────────────────────────────────────────────────┐
 3  │  │1: mcondL.........................................................▲│
 4  │  │..................................................................│
 5  │  │..................................................................│
 6  │  │..................................................................│
 7  │  │..................................................................│
 8  │  │..................................................................│
 9  │  │..................................................................│
10  │  │..................................................................│
11  │  │..................................................................│
12  │  │..................................................................│
13  │  │..................................................................│
14  │  │..................................................................│
15  │  │..................................................................│
16  │  │..................................................................│
17  │  │..................................................................│
18  │  │..................................................................│
19  │  │..................................................................▼│
20  │  └──────────────────────────────────────────────────────────────────┘
```

Window name: W_handp
Coordinates: FROM 0,0 TO 0,79
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mcondL	List	"@&N"
2:	mckhandp	Push button	"@*HN \<Symptom"
3:	chr(31)	Field	
4:	chr(30)	Field	

W12B.SCX Last updated: 07/06/92 at 10:41

```
 0 ┌──────────────────────────────<Assist>──────────────────────────────43┐
 1 │ ┌──────────────────────────────────────────────────────────────────┐
 2 │ │1: mdiagL.........................................................▲│
 3 │ │.................................................................│
 4 │ │.................................................................│
 5 │ │.................................................................│
 6 │ │.................................................................│
 7 │ │.................................................................│
 8 │ │.................................................................│
 9 │ │.................................................................│
10 │ │.................................................................│
11 │ │.................................................................│
12 │ │.................................................................│
13 │ │.................................................................│
14 │ │.................................................................│
15 │ │.................................................................│
16 │ │.................................................................│
17 │ │.................................................................│
18 │ │.................................................................▼│
19 │ └──────────────────────────────────────────────────────────────────┘
20 └──────────────────────────────────────────────────────────────────────┘
```

Window name: W_assist
Coordinates: FROM 0,0 TO 0,79
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mdiagL	List	"@&N"
2: mckassist	Push button	"@*HN \<Assist"
3: chr(30)	Field	
4: chr(31)	Field	

W12C.SCX Last updated: 07/06/92 at 10:41

Treatment

```
 0 ┌──────────────────────────────<Treatment >──────────────────────────────32┐
 1 │
 2 │ 1: me_treat.......................................................... ▲
 3 │ ......................................................................... ♦
 4 │ ......................................................................... ▌
 5 │ ......................................................................... ▌
 6 │ ......................................................................... ▌
 7 │ ......................................................................... ▌
 8 │ ......................................................................... ▌
 9 │ ......................................................................... ▌
10 │ ......................................................................... ▌
11 │ ......................................................................... ▌
12 │ ......................................................................... ▌
13 │ ......................................................................... ▌
14 │ ......................................................................... ▌
15 │ ......................................................................... ▌
16 │ ......................................................................... ▌
17 │ ......................................................................... ▌
18 │ ......................................................................... ▼
19 │
```

Window name: W_treat
Coordinates: FROM 0,0 TO 0,79
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: me_treat	Field	
2: chr(30)	Field	
3: chr(31)	Field	
4: mcktreat	Push button	"@*HN \<Treatment"

W12E.SCX Last updated: 07/20/92 at 12:20

```
0 1: ..............................................................
1 ....................................................................
```

Window name: W_warning
Coordinates: FROM 23,0 TO 24,79
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mprivateact	Field	77

BACKUP.SCX Last updated: 10/09/92 at 12:57

```
 0  2: message.............................
 1
 2
 3  5: mfile...............▲   To drive:.  1: mdrive.......
 4  ......................
 5  ......................
 6  ......................   Directory:   3: mpath........
 7  ......................
 8  ......................
 9  ......................          « Back  »
10  ......................
11  ......................▼          <Cancel>
12
13
14  Backup file name:.6: mfname...............................
```

Window name: W_backup
Coordinates: FROM 0,0 TO 0,60
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mdrive	Popup	"@^ "
2:	message	Field	
3:	mpath	Popup	"@^ "
4:	maction	Push button	"@*VT \!\<Backup;\<Cancel"
5:	mfile	List	"@&N"
6:	mfname	Field	

RESTORE.SCX Last updated: 10/09/92 at 14:09

```
 0 2: message...............................
 1
 2
 3   5: mfile...............▲  From drive:. 1: mdrive.......
 4   .......................
 5   .......................
 6   .......................  Directory:    3: mpath........
 7   .......................
 8   .......................
 9   .......................        «Restore»
10   .......................
11   .......................▼        <Cancel >
12
13
14 Restore file name:.6: mfname...............................
```

Window name: W_restore
Coordinates: FROM 0,0 TO 0,60
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mdrive	Popup	"@^ "
2: message	Field	
3: mpath	Popup	"@^ "
4: maction	Push button	"@*VT \!\<Restore;\<Cancel"
5: mfile	List	"@&N"
6: mfname	Field	

W7A.SCX Last updated: 08/03/92 at 16:04

Disease/Treatment Reference

```
 0
 1   1: mchoice......................▲
 2   ................................
 3   ................................
 4   ................................
 5   ................................
 6   ................................
 7   ................................
 8   ................................
 9   ................................
10   ................................
11   ................................
12   ................................
13   ................................▼
14
15            « View »    < Quit >
```

Window name: W_dislook
Coordinates: FROM 0,0 TO 0,39
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mchoice	List	"@&T"
2: mbuttons	Push button	"@*HT \!\<View;\<Quit"

COLORSET.SCX Last updated: 10/07/92 at 11:39

```
 0
 1   1: coloridx.............▲
 2   .......................
 3   .......................
 4   .......................
 5   .......................
 6   .......................
 7   .......................
 8   .......................▼
 9
10
11       < Load >   <Cancel>
```

Window name: W_color
Coordinates: FROM 0,0 TO 0,31
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: m.coloridx	List	"@&N"
2: m.action	Push button	"@*HT \<Load;\<Cancel"

section IV. Report Form File Summary. *The only one report forms was created in CAMD system; SF600.FRX. Last updated: 07/09/92 at 16:02*

Report Contents

No.	Field	Length	Row	Column
1	PAGENO	4	0	73
2	mdateenc	8	11	0
3	mtext	65	11	11
4	mtimeenc	8	12	0
5	msocsec	11	53	0
6	alltrim(mlastname)+ ", " + alltrim(mfirstname)+ " " + alltrim(mmiddlename)	28	53	14
7	mdatebirth	12	53	45
8	msex	6	53	60
9	mraterank	10	53	69
		152		

Database and Program References

There is no an associated database

Used by: SFPRINT *(procedure in C:\CAMD\WORK\U.PRG)*

Section V. Data Dictionary. *There are eighteen (18) database files in the CAMD system:*

SECTION.DBF	--	SF600 report section.
REPORT.DBF	--	SF600 report of each CAMD system area.
SF600TEX.DBF	--	The natural language of sign and symptom, use to generate SF600 report.
AREA.DBF	--	Diagnosis area of the CAMD System.
DICT.DBF	--	Sign & Symptom Question.
VAL.DBF	--	Range of Numeric type
ENUM.DBF	--	The response to the signs and symptoms question
DISEASE.DBF	--	Disease description and Treatment protocol.
PREMISE.DBF	--	Rule premise file.
ACTION.DBF	--	Rule action file.
RULE.DBF	--	Rules database file.
SYSTEM.DBF	--	CAMD system configure, greeting message.
PATIENT.DBF	--	Patient demographic database.
ENCOUNTR.DBF	--	Patient encounter record.
SYMPTOMS.DBF	--	Sign and Symptoms input record.
DIAGNOSE.DBF	--	Diagnosis output by CAMD system.
BACKUP.DBF	--	List of database file to be backup.
KEYLOG.DBF	--	Action record of user using CAMD system.

Database Structure Summary

```
Structure for database : SECTION.DBF
Number of data records :        81
          Last updated : 09/23/92 at 12:12

Field  Field name  Type        Width    Dec    Start      End
    1  SECTION     Numeric         3               1        3
    2  OBJECT      Character       1               4        4
    3  ID          Numeric         5               5        9
    4  SENTENCE    Numeric         3              10       12
** Total **                       13

Used by: SFSYMP                    (procedure in DEMO.PRG)
       : REINDEX                   (procedure in U.PRG)


Structure for database : REPORT.DBF
Number of data records :        59
          Last updated : 09/23/92 at 12:13
Field  Field name  Type        Width    Dec    Start      End
    1  SECTION     Numeric         3               1        3
    2  AREA        Numeric         4               4        7
** Total **                        8

Used by: STRNATURAL                (procedure in DEMO.PRG)
       : SFSYMP                    (procedure in DEMO.PRG)
       : REINDEX                   (procedure in U.PRG)
```

```
Structure for database : SF600TEX.DBF
Number of data records :        81
          Last updated : 09/23/92 at 12:12

Field  Field name  Type        Width    Dec    Start      End
    1  SENTENCE    Numeric         3                1        3
    2  OBJECT      Character       1                4        4
    3  ID          Numeric         5                5        9
    4  STARTTEXT   Character      60               10       69
    5  ENDTEXT     Character      60               70      129
    6  DEFAULT     Character     120              130      249
    7  NOPREVSEN   Character      60              250      309
** Total **                      310

Used by: SFSYMP                    (procedure in DEMO.PRG)
       : SFPRINT                   (procedure in U.PRG)
       : REINDEX                   (procedure in U.PRG)


Structure for database : AREA.DBF
Number of data records :         2
          Last updated : 08/03/92 at 11:16
Field  Field name  Type        Width    Dec    Start      End
    1  AREA        Numeric         4                1        4
    2  NAME        Character      30                5       34
    3  RULEBASE    Character      10               35       44
    4  SIGNON      Character      30               45       74
    5  START       Character      30               75      104
    6  FINISH      Character      30              105      134
    7  THRESHOLD   Numeric         6        2      135      140
    8  PROBABLE    Numeric         6        2      141      146
    9  LIKELY      Numeric         6        2      147      152
   10  RULES       Numeric         4               153      156
** Total **                      157

Used by: GETEXAREA()               (function in U.PRG)
       : GETRULEBASE()             (function in U.PRG)
       : AREACOUNT()               (function in U.PRG)
       : AREANAMES()               (function in U.PRG)
       : ENCNAMES()                (function in U.PRG)
       : SFENCNAMES                (procedure in U.PRG)
       : OPENFILE                  (procedure in QUESTION.PRG)


Structure for database : DICT.DBF
Number of data records :        80
          Last updated : 09/02/92 at 14:36
Field  Field name  Type        Width    Dec    Start      End
    1  ID          Numeric         4                1        4
    2  NAME        Character      20                5       24
    3  DATATYPE    Character       1               25       25
    4  ASKABLE     Logical         1               26       26
    5  QUESTION    Character     100               27      126
    6  PROMPT      Character      80              127      206
** Total **                      207

Used by: DICTCOUNT()               (function in U.PRG)
       : OPENDICT                  (procedure in U.PRG)
       : REINDEX                   (procedure in U.PRG)
       : OPENFILE                  (procedure in QUESTION.PRG)
```

Structure for database : VAL.DBF Alias: DICTVAL
Number of data records : 11
Last updated : 07/23/92 at 8:32

Field	Field name	Type	Width	Dec	Start	End
1	ID	Numeric	4		1	4
2	WIDTH	Numeric	2		5	6
3	DEC	Numeric	1		7	7
4	LO	Numeric	9	2	8	16
5	HI	Numeric	9	2	17	25
6	UNITS	Character	10		26	35
** Total **			36			

Used by: OPENDICT (procedure in U.PRG)
: REINDEX (procedure in U.PRG)
: OPENFILE (procedure in QUESTION.PRG)

Structure for database : ENUM.DBF
Number of data records : 251
Last updated : 07/23/92 at 11:09

Field	Field name	Type	Width	Dec	Start	End
1	ID	Numeric	4		1	4
2	ORD	Numeric	2		5	6
3	MUTEX	Character	1		7	7
4	ENUMERATE	Character	80		8	87
5	REPORT	Character	80		88	167
** Total **			168			

Used by: OPENDICT (procedure in U.PRG)
: REINDEX (procedure in U.PRG)
: OPENFILE (procedure in QUESTION.PRG)

Structure for database : DISEASE.DBF
Number of data records : 20
Last updated : 10/08/92 at 11:58

Field	Field name	Type	Width	Dec	Start	End
1	NAME	Character	60		1	60
2	DESCRIPT	Memo	10		61	70
3	ID	Numeric	5		71	75
4	SOURCE	Character	1		76	76
5	XID	Numeric	5		77	81
6	AREA	Numeric	4		82	85
7	TREATMENT	Memo	10		86	95
8	BRIEF	Memo	10		96	105
** Total **			106			

This database is associated with the memo file: DISEASE.FPT

Used by: DISEASENAM() (function in U.PRG)
: DISEASEID() (function in U.PRG)
: DISEASECNT() (function in U.PRG)
: DISNAMES() (function in U.PRG)
: DISEASEGET (procedure in U.PRG)
: DISEASEPUT (procedure in U.PRG)
: REINDEX (procedure in U.PRG)

```
Structure for database : PREMISE.DBF
Number of data records :        372
          Last updated : 07/23/92 at 11:22

Field  Field name  Type        Width    Dec    Start      End
    1  CLAUSE      Numeric         5               1        5
    2  OP          Character       2               6        7
    3  OBJECT      Character       1               8        8
    4  ID          Numeric         5               9       13
    5  VAL         Character      30              14       43
** Total **                       44

Used by: OPENRULE                  (procedure in U.PRG)
       : REINDEX                   (procedure in U.PRG)
       : OPENFILE                  (procedure in QUESTION.PRG)


Structure for database : ACTION.DBF
Number of data records :       1779
          Last updated : 06/09/92 at 15:01
Field  Field name  Type        Width    Dec    Start      End
    1  CLAUSE      Numeric         5               1        5
    2  OP          Character       2               6        7
    3  OBJECT      Character       1               8        8
    4  ID          Numeric         5               9       13
    5  VAL         Character      30              14       43
** Total **                       44

Used by: OPENRULE                  (procedure in U.PRG)
       : REINDEX                   (procedure in U.PRG)
       : OPENFILE                  (procedure in QUESTION.PRG)


Structure for database : RULE.DBF
Number of data records :        330
          Last updated : 08/04/92 at 10:27

  ld   Field name  Type        Width    Dec    Start      End
    1  RULE        Numeric         5               1        5
    2  AREA        Numeric         4               6        9
    3  SALIENCE    Numeric         3              10       12
    4  PREMISE     Numeric         5              13       17
    5  ACTION      Numeric         5              18       22
    6  TEXT        Memo           10              23       32
** Total **                       33

This database is associated with the memo file: RULE.FPT

Used by: OPENRULE                  (procedure in U.PRG)
       : REINDEX                   (procedure in U.PRG)
       : OPENFILE                  (procedure in QUESTION.PRG)
```

Structure for database : SYSTEM.DBF
Number of data records : 1
Last updated : 10/13/92 at 9:58

Field	Field name	Type	Width	Dec	Start	End
1	PASSWORD	Character	10		1	10
2	PATIENTS	Numeric	4		11	14
3	PROVIDERS	Numeric	4		15	18
4	ENCOUNTERS	Numeric	4		19	22
5	STATION	Character	30		23	52
6	DEPARTMENT	Character	30		53	82
7	SYSWARNING	Memo	10		83	92
8	SYSDISCL	Memo	10		93	102
9	PRIVACYACT	Memo	10		103	112
10	DEBUG	Logical	1		113	113
11	CONF	Logical	1		114	114
12	SHOWRULE	Logical	1		115	115
13	NOVICE	Logical	1		116	116
14	EDITHELP	Logical	1		117	117
** Total **			118			

This database is associated with the memo file: SYSTEM.FPT

Used by: PWDEDT.PRG
: NEXTENCNT() (function in U.PRG)
: GETSTATION() (function in U.PRG)
: GETOPTS (procedure in U.PRG)
: NEXTPAT (procedure in U.PRG)
: NENCNT (procedure in U.PRG)
: NEXTUSER (procedure in U.PRG)
: GETSYSCODE() (function in SECHECK.PRG)
: OPENSAMS (procedure in U.PRG)

Structure for database : PATIENT.DBF
Number of data records : 2
Last updated : 10/09/92 at 14:11

Field	Field name	Type	Width	Dec	Start	End
1	ID	Numeric	4		1	4
2	SOCSEC	Character	9		5	13
3	LASTNAME	Character	30		14	43
4	FIRSTNAME	Character	30		44	73
5	MIDLEINIT	Character	1		74	74
6	DATEBIRTH	Date	8		75	82
7	SEX	Character	1		83	83
8	RATERANK	Character	20		84	103
** Total **			104			

Used by: REINDEX (procedure in U.PRG)

```
Structure for database : ENCOUNTR.DBF
Number of data records :          1
          Last updated : 10/13/92 at 10:19

Field  Field name  Type        Width    Dec    Start      End
    1  ENCOUNTRID  Numeric       5                1        5
    2  PATIENT     Numeric       4                6        9
    3  DATE        Date          8               10       17
    4  TIME        Character     8               18       25
    5  PROVIDER    Numeric       4               26       29
    6  STATION     Character    30               30       59
    7  DEPARTMENT  Character    30               60       89
    8  AREA        Numeric       5               90       94
    9  DIAGNOSE    Numeric       5               95       99
   10  COMPLAINT   Character    30              100      129
   11  TREATMENT   Memo         10              130      139
   12  REPORT      Memo         10              140      149
   13  STATE       Character     1              150      150
** Total **                    151

This database is associated with the memo file: ENCOUNTR.FPT

Used by: REINDEX                    (procedure in U.PRG)


Structure for database : SYMPTOMS.DBF
Number of data records :          2
          Last updated : 10/13/92 at 10:19

Field  Field name  Type        Width    Dec    Start      End
    1  ID          Numeric       5                1        5
    2  SYMPTOM     Numeric       4                6        9
    3  VALUE       Character    10               10       19
** Total **                     20

Used by: REINDEX                    (procedure in U.PRG)


Structure for database : DIAGNOSE.DBF
Number of data records :          0
          Last updated : 09/03/92 at 12:45
Field  Field name  Type        Width    Dec    Start      End
    1  ID          Numeric       5                1        5
    2  DISEASE     Numeric       5                6       10
    3  PROBABILTY  Numeric       6      2        11       16
** Total **                     17

Used by: REINDEX                    (procedure in U.PRG)


Structure for database : BACKUP.DBF
Number of data records :         17
          Last updated : 09/03/92 at 12:44
Field  Field name  Type        Width    Dec    Start      End
    1  FILE_NAME   Character    12                1       12
** Total **                     13

This database appears to be associated with multiple index file(s):
       : BACKUP.CDX
       : PKUNZIP                    (procedure in RESTORE.SPR)
```

Structure for database : KEYLOG.DBF
Number of data records : 34
Last updated : 10/13/92 at 10:29

Field	Field name	Type	Width	Dec	Start	End
1	ACTION	Character	60		1	60
2	DATE	Date	8		61	68
3	TIME	Character	8		69	76
** Total **			77			

Used by: KEYLOG.PRG

Structure for database : COLORSET.DBF
Number of data records : 7
Last updated : 10/07/92 at 11:52

Field	Field name	Type	Width	Dec	Start	End
1	COLOR_SET	Character	30		1	30
** Total **			31			

Used by: COLORSET.SPR

Database Field Summary: A listing of alphabetical database's field name was used in the CAMD system.

Field Name	Type	Len	Dec	Database
ACTION	C	60	0	KEYLOG.DBF
ACTION	N	5	0	RULE.DBF
AREA	N	4	0	RULE.DBF
				REPORT.DBF
AREA	N	5	0	ENCOUNTR.DBF
AREA	N	4	0	AREA.DBF
				DISEASE.DBF
ASKABLE	L	1	0	DICT.DBF
BRIEF	M	10	0	DISEASE.DBF
CLAUSE	N	5	0	ACTION.DBF
				PREMISE.DBF
COLOR_SET	C	30	0	COLORSET.DBF
COMPLAINT	C	30	0	ENCOUNTR.DBF
CONF	L	1	0	SYSTEM.DBF
DATATYPE	C	1	0	DICT.DBF
DATE	D	8	0	KEYLOG.DBF
				ENCOUNTR.DBF
DATEBIRTH	D	8	0	PATIENT.DBF
DEBUG	L	1	0	SYSTEM.DBF
DEC	N	1	0	VAL.DBF
DEFAULT	C	120	0	SF600TEX.DBF
DEPARTMENT	C	30	0	ENCOUNTR.DBF
				SYSTEM.DBF
DESCRIPT	M	10	0	DISEASE.DBF
DIAGNOSE	N	5	0	ENCOUNTR.DBF
DISEASE	N	5	0	DIAGNOSE.DBF
EDITHELP	L	1	0	SYSTEM.DBF
ENCOUNTERS	N	4	0	SYSTEM.DBF
ENCOUNTRID	N	5	0	ENCOUNTR.DBF
ENDTEXT	C	60	0	SF600TEX.DBF
ENUMERATE	C	80	0	ENUM.DBF
FILE_NAME	C	12	0	BACKUP.DBF
FINISH	C	30	0	AREA.DBF
FIRSTNAME	C	30	0	PATIENT.DBF
HI	N	9	2	VAL.DBF

ID	N	4	0	VAL.DBF
ID	N	5	0	SECTION.DBF
				DISEASE.DBF
				SYMPTOMS.DBF
				SF600TEX.DBF
ID	N	4	0	DICT.DBF
				ENUM.DBF
ID	N	5	0	DIAGNOSE.DBF
				PREMISE.DBF
ID	N	4	0	PATIENT.DBF
LASTNAME	C	30	0	PATIENT.DBF
LIKELY	N	6	2	AREA.DBF
LO	N	9	2	VAL.DBF
MIDLEINIT	C	1	0	PATIENT.DBF
MUTEX	C	1	0	ENUM.DBF
NAME	C	60	0	DISEASE.DBF
NAME	C	30	0	AREA.DBF
NAME	C	20	0	DICT.DBF
NOPREVSEN	C	60	0	SF600TEX.DBF
NOVICE	L	1	0	SYSTEM.DBF
OBJECT	C	1	0	SECTION.DBF
				SF600TEX.DBF
				ACTION.DBF
				PREMISE.DBF
OP	C	2	0	ACTION.DBF
				PREMISE.DBF
ORD	N	2	0	ENUM.DBF
PASSWORD	C	10	0	SYSTEM.DBF
PATIENT	N	4	0	ENCOUNTR.DBF
PATIENTS	N	4	0	SYSTEM.DBF
PREMISE	N	5	0	RULE.DBF
PRIVACYACT	M	10	0	SYSTEM.DBF
PROBABILTY	N	6	2	DIAGNOSE.DBF
PROBABLE	N	6	2	AREA.DBF
PROMPT	C	80	0	DICT.DBF
PROVIDER	N	4	0	ENCOUNTR.DBF
PROVIDERS	N	4	0	SYSTEM.DBF
QUESTION	C	100	0	DICT.DBF
RATERANK	C	20	0	PATIENT.DBF
REPORT	M	10	0	ENCOUNTR.DBF
REPORT	C	80	0	ENUM.DBF
RULE	N	5	0	RULE.DBF
RULEBASE	C	10	0	AREA.DBF
RULES	N	4	0	AREA.DBF
SALIENCE	N	3	0	RULE.DBF
SECTION	N	3	0	REPORT.DBF
				SECTION.DBF
				SF600TEX.DBF
SEX	C	1	0	PATIENT.DBF
SHOWRULE	L	1	0	SYSTEM.DBF
SIGNON	C	30	0	AREA.DBF
SOCSEC	C	9	0	PATIENT.DBF
SOURCE	C	1	0	DISEASE.DBF
START	C	30	0	AREA.DBF
STARTTEXT	C	60	0	SF600TEX.DBF
STATE	C	1	0	ENCOUNTR.DBF
STATION	C	30	0	SYSTEM.DBF
				ENCOUNTR.DBF
SYMPTOM	N	4	0	SYMPTOMS.DBF
SYSDISCL	M	10	0	SYSTEM.DBF
SYSWARNING	M	10	0	SYSTEM.DBF
TEXT	M	10	0	RULE.DBF

TEXT	M	10	0	RULE.DBF
THRESHOLD	N	6	2	AREA.DBF
TIME	C	8	0	KEYLOG.DBF
				ENCOUNTR.DBF
TREATMENT	M	10	0	ENCOUNTR.DBF
				DISEASE.DBF
UNITS	C	10	0	VAL.DBF
VAL	C	30	0	PREMISE.DBF
				ACTION.DBF
VALUE	C	10	0	SYMPTOMS.DBF
WIDTH	N	2	0	VAL.DBF
XID	N	5	0	DISEASE.DBF

Index Parameter Summary: *There are twenty five index associate to the database files; SECTION.IDX, REPORT.IDX, SF600TEX.IDX, AREA.IDX, DICTID.IDX, DICTNAME.IDX, VAL.IDX, ENUM.IDX, DISEASE.IDX, DISID.IDX, PREMISE.IDX, ACTION.IDX, SALIENCE.IDX, RULE.IDX, PATIENT.IDX, PATNAME.IDX, PATID.IDX, ENCOUNTR.IDX, ENCOUNTD.IDX, ENCOUNTI.IDX, SYMPTOMS.IDX, DISAREA.IDX, DIAGNOSE.IDX, KEY.IDX, KEYDATE.IDX.*

```
SECTION.IDX -- Indexed on: SECTION

This index file appears to be associated with database(s):
       : SECTION.DBF

Used by: SFSYMP()                  (function in DEMO.PRG)
       : REINDEX()                 (function in U.PRG)

REPORT.IDX -- Indexed on: AREA

This index file appears to be associated with database(s):
       : REPORT.DBF

Used by: SFSYMP()                  (function in DEMO.PRG)

SF600TEX.IDX -- Indexed on: SENTENCE

This index file appears to be associated with database(s):
       : SF600TEX.DBF

Used by: SFSYMP()                  (function in DEMO.PRG)
       : SFPRINT                   (procedure in U.PRG)
       : REINDEX()                 (function in U.PRG)

AREA.IDX -- Indexed on: AREA

This index file appears to be associated with database(s):
       : AREA.DBF

Used by: GETEXAREA()               (function in U.PRG)
       : GETRULEBASE()             (function in U.PRG)
       : AREANAMES()               (function in U.PRG)
       : ENCNAMES()                (function in U.PRG)
       : SFENCNAMES()              (function in U.PRG)
       : OPENFILE()                (function in QUESTION.PRG)

DICTID.IDX -- Indexed on: ID

This index file appears to be associated with database(s):
       : DICT.DBF

Used by: OPENDICT()                (function in U.PRG)
```

```
        : OPENFILE()                    (function in QUESTION.PRG)

DICTNAME.IDX -- Indexed on: NAME

This index file appears to be associated with database(s):
        : DICT.DBF

Used by: OPENDICT()                     (function in U.PRG)
        : REINDEX()                     (function in U.PRG)
        : OPENFILE()                    (function in QUESTION.PRG)

VAL.IDX -- Indexed on: ID

This index file appears to be associated with database(s):
        : VAL.DBF

Used by: OPENDICT()                     (function in U.PRG)
        : REINDEX()                     (function in U.PRG)
        : OPENFILE()                    (function in QUESTION.PRG)

ENUM.IDX -- Indexed on: ID

This index file appears to be associated with database(s):
        : ENUM.DBF

Used by: OPENDICT()                     (function in U.PRG)
        : REINDEX()                     (function in U.PRG)
        : OPENFILE()                    (function in QUESTION.PRG)

DISEASE.IDX -- Indexed on: NAME

This index file appears to be associated with database(s):
        : DISEASE.DBF

Used by: DISEASEID()                    (function in U.PRG)
        : DISNAMES()                    (function in U.PRG)
        : DISEASEGET()                  (function in U.PRG)
        : DISEASEPUT()                  (function in U.PRG)
        : REINDEX()                     (function in U.PRG)

DISID.IDX -- Indexed on: ID

This index file appears to be associated with database(s):
        : DISEASE.DBF

Used by: DISEASENAM()                   (function in U.PRG)
        : DISNAMES()                    (function in U.PRG)
        : REINDEX()                     (function in U.PRG)

PREMISE.IDX -- Indexed on: CLAUSE

This index file appears to be associated with database(s):
        : PREMISE.DBF

Used by: OPENRULE()                     (function in U.PRG)
        : REINDEX()                     (function in U.PRG)
        : OPENFILE()                    (function in QUESTION.PRG)

ACTION.IDX -- Indexed on: CLAUSE

This index file appears to be associated with database(s):
        : ACTION.DBF
```

```
Used by: OPENRULE()                (function in U.PRG)
       : REINDEX()                 (function in U.PRG)
       : OPENFILE()                (function in QUESTION.PRG)

SALIENCE.IDX -- Indexed on: SALIENCE

This index file appears to be associated with database(s):
       : RULE.DBF

Used by: OPENRULE()                (function in U.PRG)
       : REINDEX()                 (function in U.PRG)
       : OPENFILE()                (function in QUESTION.PRG)

RULE.IDX  -- Indexed on: AREA

This index file appears to be associated with database(s):
       : RULE.DBF

Used by: OPENRULE()                (function in U.PRG)
       : REINDEX()                 (function in U.PRG)
       : OPENFILE()                (function in QUESTION.PRG)

PATIENT.IDX -- Index on: SOCSEC

This index file appears to be associated with database(s):
       : PATIENT.DBF

Used by: REINDEX()                 (function in U.PRG)

PATNAME.IDX -- Index on: LASTNAME

This index file appears to be associated with database(s):
       : PATIENT.DBF

Used by: REINDEX()                 (function in U.PRG)

PATID.IDX -- Index on: ID

This index file appears to be associated with database(s):
       : PATIENT.DBF

Used by: REINDEX()                 (function in U.PRG)

ENCOUNTR.IDX -- Index on: PATIENT

This index file appears to be associated with database(s):
       : ENCOUNTR.DBF

Used by: REINDEX()                 (function in U.PRG)

ENCOUNTD.IDX -- Index on: DATE

This index file appears to be associated with database(s):
       : ENCOUNTR.DBF

Used by: REINDEX()                 (function in U.PRG)

ENCOUNTI.IDX -- Index on: ENCOUNTRID

This index file appears to be associated with database(s):
       : ENCOUNTR.DBF
```

```
Used by: REINDEX()
SYMPTOMS.IDX

This index file appears to be associated with database(s):
       : SYMPTOMS.DBF

Used by: REINDEX()                  (function in U.PRG)

DISAREA.IDX -- Index on: AREA

This index file appears to be associated with database(s):
       : DISEASE.DBF

Used by: REINDEX()                  (function in U.PRG)

DIAGNOSE.IDX -- Index on: ID

This index file appears to be associated with database(s):
       : DIAGNOSE.DBF

Used by: REINDEX()                  (function in U.PRG)

KEY.IDX -- Index on: ACTION

This index file appears to be associated with database(s):
       : KEYLOG.DBF

Used by: KEYLOG.PRG

KEYDATE.IDX -- Index on: DATE

This index file appears to be associated with database(s):
       : KEYLOG.DBF

Used by: KEYLOG.PRG

COLORSET.IDX -- Index on: COLOR_SET

This index file appears to be associated with database(s):
       : COLORSET.DBF

Used by: COLORSET.SPR

BACKUP.CDX -- Index on: FILE_NAME

This multiple index file appears to be associated with database(s):
       : BACKUP.DBF

Used by: PKZIP                      (procedure in BACKUP.SPR)
       : PKUNZIP                    (procedure in RESTORE.SPR)
```

Section VI. Tree Diagram. *The tree diagram lists each program that is called by the order in which it is used. Under each program there is a list of all functions called.*

```
DEMO.PRG
├──SET() (FoxPro function)
├──SECHECK.PRG
│   ├──SET() ... (FoxPro function)
│   ├──FULLPATH() (FoxPro function)
│   ├──SYS() (FoxPro function)
│   ├──GETENV() (FoxPro function)
│   ├──GETSYSCODE() (function in SECHECK.PRG)
│   │      SYSTEM.DBF (database)
│   │   ├──SELECT() (FoxPro function)
│   │   ├──USED() (FoxPro function)
│   │   ├──FILE() (FoxPro function)
│   │   ├──FILEMISSING     (procedure in SECHECK.PRG)
│   │   │   └──ERRMSG.PRG
│   │   │       ├──PARAMETERS() (FoxPro function)
│   │   │       ├──EMPTY() (FoxPro function)
│   │   │       ├──TYPE() (FoxPro function)
│   │   │       ├──IIF() (FoxPro function)
│   │   │       ├──INT() (FoxPro function)
│   │   │       ├──VAL() (FoxPro function)
│   │   │       ├──SET() ... (FoxPro function)
│   │   │       ├──LEN() (FoxPro function)
│   │   │       ├──SUBSTR() (FoxPro function)
│   │   │       ├──WOUTPUT() (FoxPro function)
│   │   │       └──WCOL() (FoxPro function)
│   │   ├──LOCFILE() (FoxPro function)
│   │   └──ALLTRIM() (FoxPro function)
│   ├──SYSWARN() (function in SECHECK.PRG)
│   │   └──PARAMETER() (FoxPro function)
│   ├──GETPWD.PRG
│   │   ├──EMPTY() ... (FoxPro function)
│   │   ├──SET() ... (FoxPro function)
│   │   ├──SPACE() (FoxPro function)
│   │   ├──INKEY() (FoxPro function)
│   │   ├──LASTKEY() (FoxPro function)
│   │   ├──SUBSTR() ... (FoxPro function)
│   │   ├──LEN() ... (FoxPro function)
│   │   ├──UPPER() (FoxPro function)
│   │   ├──ERRMSG.PRG...
│   │   └──CHR() (FoxPro function)
│   ├──BIGCHARS.PRG
│   │   ├──PARAMETERS() ... (FoxPro function)
│   │   ├──UPPER() ... (FoxPro function)
│   │   ├──LEN() ... (FoxPro function)
│   │   ├──SYS() ... (FoxPro function)
│   │   ├──SUBSTR() ... (FoxPro function)
│   │   ├──ALLTRIM() ... (FoxPro function)
│   │   ├──STR() (FoxPro function)
│   │   └──EMPTY() ... (FoxPro function)
│   └──SYSCLAIM() (function in SECHECK.PRG)
│       ├──WEXIST() (FoxPro function)
│       ├──INT() ... (FoxPro function)
│       ├──SROW() (FoxPro function)
│       ├──SCOL() (FoxPro function)
│       └──WVISIBLE() (FoxPro function)
└──SYSINIT() (function in DEMO.PRG)
    ├──SET() ... (FoxPro function)
```

```
│      ├──EMPTY() ... (FoxPro function)
│      ├──GETENV() ... (FoxPro function)
│      ├──SETPATH      (procedure in U.PRG)
│      │    ├──EMPTY() ... (FoxPro function)
│      │    ├──FULLPATH() ... (FoxPro function)
│      │    ├──IIF() ... (FoxPro function)
│      │    ├──SET() ... (FoxPro function)
│      │    └──CURDIR() (FoxPro function)
│      └──GETOPTS() (function in U.PRG)
│           │    SYSTEM.DBF (database)
│           ├──SELECT() ... (FoxPro function)
│           └──ALLTRIM() ... (FoxPro function)
├──USERINIT() (function in U.PRG)
│      ├──SET() ... (FoxPro function)
│      ├──INITHELP() (function in U.PRG)
│      │    └──OLMR() (function in U.PRG)
│      │         ├──GETENV() ... (FoxPro function)
│      │         └──SYS() ... (FoxPro function)
│      ├──PUSHHELP() (function in U.PRG)
│      │    ├──ALEN() (FoxPro function)
│      │    └──SETHELP() (function in U.PRG)
│      ├──CHR() ... (FoxPro function)
│      ├──GETENV() ... (FoxPro function)
│      ├──EMPTY() ... (FoxPro function)
│      ├──GETCOLORS() (function in U.PRG)
│      │    ├──SELECT() ... (FoxPro function)
│      │    ├──EOF() (FoxPro function)
│      │    └──TRIM() (FoxPro function)
│      └──PADC() (FoxPro function)
├──POPUPSHOW() (function in U.PRG)
│      ├──WEXIST() ... (FoxPro function)
│      ├──INT() ... (FoxPro function)
│      ├──SROW() ... (FoxPro function)
│      ├──SCOL() ... (FoxPro function)
│      └──WVISIBLE() ... (FoxPro function)
├──INIENCOUNT() (function in DEMO.PRG)
│      ├──CTOD() (FoxPro function)
│      ├──DATE() (FoxPro function)
│      ├──TIME(, (FoxPro function)
│      └──GETSTATION() (function in U.PRG)
│           │    SYSTEM.DBF (database)
│           └──SELECT() ... (FoxPro function)
├──INISYMPTOM() (function in DEMO.PRG)
│      ├──DICTCOUNT() (function in U.PRG)
│      │    │    DICT.DBF (database)
│      │    ├──SELECT() ... (FoxPro function)
│      │    └──RECCOUNT() (FoxPro function)
│      └──DELSYMPTOM() (function in DEMO.PRG)
├──INIDIAGNOS() (function in DEMO.PRG)
│      ├──DISEASECNT() (function in U.PRG)
│      │    │    DISEASE.DBF (database)
│      │    ├──SELECT() ... (FoxPro function)
│      │    └──RECCOUNT() ... (FoxPro function)
│      └──DELDIAGNOS() (function in DEMO.PRG)
│           └──RESETDIAG() (function in DEMO.PRG)
├──POPUPHIDE() (function in U.PRG)
├──IIF() ... (FoxPro function)
├──EMPTY() ... (FoxPro function)
├──OPENFILES() (function in U.PRG)
│      ├──EMPTY() ... (FoxPro function)
│      ├──POPUPSHOW() ... (function in U.PRG)
│      ├──RIGHT() (FoxPro function)
```

```
│   ├──OPENPAT() (function in U.PRG)
│   │   ├──FILEFIND() (function in U.PRG)
│   │   │   ├──PARAMETER() ... (FoxPro function)
│   │   │   ├──FILE() ... (FoxPro function)
│   │   │   ├──DP.PRG
│   │   │   │   ├──PARAMETER() ... (FoxPro function)
│   │   │   │   ├──TYPE() ... (FoxPro function)
│   │   │   │   ├──IIF() ... (FoxPro function)
│   │   │   │   ├──STR() ... (FoxPro function)
│   │   │   │   ├──OCCURS() (FoxPro function)
│   │   │   │   ├──AT() (FoxPro function)
│   │   │   │   ├──LEN() ... (FoxPro function)
│   │   │   │   └──SUBSTR() ... (FoxPro function)
│   │   │   └──LOCFILE() ... (FoxPro function)
│   │   ├──ERRHAND     (procedure in U.PRG)
│   │   │   ├──PARAMETER() ... (FoxPro function)
│   │   │   ├──EMPTY() ... (FoxPro function)
│   │   │   └──ERRMSG.PRG...
│   │   ├──EMPTY() ... (FoxPro function)
│   │   ├──USED() ... (FoxPro function)
│   │   ├──SELECT() ... (FoxPro function)
│   │   └──OPENSAMS() (function in U.PRG)
│   │       ├──EMPTY() ... (FoxPro function)
│   │       ├──GETENV() ... (FoxPro function)
│   │       ├──SETPATH...     (procedure in U.PRG)
│   │       ├──FILEFIND() ... (function in U.PRG)
│   │       ├──USED() ... (FoxPro function)
│   │       └──SELECT() ... (FoxPro function)
│   ├──OPENENCNT() (function in U.PRG)
│   │   ├──USED() ... (FoxPro function)
│   │   └──SELECT() ... (FoxPro function)
│   ├──OPENDICT() (function in U.PRG)
│   │       DICT.DBF (database)
│   │       VAL.DBF (database)
│   │       ENUM.DBF (database)
│   ├──OPENRULE() (function in U.PRG)
│   │       PREMISE.DBF (database)
│   │       ACTION.DBF (database)
│   │       RULE.DBF (database)
│   └──POPUPHIDE() ... (function in U.PRG)
├──BATCHTEST() (function in DEMO.PRG)
│   ├──RIGHT() ... (FoxPro function)
│   ├──EMPTY() ... (FoxPro function)
│   ├──TIME() ... (FoxPro function)
│   ├──DATE() ... (FoxPro function)
│   ├──EOF() ... (FoxPro function)
│   ├──DELSYMPTOM() ... (function in DEMO.PRG)
│   ├──DELDIAGNOS() ... (function in DEMO.PRG)
│   ├──M_ENCRULE() (function in DEMO.PRG)
│   │   └──LODRULE() (function in DEMO.PRG)
│   │       └──GETRULEBASE() (function in U.PRG)
│   │           │   AREA.DBF (database)
│   │           ├──SELECT() ... (FoxPro function)
│   │           ├──EMPTY() ... (FoxPro function)
│   │           ├──FOUND() (FoxPro function)
│   │           └──TRIM() ... (FoxPro function)
│   ├──ENC2STR() (function in DEMO.PRG)
│   │   ├──SELECT() ... (FoxPro function)
│   │   ├──LEN() ... (FoxPro function)
│   │   ├──SUBSTR() ... (FoxPro function)
│   │   ├──INVERTCOND() (function in DEMO.PRG)
│   │   │   ├──VAL() ... (FoxPro function)
```

```
IIF() ... (FoxPro function)
ALLTRIM() ... (FoxPro function)
STR() ... (FoxPro function)
EVALCOND() (function in DEMO.PRG)
    MULFACTOR() (function in DEMO.PRG)
        FACTORID() (function in DEMO.PRG)
            SET() ... (FoxPro function)
            ASCAN() (FoxPro function)
            ERRMESSAGE() (function in U.PRG)
            AINS() (FoxPro function)
        IIF() ... (FoxPro function)
    VAL() ... (FoxPro function)
    SETQ() (function in DEMO.PRG)
        DEFQ() (function in DEMO.PRG)
            QID() (function in DEMO.PRG)
                SET() ... (FoxPro function)
                ASCAN() ... (FoxPro function)
                ERRMESSAGE() ... (function in
                                  U.PRG)
                AINS() ... (FoxPro function)
            EMPTY() ... (FoxPro function)
            LEN() ... (FoxPro function)
    TRIM() ... (FoxPro function)
    ASK() (function in DEMO.PRG)
        EVALQ() (function in DEMO.PRG)
            QID() ... (function in DEMO.PRG)
            EMPTY() ... (FoxPro function)
            EDITSYMP() (function in DEMO.PRG)
                DICTQ() (function in U.PRG)
                    SELECT() ... (FoxPro function)
                    VAL() ... (FoxPro function)
                    FOUND() ... (FoxPro function)
                    PARAMETERS() ... (FoxPro
                                      function)
                    TRIM() ... (FoxPro function)
                EMPTY() ... (FoxPro function)
                ERRMESSAGE() ... (function in
                                  U.PRG)
                DICTQV() (function in U.PRG)
                    SELECT() ... (FoxPro function)
                    VAL() ... (FoxPro function)
                    FOUND() ... (FoxPro function)
                    TRIM() ... (FoxPro function)
                MAX() (FoxPro function)
                EDNUM() (function in U.PRG)
                    ALEN() ... (FoxPro function)
                    ACOPY() (FoxPro function)
                    PUSHHELP() ... (function in
                                    U.PRG)
                    VAL() ... (FoxPro function)
                    W6.SPR
                        SET() ... (FoxPro
                                   function)
                        WEXIST() ... (FoxPro
                                      function)
                        INT() ... (FoxPro
                                   function)
                        SROW() ... (FoxPro
                                    function)
                        SCOL() ... (FoxPro
                                    function)
```

```
│   │   │   │   │   │   │   │   │   │   └──WVISIBLE() ... (FoxPro
│   │   │   │   │   │   │   │   │   │                   function)
│   │   │   │   │   │   │   │   │   ├──LTRIM() (FoxPro function)
│   │   │   │   │   │   │   │   │   ├──STR() ... (FoxPro function)
│   │   │   │   │   │   │   │   │   └──POPHELP() (function in U.PRG)
│   │   │   │   │   │   │   │   │       └──SETHELP() ... (function
│   │   │   │   │   │   │   │   │                          in U.PRG)
│   │   │   │   │   │   │   ├──EDBOOL() (function in U.PRG)
│   │   │   │   │   │   │   │   ├──ALEN() ... (FoxPro function)
│   │   │   │   │   │   │   │   ├──ACOPY() ... (FoxPro function)
│   │   │   │   │   │   │   │   ├──PUSHHELP() ... (function in
│   │   │   │   │   │   │   │   │                U.PRG)
│   │   │   │   │   │   │   │   ├──IIF() ... (FoxPro function)
│   │   │   │   │   │   │   │   ├──W6C.SPR
│   │   │   │   │   │   │   │   │   ├──SET() ... (FoxPro
│   │   │   │   │   │   │   │   │   │              function)
│   │   │   │   │   │   │   │   │   ├──WEXIST() ... (FoxPro
│   │   │   │   │   │   │   │   │   │                function)
│   │   │   │   │   │   │   │   │   ├──INT() ... (FoxPro
│   │   │   │   │   │   │   │   │   │              function)
│   │   │   │   │   │   │   │   │   ├──SROW() ... (FoxPro
│   │   │   │   │   │   │   │   │   │                function)
│   │   │   │   │   │   │   │   │   ├──SCOL() ... (FoxPro
│   │   │   │   │   │   │   │   │   │                function)
│   │   │   │   │   │   │   │   │   └──WVISIBLE() ... (FoxPro
│   │   │   │   │   │   │   │   │                      function)
│   │   │   │   │   │   │   │   └──POPHELP() ... (function in
│   │   │   │   │   │   │   │                    U.PRG)
│   │   │   │   │   │   │   ├──IIF() ... (FoxPro function)
│   │   │   │   │   │   │   ├──LEFT() (FoxPro function)
│   │   │   │   │   │   │   ├──DICTECOUNT() (function in U.PRG)
│   │   │   │   │   │   │   │   ├──SELECT() ... (FoxPro function)
│   │   │   │   │   │   │   │   ├──VAL() ... (FoxPro function)
│   │   │   │   │   │   │   │   └──EOF() ... (FoxPro function)
│   │   │   │   │   │   │   ├──DICTQE() (function in U.PRG)
│   │   │   │   │   │   │   │   ├──SELECT() ... (FoxPro function)
│   │   │   │   │   │   │   │   ├──VAL() ... (FoxPro function)
│   │   │   │   │   │   │   │   ├──FOUND() ... (FoxPro function)
│   │   │   │   │   │   │   │   ├──EOF() ... (FoxPro function)
│   │   │   │   │   │   │   │   └──TRIM() ... (FoxPro function)
│   │   │   │   │   │   │   ├──AT() ... (FoxPro function)
│   │   │   │   │   │   │   └──EDCHECK() (function in U.PRG)
│   │   │   │   │   │   │       ├──ALEN() ... (FoxPro function)
│   │   │   │   │   │   │       ├──ACOPY() ... (FoxPro function)
│   │   │   │   │   │   │       ├──PUSHHELP() ... (function in
│   │   │   │   │   │   │       │                U.PRG)
│   │   │   │   │   │   │       ├──LTRIM() ... (FoxPro function)
│   │   │   │   │   │   │       ├──SETCHECK() (function in U.PRG)
│   │   │   │   │   │   │       │   └──ALEN() ... (FoxPro
│   │   │   │   │   │   │       │                  function)
│   │   │   │   │   │   │       ├──W6A.PRG
│   │   │   │   │   │   │       │   ├──SET() ... (FoxPro
│   │   │   │   │   │   │       │   │              function)
│   │   │   │   │   │   │       │   ├──SELECT() ... (FoxPro
│   │   │   │   │   │   │       │   │                function)
│   │   │   │   │   │   │       │   ├──WEXIST() ... (FoxPro
│   │   │   │   │   │   │       │   │                function)
│   │   │   │   │   │   │       │   ├──INT() ... (FoxPro
│   │   │   │   │   │   │       │   │              function)
│   │   │   │   │   │   │       │   ├──SROW() ... (FoxPro
│   │   │   │   │   │   │       │   │                function)
```

```
SCOL() ... (FoxPro function)
WVISIBLE() ... (FoxPro function)
ALEN() ... (FoxPro function)
LEN() ... (FoxPro function)
MIN() (FoxPro function)
IIF() ... (FoxPro function)
EDCHECK() (function in U.PRG)
    VAL() ... (FoxPro function)
    SUBSTR() ... (FoxPro function)
    VARREAD() (FoxPro function)
    AT() ... (FoxPro function)
    ALEN() ... (FoxPro function)
    SETCHECK() ... (function in U.PRG)
IIF() ... (FoxPro function)
POPHELP() ... (function in U.PRG)
DEFQ() ... (function in DEMO.PRG)
PUSHQ() (function in DEMO.PRG)
    SET() ... (FoxPro function)
    IIF() ... (FoxPro function)
    ASCAN() ... (FoxPro function)
IF() ... (FoxPro function)
SET() ... (FoxPro function)
EVALUATE() (FoxPro function)
LEN() ... (FoxPro function)
AT() ... (FoxPro function)
LEFT() ... (FoxPro function)
SUBSTR() ... (FoxPro function)
ALLTRIM() ... (FoxPro function)
STR() ... (FoxPro function)
ALLTRIM() ... (FoxPro function)
M_ENCGET() (function in DEMO.PRG)
    POPUPSHOW() ... (function in U.PRG)
    LODENC() (function in DEMO.PRG)
        ENCGETINFO() (function in U.PRG)
            SELECT() ... (FoxPro function)
            FOUND() ... (FoxPro function)
        DELDIAGNOS() ... (function in DEMO.PRG)
        GETDIAGNOS() (function in U.PRG)
            SELECT() ... (FoxPro function)
            OPENDIAG() (function in U.PRG)
                SELECT() ... (FoxPro function)
            ADDDIAGNOS() (function in DEMO.PRG)
                ALLTRIM() ... (FoxPro function)
                DEFFACTOR() (function in DEMO.PRG)
                    FACTORID() ... (function in DEMO.PRG)
            STR() ... (FoxPro function)
        DELSYMPTOM() ... (function in DEMO.PRG)
        GETSYMPTOM() (function in U.PRG)
            SELECT() ... (FoxPro function)
```

```
│   │   │   │   ├──OPENSYMP() (function in U.PRG)
│   │   │   │   │   └──SELECT() ... (FoxPro function)
│   │   │   │   ├──ALLTRIM() ... (FoxPro function)
│   │   │   │   ├──TRIM() ... (FoxPro function)
│   │   │   │   ├──STR() ... (FoxPro function)
│   │   │   │   ├──DICTQ() ... (function in U.PRG)
│   │   │   │   ├──DICTENUM() (function in U.PRG)
│   │   │   │   │   ├──SELECT() ... (FoxPro function)
│   │   │   │   │   ├──VAL() ... (FoxPro function)
│   │   │   │   │   ├──FOUND() ... (FoxPro function)
│   │   │   │   │   ├──EOF() ... (FoxPro function)
│   │   │   │   │   └──TRIM() ... (FoxPro function)
│   │   │   │   └──ADDSYMPTOM() (function in DEMO.PRG)
│   │   │   │       ├──ALLTRIM() ... (FoxPro function)
│   │   │   │       └──DEFQ() ... (function in DEMO.PRG)
│   │   │   ├──M_ENCRULE() ... (function in DEMO.PRG)
│   │   │   ├──AGEAT() (function in DEMO.PRG)
│   │   │   │   ├──YEAR() (FoxPro function)
│   │   │   │   ├──MONTH() (FoxPro function)
│   │   │   │   ├──DAY() (FoxPro function)
│   │   │   │   ├──LTRIM() ... (FoxPro function)
│   │   │   │   └──STR() ... (FoxPro function)
│   │   │   ├──EDSEXAGE() (function in DEMO.PRG)
│   │   │   │   └──ADDHISTORY() (function in DEMO.PRG)
│   │   │   │       ├──ALEN() ... (FoxPro function)
│   │   │   │       ├──DICTQNAME() (function in U.PRG)
│   │   │   │       │   ├──SELECT() ... (FoxPro function)
│   │   │   │       │   ├──FOUND() ... (FoxPro function)
│   │   │   │       │   ├──ALLTRIM() ... (FoxPro function)
│   │   │   │       │   └──STR() ... (FoxPro function)
│   │   │   │       ├──EMPTY() ... (FoxPro function)
│   │   │   │       ├──ERASEQ() (function in DEMO.PRG)
│   │   │   │       │   └──QID() ... (function in DEMO.PRG)
│   │   │   │       └──ADDSYMPTOM() ... (function in DEMO.PRG)
│   │   │   └──POPUPHIDE() ... (function in U.PRG)
│   │   ├──M_ASSIST     (procedure in DEMO.PRG)
│   │   │   ├──EMPTY() ... (FoxPro function)
│   │   │   ├──POPUPSHOW() ... (function in U.PRG)
│   │   │   ├──RESETDIAG() ... (function in DEMO.PRG)
│   │   │   ├──TESTRULES() (function in DEMO.PRG)
│   │   │   │   ├──SELECT() ... (FoxPro function)
│   │   │   │   ├──EOF() ... (FoxPro function)
│   │   │   │   ├──EVALCOND() ... (function in DEMO.PRG)
│   │   │   │   ├──ALLTRIM() ... (FoxPro function)
│   │   │   │   └──STR() ... (FoxPro function)
│   │   │   ├──EVALRULES() (function in DEMO.PRG)
│   │   │   │   ├──SELECT() ... (FoxPro function)
│   │   │   │   ├──ALEN() ... (FoxPro function)
│   │   │   │   ├──EVALCOND() ... (function in DEMO.PRG)
│   │   │   │   ├──ALLTRIM() ... (FoxPro function)
│   │   │   │   └──STR() ... (FoxPro function)
│   │   │   ├──MAKEDIAG() (function in DEMO.PRG)
│   │   │   │   └──NORMFACTOR() (function in DEMO.PRG)
│   │   │   │       └──SUMFACTOR() (function in DEMO.PRG)
│   │   │   ├──POPMESSAGE() (function in U.PRG)
│   │   │   │   ├──PUSHHELP() ... (function in U.PRG)
│   │   │   │   ├──W9.SPR
│   │   │   │   │   ├──SET() ... (FoxPro function)
│   │   │   │   │   ├──WEXIST() ... (FoxPro function)
│   │   │   │   │   ├──INT() ... (FoxPro function)
│   │   │   │   │   ├──SROW() ... (FoxPro function)
│   │   │   │   │   ├──SCOL() ... (FoxPro function)
```

```
                    └──WVISIBLE() ... (FoxPro function)
                 └──POPHELP() ... (function in U.PRG)
              ├──POPUPHIDE() ... (function in U.PRG)
              ├──RELEASE() (function in U.PRG)
                 ├──PARAMETER() ... (FoxPro function)
                 └──WONTOP() (FoxPro function)
              └──OBJNUM() (FoxPro function)
        ├──M_ENCSAV() (function in DEMO.PRG)
              ├──POPMESSAGE() ... (function in U.PRG)
              ├──M_ENCVALID() (function in DEMO.PRG)
                 └──EMPTY() ... (FoxPro function)
              ├──ADDENCNT() (function in U.PRG)
                 └──NEXTENCNT() (function in U.PRG)
                        SYSTEM.DBF (database)
                        ├──SELECT() ... (FoxPro function)
                        ├──IIF() ... (FoxPro function)
                        └──TYPE() ... (FoxPro function)
              ├──NEWENCNT() (function in U.PRG)
                 └──SELECT() ... (FoxPro function)
              ├──PUTENCNT() (function in U.PRG)
                 ├──SELECT() ... (FoxPro function)
                 └──FOUND() ... (FoxPro function)
              ├──PUTSYMPTOM() (function in U.PRG)
                 ├──SELECT() ... (FoxPro function)
                 ├──OPENSYMP() ... (function in U.PRG)
                 ├──FOUND() ... (FoxPro function)
                 ├──ALEN() ... (FoxPro function)
                 ├──IIF() ... (FoxPro function)
                 ├──EMPTY() ... (FoxPro function)
                 ├──SPLIT() (function in U.PRG)
                    ├──AT() ... (FoxPro function)
                    ├──LEFT() ... (FoxPro function)
                    ├──SUBSTR() ... (FoxPro function)
                    └──LEN() ... (FoxPro function)
                 ├──VAL() ... (FoxPro function)
                 ├──DICTQ() ... (function in U.PRG)
                 ├──DICTORD() (function in U.PRG)
                    ├──SELECT() ... (FoxPro function)
                    ├──VAL() ... (FoxPro function)
                    ├──FOUND() ... (FoxPro function)
                    ├──EOF() ... (FoxPro function)
                    ├──ALLTRIM() ... (FoxPro function)
                    └──STR() ... (FoxPro function)
                 └──EOF() ... (FoxPro function)
              └──PUTDIAGNOS() (function in U.PRG)
                 ├──SELECT() ... (FoxPro function)
                 ├──OPENDIAG() ... (function in U.PRG)
                 ├──FOUND() ... (FoxPro function)
                 ├──ALEN() ... (FoxPro function)
                 ├──EMPTY() ... (FoxPro function)
                 ├──VAL() ... (FoxPro function)
                 └──EOF() ... (FoxPro function)
        ├──STR() ... (FoxPro function)
        ├──ENC2BIT() (function in DEMO.PRG)
              ├──ALEN() ... (FoxPro function)
              └──IIF() ... (FoxPro function)
        ├──ALEN() ... (FoxPro function)
        ├──DISEASENAM() (function in U.PRG)
              DISEASE.DBF (database)
              ├──SELECT() ... (FoxPro function)
              ├──PARAMETERS() ... (FoxPro function)
              ├──ALEN() ... (FoxPro function)
```

```
│       │       ├──IIF() ... (FoxPro function)
│       │       ├──FOUND() ... (FoxPro function)
│       │       └──TRIM() ... (FoxPro function)
│       ├──VAL() ... (FoxPro function)
│       ├──IIF() ... (FoxPro function)
│       ├──LEFT() ... (FoxPro function)
│       └──INKEY() ... (FoxPro function)
├──MAINMENU() (function in U.PRG)
│       ├──CAMD.MPR
│       │       ├──EMPTY() ... (FoxPro function)
│       │       ├──WONTOP() ... (FoxPro function)
│       │       ├──LOCFILE() ... (FoxPro function)
│       │       ├── Q9POPFLOY     (procedure in CAMD.MPR)
│       │       │       ├──PUSHHELP() ... (function in U.PRG)
│       │       │       ├──PWDEDT.PRG
│       │       │       │       SYSTEM.DBF (database)
│       │       │       │       ├──SET() ... (FoxPro function)
│       │       │       │       ├──FULLPATH() ... (FoxPro function)
│       │       │       │       ├──SYS() ... (FoxPro function)
│       │       │       │       ├──GETENV() ... (FoxPro function)
│       │       │       │       ├──SPACE() ... (FoxPro function)
│       │       │       │       ├──INKEY() ... (FoxPro function)
│       │       │       │       ├──LASTKEY() ... (FoxPro function)
│       │       │       │       ├──SUBSTR() ... (FoxPro function)
│       │       │       │       ├──LEN() ... (FoxPro function)
│       │       │       │       ├──EMPTY() ... (FoxPro function)
│       │       │       │       ├──UPPER() ... (FoxPro function)
│       │       │       │       └──CHR() ... (FoxPro function)
│       │       │       └──POPHELP() ... (function in U.PRG)
│       │       ├── MHIT      (procedure in U.PRG)
│       │       │       └──RDLEVEL() (FoxPro function)
│       │       ├──REINDEX() (function in U.PRG)
│       │       │       PATIENT.DBF (database)
│       │       │       ENCOUNTR.DBF (database)
│       │       │       SYMPTOMS.DBF (database)
│       │       │       DICT.DBF (database)
│       │       │       VAL.DBF (database)
│       │       │       ENUM.DBF (database)
│       │       │       PREMISE.DBF (database)
│       │       │       ACTION.DBF (database)
│       │       │       RULE.DBF (database)
│       │       │       DISEASE.DBF (database)
│       │       │       DIAGNOSE.DBF (database)
│       │       │       REPORT.DBF (database)
│       │       │       SECTION.DBF (database)
│       │       │       SF600TEX.DBF (database)
│       │       │       ├──EMPTY() ... (FoxPro function)
│       │       │       ├──WOUTPUT() ... (FoxPro function)
│       │       │       ├──POPUPSHOW() ... (function in U.PRG)
│       │       │       ├──CLOSEFILES() (function in U.PRG)
│       │       │       └──POPUPHIDE() ... (function in U.PRG)
│       │       ├── QUIT      (procedure in U.PRG)
│       │       │       ├──EMPTY() ... (FoxPro function)
│       │       │       ├──WOUTPUT() ... (FoxPro function)
│       │       │       └──POPMESSAGE() ... (function in U.PRG)
│       │       ├── Q9POPFLR8     (procedure in CAMD.MPR)
│       │       │       ├──CAMDTR.MPR
│       │       │       │       ├──EMPTY() ... (FoxPro function)
│       │       │       │       ├──WONTOP() ... (FoxPro function)
│       │       │       │       ├──PWDEDT.PRG...
│       │       │       │       ├── MHIT...     (procedure in U.PRG)
│       │       │       │       ├──REINDEX() ... (function in U.PRG)
```

```
└──── _QUIT...      (procedure in U.PRG)
├──── _MHIT...      (procedure in U.PRG)
└──── CAMD.MPR...
└──── COLORSET.SPR
      COLORSET.DBF (database)
      ├──── SET() ... (FoxPro function)
      ├──── WEXIST() ... (FoxPro function)
      ├──── INT() ... (FoxPro function)
      ├──── SROW() ... (FoxPro function)
      ├──── SCOL() ... (FoxPro function)
      ├──── USED() ... (FoxPro function)
      ├──── WVISIBLE() ... (FoxPro function)
      └──── _Q9P0PFX2H() (function in COLORSET.SPR)
            └──── TRIM() ... (FoxPro function)
└──── MYHANDLER() (function in U.PRG)
      ├──── LEN() ... (FoxPro function)
      ├──── WONTOP() ... (FoxPro function)
      ├──── M_RESUME      (procedure in DEMO.PRG)
      ├──── ALEN() ... (FoxPro function)
      ├──── STRCOND() (function in DEMO.PRG)
      ├──── PARAMETERS() ... (FoxPro function)
      ├──── TYPE() ... (FoxPro function)
      ├──── LEN() ... (FoxPro function)
      ├──── LOOKUPNAM() (function in DEMO.PRG)
      ├──── DICTQ() ... (function in U.PRG)
      └──── EMPTY() ... (FoxPro function)
      └──── LEFT() ... (FoxPro function)
      ├──── STRDIAG() (function in DEMO.PRG)
      ├──── AINS() ... (FoxPro function)
      ├──── IIF() ... (FoxPro function)
      ├──── TYPE() ... (FoxPro function)
      ├──── DISEASENAM() ... (function in U.PRG)
      ├──── VAL() ... (FoxPro function)
      ├──── PARAMETERS() ... (FoxPro function)
      ├──── LTRIM() ... (FoxPro function)
      └──── STR() ... (FoxPro function)
      ├──── GETEXAREA() (function in U.PRG)
      AREA.DBF (database)
      ├──── EMPTY() ... (FoxPro function)
      ├──── SELECT() ... (FoxPro function)
      ├──── IIF() ... (FoxPro function)
      ├──── FOUND() ... (FoxPro function)
      └──── TRIM() ... (FoxPro function)
      ├──── DISEASENAM() ... (function in U.PRG)
      ├──── AGEAT() ... (function in DEMO.PRG)
      ├──── EDSEXAGE() ... (function in DEMO.PRG)
      ├──── ENCNTTITLE() (function in U.PRG)
      ├──── PUSHHELP() ... (function in U.PRG)
      ├──── W12.SPR
      ├──── SET() ... (FoxPro function)
      ├──── WEXIST() ... (FoxPro function)
      ├──── WVISIBLE() ... (FoxPro function)
      ├──── _EDITCOMPL() (function in U.PRG)
      ├──── M_COMPL      (procedure in DEMO.PRG)
      ├──── EMPTY() ... (FoxPro function)
      ├──── GETDIAGAREA() (function in
                                DEMO.PRG)
      ├──── AREACOUNT() (function in
                                U.PRG)
      AREA.DBF (database)
      ├──── SELECT() ... (FoxPro
                                function)
```

```
└──RECCOUNT() ... (FoxPro
                    function)
──AREANAMES() (function in
                U.PRG)
    AREA.DBF (database)
    ├──SELECT() ... (FoxPro
                      function)
    ├──EOF() ... (FoxPro
                   function)
    └──TRIM() ... (FoxPro
                    function)
──GETAREALIST() (function in
                  U.PRG)
    ├──ALEN() ... (FoxPro
                    function)
    ├──PUSHHELP() ... (function
                        in U.PRG)
    ├──W5.SPR
    │   ├──SET() ... (FoxPro
                       function)
    │   ├──WEXIST() ... (FoxPro
                         function)
    │   ├──INT() ... (FoxPro
                       function)
    │   ├──SROW() ... (FoxPro
                       function)
    │   ├──SCOL() ... (FoxPro
                       function)
    │   └──WVISIBLE() ...
            (FoxPro function)
    └──POPHELP() ... (function
                       in U.PRG)
──IIF() ... (FoxPro function)
──M_ENCRULE() ... (function in
                    DEMO.PRG)
──POPMESSAGE() ... (function in
                     U.PRG)
──EMPTY() ... (FoxPro function)
──GETEXAREA() ... (function in U.PRG)
──OBJNUM() ... (FoxPro function)
EDITHANDP() (function in U.PRG)
──M_HANDP      (procedure in DEMO.PRG)
    ├──EMPTY() ... (FoxPro function)
    ├──POPUPSHOW() ... (function in U.PRG)
    ├──TESTRULES() ... (function in
                         DEMO.PRG)
    ├──POPUPHIDE() ... (function in U.PRG)
    ├──POPMESSAGE() ... (function in
                          U.PRG)
    ├──RELEASE() ... (function in U.PRG)
    └──OBJNUM() ... (FoxPro function)
──STRCOND() ... (function in DEMO.PRG)
EDITCOND() (function in U.PRG)
──M_REDOQ() (function in DEMO.PRG)
    ├──GETQ() (function in DEMO.PRG)
    │   ├──QID() ... (function in
                       DEMO.PRG)
    │   └──IIF() ... (FoxPro function)
    ├──ERASEQ() ... (function in DEMO.PRG)
    ├──ASK() ... (function in DEMO.PRG)
    └──RESTOREQ() (function in DEMO.PRG)
```

```
└──QID() ... (function in
                DEMO.PRG)
└──STRCOND() ... (function in DEMO.PRG)
├── EDITDIAG() (function in U.PRG)
├──M_DIAG     (procedure in DEMO.PRG)
├──EMPTY() ... (FoxPro function)
├──ASKDIAG() (function in DEMO.PRG)
│  ├──DISEASECNT() ... (function in
│  │                    U.PRG)
│  ├──DISNAMES() (function in U.PRG)
│  │     DISEASE.DBF (database)
│  │  ├──SELECT() ... (FoxPro
│  │  │                function)
│  │  ├──PARAMETERS() ... (FoxPro
│  │  │                    function)
│  │  ├──EOF() ... (FoxPro
│  │  │             function)
│  │  └──TRIM() ... (FoxPro
│  │                function)
│  ├──IIF() ... (FoxPro function)
│  └──GETDISLIST() (function in
│                   U.PRG)
│     ├──ALEN() ... (FoxPro
│     │              function)
│     ├──PUSHHELP() ... (function
│     │                  in U.PRG)
│     ├──W7.SPR
│     │  ├──SET() ... (FoxPro
│     │  │             function)
│     │  ├──WEXIST() ... (FoxPro
│     │  │               function)
│     │  ├──INT() ... (FoxPro
│     │  │              function)
│     │  ├──SROW() ... (FoxPro
│     │  │               function)
│     │  ├──SCOL() ... (FoxPro
│     │  │               function)
│     │  └──WVISIBLE() ...
│     │     (FoxPro function)
│     ├──POPHELP() ... (function
│     │                  in U.PRG)
│     ├──DISEASEGET() (function in
│     │                U.PRG)
│     │     DISEASE.DBF
│     │     (database)
│     │  ├──SELECT() ... (FoxPro
│     │  │               function)
│     │  ├──UPPER() ... (FoxPro
│     │  │               function)
│     │  ├──FOUND() ... (FoxPro
│     │  │               function)
│     │  └──CHR() ... (FoxPro
│     │                function)
│     └──W8.PRG
│        ├──SET() ... (FoxPro
│        │             function)
│        ├──SELECT() ... (FoxPro
│        │               function)
│        ├──WEXIST() ... (FoxPro
│        │               function)
```

```
                                    ├──INT() ... (FoxPro
                                                  function)
                                    ├──SROW() ... (FoxPro
                                                  function)
                                    ├──SCOL() ... (FoxPro
                                                  function)
                                    └──WVISIBLE() ...
                                       (FoxPro function)
                        ├──DISEASENAM() ... (function in
                                              U.PRG)
                        ├──DISEASEID() (function in U.PRG)
                        │    DISEASE.DBF (database)
                        │   ├──SELECT() ... (FoxPro function)
                        │   ├──PARAMETERS() ... (FoxPro
                        │                        function)
                        │   ├──ALEN() ... (FoxPro function)
                        │   ├──IIF() ... (FoxPro function)
                        │   └──FOUND() ... (FoxPro function)
                        ├──POPMESSAGE() ... (function in
                                              U.PRG)
                        └──OBJNUM() ... (FoxPro function)
                  ├──EMPTY() ... (FoxPro function)
                  ├──DISEASENAM() ... (function in U.PRG)
                  └──OBJNUM() ... (FoxPro function)
            ├── EDITASSIS() (function in U.PRG)
            │   ├──M_ASSIST...      (procedure in DEMO.PRG)
            │   └──STRDIAG() ... (function in DEMO.PRG)
            ├── EDITVDIAG() (function in U.PRG)
            │   ├──DISEASEGET() ... (function in U.PRG)
            │   ├──PUSHHELP() ... (function in U.PRG)
            │   ├──W8.PRG...
            │   └──POPHELP() ... (function in U.PRG)
            ├── EDITTREAT() (function in U.PRG)
            │   └──M_TREAT      (procedure in DEMO.PRG)
            │       ├──EMPTY() ... (FoxPro function)
            │       ├──OBJNUM() ... (FoxPro function)
            │       ├──POPMESSAGE() ... (function in
            │                              U.PRG)
            │       └──RELEASE() ... (function in U.PRG)
            ├── Q9V0UMCRG() (function in W12.SPR)
            │   ├──WONTOP() ... (FoxPro function)
            │   └──W12C.SPR
            │       ├──SET() ... (FoxPro function)
            │       ├──WEXIST() ... (FoxPro function)
            │       ├──INT() ... (FoxPro function)
            │       ├──SROW() ... (FoxPro function)
            │       ├──SCOL() ... (FoxPro function)
            │       ├──WVISIBLE() ... (FoxPro function)
            │       ├── Q740N41W1() (function in W12C.SPR)
            │       │   ├──WONTOP() ... (FoxPro function)
            │       │   └──W12C.SPR...
            │       ├── Q740N423H() (function in W12C.SPR)
            │       │   └──RELEASE() ... (function in
            │       │                       U.PRG)
            │       └──CHR() ... (FoxPro function)
            ├──CHR() ... (FoxPro function)
            ├── EVALID() (function in U.PRG)
            │   ├──PUSHHELP() ... (function in U.PRG)
            │   └──W2.SPR
            │       ├──SET() ... (FoxPro function)
            │       ├──WEXIST() ... (FoxPro function)
```

```
INT() ... (FoxPro function)
SROW() ... (FoxPro function)
SCOL() ... (FoxPro function)
WVISIBLE() ... (FoxPro function)
_Q7I0LH8OT() (function in W2.SPR)
_Q7I0LH8WD() (function in W2.SPR)
    _PRINTENC() (function in U.PRG)
        ALLTRIM() ... (FoxPro function)
        EMPTY() ... (FoxPro function)
        DTOC() (FoxPro function)
        STR() ... (FoxPro function)
        TRIM() ... (FoxPro function)
        IIF() ... (FoxPro function)
        ALEN() ... (FoxPro function)
        PROPER() (FoxPro function)
    ALLTRIM() ... (FoxPro function)
_Q7I0LH94Q() (function in W2.SPR)
    OBJNUM() ... (FoxPro function)
POPHELP() ... (function in U.PRG)
_Q9V0UMD2M() (function in W12.SPR)
    RELEASE() ... (function in U.PRG)
_Q9V0UMDDI() (function in W12.SPR)
    RELEASE() ... (function in U.PRG)
_Q9V0UMDJJ() (function in W12.SPR)
    WONTOP() ... (FoxPro function)
    W12A.SPR
        SET() ... (FoxPro function)
        WEXIST(, ... (FoxPro function)
        INT() ... (FoxPro function)
        SROW() ... (FoxPro function)
        SCOL() ... (FoxPro function)
        WVISIBLE() ... (FoxPro function)
        _EDITCOND() ... (function in U.PRG)
        _EDITHANDP() ... (function in U.PRG)
        _Q740N3U9Y() (function in W12A.SPR)
            RELEASE() ... (function in U.PRG)
        CHR() ... (FoxPro function)
        _Q740N3UHF() (function in W12A.SPR)
            WONTOP() ... (FoxPro function)
            W12A.SPR...
_Q9V0UMDQF() (function in W12.SPR)
    RELEASE() ... (function in U.PRG)
_Q9V0UMDWG() (function in W12.SPR)
    WONTOP() ... (FoxPro function)
    W12B.SPR
        SET() ... (FoxPro function)
        WEXIST() ... (FoxPro function)
        INT() ... (FoxPro function)
        SROW() ... (FoxPro function)
        SCOL() ... (FoxPro function)
        WVISIBLE() ... (FoxPro function)
```

```
                                    Q740N3XWA() (function in W12B.SPR)
                                        RELEASE() ... (function in
                                                        U.PRG)
                                    _EDITVDIAG() ... (function in
                                                        U.PRG)
                                    _EDITASSIS() ... (function in
                                                        U.PRG)
                                    Q740N3Y5O() (function in W12B.SPR)
                                        WONTOP() ... (FoxPro function)
                                        W12B.SPR...
                                    CHR() ... (FoxPro function)
                            Q9V0UME80() (function in W12.SPR)
                                WONTOP() ... (FoxPro function)
                            STOPREAD() (function in U.PRG)
                                WVISIBLE() ... (FoxPro function)
                                WREAD() (FoxPro function)
                        POPHELP() ... (function in U.PRG)
                M_ENCSAV() ... (function in DEMO.PRG)
                M_ENCCLO() (function in DEMO.PRG)
                M_ENCVALID() ... (function in DEMO.PRG)
                WCONFIRM() (function in U.PRG)
                        PUSHHELP() ... (function in U.PRG)
                        W10.SPR
                            SET() ... (FoxPro function)
                            WEXIST() ... (FoxPro function)
                            INT() ... (FoxPro function)
                            SROW() ... (FoxPro function)
                            SCOL() ... (FoxPro function)
                            WVISIBLE() ... (FoxPro function)
                        POPHELP() ... (function in U.PRG)
        M_REGPAT      (procedure in DEMO.PRG)
                CTOD() ... (FoxPro function)
                GETCURPAT() (function in DEMO.PRG)
                PATID() (function in U.PRG)
                        PUSHHELP() ... (function in U.PRG)
                        W4.SPR
                            SET() ... (FoxPro function)
                            WEXIST() ... (FoxPro function)
                            INT() ... (FoxPro function)
                            SROW() ... (FoxPro function)
                            SCOL() ... (FoxPro function)
                            WVISIBLE() ... (FoxPro function)
                            _PVALID() (function in U.PRG)
                            _Q7Q0ZD5UR() (function in W4.SPR)
                            _Q7Q0ZD65J() (function in W4.SPR)
                                M_PATLIST     (procedure in DEMO.PRG)
                                    GETPATIENT() (function in DEMO.PRG)
                                        PATCOUNT() (function in U.PRG)
                                            SELECT() ... (FoxPro
                                                            function)
                                            RECCOUNT() ... (FoxPro
                                                            function)
                                        PATNAMES() (function in U.PRG)
                                            SELECT() ... (FoxPro
                                                            function)
                                            SET() ... (FoxPro
                                                            function)
                                            TYPE() ... (FoxPro
                                                            function)
                                            ALLTRIM() ... (FoxPro
                                                            function)
```

```
                    ├──ALEN() ... (FoxPro function)
                    ├──EOF() ... (FoxPro function)
                    └──STRNAME() (function in U.PRG)
                        ├──IIF() ... (FoxPro function)
                        ├──EMPTY() ... (FoxPro function)
                        └──TRIM() ... (FoxPro function)
                ├──IIF() ... (FoxPro function)
                └──GETPATLIST() (function in U.PRG)
                    ├──ALEN() ... (FoxPro function)
                    ├──IIF() ... (FoxPro function)
                    ├──TYPE() ... (FoxPro function)
                    ├──ALLTRIM() ... (FoxPro function)
                    ├──PUSHHELP() ... (function in U.PRG)
                    ├──W3.SPR
                    │   ├──SET() ... (FoxPro function)
                    │   ├──WEXIST() ... (FoxPro function)
                    │   ├──INT() ... (FoxPro function)
                    │   ├──SROW() ... (FoxPro function)
                    │   ├──SCOL() ... (FoxPro function)
                    │   └──WVISIBLE() ... (FoxPro function)
                    └──POPHELP() ... (function in U.PRG)
            ├──SUBSTR() ... (FoxPro function)
            └──AT() ... (FoxPro function)
        ├──EMPTY() ... (FoxPro function)
        └──M_PATGET() (function in DEMO.PRG)
            ├──PATGETINFO() (function in U.PRG)
            ├──SELECT() ... (FoxPro function)
            └──FOUND() ... (FoxPro function)
            └──ERRMESSAGE() ... (function in U.PRG)
    └──STOPREAD() ... (function in U.PRG)
└──POPHELP() ... (function in U.PRG)
M_PATPUT() (function in DEMO.PRG)
├──IIF() ... (FoxPro function)
├──TYPE() ... (FoxPro function)
├──POPMESSAGE() ... (function in U.PRG)
├──ISVALIDSSN() (function in U.PRG)
│   ├──SELECT() ... (FoxPro function)
│   └──FOUND() ... (FoxPro function)
```

```
│                │   └──PATPUTINFO() (function in U.PRG)
│                │       ├──SELECT() ... (FoxPro function)
│                │       └──FOUND() ... (FoxPro function)
│                ├──M_PATADD() (function in DEMO.PRG)
│                │   ├──IIF() ... (FoxPro function)
│                │   ├──TYPE() ... (FoxPro function)
│                │   ├──POPMESSAGE() ... (function in U.PRG)
│                │   ├──ISVALIDSSN() ... (function in U.PRG)
│                │   ├──ADDPAT() (function in U.PRG)
│                │   │   └──NEXTPAT() (function in U.PRG)
│                │   │       │   SYSTEM.DBF (database)
│                │   │       ├──SELECT() ... (FoxPro function)
│                │   │       ├──IIF() ... (FoxPro function)
│                │   │       └──TYPE() ... (FoxPro function)
│                │   └──NEWPATIENT() (function in U.PRG)
│                │       └──SELECT() ... (FoxPro function)
│                ├──M_ENCVALID() ... (function in DEMO.PRG)
│                ├──SETCURPAT() (function in DEMO.PRG)
│                └──AGEAT() ... (function in DEMO.PRG)
├──USERCLOSE() (function in U.PRG)
│   └──EMPTY() ... (FoxPro function)
└──SYSCLOSE() (function in DEMO.PRG)
```

Section VII. Procedure and Function Summary. *There are twenty one (21) files containing procedures in the CAMD system: DEMO.PRG, U.PRG, W12.SPR, W2.SPR, WP1.PRG, WP2.PRG, WPATS.PRG, WP3.SPR, WP7.SPR, SECHECK.PRG, W12A.SPR, W12B.SPR, W12C.SPR, W4.SPR, BACKUP.SPR, RESTORE.SPR, W7A.SPR, QUESTION.PRG, Q2.PRG, COLORSET.SPR, CAMD.MPR.*

DEMO.PRG -- Last updated: 10/05/92 at 16:36

```
Contains: M_RESUME                    (Params: none)
   Called by: M_ENC                   (procedure in DEMO.PRG)
   Called by: MYHANDLER               (procedure in U.PRG)
       Calls: STRCOND                 (procedure in DEMO.PRG)
       Calls: STRDIAG                 (procedure in DEMO.PRG)
       Calls: GETEXAREA()             (function in U.PRG)
       Calls: DISEASENAM()            (function in U.PRG)
       Calls: AGEAT                   (procedure in DEMO.PRG)
       Calls: EDSEXAGE                (procedure in DEMO.PRG)
       Calls: ENCNTTITLE()            (function in U.PRG)
       Calls: M_ENCSAV                (procedure in DEMO.PRG)
       Calls: M_ENCCLO                (procedure in DEMO.PRG)
       Calls: M_ENCVALID              (procedure in DEMO.PRG)
       Calls: WCONFIRM()              (function in U.PRG)
Contains: M_REGPAT                    (Params: none)
   Called by: MYHANDLER               (procedure in U.PRG)
       Calls: GETCURPAT               (procedure in DEMO.PRG)
       Calls: PATID()                 (function in U.PRG)
       Calls: M_PATPUT                (procedure in DEMO.PRG)
       Calls: M_PATADD                (procedure in DEMO.PRG)
       Calls: M_ENCVALID              (procedure in DEMO.PRG)
       Calls: SETCURPAT               (procedure in DEMO.PRG)
       Calls: AGEAT                   (procedure in DEMO.PRG)
Contains: M_PAT                       (Params: none)
       Calls: OPENFILES()             (function in U.PRG)
       Calls: KEYLOG.PRG
       Calls: WPAT                    (procedure in DEMO.PRG)
       Calls: FLUSHFILES()            (function in U.PRG)
Contains: M_ENC                       (Params: none)
   Called by: M_NEWENC                (procedure in DEMO.PRG)
   Called by: M_TRAINING              (procedure in DEMO.PRG)
       Calls: ENCOUNT                 (procedure in DEMO.PRG)
       Calls: M_RESUME                (procedure in DEMO.PRG)
       Calls: POPMESSAGE()            (function in U.PRG)
Contains: M_COMPL                     (Params: none)
   Called by: _EDITCOMPL              (procedure in U.PRG)
       Calls: GETDIAGAREA             (procedure in DEMO.PRG)
       Calls: M_ENCRULE               (procedure in DEMO.PRG)
       Calls: POPMESSAGE()            (function in U.PRG)
Contains: M_DIAG                      (Params: none)
   Called by: _EDITDIAG               (procedure in U.PRG)
       Calls: ASKDIAG                 (procedure in DEMO.PRG)
       Calls: DISEASENAM()            (function in U.PRG)
       Calls: DISEASEID()             (function in U.PRG)
       Calls: POPMESSAGE()            (function in U.PRG)
Contains: M_ASSIST                    (Params: none)
   Called by: BATCHTEST()             (function in DEMO.PRG)
   Called by: _EDITASSIS              (procedure in U.PRG)
       Calls: POPUPSHOW()             (function in U.PRG)
       Calls: RESETDIAG               (procedure in DEMO.PRG)
       Calls: TESTRULES               (procedure in DEMO.PRG)
       Calls: EVALRULES               (procedure in DEMO.PRG)
       Calls: MAKEDIAG                (procedure in DEMO.PRG)
```

```
        Calls: POPMESSAGE()          (function in U.PRG)
        Calls: POPUPHIDE()           (function in U.PRG)
Contains: M_HANDP                    (Params: none)
   Called by: _EDITHANDP             (procedure in U.PRG)
        Calls: POPUPSHOW()           (function in U.PRG)
        Calls: TESTRULES             (procedure in DEMO.PRG)
        Calls: POPUPHIDE()           (function in U.PRG)
        Calls: POPMESSAGE()          (function in U.PRG)
Contains: M_TREAT                    (Params: NEWTREAT)
   Called by: _EDITTREAT             (procedure in U.PRG)
        Calls: POPMESSAGE()          (function in U.PRG)
Contains: M_REDOQ                    (Params: MITEM)
   Called by: _EDITCOND              (procedure in U.PRG)
        Calls: GETQ                  (procedure in DEMO.PRG)
        Calls: ERASEQ                (procedure in DEMO.PRG)
        Calls: ASK                   (procedure in DEMO.PRG)
        Calls: RESTOREQ              (procedure in DEMO.PRG)
Contains: ENCOUNT                    (Params: none)
   Called by: M_ENC                  (procedure in DEMO.PRG)
        Calls: GETENCNT              (procedure in DEMO.PRG)
        Calls: PATSSN()              (function in U.PRG)
        Calls: M_PATGET              (procedure in DEMO.PRG)
        Calls: M_ENCGET              (procedure in DEMO.PRG)
        Calls: M_ENCNEW              (procedure in DEMO.PRG)
Contains: M_ENCCLO                   (Params: none)
   Called by: M_RESUME               (procedure in DEMO.PRG)
Contains: M_ENCSAV                   (Params: none)
   Called by: BATCHTEST()            (function in DEMO.PRG)
   Called by: M_RESUME               (procedure in DEMO.PRG)
        Calls: POPMESSAGE()          (function in U.PRG)
        Calls: M_ENCVALID            (procedure in DEMO.PRG)
        Calls: ADDENCNT()            (function in U.PRG)
        Calls: NEWENCNT()            (function in U.PRG)
        Calls: PUTENCNT()            (function in U.PRG)
        Calls: PUTSYMPTOM()          (function in U.PRG)
        Calls: PUTDIAGNOS()          (function in U.PRG)
Contains: M_ENCVALID                 (Params: none)
   Called by: M_RESUME               (procedure in DEMO.PRG)
   Called by: M_REGPAT               (procedure in DEMO.PRG)
   Called by: M_ENCSAV               (procedure in DEMO.PRG)
Contains: M_ENCGET                   (Params: none)
   Called by: BATCHTEST()            (function in DEMO.PRG)
   Called by: ENCOUNT                (procedure in DEMO.PRG)
        Calls: POPUPSHOW()           (function in U.PRG)
        Calls: LODENC                (procedure in DEMO.PRG)
        Calls: M_ENCRULE             (procedure in DEMO.PRG)
        Calls: AGEAT                 (procedure in DEMO.PRG)
        Calls: EDSEXAGE              (procedure in DEMO.PRG)
        Calls: POPUPHIDE()           (function in U.PRG)
Contains: M_ENCNEW                   (Params: none)
   Called by: ENCOUNT                (procedure in DEMO.PRG)
        Calls: NEXTENCNT()           (function in U.PRG)
        Calls: AGEAT                 (procedure in DEMO.PRG)
        Calls: DELSYMPTOM            (procedure in DEMO.PRG)
        Calls: DELDIAGNOS            (procedure in DEMO.PRG)
        Calls: EDSEXAGE              (procedure in DEMO.PRG)
Contains: EDSEXAGE                   (Params: MAGE, MSEX)
   Called by: M_RESUME               (procedure in DEMO.PRG)
   Called by: M_ENCGET               (procedure in DEMO.PRG)
   Called by: M_ENCNEW               (procedure in DEMO.PRG)
        Calls: ADDHISTORY            (procedure in DEMO.PRG)
Contains: M_ENCRULE                  (Params: none)
```

```
   Called by: BATCHTEST()          (function in DEMO.PRG)
   Called by: M_COMPL              (procedure in DEMO.PRG)
   Called by: M_ENCGET             (procedure in DEMO.PRG)
       Calls: LODRULE              (procedure in DEMO.PRG)
Contains: LODRULE                  (Params: MAREA)
   Called by: M_ENCRULE            (procedure in DEMO.PRG)
       Calls: GETRULEBASE()        (function in U.PRG)
Contains: TESTRULES                (Params: MAREA)
   Called by: M_ASSIST             (procedure in DEMO.PRG)
   Called by: M_HANDP              (procedure in DEMO.PRG)
       Calls: EVALCOND             (procedure in DEMO.PRG)
Contains: EVALRULES                (Params: MAREA)
   Called by: M_ASSIST             (procedure in DEMO.PRG)
       Calls: EVALCOND             (procedure in DEMO.PRG)
Contains: OBJTYPE                  (Params: MOBJECT)
Contains: EVALCOND                 (Params: MOBJECT, MID, MOP, MVAL)
   Called by: TESTRULES            (procedure in DEMO.PRG)
   Called by: EVALRULES            (procedure in DEMO.PRG)
   Called by: INVERTCOND           (procedure in DEMO.PRG)
       Calls: MULFACTOR            (procedure in DEMO.PRG)
       Calls: SETQ                 (procedure in DEMO.PRG)
       Calls: ASK                  (procedure in DEMO.PRG)
Contains: UNDORULES                (Params: none)
   Called by: M_UNDORULE           (procedure in DEMO.PRG)
       Calls: UNDOFACTOR           (procedure in DEMO.PRG)
       Calls: UNDOQUAL             (procedure in DEMO.PRG)
Contains: UNDOFACTOR               (Params: STATESTR)
   Called by: UNDORULES            (procedure in DEMO.PRG)
       Calls: FACTORID             (procedure in DEMO.PRG)
Contains: UNDOQUAL                 (Params: STATESTR)
   Called by: UNDORULES            (procedure in DEMO.PRG)
   Called by: POPQ                 (procedure in DEMO.PRG)
       Calls: QID                  (procedure in DEMO.PRG)
Contains: ASKDIAG                  (Params: MDIAGNOSIS)
   Called by: M_DIAG               (procedure in DEMO.PRG)
       Calls: DISEASECNT()         (function in U.PRG)
       Calls: DISNAMES()           (function in U.PRG)
       Calls: GETDISLIST()         (function in U.PRG)
Contains: GETDIAGAREA              (Params: none)
   Called by: M_COMPL              (procedure in DEMO.PRG)
   Called by: M_DATA               (procedure in DEMO.PRG)
       Calls: AREACOUNT()          (function in U.PRG)
       Calls: AREANAMES()          (function in U.PRG)
       Calls: GETAREALIST()        (function in U.PRG)
Contains: GETPATIENT               (Params: MKEY)
   Called by: M_PATLIST            (procedure in DEMO.PRG)
       Calls: PATCOUNT()           (function in U.PRG)
       Calls: PATNAMES()           (function in U.PRG)
       Calls: GETPATLIST()         (function in U.PRG)
Contains: GETENCNT                 (Params: MPATIENTID, MDATE)
   Called by: ENCOUNT              (procedure in DEMO.PRG)
       Calls: ENCCOUNT()           (function in U.PRG)
       Calls: ENCNAMES()           (function in U.PRG)
       Calls: GETENCLIST()         (function in U.PRG)
       Calls: PATSSN()             (function in U.PRG)
Contains: INIENCOUNT()             (Params: none)
   Called by: DEMO.PRG
       Calls: GETSTATION()         (function in U.PRG)
Contains: LODENC                   (Params: MENCOUNTID)
   Called by: M_ENCGET             (procedure in DEMO.PRG)
   Called by: SFENCOUNT            (procedure in U.PRG)
       Calls: ENCGETINFO()         (function in U.PRG)
```

```
       Calls: DELDIAGNOS              (procedure in DEMO.PRG)
       Calls: GETDIAGNOS()            (function in U.PRG)
       Calls: DELSYMPTOM              (procedure in DEMO.PRG)
       Calls: GETSYMPTOM()            (function in U.PRG)
Contains: M_CHANGE                    (Params: none)
       Calls: CHANGEQ                 (procedure in DEMO.PRG)
       Calls: POPQ                    (procedure in DEMO.PRG)
       Calls: M_UNDORULE              (procedure in DEMO.PRG)
       Calls: POPMESSAGE()            (function in U.PRG)
Contains: M_UNDO                      (Params: none)
       Calls: LASTQ                   (procedure in DEMO.PRG)
       Calls: POPQ                    (procedure in DEMO.PRG)
       Calls: M_UNDORULE              (procedure in DEMO.PRG)
Contains: M_UNDORULE                  (Params: none)
   Called by: M_CHANGE                (procedure in DEMO.PRG)
   Called by: M_UNDO                  (procedure in DEMO.PRG)
       Calls: UNDORULES               (procedure in DEMO.PRG)
Contains: MYUNDO                      (Params: none)
       Calls: UNDOMENU()              (function in U.PRG)
Contains: MAKEDIAG                    (Params: none)
   Called by: M_ASSIST                (procedure in DEMO.PRG)
       Calls: NORMFACTOR              (procedure in DEMO.PRG)
Contains: GETCURPAT                   (Params: MPID, MSSN, MLAST, MFIRST,
                                      MBIRTH, MSX, MRATE)
   Called by: M_REGPAT                (procedure in DEMO.PRG)
Contains: SETCURPAT                   (Params: MPID, MSSN, MLAST, MFIRST,
                                      MBIRTH, MSX, MRATE)
   Called by: M_REGPAT                (procedure in DEMO.PRG)
Contains: M_PATLIST                   (Params: none)
   Called by: _Q7Q0ZD65J()            (function in W4.SPR)
       Calls: GETPATIENT              (procedure in DEMO.PRG)
       Calls: M_PATGET                (procedure in DEMO.PRG)
Contains: M_PATGET                    (Params: MPID, MSSN, MLAST, MFIRST,
                                      MBIRTH, MSX, MRATE)
   Called by: ENCOUNT                 (procedure in DEMO.PRG)
   Called by: M_PATLIST               (procedure in DEMO.PRG)
   Called by: SFENCOUNT               (procedure in U.PRG)
       Calls: PATGETINFO()            (function in U.PRG)
       Calls: ERRMESSAGE()            (function in U.PRG)
Contains: M_PATPUT                    (Params: MPID, MSSN, MLAST, MFIRST,
                                      MBIRTH, MSX, MRATE)
   Called by: M_REGPAT                (procedure in DEMO.PRG)
       Calls: POPMESSAGE()            (function in U.PRG)
       Calls: ISVALIDSSN()            (function in U.PRG)
       Calls: PATPUTINFO()            (function in U.PRG)
Contains: M_PATADD                    (Params: MPID, MSSN, MLAST, MFIRST,
                                      MBIRTH, MSX, MRATE)
   Called by: M_REGPAT                (procedure in DEMO.PRG)
       Calls: POPMESSAGE()            (function in U.PRG)
       Calls: ISVALIDSSN()            (function in U.PRG)
       Calls: ADDPAT()                (function in U.PRG)
       Calls: NEWPATIENT()            (function in U.PRG)
Contains: AGEAT                       (Params: MDATE, MDATEBIRTH)
   Called by: M_RESUME                (procedure in DEMO.PRG)
   Called by: M_REGPAT                (procedure in DEMO.PRG)
   Called by: M_ENCGET                (procedure in DEMO.PRG)
   Called by: M_ENCNEW                (procedure in DEMO.PRG)
   Called by: WPAT                    (procedure in DEMO.PRG)
Contains: ADDDIAGNOS                  (Params: DNAME, DVAL)
   Called by: GETDIAGNOS()            (function in U.PRG)
       Calls: DEFFACTOR               (procedure in DEMO.PRG)
Contains: ADDHISTORY                  (Params: HISTNAME, HISTVAL)
```

```
   Called by: EDSEXAGE                    (procedure in DEMO.PRG)
       Calls: DICTONAME()                 (function in U.PRG)
       Calls: ERASEQ                      (procedure in DEMO.PRG)
       Calls: ADDSYMPTOM                  (procedure in DEMO.PRG)
Contains: ADDSYMPTOM                      (Params: MNAME, MVAL)
   Called by: ADDHISTORY                  (procedure in DEMO.PRG)
   Called by: GETSYMPTOM()                (function in U.PRG)
       Calls: DEFQ                        (procedure in DEMO.PRG)
Contains: DELDIAGNOS                      (Params: none)
   Called by: INIDIAGNOS()                (function in DEMO.PRG)
   Called by: BATCHTEST()                 (function in DEMO.PRG)
   Called by: M_ENCNEW                    (procedure in DEMO.PRG)
   Called by: LODENC                      (procedure in DEMO.PRG)
       Calls: RESETDIAG                   (procedure in DEMO.PRG)
Contains: RESETDIAG                       (Params: none)
   Called by: M_ASSIST                    (procedure in DEMO.PRG)
   Called by: DELDIAGNOS                  (procedure in DEMO.PRG)
Contains: DELSYMPTOM                      (Params: none)
   Called by: INISYMPTOM()                (function in DEMO.PRG)
   Called by: BATCHTEST()                 (function in DEMO.PRG)
   Called by: M_ENCNEW                    (procedure in DEMO.PRG)
   Called by: LODENC                      (procedure in DEMO.PRG)
Contains: INCR                            (Params: MSTR, MINCR)
   Called by: DECR                        (procedure in DEMO.PRG)
       Calls: GETFACTOR                   (procedure in DEMO.PRG)
       Calls: SETFACTOR                   (procedure in DEMO.PRG)
Contains: DECR                            (Params: MSTR, MINCR)
       Calls: INCR                        (procedure in DEMO.PRG)
Contains: MULFACTOR                       (Params: MSTR, MVAL)
   Called by: EVALCOND                    (procedure in DEMO.PRG)
       Calls: FACTORID                    (procedure in DEMO.PRG)
Contains: SUMFACTOR                       (Params: none)
   Called by: NORMFACTOR                  (procedure in DEMO.PRG)
Contains: NORMFACTOR                      (Params: none)
   Called by: MAKEDIAG                    (procedure in DEMO.PRG)
       Calls: SUMFACTOR                   (procedure in DEMO.PRG)
Contains: GETFACTOR                       (Params: MSTR)
   Called by: INCR                        (procedure in DEMO.PRG)
       Calls: FACTORID                    (procedure in DEMO.PRG)
Contains: SETFACTOR                       (Params: MSTR, MVAL)
   Called by: INCR                        (procedure in DEMO.PRG)
       Calls: DEFFACTOR                   (procedure in DEMO.PRG)
Contains: DEFFACTOR                       (Params: MSTR, MVAL)
   Called by: ADDDIAGNOS                  (procedure in DEMO.PRG)
   Called by: SETFACTOR                   (procedure in DEMO.PRG)
       Calls: FACTORID                    (procedure in DEMO.PRG)
Contains: FACTORID                        (Params: MSTR)
   Called by: UNDOFACTOR                  (procedure in DEMO.PRG)
   Called by: MULFACTOR                   (procedure in DEMO.PRG)
   Called by: GETFACTOR                   (procedure in DEMO.PRG)
   Called by: DEFFACTOR                   (procedure in DEMO.PRG)
   Called by: ADDFACTOR                   (procedure in DEMO.PRG)
       Calls: ERRMESSAGE()                (function in U.PRG)
Contains: ADDFACTOR                       (Params: MSTR)
       Calls: FACTORID                    (procedure in DEMO.PRG)
Contains: INIDIAGNOS()                    (Params: none)
   Called by: DEMO.PRG
       Calls: DISEASECNT()                (function in U.PRG)
       Calls: DELDIAGNOS                  (procedure in DEMO.PRG)
Contains: INISYMPTOM()                    (Params: none)
   Called by: DEMO.PRG
       Calls: DICTCOUNT()                 (function in U.PRG)
```

```
       Calls: DELSYMPTOM                  (procedure in DEMO.PRG)
Contains: LOOKUPQ                         (Params: MID)
   Called by: CHANGEQ                     (procedure in DEMO.PRG)
   Called by: LASTQ                       (procedure in DEMO.PRG)
       Calls: DICTQ()                     (function in U.PRG)
Contains: LOOKUPNAM                       (Params: MID)
   Called by: STRCOND                     (procedure in DEMO.PRG)
       Calls: DICTQ()                     (function in U.PRG)
Contains: ASK                             (Params: MSTR, MQ)
   Called by: M_REDOQ                     (procedure in DEMO.PRG)
   Called by: EVALCOND                    (procedure in DEMO.PRG)
       Calls: EVALQ                       (procedure in DEMO.PRG)
Contains: ISQ                             (Params: MSTR, MVAL)
       Calls: EVALQ                       (procedure in DEMO.PRG)
Contains: VALQ                            (Params: MSTR)
       Calls: EVALQ                       (procedure in DEMO.PRG)
Contains: EVALQ                           (Params: MSTR)
   Called by: ASK                         (procedure in DEMO.PRG)
   Called by: ISQ                         (procedure in DEMO.PRG)
   Called by: VALQ                        (procedure in DEMO.PRG)
       Calls: QID                         (procedure in DEMO.PRG)
       Calls: EDITSYMP                    (procedure in DEMO.PRG)
       Calls: DEFQ                        (procedure in DEMO.PRG)
       Calls: PUSHQ                       (procedure in DEMO.PRG)
Contains: EDITSYMP                        (Params: DICTID, MSTRVAL)
   Called by: EVALQ                       (procedure in DEMO.PRG)
       Calls: DICTQ()                     (function in U.PRG)
       Calls: ERRMESSAGE()                (function in U.PRG)
       Calls: DICTQV()                    (function in U.PRG)
       Calls: EDNUM()                     (function in U.PRG)
       Calls: EDBOOL()                    (function in U.PRG)
       Calls: DICTECOUNT()                (function in U.PRG)
       Calls: DICTQE()                    (function in U.PRG)
       Calls: EDCHECK()                   (function in U.PRG)
Contains: SETQ                            (Params: MSTR, MVAL)
   Called by: EVALCOND                    (procedure in DEMO.PRG)
       Calls: DEFQ                        (procedure in DEMO.PRG)
Contains: DEFQ                            (Params: MSTR, MVAL)
   Called by: ADDSYMPTOM                  (procedure in DEMO.PRG)
   Called by: EVALQ                       (procedure in DEMO.PRG)
   Called by: SETQ                        (procedure in DEMO.PRG)
       Calls: QID                         (procedure in DEMO.PRG)
Contains: GETQ                            (Params: MSTR)
   Called by: M_REDOQ                     (procedure in DEMO.PRG)
   Called by: STRNATURAL                  (procedure in DEMO.PRG)
   Called by: SFSYMP                      (procedure in DEMO.PRG)
       Calls: QID                         (procedure in DEMO.PRG)
Contains: ERASEQ                          (Params: MSTR)
   Called by: M_REDOQ                     (procedure in DEMO.PRG)
   Called by: ADDHISTORY                  (procedure in DEMO.PRG)
       Calls: QID                         (procedure in DEMO.PRG)
Contains: RESTOREQ                        (Params: MSTR, MVAL)
   Called by: M_REDOQ                     (procedure in DEMO.PRG)
       Calls: QID                         (procedure in DEMO.PRG)
Contains: PUSHQ                           (Params: MSTR, MVAL)
   Called by: EVALQ                       (procedure in DEMO.PRG)
Contains: CHANGEQ                         (Params: N)
   Called by: M_CHANGE                    (procedure in DEMO.PRG)
       Calls: LOOKUPQ                     (procedure in DEMO.PRG)
       Calls: CHANGELIST()                (function in U.PRG)
Contains: POPQ                            (Params: N)
   Called by: M_CHANGE                    (procedure in DEMO.PRG)
```

```
   Called by: M_UNDO                    (procedure in DEMO.PRG)
       Calls: UNDOQUAL                  (procedure in DEMO.PRG)
Contains: LASTQ                         (Params: none)
   Called by: M_UNDO                    (procedure in DEMO.PRG)
       Calls: LOOKUPQ                   (procedure in DEMO.PRG)
Contains: QID                           (Params: MSTR)
   Called by: UNDOQUAL                  (procedure in DEMO.PRG)
   Called by: EVALQ                     (procedure in DEMO.PRG)
   Called by: DEFQ                      (procedure in DEMO.PRG)
   Called by: GETQ                      (procedure in DEMO.PRG)
   Called by: ERASEQ                    (procedure in DEMO.PRG)
   Called by: RESTOREQ                  (procedure in DEMO.PRG)
   Called by: ADDQ                      (procedure in DEMO.PRG)
       Calls: ERRMESSAGE()              (function in U.PRG)
Contains: ADDQ                          (Params: MSTR)
       Calls: QID                       (procedure in DEMO.PRG)
Contains: STRCOND                       (Params: MCOND, MQID, MNAME, MVAL,
                                        DELIM)
   Called by: M_RESUME                  (procedure in DEMO.PRG)
   Called by: _EDITCOND                 (procedure in U.PRG)
   Called by: _EDITHANDP                (procedure in U.PRG)
       Calls: LOOKUPNAM                 (procedure in DEMO.PRG)
Contains: STRDIAG                       (Params:  MSORTED,  MDIAG,  MCONF,
                                        THRESHOLD, LIMIT, SHOWCONF)
   Called by: M_RESUME                  (procedure in DEMO.PRG)
   Called by: _EDITASSIS                (procedure in U.PRG)
       Calls: DISEASENAM()              (function in U.PRG)
Contains: STRNATURAL                    (Params: MAREA)
       Calls: GETQ                      (procedure in DEMO.PRG)
       Calls: DICTREPORT()              (function in U.PRG)
       Calls: SPLIT()                   (function in U.PRG)
Contains: BATCHTEST()                   (Params: TESTING, TESTFILE, LOGFILE)
   Called by: DEMO.PRG
       Calls: DELSYMPTOM                (procedure in DEMO.PRG)
       Calls: DELDIAGNOS                (procedure in DEMO.PRG)
       Calls: M_ENCRULE                 (procedure in DEMO.PRG)
       Calls: ENC2STR                   (procedure in DEMO.PRG)
       Calls: M_ENCGET                  (procedure in DEMO.PRG)
       Calls: M_ASSIST                  (procedure in DEMO.PRG)
       Calls: M_ENCSAV                  (procedure in DEMO.PRG)
       Calls: ENC2BIT                   (procedure in DEMO.PRG)
       Calls: DISEASENAM()              (function in U.PRG)
Contains: ENC2STR                       (Params: MAREA, MBIT)
   Called by: BATCHTEST()               (function in DEMO.PRG)
       Calls: INVERTCOND                (procedure in DEMO.PRG)
Contains: ENC2BIT                       (Params: FIRE)
   Called by: BATCHTEST()               (function in DEMO.PRG)
Contains: INVERTCOND                    (Params: MOBJECT, MID, MOP, MVAL)
   Called by: ENC2STR                   (procedure in DEMO.PRG)
       Calls: EVALCOND                  (procedure in DEMO.PRG)
Contains: SYSINIT()                     (Params: none)
   Called by: DEMO.PRG
       Calls: SETPATH                   (procedure in U.PRG)
       Calls: GETOPTS                   (procedure in U.PRG)
Contains: SYSCLOSE()                    (Params: none)
   Called by: DEMO.PRG
Contains: WPAT                          (Params: none)
   Called by: M_PAT                     (procedure in DEMO.PRG)
   Called by: M_NEWENC                  (procedure in DEMO.PRG)
   Called by: M_TRAINING                (procedure in DEMO.PRG)
       Calls: PUSHHELP                  (procedure in U.PRG)
       Calls: WP1.PRG
```

```
        Calls: DP.PRG
        Calls: AGEAT                    (procedure in DEMO.PRG)
        Calls: WPCLEAN                  (procedure in DEMO.PRG)
        Calls: POPHELP                  (procedure in U.PRG)
Contains: WPCLEAN                       (Params: none)
   Called by: WPAT                      (procedure in DEMO.PRG)
Contains: M_SF600                       (Params: none)
        Calls: OPENFILES()              (function in U.PRG)
        Calls: SFENCOUNT                (procedure in U.PRG)
        Calls: SFREPORT                 (procedure in DEMO.PRG)
        Calls: FLUSHFILES()             (function in U.PRG)
Contains: SFREPORT                      (Params: MAREA)
   Called by: M_SF600                   (procedure in DEMO.PRG)
        Calls: YESNO.PRG
        Calls: POPUPSHOW()              (function in U.PRG)
        Calls: SFSYMP                   (procedure in DEMO.PRG)
        Calls: POPUPHIDE()              (function in U.PRG)
        Calls: SFDIAG                   (procedure in DEMO.PRG)
        Calls: SFTREAT                  (procedure in DEMO.PRG)
        Calls: EDITTEXT                 (procedure in DEMO.PRG)
Contains: SFSYMP                        (Params: MAREA)
   Called by: SFREPORT                  (procedure in DEMO.PRG)
        Calls: CASECONVERT              (procedure in U.PRG)
        Calls: GETQ                     (procedure in DEMO.PRG)
        Calls: DICTREPORT()             (function in U.PRG)
        Calls: DP.PRG
Contains: EDITTEXT                      (Params: MTEXT)
   Called by: SFREPORT                  (procedure in DEMO.PRG)
        Calls: PUSHHELP                 (procedure in U.PRG)
        Calls: WP6.SPR
        Calls: POPHELP                  (procedure in U.PRG)
        Calls: WP7.SPR
Contains: SFDIAG                        (Params: none)
   Called by: SFREPORT                  (procedure in DEMO.PRG)
        Calls: DISEASENAME()            (function in U.PRG)
Contains: SFTREAT                       (Params: none)
   Called by: SFREPORT                  (procedure in DEMO.PRG)
Contains: M_NEWENC                      (Params: none)
        Calls: OPENFILES()              (function in U.PRG)
        Calls: WPAT                     (procedure in DEMO.PRG)
        Calls: M_ENC                    (procedure in DEMO.PRG)
        Calls: KEYLOG.PRG
        Calls: FLUSHFILES()             (function in U.PRG)
Contains: M_TRAINING                    (Params: none)
        Calls: OPENFILES()              (function in U.PRG)
        Calls: WPAT                     (procedure in DEMO.PRG)
        Calls: KEYLOG.PRG
        Calls: M_ENC                    (procedure in DEMO.PRG)
        Calls: FLUSHFILES()             (function in U.PRG)
Contains: BACKUP                        (Params: MESSAGE, WILDCARD, FILENAME)
        Calls: PUSHHELP                 (procedure in U.PRG)
        Calls: BACKUP.SPR
        Calls: POPHELP                  (procedure in U.PRG)
        Calls: KEYLOG.PRG
Contains: RESTORE                       (Params: MESSAGE, WILDCARD, FILENAME)
        Calls: PUSHHELP                 (procedure in U.PRG)
        Calls: RESTORE.SPR
        Calls: POPHELP                  (procedure in U.PRG)
        Calls: KEYLOG.PRG
Contains: M_LOOKUP                      (Params: none)
        Calls: DISEASECNT()             (function in U.PRG)
        Calls: DISNAMES()               (function in U.PRG)
```

```
        Calls: GETDISLOOKUPLIST          (procedure in U.PRG)
        Calls: KEYLOG.PRG
Contains: M_DATA                         (Params: none)
        Calls: GETDIAGAREA               (procedure in DEMO.PRG)
        Calls: QUESTION.PRG
        Calls: KEYLOG.PRG


U.PRG -- Last updated:  10/13/92 at 10:21

Contains: WCHOICE                        (Params: TOP, LEFT, BOTTOM, RIGHT,
                                         MPAD, MCOLOR, MTITLE, MHELP, MCHOICE)
   Called by: CHANGELIST()               (function in U.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: W14.SPR
        Calls: POPHELP                   (procedure in U.PRG)
Contains: WCONFIRM()                     (Params: M.PROMPT, MHELP)
   Called by: M_RESUME                   (procedure in DEMO.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: W10.SPR
        Calls: POPHELP                   (procedure in U.PRG)
Contains: USERINIT()                     (Params: NOTIME)
   Called by: DEMO.PRG
        Calls: INITHELP                  (procedure in U.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: GETCOLORS                 (procedure in U.PRG)
Contains: USERCLOSE()                    (Params: none)
   Called by: DEMO.PRG
Contains: CHANGELIST()                   (Params: QNAMES, MCHOICE)
   Called by: CHANGEQ                    (procedure in DEMO.PRG)
        Calls: WCHOICE                   (procedure in U.PRG)
Contains: GETDISLIST()                   (Params: MLIST)
   Called by: ASKDIAG                    (procedure in DEMO.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: W7.SPR
        Calls: POPHELP                   (procedure in U.PRG)
        Calls: DISEASEGET                (procedure in U.PRG)
        Calls: W8.PRG
Contains: GETDISLOOKUPLIST               (Params: MLIST)
   Called by: M_LOOKUP                   (procedure in DEMO.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: W7A.SPR
        Calls: POPHELP                   (procedure in U.PRG)
        Calls: DISEASEGET                (procedure in U.PRG)
        Calls: W8.PRG
Contains: EDNUM()                        (Params:  MID,  MNAME,  MQUESTION,
                                         MEXTRA,  MHI,  MLO,  MWIDTH,  MDEC,
                                         MUNITS, MSTRVAL)
   Called by: EDITSYMP                   (procedure in DEMO.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: W6.SPR
        Calls: POPHELP                   (procedure in U.PRG)
Contains: EDBOOL()                       (Params:  ID,  MNAME,  MQUESTION,
                                         MEXTRA, MSTRVAL)
   Called by: EDITSYMP                   (procedure in DEMO.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: W6C.SPR
        Calls: POPHELP                   (procedure in U.PRG)
Contains: EDLIST                         (Params:  ID,  MNAME,  MQUESTION,
                                         MEXTRA, CHOICES, MSTRVAL)
        Calls: PUSHHELP                  (procedure in U.PRG)
```

```
        Calls: W6B.SPR
        Calls: POPHELP                    (procedure in U.PRG)
Contains: EDCHECK()                       (Params:   ID,   MNAME,   MQUESTION,
                                          MEXTRA, CHOICES, INCLUSIVE, CHECK,
                                          MSTRVAL)
   Called by: EDITSYMP                    (procedure in DEMO.PRG)
        Calls: PUSHHELP                   (procedure in U.PRG)
        Calls: SETCHECK                   (procedure in U.PRG)
        Calls: W6A.PRG
        Calls: POPHELP                    (procedure in U.PRG)
Contains:  _EDCHECK                       (Params: none)
   Called by: W6A.PRG
        Calls: SETCHECK                   (procedure in U.PRG)
Contains: SETCHECK                        (Params: N, CHECK, INCLUSIVE)
   Called by: EDCHECK()                   (function in U.PRG)
   Called by:  _EDCHECK                   (procedure in U.PRG)
Contains: GETENCLIST()                    (Params: MLIST, MKEY)
   Called by: GETENCNT                    (procedure in DEMO.PRG)
        Calls: PUSHHELP                   (procedure in U.PRG)
        Calls: W11.SPR
        Calls: POPHELP                    (procedure in U.PRG)
Contains: ENCNTTITLE()                    (Params:    ME_SSN,    ME_LASTNAM,
                                          ME_FIRSTNA,    ME_SEX,    ME_AGE,
                                          ME_DATEENC,   ME_TIMEENC,   ME_AREA,
                                          ME_PROVIDR,   ME_DIAGNOS,   ME_DIAG,
                                          ME_COND, ME_TREAT, ME_STATE)
   Called by: M_RESUME                    (procedure in DEMO.PRG)
        Calls: PUSHHELP                   (procedure in U.PRG)
        Calls: W12.SPR
        Calls: POPHELP                    (procedure in U.PRG)
Contains:  _EVALID                        (Params: none)
   Called by: W12.SPR
        Calls: PUSHHELP                   (procedure in U.PRG)
        Calls: W2.SPR
        Calls: POPHELP                    (procedure in U.PRG)
Contains:  _EDITCOND                      (Params: MITEM)
   Called by: W12.SPR
   Called by: W12A.SPR
        Calls: M_REDOQ                    (procedure in DEMO.PRG)
        Calls: STRCOND                    (procedure in DEMO.PRG)
Contains:  _EDITVDIAG                     (Params: MITEM)
   Called by: W12.SPR
   Called by: W12B.SPR
        Calls: DISEASEGET                 (procedure in U.PRG)
        Calls: PUSHHELP                   (procedure in U.PRG)
        Calls: W8.PRG
        Calls: POPHELP                    (procedure in U.PRG)
Contains:  _EDITCOMPL                     (Params: none)
   Called by: W12.SPR
        Calls: M_COMPL                    (procedure in DEMO.PRG)
        Calls: GETEXAREA()                (function in U.PRG)
Contains:  _EDITDIAG                      (Params: none)
   Called by: W12.SPR
        Calls: M_DIAG                     (procedure in DEMO.PRG)
        Calls: DISEASENAM()               (function in U.PRG)
Contains:  _EDITHANDP                     (Params: none)
   Called by: W12.SPR
   Called by: W12A.SPR
        Calls: M_HANDP                    (procedure in DEMO.PRG)
        Calls: STRCOND                    (procedure in DEMO.PRG)
Contains:  _EDITASSIS                     (Params: none)
   Called by: W12.SPR
```

```
   Called by: W12B.SPR
       Calls: M_ASSIST                    (procedure in DEMO.PRG)
       Calls: STRDIAG                     (procedure in DEMO.PRG)
Contains:  _EDITTREAT                     (Params: none)
   Called by: W12.SPR
       Calls: M_TREAT                     (procedure in DEMO.PRG)
Contains: RELEASE                         (Params: MWINS)
Contains:  _PRINTENC                      (Params: MPRINTFILE)
   Called by:  _Q7IOLH8WD()               (function in W2.SPR)
Contains:  _STOPREAD                      (Params: MWINDOW)
   Called by: W12.SPR
   Called by: W4.SPR
Contains: ERRMESSAGE()                    (Params: ERRSTR)
   Called by: M_PATGET                    (procedure in DEMO.PRG)
   Called by: FACTORID                    (procedure in DEMO.PRG)
   Called by: EDITSYMP                    (procedure in DEMO.PRG)
   Called by: QID                         (procedure in DEMO.PRG)
Contains: POPMESSAGE()                    (Params: ERRSTR)
   Called by: M_ENC                       (procedure in DEMO.PRG)
   Called by: M_COMPL                     (procedure in DEMO.PRG)
   Called by: M_DIAG                      (procedure in DEMO.PRG)
   Called by: M_ASSIST                    (procedure in DEMO.PRG)
   Called by: M_HANDP                     (procedure in DEMO.PRG)
   Called by: M_TREAT                     (procedure in DEMO.PRG)
   Called by: M_ENCSAV                    (procedure in DEMO.PRG)
   Called by: M_CHANGE                    (procedure in DEMO.PRG)
   Called by: M_PATPUT                    (procedure in DEMO.PRG)
   Called by: M_PATADD                    (procedure in DEMO.PRG)
   Called by:  _QUIT                      (procedure in U.PRG)
   Called by: SEARCH                      (procedure in U.PRG)
   Called by: MORESEARCH                  (procedure in U.PRG)
   Called by: SFENCOUNT                   (procedure in U.PRG)
   Called by: SFGETENC                    (procedure in U.PRG)
   Called by:  _Q8D0TWCZ4()               (function in WP2.PRG)
   Called by:  _Q8D0TWDV1()               (function in WP2.PRG)
       Calls: PUSHHELP                    (procedure in U.PRG)
       Calls: W9.SPR
       Calls: POPHELP                     (procedure in U.PRG)
Contains: POPUPSHOW()                     (Params: ERRSTR)
   Called by: DEMO.PRG
   Called by: OPENFILES()                 (function in U.PRG)
   Called by: M_ASSIST                    (procedure in DEMO.PRG)
   Called by: M_HANDP                     (procedure in DEMO.PRG)
   Called by: M_ENCGET                    (procedure in DEMO.PRG)
   Called by: SFREPORT                    (procedure in DEMO.PRG)
   Called by: REINDEX                     (procedure in U.PRG)
Contains: POPUPHIDE()                     (Params: W)
   Called by: DEMO.PRG
   Called by: OPENFILES()                 (function in U.PRG)
   Called by: M_ASSIST                    (procedure in DEMO.PRG)
   Called by: M_HANDP                     (procedure in DEMO.PRG)
   Called by: M_ENCGET                    (procedure in DEMO.PRG)
   Called by: SFREPORT                    (procedure in DEMO.PRG)
   Called by: SEARCH                      (procedure in U.PRG)
   Called by: MORESEARCH                  (procedure in U.PRG)
   Called by: REINDEX                     (procedure in U.PRG)
Contains: GETAREALIST()                   (Params: MLIST)
   Called by: GETDIAGAREA                 (procedure in DEMO.PRG)
       Calls: PUSHHELP                    (procedure in U.PRG)
       Calls: W5.SPR
       Calls: POPHELP                     (procedure in U.PRG)
Contains: GETPATLIST()                    (Params: MKEY, MLIST)
```

```
   Called by: GETPATIENT                    (procedure in DEMO.PRG)
       Calls: PUSHHELP                      (procedure in U.PRG)
       Calls: W3.SPR
       Calls: POPHELP                       (procedure in U.PRG)
Contains: MAINMENU()                        (Params: none)
   Called by: DEMO.PRG
       Calls: CAMD.MPR
       Calls: MYHANDLER                     (procedure in U.PRG)
Contains: MYHANDLER                         (Params: none)
   Called by: MAINMENU()                    (function in U.PRG)
       Calls: M_RESUME                      (procedure in DEMO.PRG)
       Calls: M_REGPAT                      (procedure in DEMO.PRG)
Contains: _MHIT                             (Params: PROG)
   Called by: CAMD.MPR
   Called by: _Q9POPFLR8                    (procedure in CAMD.MPR)
   Called by: CAMDTR.MPR
Contains: _QUIT                             (Params: none)
   Called by: CAMD.MPR
   Called by: CAMDTR.MPR
       Calls: POPMESSAGE()                  (function in U.PRG)
Contains: PATID()                           (Params:    MPATIENTID,    MSOCSEC,
                                            MLASTNAME,  MFIRSTNAME,  MDATEBIRTH,
                                            MSEX, MRATERANK, ISEDIT)
   Called by: M_REGPAT                      (procedure in DEMO.PRG)
       Calls: PUSHHELP                      (procedure in U.PRG)
       Calls: W4.SPR
       Calls: POPHELP                       (procedure in U.PRG)
Contains: _PVALID                           (Params: none)
   Called by: W4.SPR
Contains: UNDOMENU()                        (Params: none)
   Called by: MYUNDO                        (procedure in DEMO.PRG)
       Calls: W13.SPR
Contains: PUSHHELP                          (Params: HELPTOPIC)
   Called by: WP2.PRG
   Called by: USERINIT()                    (function in U.PRG)
   Called by: WPAT                          (procedure in DEMO.PRG)
   Called by: EDITTEXT                      (procedure in DEMO.PRG)
   Called by: BACKUP                        (procedure in DEMO.PRG)
   Called by: RESTORE                       (procedure in DEMO.PRG)
   Called by: ENCNTTITLE()                  (function in U.PRG)
   Called by: WCONFIRM()                    (function in U.PRG)
   Called by: PATID()                       (function in U.PRG)
   Called by: POPMESSAGE()                  (function in U.PRG)
   Called by: GETDISLIST()                  (function in U.PRG)
   Called by: GETAREALIST()                 (function in U.PRG)
   Called by: GETPATLIST()                  (function in U.PRG)
   Called by: GETENCLIST()                  (function in U.PRG)
   Called by: EDNUM()                       (function in U.PRG)
   Called by: EDBOOL()                      (function in U.PRG)
   Called by: EDCHECK()                     (function in U.PRG)
   Called by: WCHOICE                       (procedure in U.PRG)
   Called by: GETDISLOOKUPLIST              (procedure in U.PRG)
   Called by: EDLIST                        (procedure in U.PRG)
   Called by: _EVALID                       (procedure in U.PRG)
   Called by: _EDITVDIAG                    (procedure in U.PRG)
   Called by: WPGETPATLIST                  (procedure in U.PRG)
   Called by: SEARCH                        (procedure in U.PRG)
   Called by: SFENCOUNT                     (procedure in U.PRG)
   Called by: SFENCLIST                     (procedure in U.PRG)
   Called by: SFGETPATLIST                  (procedure in U.PRG)
   Called by: _Q9POPFLOY                    (procedure in CAMD.MPR)
       Calls: SETHELP                       (procedure in U.PRG)
```

```
Contains: POPHELP                        (Params: none)
   Called by: WP2.PRG
   Called by: WPAT                       (procedure in DEMO.PRG)
   Called by: EDITTEXT                   (procedure in DEMO.PRG)
   Called by: BACKUP                     (procedure in DEMO.PRG)
   Called by: RESTORE                    (procedure in DEMO.PRG)
   Called by: ENCNTTITLE()               (function in U.PRG)
   Called by: WCONFIRM()                 (function in U.PRG)
   Called by: PATID()                    (function in U.PRG)
   Called by: POPMESSAGE()               (function in U.PRG)
   Called by: GETDISLIST()               (function in U.PRG)
   Called by: GETAREALIST()              (function in U.PRG)
   Called by: GETPATLIST()               (function in U.PRG)
   Called by: GETENCLIST()               (function in U.PRG)
   Called by: EDNUM()                    (function in U.PRG)
   Called by: EDBOOL()                   (function in U.PRG)
   Called by: EDCHECK()                  (function in U.PRG)
   Called by: WCHOICE                    (procedure in U.PRG)
   Called by: GETDISLOOKUPLIST           (procedure in U.PRG)
   Called by: EDLIST                     (procedure in U.PRG)
   Called by: _EVALID                    (procedure in U.PRG)
   Called by: _EDITVDIAG                 (procedure in U.PRG)
   Called by: WPGETPATLIST               (procedure in U.PRG)
   Called by: SEARCH                     (procedure in U.PRG)
   Called by: SFENCOUNT                  (procedure in U.PRG)
   Called by: SFENCLIST                  (procedure in U.PRG)
   Called by: SFGETPATLIST               (procedure in U.PRG)
   Called by: _Q9POPFLOY                 (procedure in CAMD.MPR)
       Calls: SETHELP                    (procedure in U.PRG)
Contains: SETHELP                        (Params: none)
   Called by: PUSHHELP                   (procedure in U.PRG)
   Called by: POPHELP                    (procedure in U.PRG)
Contains: OLMR                           (Params: none)
   Called by: INITHELP                   (procedure in U.PRG)
Contains: INITHELP                       (Params: none)
   Called by: USERINIT()                 (function in U.PRG)
       Calls: OLMR                       (procedure in U.PRG)
Contains: AREACOUNT()                    (Params: none)
   Called by: GETDIAGAREA                (procedure in DEMO.PRG)
Contains: AREANAMES()                    (Params: MAREA, MNAME)
   Called by: GETDIAGAREA                (procedure in DEMO.PRG)
Contains: GETEXAREA()                    (Params: MAREA)
   Called by: M_RESUME                   (procedure in DEMO.PRG)
   Called by: _EDITCOMPL                 (procedure in U.PRG)
Contains: GETRULEBASE()                  (Params:    MAREA,    CTHRESHOLD,
                                         CTHRESHPRO, CTHRESHLIK, NRULES)
   Called by: LODRULE                    (procedure in DEMO.PRG)
Contains: GETCOLORS                      (Params: COLORSET)
   Called by: USERINIT()                 (function in U.PRG)
Contains: PUTDIAGNOS()                   (Params: MENCOUNTID, MDIAG, MCONF)
   Called by: M_ENCSAV                   (procedure in DEMO.PRG)
       Calls: OPENDIAG                   (procedure in U.PRG)
Contains: GETDIAGNOS()                   (Params: MENCOUNTID)
   Called by: LODENC                     (procedure in DEMO.PRG)
       Calls: OPENDIAG                   (procedure in U.PRG)
       Calls: ADDDIAGNOS                 (procedure in DEMO.PRG)
Contains: OPENDIAG                       (Params: none)
   Called by: PUTDIAGNOS()               (function in U.PRG)
   Called by: GETDIAGNOS()               (function in U.PRG)
Contains: OPENDICT                       (Params: none)
   Called by: OPENFILES()                (function in U.PRG)
Contains: CLOSEDICT                      (Params: none)
```

Called by: FLUSHFILES()	(function in U.PRG)
Contains: DICTCOUNT()	(Params: none)
Called by: INISYMPTOM()	(function in DEMO.PRG)
Contains: DICTQ()	(Params: MID, MASK, MQ, MNAME, MPROMPT)
Called by: LOOKUPQ	(procedure in DEMO.PRG)
Called by: LOOKUPNAM	(procedure in DEMO.PRG)
Called by: EDITSYMP	(procedure in DEMO.PRG)
Called by: PUTSYMPTOM()	(function in U.PRG)
Called by: GETSYMPTOM()	(function in U.PRG)
Contains: DICTQV()	(Params: MID, MHI, MLO, MWIDTH, MDEC, MUNITS)
Called by: EDITSYMP	(procedure in DEMO.PRG)
Contains: DICTECOUNT()	(Params: MID)
Called by: EDITSYMP	(procedure in DEMO.PRG)
Contains: DICTQE()	(Params: MID, NAMES, MATTRIB)
Called by: EDITSYMP	(procedure in DEMO.PRG)
Contains: DICTORD	(Params: MID, MENUM)
Called by: PUTSYMPTOM()	(function in U.PRG)
Contains: DICTENUM	(Params: MID, MORD)
Called by: GETSYMPTOM()	(function in U.PRG)
Contains: DICTREPORT()	(Params: MID, MENUM)
Called by: STRNATURAL	(procedure in DEMO.PRG)
Called by: SFSYMP	(procedure in DEMO.PRG)
Contains: DICTQID	(Params: MNAME)
Contains: DICTQNAME()	(Params: MNAME)
Called by: ADDHISTORY	(procedure in DEMO.PRG)
Contains: OPENENCNT	(Params: none)
Called by: OPENFILES()	(function in U.PRG)
Contains: FLUSHENCNT	(Params: none)
Called by: FLUSHFILES()	(function in U.PRG)
Contains: NEWENCNT()	(Params: MENCOUNTID, MPATIENTID, MDATEENC, MTIMEENC, MPROVIDER, MSTATION, MDEPT, MAREA, MDIAGNOSIS, MTREATMENT, MSTATE)
Called by: M_ENCSAV	(procedure in DEMO.PRG)
Contains: ENCID	(Params: MPATIENTID, MDATE, MTIME)
Contains: PUTENCNT()	(Params: MENCOUNTID, MPATIENT, MDATEENC, MTIMEENC, MPROVIDER, MSTATION, MDEPT, MAREA, MDIAGNOSIS, MTREATMENT, MSTATE)
Called by: M_ENCSAV	(procedure in DEMO.PRG)
Contains: ENCGETINFO()	(Params: MENCOUNTID, MPATIENTID, MDATEENC, MTIMEENC, MPROVIDER, MSTATION, MDEPT, MAREA, MDIAGNOSIS, MTREATMENT, MSTATE, MREPORT)
Called by: LODENC	(procedure in DEMO.PRG)
Contains: ENCCOUNT()	(Params: none)
Called by: GETENCNT	(procedure in DEMO.PRG)
Called by: SFGETENC	(procedure in U.PRG)
Contains: ENCNAMES()	(Params: NAMES, IDS, PATIENTS, MPATIENT, MDATE)
Called by: GETENCNT	(procedure in DEMO.PRG)
Calls: PATSSN()	(function in U.PRG)
Contains: DISEASEGET	(Params: MDISEASE)
Called by: GETDISLIST()	(function in U.PRG)
Called by: GETDISLOOKUPLIST	(procedure in U.PRG)
Called by: _EDITVDIAG	(procedure in U.PRG)
Contains: DISEASECNT()	(Params: none)
Called by: INIDIAGNOS()	(function in DEMO.PRG)
Called by: ASKDIAG	(procedure in DEMO.PRG)
Called by: M_LOOKUP	(procedure in DEMO.PRG)

```
Contains: DISNAMES()                    (Params: MDISEASES, MAREA)
   Called by: ASKDIAG                   (procedure in DEMO.PRG)
   Called by: M_LOOKUP                  (procedure in DEMO.PRG)
Contains: DISEASEPUT                    (Params: MDISEASE, MHELP)
Contains: DISEASENAM()                  (Params: MID, MNAME)
   Called by: BATCHTEST()               (function in DEMO.PRG)
   Called by: M_RESUME                  (procedure in DEMO.PRG)
   Called by: M_DIAG                    (procedure in DEMO.PRG)
   Called by: STRDIAG                   (procedure in DEMO.PRG)
   Called by: _EDITDIAG                 (procedure in U.PRG)
Contains: DISEASEID()                   (Params: DNAME, DID)
   Called by: M_DIAG                    (procedure in DEMO.PRG)
Contains: OPENFILES()                   (Params: TESTING)
   Called by: DEMO.PRG
   Called by: M_PAT                     (procedure in DEMO.PRG)
   Called by: M_SF600                   (procedure in DEMO.PRG)
   Called by: M_NEWENC                  (procedure in DEMO.PRG)
   Called by: M_TRAINING                (procedure in DEMO.PRG)
       Calls: POPUPSHOW()               (function in U.PRG)
       Calls: OPENPAT                   (procedure in U.PRG)
       Calls: OPENENCNT                 (procedure in U.PRG)
       Calls: OPENDICT                  (procedure in U.PRG)
       Calls: OPENRULE                  (procedure in U.PRG)
       Calls: POPUPHIDE()               (function in U.PRG)
Contains: FLUSHFILES()                  (Params: none)
   Called by: M_PAT                     (procedure in DEMO.PRG)
   Called by: M_SF600                   (procedure in DEMO.PRG)
   Called by: M_NEWENC                  (procedure in DEMO.PRG)
   Called by: M_TRAINING                (procedure in DEMO.PRG)
       Calls: FLUSHPAT                  (procedure in U.PRG)
       Calls: FLUSHENCNT                (procedure in U.PRG)
       Calls: CLOSEDICT                 (procedure in U.PRG)
       Calls: CLOSERULE                 (procedure in U.PRG)
Contains: CLOSEFILES                    (Params: none)
   Called by: BACKUP.SPR
   Called by: RESTORE.SPR
   Called by: REINDEX                   (procedure in U.PRG)
Contains: OPENPAT                       (Params: none)
   Called by: OPENFILES()               (function in U.PRG)
       Calls: FILEFIND                  (procedure in U.PRG)
       Calls: ERRHAND                   (procedure in U.PRG)
       Calls: OPENSAMS                  (procedure in U.PRG)
Contains: FLUSHPAT                      (Params: none)
   Called by: FLUSHFILES()              (function in U.PRG)
Contains: PATSSN()                      (Params: MID)
   Called by: ENCOUNT                   (procedure in DEMO.PRG)
   Called by: GETENCNT                  (procedure in DEMO.PRG)
   Called by: ENCNAMES()                (function in U.PRG)
   Called by: SFENCOUNT                 (procedure in U.PRG)
   Called by: SFENCLIST                 (procedure in U.PRG)
   Called by: SFENCNAMES                (procedure in U.PRG)
Contains: PATGETINFO()                  (Params:    MPATIENTID,    MSOCSEC,
                                        MLASTNAME,  MFIRSTNAME,  MDATEBIRTH,
                                        MSEX, MRATERANK)
   Called by: M_PATGET                  (procedure in DEMO.PRG)
Contains: PATPUTINFO()                  (Params:    MPATIENTID,    MSOCSEC,
                                        MLASTNAME,  MFIRSTNAME,  MDATEBIRTH,
                                        MSEX, MRATERANK)
   Called by: M_PATPUT                  (procedure in DEMO.PRG)
Contains: NEWPATIENT()                  (Params:    MPATIENTID,    MSOCSEC,
                                        MLASTNAME,  MFIRSTNAME,  MDATEBIRTH,
                                        MSEX, MRATERANK)
```

```
   Called by: M_PATADD                    (procedure in DEMO.PRG)
Contains: DELPATIENT                       (Params: MPATIENTID)
Contains: ISVALIDPAT                      (Params: MPATIENTID)
Contains: ISVALIDSSN()                    (Params: MSOCSEC)
   Called by: M_PATPUT                    (procedure in DEMO.PRG)
   Called by: M_PATADD                    (procedure in DEMO.PRG)
Contains: PATCOUNT()                      (Params: none)
   Called by: GETPATIENT                  (procedure in DEMO.PRG)
Contains: PATNAMES()                      (Params: NAMES, DELIM, MKEY)
   Called by: GETPATIENT                  (procedure in DEMO.PRG)
       Calls: STRNAME                     (procedure in U.PRG)
Contains: STRNAME                         (Params:    MSOCSEC,    MLASTNAM,
                                          MFIRSTNAME, DELIM)
   Called by: PATNAMES()                  (function in U.PRG)
Contains: OPENRULE                        (Params: none)
   Called by: OPENFILES()                 (function in U.PRG)
Contains: CLOSERULE                       (Params: none)
   Called by: FLUSHFILES()                (function in U.PRG)
Contains: GETRULE                         (Params: RULEID)
Contains: PUTSYMPTOM()                    (Params: MENCOUNTID, MCOND, MVAL)
   Called by: M_ENCSAV                    (procedure in DEMO.PRG)
       Calls: OPENSYMP                    (procedure in U.PRG)
       Calls: SPLIT()                     (function in U.PRG)
       Calls: DICTQ()                     (function in U.PRG)
       Calls: DICTORD                     (procedure in U.PRG)
Contains: SPLIT()                         (Params: MQ)
   Called by: STRNATURAL                  (procedure in DEMO.PRG)
   Called by: PUTSYMPTOM()                (function in U.PRG)
Contains: GETSYMPTOM()                    (Params: MENCOUNTID)
   Called by: LODENC                      (procedure in DEMO.PRG)
       Calls: OPENSYMP                    (procedure in U.PRG)
       Calls: DICTQ()                     (function in U.PRG)
       Calls: DICTENUM                    (procedure in U.PRG)
       Calls: ADDSYMPTOM                  (procedure in DEMO.PRG)
Contains: OPENSYMP                        (Params: none)
   Called by: PUTSYMPTOM()                (function in U.PRG)
   Called by: GETSYMPTOM()                (function in U.PRG)
Contains: GETSTATION()                    (Params: MSTATION, MDEPT)
   Called by: INIENCOUNT()                (function in DEMO.PRG)
Contains: GETOPTS                         (Params: IS_DEBUG, IS_CONF, IS_RULE,
                                          IS_NOVICE,   IS_EDIT,    MPRIVATE,
                                          MDISCLAIM)
   Called by: SYSINIT()                   (function in DEMO.PRG)
Contains: NEXTPAT                         (Params: INC)
   Called by: ADDPAT()                    (function in U.PRG)
Contains: ADDPAT()                        (Params: none)
   Called by: M_PATADD                    (procedure in DEMO.PRG)
       Calls: NEXTPAT                     (procedure in U.PRG)
Contains: NEXTENCNT()                     (Params: INC)
   Called by: M_ENCNEW                    (procedure in DEMO.PRG)
   Called by: ADDENCNT()                  (function in U.PRG)
Contains: ADDENCNT()                      (Params: none)
   Called by: M_ENCSAV                    (procedure in DEMO.PRG)
       Calls: NEXTENCNT()                 (function in U.PRG)
Contains: NENCNT                          (Params: none)
Contains: NEXTUSER                        (Params: INC)
   Called by: ADDUSER                     (procedure in U.PRG)
Contains: ADDUSER                         (Params: none)
       Calls: NEXTUSER                    (procedure in U.PRG)
Contains: BROWSEPAT                       (Params: MSOC, MLAST, MFIRST, MMID)
   Called by: _Q8D0YAR6U()                (function in WP1.PRG)
       Calls: ERRMSG.PRG
```

```
        Calls: WPGETPATLIST              (procedure in U.PRG)
Contains: WPGETPATLIST                   (Params: MLIST)
   Called by: BROWSEPAT                  (procedure in U.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: WP3.SPR
        Calls: POPHELP                   (procedure in U.PRG)
        Calls: DP.PRG
        Calls: GETPAT                    (procedure in U.PRG)
        Calls: WP2.PRG
Contains: SEARCH                         (Params: MSOC, MLAST, MFIRST)
   Called by: _Q8D0YAR6U()               (function in WP1.PRG)
        Calls: POPMESSAGE()              (function in U.PRG)
        Calls: GETCAMDPAT                (procedure in U.PRG)
        Calls: GETSAMSPAT                (procedure in U.PRG)
        Calls: DP.PRG
        Calls: GETSAMS                   (procedure in U.PRG)
        Calls: POPUPHIDE()               (function in U.PRG)
        Calls: PUSHHELP                  (procedure in U.PRG)
        Calls: WPATS.PRG
        Calls: POPHELP                   (procedure in U.PRG)
        Calls: WP2.PRG
Contains: MORESEARCH                     (Params: MITEM, CHOICE)
   Called by: _Q7I0L5TJ5()               (function in WPATS.PRG)
        Calls: DP.PRG
        Calls: POPMESSAGE()              (function in U.PRG)
        Calls: GETSAMSPAT                (procedure in U.PRG)
        Calls: POPUPHIDE()               (function in U.PRG)
Contains: GETCAMDPAT                     (Params: MSSN, MLAST, MFIRST)
   Called by: SEARCH                     (procedure in U.PRG)
   Called by: SFGETPAT                   (procedure in U.PRG)
Contains: GETSAMSPAT                     (Params: MSSN, MLAST, MFIRST)
   Called by: SEARCH                     (procedure in U.PRG)
   Called by: MORESEARCH                 (procedure in U.PRG)
Contains: GETPATDATA                     (Params: MITEM)
   Called by: RESULT                     (procedure in WPATS.PRG)
   Called by: PATEDIT                    (procedure in WPATS.PRG)
        Calls: DP.PRG
        Calls: GETPATTEMP                (procedure in U.PRG)
        Calls: GETSAMS                   (procedure in U.PRG)
Contains: GETSAMS                        (Params: MREC)
   Called by: SEARCH                     (procedure in U.PRG)
   Called by: GETPATDATA                 (procedure in U.PRG)
        Calls: SAMSDATA                  (procedure in U.PRG)
Contains: SAMSDATA                       (Params: none)
   Called by: GETSAMS                    (procedure in U.PRG)
        Calls: DP.PRG
        Calls: COMBINED                  (procedure in U.PRG)
Contains: GETPAT                         (Params: MREC)
   Called by: WPGETPATLIST               (procedure in U.PRG)
        Calls: PATDATA                   (procedure in U.PRG)
Contains: GETPATTEMP                     (Params: MREC)
   Called by: GETPATDATA                 (procedure in U.PRG)
   Called by: SFGETPATLIST               (procedure in U.PRG)
        Calls: PATDATA                   (procedure in U.PRG)
Contains: PATDATA                        (Params: none)
   Called by: GETPAT                     (procedure in U.PRG)
   Called by: GETPATTEMP                 (procedure in U.PRG)
        Calls: COMBINED                  (procedure in U.PRG)
Contains: COMBINED                       (Params: MT1, MT2, MT3, MT4, MT5,
                                         MT6, MT7, MT8, MT9)
   Called by: SAMSDATA                   (procedure in U.PRG)
   Called by: PATDATA                    (procedure in U.PRG)
```

```
   Called by: SAVREC                    (procedure in U.PRG)
   Called by: _Q8D0TWDV1()              (function in WP2.PRG)
Contains: SAVREC                        (Params: MFLAG, MITEM)
   Called by: _Q8D0TWDV1()              (function in WP2.PRG)
   Called by: RESULT                    (procedure in WPATS.PRG)
       Calls: DP.PRG
       Calls: COMBINED                  (procedure in U.PRG)
Contains: OPENSAMS                      (Params: none)
   Called by: OPENPAT                   (procedure in U.PRG)
       Calls: SETPATH                   (procedure in U.PRG)
       Calls: FILEFIND                  (procedure in U.PRG)
Contains: CASECONVERT                   (Params: MSTR, MCASE)
   Called by: SFSYMP                    (procedure in DEMO.PRG)
Contains: SFENCOUNT                     (Params: none)
   Called by: M_SF600                   (procedure in DEMO.PRG)
       Calls: M_PATGET                  (procedure in DEMO.PRG)
       Calls: PUSHHELP                  (procedure in U.PRG)
       Calls: WP4.SPR
       Calls: POPHELP                   (procedure in U.PRG)
       Calls: SFGETPAT                  (procedure in U.PRG)
       Calls: SFGETENC                  (procedure in U.PRG)
       Calls: POPMESSAGE()              (function in U.PRG)
       Calls: PATSSN()                  (function in U.PRG)
       Calls: LODENC                    (procedure in DEMO.PRG)
Contains: SFGETENC                      (Params: PAR1, PAR2)
   Called by: SFENCOUNT                 (procedure in U.PRG)
       Calls: ENCCOUNT()                (function in U.PRG)
       Calls: SFENCNAMES                (procedure in U.PRG)
       Calls: SFENCLIST                 (procedure in U.PRG)
       Calls: POPMESSAGE()              (function in U.PRG)
Contains: SFENCLIST                     (Params: MLIST, MPAT, MDATENC)
   Called by: SFGETENC                  (procedure in U.PRG)
       Calls: PATSSN()                  (function in U.PRG)
       Calls: PUSHHELP                  (procedure in U.PRG)
       Calls: WP5.SPR
       Calls: POPHELP                   (procedure in U.PRG)
Contains: SFGETPAT                      (Params: MSOC, MLASTT, MFIRSTT)
   Called by: SFENCOUNT                 (procedure in U.PRG)
       Calls: GETCAMDPAT                (procedure in U.PRG)
       Calls: SFGETPATLIST              (procedure in U.PRG)
       Calls: DP.PRG
Contains: SFENCNAMES                    (Params:  NAMES,  IDS,  PATIENTS,
                                        MPATIENT, MDATE)
   Called by: SFGETENC                  (procedure in U.PRG)
       Calls: PATSSN()                  (function in U.PRG)
Contains: SFGETPATLIST                  (Params: MLIST)
   Called by: SFGETPAT                  (procedure in U.PRG)
       Calls: PUSHHELP                  (procedure in U.PRG)
       Calls: WP8.SPR
       Calls: POPHELP                   (procedure in U.PRG)
       Calls: DP.PRG
       Calls: GETPATTEMP                (procedure in U.PRG)
Contains: SFPRINT                       (Params: MPRINTFILE, MSFSTR)
   Called by: _Q7I0LI7F7()              (function in WP7.SPR)
Contains: SETPATH                       (Params: NEWPATH)
   Called by: SYSINIT()                 (function in DEMO.PRG)
   Called by: OPENSAMS                  (procedure in U.PRG)
   Called by: PKUNZIP                   (procedure in RESTORE.SPR)
Contains: ERRHAND                       (Params: ERRMSG, QUITFLAG)
   Called by: OPENPAT                   (procedure in U.PRG)
       Calls: ERRMSG.PRG
Contains: FILEFIND                      (Params: MFILENAME)
```

```
      Called by: KEYLOG.PRG
      Called by: OPENPAT                    (procedure in U.PRG)
      Called by: OPENSAMS                   (procedure in U.PRG)
          Calls: DP.PRG
   Contains: REINDEX                        (Params: none)
      Called by: CAMD.MPR
      Called by: CAMDTR.MPR
          Calls: POPUPSHOW()                (function in U.PRG)
          Calls: CLOSEFILES                 (procedure in U.PRG)
          Calls: POPUPHIDE()                (function in U.PRG)


W12.SPR -- Last updated:  10/13/92 at 14:17

   Contains: _Q9V0UMCRG()                   (Params: none)
      Called by: W12.SPR
          Calls: W12C.SPR
   Contains: _Q9V0UMD2M()                   (Params: none)
      Called by: W12.SPR
   Contains: _Q9V0UMDDI()                   (Params: none)
      Called by: W12.SPR
   Contains: _Q9V0UMDJJ()                   (Params: none)
      Called by: W12.SPR
          Calls: W12A.SPR
   Contains: _Q9V0UMDQF()                   (Params: none)
      Called by: W12.SPR
   Contains: _Q9V0UMDWG()                   (Params: none)
      Called by: W12.SPR
          Calls: W12B.SPR
   Contains: _Q9V0UME80()                   (Params: none)
      Called by: W12.SPR


W2.SPR -- Last updated:  07/20/92 at 10:01

   Contains: _Q7I0LH8OT()                   (Params: none)
      Called by: W2.SPR
      Called by: Q2.PRG
   Contains: _Q7I0LH8WD()                   (Params: none)
      Called by: W2.SPR
      Called by: Q2.PRG
          Calls: _PRINTENC                  (procedure in U.PRG)
   Contains: _Q7I0LH94Q()                   (Params: none)
      Called by: W2.SPR
      Called by: Q2.PRG


WP1.PRG -- Last updated:  08/20/92 at 16:00

   Contains: _Q8D0YAR6U()                   (Params: none)
      Called by: WP1.PRG
          Calls: SEARCH                     (procedure in U.PRG)
          Calls: BROWSEPAT                  (procedure in U.PRG)


WP2.PRG -- Last updated:  10/09/92 at 14:13

   Contains: _Q8D0TWCZ4()                   (Params: none)
      Called by: WP2.PRG
          Calls: POPMESSAGE()               (function in U.PRG)
   Contains: _Q8D0TWDKT()                   (Params: none)
      Called by: WP2.PRG
```

```
   Contains:  _Q8D0TWDV1()                    (Params: none)
      Called by: WP2.PRG
          Calls: POPMESSAGE()                 (function in U.PRG)
          Calls: COMBINED                     (procedure in U.PRG)
          Calls: SAVREC                       (procedure in U.PR

WPATS.PRG -- Last updated:  07/31/92 at 14:41

   Contains: RESULT                           (Params: MITEM)
      Called by:  _Q7I0L5TJ5()                (function in WPATS.PRG)
          Calls: DP.PRG
          Calls: GETPATDATA                   (procedure in U.PRG)
          Calls: SAVREC                       (procedure in U.PRG)
   Contains: PATEDIT                          (Params: MITEM)
      Called by:  _Q7I0L5TJ5()                (function in WPATS.PRG)
          Calls: DP.PRG
          Calls: GETPATDATA                   (procedure in U.PRG)
          Calls: WP2.PRG
   Contains: PATNEW                           (Params: none)
      Called by:  _Q7I0L5TJ5()                (function in WPATS.PRG)
          Calls: WP2.PRG
   Contains:  _Q7I0L5TB2()                    (Params: none)
      Called by: WPATS.PRG
   Contains:  _Q7I0L5TJ5()                    (Params: none)
      Called by: WPATS.PRG
          Calls: RESULT                       (procedure in WPATS.PRG)
          Calls: PATEDIT                      (procedure in WPATS.PRG)
          Calls: PATNEW                       (procedure in WPATS.PRG)
          Calls: MORESEARCH                   (procedure in U.PRG)
   Contains:  _Q7I0L5TXD()                    (Params: none)
      Called by: WPATS.PRG


WP3.SPR -- Last updated:  08/03/92 at 16:06

   Contains:  _Q7W0YICA9()                    (Params: none)
      Called by: WP3.SPR


WP7.SPR -- Last updated:  07/20/92 at 10:02

   Contains:  _Q7I0LI77M()                    (Params: none)
      Called by: WP7.SPR
   Contains:  _Q7I0LI7F7()                    (Params: none)
      Called by: WP7.SPR
          Calls: SFPRINT                      (procedure in U.PRG)
   Contains:  _Q7I0LI7OJ()                    (Params: none)
      Called by: WP7.SPR


SECHECK.PRG -- Last updated:  09/03/92 at 12:01

   Contains: SYSWARN()                        (Params: MSYSWAR)
      Called by: SECHECK.PRG
   Contains: GETSYSCODE()                     (Params:  MWARN,  MPRIVATE,  MDIS,
                                              MPASS)
      Called by: SECHECK.PRG
          Calls: FILEMISSING                  (procedure in SECHECK.PRG)
   Contains: FILEMISSING                      (Params: none)
      Called by: GETSYSCODE()                 (function in SECHECK.PRG)
          Calls: ERRMSG.PRG
   Contains: SYSCLAIM()                       (Params: M.TXT)
```

```
      Called by: SECHECK.PRG


W12A.SPR -- Last updated:  07/06/92 at 10:46

   Contains: _Q740N3U9Y()                    (Params: none)
      Called by: W12A.SPR
   Contains: _Q740N3UHF()                    (Params: none)
      Called by: W12A.SPR
          Calls: W12A.SPR


W12B.SPR -- Last updated:  07/06/92 at 10:46

   Contains: _Q740N3XWA()                    (Params: none)
      Called by: W12B.SPR
   Contains: _Q740N3Y5O()                    (Params: none)
      Called by: W12B.SPR
          Calls: W12B.SPR


W12C.SPR -- Last updated:  07/06/92 at 10:47

   Contains: _Q740N41W1()                    (Params: none)
      Called by: W12C.SPR
          Calls: W12C.SPR
   Contains: _Q740N423H()                    (Params: none)
      Called by: W12C.SPR


W4.SPR -- Last updated:  07/28/92 at 16:30

   Contains: _Q7Q0ZD5UR()                    (Params: none)
      Called by: W4.SPR
   Contains: _Q7Q0ZD65J()                    (Params: none)
      Called by: W4.SPR
          Calls: M_PATLIST                   (procedure in DEMO.PRG)


BACKUP.SPR -- Last updated:  10/09/92 at 13:00

   Contains: NEWDRIVEPOP                     (Params: none)
      Called by: BACKUP.SPR
      Called by: RESTORE.SPR
   Contains: NEWPATHPOP                      (Params: none)
      Called by: BACKUP.SPR
      Called by: RESTORE.SPR
      Called by: _Q9R0RVE7X()                (function in BACKUP.SPR)
      Called by: _Q9R0RVEMQ()                (function in BACKUP.SPR)
      Called by: _Q9R0RVF7K()                (function in BACKUP.SPR)
      Called by: _Q9R0UD7U5()                (function in RESTORE.SPR)
      Called by: _Q9R0UD888()                (function in RESTORE.SPR)
      Called by: _Q9R0UD8P2()                (function in RESTORE.SPR)
   Contains: NEWFILEPOP                      (Params: none)
      Called by: BACKUP.SPR
      Called by: RESTORE.SPR
      Called by: _Q9R0RVE7X()                (function in BACKUP.SPR)
      Called by: _Q9R0RVEMQ()                (function in BACKUP.SPR)
      Called by: _Q9R0RVF7K()                (function in BACKUP.SPR)
      Called by: _Q9R0UD7U5()                (function in RESTORE.SPR)
      Called by: _Q9R0UD888()                (function in RESTORE.SPR)
```

```
      Called by: _Q9ROUD8P2()              (function in RESTORE.SPR)
   Contains: PATHSTRING                    (Params: none)
      Called by: _Q9RORVEMQ()              (function in BACKUP.SPR)
      Called by: PKZIP                     (procedure in BACKUP.SPR)
      Called by: _Q9ROUD888()              (function in RESTORE.SPR)
      Called by: PKUNZIP                   (procedure in RESTORE.SPR)
   Contains: PKZIP                         (Params: none)
      Called by: _Q9RORVEV4()              (function in BACKUP.SPR)
          Calls: PATHSTRING                (procedure in BACKUP.SPR)
   Contains: _Q9RORVDY4()                  (Params: none)
      Called by: BACKUP.SPR
   Contains: _Q9RORVE7X()                  (Params: none)
      Called by: BACKUP.SPR
          Calls: YESNO.PRG
          Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
          Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
   Contains: _Q9RORVEMQ()                  (Params: none)
      Called by: BACKUP.SPR
          Calls: PATHSTRING                (procedure in BACKUP.SPR)
          Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
          Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
   Contains: _Q9RORVEV4()                  (Params: none)
      Called by: BACKUP.SPR
          Calls: YESNO.PRG
          Calls: PKZIP                     (procedure in BACKUP.SPR)
   Contains: _Q9RORVF7K()                  (Params: none)
      Called by: BACKUP.SPR
          Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
          Calls: NEWFILEPOP                (procedure in BACKUP.SPR)


RESTORE.SPR -- Last updated:  10/09/92 at 14:10

   Contains: NEWDRIVEPOP                   (Params: none)
      Called by: BACKUP.SPR
      Called by: RESTORE.SPR
   Contains: NEWPATHPOP                    (Params: none)
      Called by: BACKUP.SPR
      Called by: RESTORE.SPR
      Called by: _Q9RORVE7X()              (function in BACKUP.SPR)
      Called by: _Q9RORVEMQ()              (function in BACKUP.SPR)
      Called by: _Q9RORVF7K()              (function in BACKUP.SPR)
      Called by: _Q9ROUD7U5()              (function in RESTORE.SPR)
      Called by: _Q9ROUD888()              (function in RESTORE.SPR)
      Called by: _Q9ROUD8P2()              (function in RESTORE.SPR)
   Contains: NEWFILEPOP                    (Params: none)
      Called by: BACKUP.SPR
      Called by: RESTORE.SPR
      Called by: _Q9RORVE7X()              (function in BACKUP.SPR)
      Called by: _Q9RORVEMQ()              (function in BACKUP.SPR)
      Called by: _Q9RORVF7K()              (function in BACKUP.SPR)
      Called by: _Q9ROUD7U5()              (function in RESTORE.SPR)
      Called by: _Q9ROUD888()              (function in RESTORE.SPR)
      Called by: _Q9ROUD8P2()              (function in RESTORE.SPR)
   Contains: PATHSTRING                    (Params: none)
      Called by: _Q9RORVEMQ()              (function in BACKUP.SPR)
      Called by: PKZIP                     (procedure in BACKUP.SPR)
      Called by: _Q9ROUD888()              (function in RESTORE.SPR)
      Called by: PKUNZIP                   (procedure in RESTORE.SPR)
   Contains: PKUNZIP                       (Params: none)
      Called by: _Q9ROUD8G7()              (function in RESTORE.SPR)
          Calls: SETPATH                   (procedure in U.PRG)
```

```
        Calls: PATHSTRING                    (procedure in BACKUP.SPR)
   Contains:  _Q9ROUD7NA()                   (Params: none)
      Called by: RESTORE.SPR
   Contains:  _Q9ROUD7U5()                   (Params: none)
      Called by: RESTORE.SPR
        Calls: YESNO.PRG
        Calls: NEWPATHPOP                    (procedure in BACKUP.SPR)
        Calls: NEWFILEPOP                    (procedure in BACKUP.SPR)
   Contains:  _Q9ROUD888()                   (Params: none)
      Called by: RESTORE.SPR
        Calls: PATHSTRING                    (procedure in BACKUP.SPR)
        Calls: NEWPATHPOP                    (procedure in BACKUP.SPR)
        Calls: NEWFILEPOP                    (procedure in BACKUP.SPR)
   Contains:  _Q9ROUD8G7()                   (Params: none)
      Called by: RESTORE.SPR
        Calls: PKUNZIP                       (procedure in RESTORE.SPR)
   Contains:  _Q9ROUD8P2()                   (Params: none)
      Called by: RESTORE.SPR
        Calls: NEWPATHPOP                    (procedure in BACKUP.SPR)
        Calls: NEWFILEPOP                    (procedure in BACKUP.SPR)
   Contains:  _Q9ROUD8ZO()                   (Params: none)
      Called by: RESTORE.SPR


W7A.SPR -- Last updated:  08/03/92 at 16:06

   Contains:  _Q7WOYIII2()                   (Params: none)
      Called by: W7A.SPR


QUESTION.PRG -- Last updated:  08/04/92 at 14:48

   Contains: WRITE                           (Params: none)
      Called by: PRINTQUES()                 (function in QUESTION.PRG)
        Calls: OPENFILE                      (procedure in QUESTION.PRG)
   Contains: OPENFILE                        (Params: none)
      Called by: WRITE                       (procedure in QUESTION.PRG)
   Contains: PRINTQUES()                     (Params: MPRINTFILE)
      Called by: QUESTION.PRG
        Calls: WRITE                         (procedure in QUESTION.PRG)


Q2.PRG -- Last updated:  08/04/92 at 16:33

   Contains:  _Q7IOLH8OT()                   (Params: none)
      Called by: W2.SPR
      Called by: Q2.PRG
   Contains:  _Q7IOLH8WD()                   (Params: none)
      Called by: W2.SPR
      Called by: Q2.PRG
        Calls:  _PRINTENC                    (procedure in U.PRG)
   Contains:  _Q7IOLH94Q()                   (Params: none)
      Called by: W2.SPR
      Called by: Q2.PRG


COLORSET.SPR -- Last updated:  10/07/92 at 11:52

   Contains:  _Q9POPFX2H()                   (Params: none)
      Called by: COLORSET.SPR
```

```
CAMD.MPR -- Last updated:  10/07/92 at 11:51

   Contains: MHIT                          (Params: PROG)
   Contains: MEXIT                         (Params: none)
   Contains: _Q9POPFLOY                    (Params: none)
      Called by: CAMD.MPR
          Calls: PUSHHELP                  (procedure in U.PRG)
          Calls: PWDEDT.PRG
          Calls: POPHELP                   (procedure in U.PRG)
   Contains: _Q9POPFLR8                    (Params: none)
      Called by: CAMD.MPR
          Calls: CAMDTR.MPR
          Calls: _MHIT                     (procedure in U.PRG)
          Calls: CAMD.MPR

CAMDTR.MPR -- Last updated:  09/28/92 at 14:22

   Contains: MHIT                          (Params: PROG)
   Contains: MEXIT                         (Params: none)
```

Section VIII. Program Source Code

```
 1      *
 2      *
 3      *        10/09/92              BACKUP.SPR                13:00:12
 4      *
 5      *
 6      *
 7      *        Author's Name
 8      *
 9      *        Copyright (c) 1992 Company Name
10      *        Address
11      *        City,     Zip
12      *
13      *        Description:
14      *        This program was automatically generated by GENSCRN.
15      *
16      *
17
18
19      *
20      *
21      *                 BACKUP Setup Code - SECTION 1
22      *
23      *
24      *
25
26      #REGION 1
27      PARAMETERS MESSAGE, wildcard, filename
28      PRIVATE mfanme, mpath, olddrive, predrive, oldpath, maction, mdrive,
=> mfiles
29      DO CASE
30      CASE PARAMETERS() = 0
31          m.message = ""
32          m.wildcard = "*.ZIP"
33          mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
34      CASE PARAMETERS() = 1 OR EMPTY(m.wildcard)
35          m.wildcard = "*.ZIP"
36          mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
37      CASE PARAMETERS() = 2 OR EMPTY(m.filename)
38          mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
39      OTHERWISE
40          mfname = filename
41      ENDCASE
42
43
44      #REGION 0
45      REGIONAL m.currarea, m.talkstat, m.compstat
46
47      IF SET("TALK") = "ON"
48          SET TALK OFF
49          m.talkstat = "ON"
50      ELSE
51          m.talkstat = "OFF"
52      ENDIF
53      m.compstat = SET("COMPATIBLE")
54      SET COMPATIBLE FOXPLUS
55
56      *
57      *
58      *                 Window definitions
59      *
60      *
61      *
62
63      IF NOT WEXIST("w_backup")
64          DEFINE WINDOW w_backup ;
65              FROM INT((SROW()-18)/2),INT((SCOL()-61)/2) ;
66              TO INT((SROW()-18)/2)+17,INT((SCOL()-61)/2)+60 ;
67              NOFLOAT ;
68              NOCLOSE ;
69              SHADOW ;
70              DOUBLE ;
71              COLOR SCHEME 5
72      ENDIF
73
74
75      *
76      *
77      *                 BACKUP Setup Code - SECTION 2
78      *
79      *
80      *
81
82      #REGION 1
83      m.olddrive = SET("DEFAULT")
84      m.prevdrive = 1
85      m.oldpath = CURDIR()
86      maction = 1
87
88      DECLARE drivearray[1,1]
89      mdrive = 1
90      DO newdrivepop
91
92      DECLARE patharray[1,1]
93      mpath = 1
94      DO newpathpop
95
96      DECLARE filearray[1,1]
97      mfiles = 1
98      DO newfilepop
99
100
101     =closefiles()
102
103
104
105
106
107     *
108     *
109     *                 BACKUP Screen Layout
110     *
111     *
112     *
113
114     #REGION 1
115     IF WVISIBLE("w_backup")
116         ACTIVATE WINDOW w_backup SAME
117     ELSE
118         ACTIVATE WINDOW w_backup NOSHOW
119     ENDIF
120     @ 3,29 SAY "To drive: "
121     @ 2,40 GET mdrive ;
122         PICTURE "@^" ;
123         FROM drivearray ;
124         SIZE 3,18 ;
125         DEFAULT 1 ;
126         WHEN _q9r0rvdy4() ;
127         VALID _q9r0rve7x() ;
128         COLOR SCHEME 5, 6
129     @ 6,29 SAY "Directory:"
130     @ 0,1 SAY MESSAGE ;
131         SIZE 1,40
```

```
132     @ 5,40 GET mpath ;
133         PICTURE "@" ;
134         FROM patharray ;
135         SIZE 3,18 ;
136         DEFAULT 1 ;
137         VALID _q9r0rvemq() ;
138         COLOR SCHEME 5, 6
139     @ 9,45 GET maction ;
140         PICTURE "@*VT \i\<Backup;\<Cancel" ;
141         SIZE 1,8,1 ;
142         DEFAULT 1 ;
143         VALID _q9r0rvev4()
144     @ 2,1 GET mfile ;
145         PICTURE "@&N" ;
146         FROM filearray ;
147         SIZE 11,26 ;
148         DEFAULT 1 ;
149         VALID _q9r0rvf7k() ;
150         COLOR SCHEME 6
151     @ 14,1 SAY "Backup file name: "
152     @ 14,19 GET mfname ;
153         SIZE 1,39 ;
154         DEFAULT " "
155
156     IF NOT WVISIBLE("w_backup")
157         ACTIVATE WINDOW w_backup
158     ENDIF
159
160     READ CYCLE MODAL
161
162     RELEASE WINDOW w_backup
163
164     #REGION 0
165     IF m.talkstat = "ON"
166         SET TALK ON
167     ENDIF
168     IF m.compstat = "ON"
169         SET COMPATIBLE ON
170     ENDIF
171
172
173     *
174     *                    BACKUP Cleanup Code
175     *
176     *
177     *
178     *
179
180     #REGION 1
181     SET DEFAULT TO (m.olddrive + m.oldpath)
182     RETURN
183
184     ********************************************************
185
186     *********************
187     PROCEDURE newdrivepop
188     *********************
189     * Create a popup with each of the legal drive names
190     * The popup will be based on an array of the drive names
191     PRIVATE olddefault, olderror, drivename, ERROR
192
193     * We will change drives, so remember where we started
194     m.olddefault = SET( "DEFAULT" )
195
196     * To test for legal drives this program SET DEFAULT TO each drive
197     * If no error occurs the drive name will be added to an array
198
199     * Create the error trap
200     m.olderror = ON( "ERROR" )
201     m.error = .F.
202     ON ERROR m.error = .T.
203
204     * Create the array of legal drive names
205     DECLARE drivearray[ 1 ]
206     drivearray[ 1 ] = ""
207
208     * Loop from A to Z
209     m.drivename = "A"
210     FOR i = 1 TO 26
211
212         * Try to switch to the drive
213         SET DEFAULT TO (m.drivename)
214
215         * If it worked
216         IF NOT m.error
217
218             * Add the name to the last element in the array
219             drivearray[ ALEN( DriveArray ) ] = m.drivename + ":"
220
221             * Add another element at the end of the array
222             DECLARE drivearray[ ALEN( DriveArray ) + 1 ]
223
224         ENDIF
225
226         * Change to the next letter in the alphabet
227         m.drivename = CHR( ASC( m.drivename ) + 1 )
228
229         * Reset the error trap
230         m.error = .F.
231     NEXT
232
233     * If the first element is empty, no drives were found
234     IF EMPTY( drivearray[ 1 ] )
235         SHOW GET mdrive disabled
236     ELSE
237         * Drives were found so cut off the last empty element
238         * added to the array in the loop
239         DECLARE drivearray[ ALEN( DriveArray ) - 1 ]
240     ENDIF
241
242     * Reset the error trap and return to home
243     ON ERROR &olderror
244     SET DEFAULT TO (m.olddefault)
245
246     * Initialize the popup variable to the element in the
247     * drive array which contains the current drive
248     mdrive = ASCAN( drivearray, m.olddefault )
249     RETURN
250
251     ********************
252     PROCEDURE newpathpop
253     ********************
254     * Create a popup with one bar for each subdirectory
255     * in the current path
256     PRIVATE dirstring, dircount
257
258     * Base the popup on an array with one element for
259     * each subdirectory in the current path
260     *
261     * Start the array with the current drive
262     DECLARE patharray[ 1 ]
263     patharray[ 1 ] = SET( "DEFAULT" )
```

```
264
265     * Get the current path string
266     m.dirstring = CURDIR()
267
268     * If we are not in the root directory
269     IF LEN( m.dirstring ) > 1
270
271        * Start parsing the path string for each directory name
272        m.dircount = 1
273
274        * Continue as long as there is a pair of slashes
275        * Surrounding the next part of the string
276        DO WHILE AT( "\", m.dirstring, m.dircount ) <> 0 AND ;
277              AT( "\", m.dirstring, m.dircount + 1 ) <> 0
278
279           * Add another element at the end of the array
280           DECLARE    patharray[ m.DirCount + 1 ]
281
282           * Cut out the next set of characters between the slashes
283           m.firstchar = AT( "\", m.dirstring, m.dircount ) + 1
284
285           m.pathlength = AT( "\", m.dirstring, m.dircount + 1 )  ;
286                - m.firstchar
287
288           * And add them to the next array element
289           patharray[ m.DirCount + 1 ] = SUBSTR( m.dirstring, ;
290                m.firstchar, m.pathlength )
291
292           * Look for the next directory name in the path string
293           m.dircount = m.dircount + 1
294
295        ENDDO
296     ENDIF
297
298     * Point the popup variable at the last element in the array
299     mpath = ALEN( patharray )
300     RETURN
301
302     ********************
303     PROCEDURE newfilepop
304     ********************
305
306     * Create a popup with all of the file names and its subdirection in t
=> he current directory
307     * This will be based on an array as well
308
309     * Create an array with all the files matching the current wild card
310     * ADIR() creates an array with 5 columns.
311
312     PRIVATE startsort
313     * Fill an array with directory names only
314     =ADIR( dirarray, "", "D")
315     SIZE = ALEN(dirarray,1)
316     IF dirarray[ 1, 1 ] = "."
317        =ADEL(dirarray,1)
318        SIZE = SIZE - 1
319        DECLARE dirarray[size,5]
320     ELSE
321        * We must be in the root directory "\"
322        * Add one more row to the directory array
323        SIZE = SIZE + 1
324        DECLARE dirarray[ size, 5 ]
325
326        * Push all the rows down by one
327        =AINS( dirarray, 1 )
328
329        * Fill in the first directory name in the array
330        * a bug in the popups makes it refuse to display a
331        * prompt of "\", use "\\" to make one \ appear
332        * even then the bar will automatically be non-selectable
333        * which in this case is fine
334        dirarray[ 1,1 ] = "\\"
335
336        * Start the sort after the root name
337     ENDIF
338
339     IF ALEN( dirarray, 1 ) > 2
340
341        * Sort the array starting at the starting row
342        =ASORT( dirarray, AELEMENT( dirarray, 2, 1 ) )
343     ENDIF
344     FOR i = 1 TO ALEN(dirarray,1)
345        dirarray[i,1] = "[" + dirarray[i,1] + "]"
346     ENDFOR
347
348     IF ADIR(filearray, m.wildcard) = 0
349        SIZE = ALEN(dirarray,1)
350        DECLARE filearray[size,5]
351        =ACOPY(dirarray, filearray)
352     ELSE
353        stop = ALEN(dirarray,1)
354        FOR i = 1 TO stop
355           SIZE = ALEN(filearray,1)
356           DECLARE filearray[size + 1,5]
357           =AINS(filearray,1)
358           filearray[1,1] = dirarray(ALEN[dirarray,1],1)
359           IF ALEN(dirarray,1) > 1
360              DECLARE dirarray[alen(dirarray,1)-1,5]
361           ENDIF
362        ENDFOR
363     ENDIF
364     mfile = 1
365     RETURN
366
367     ********************
368     FUNCTION pathstring
369     ********************
370     * Convert an array of subdirectories, such as PathArray,
371     * into a legal path string for use with SET DEFAULT
372     * Use the current value of the path popup as the
373     * ending point on the path
374     PRIVATE ppath
375
376     * Start with the drive name
377     m.ppath = patharray[ 1 ]
378
379     * If the path popup is pointing to a subdirectory
380     IF mpath > 1
381
382        * Add all the subdirectories to the path string
383        FOR i = 2 TO mpath
384           m.ppath = m.ppath + "\" + patharray[ i ]
385
386        NEXT
387     ENDIF
388
389     * End the path with one last slash for good luck
390     m.ppath = m.ppath + "\"
391
392     RETURN m.ppath
393
394     ********************
```

```
395     PROCEDURE pkzip
396     ******************
397
398     PRIVATE CURDIR, datadir, msel
399     CURDIR = FULLPATH(CURDIR())
400     IF !EMPTY(GETENV("CAMD"))
401        datadir = FULLPATH(GETENV("CAMD"))
402     ELSE
403        datadir = CURDIR()
404     ENDIF
405     desdir = pathstring() + mfname
406     SET DEFA TO (datadir)
407     msel = SELECT()
408     SELECT 0
409     USE backup AGAIN
410     COPY TO t0000000.txt TYPE SDF
411     IF !EMPTY(GETENV("CAMDUTIL"))
412        pkzipcom = "!" + GETENV("CAMDUTIL") + "\PKZIP &desdir @t0000000.tx
=> t"
413     ELSE
414        pkzipcom = "!PKZIP &desdir @t0000000.txt"
415     ENDIF
416     &pkzipcom
417     USE
418     DELETE FILE t0000000.txt
419     SET DEFA TO (CURDIR)
420     RETURN
421
422     *
423     *
424     *        _Q9R0RVDY4          mdrive WHEN
425     *
426     *        Function Origin:
427     *
428     *        From Screen:        BACKUP,     Record Number:    3
429     *        Variable:           mdrive
430     *        Called By:          WHEN Clause
431     *        Object Type:        Popup
432     *        Snippet Number:     1
433     *
434     *
435     *
436     FUNCTION _q9r0rvdy4      && mdrive WHEN
437     #REGION 1
438     m.prevdrive = mdrive
439
440     *
441     *
442     *        _Q9R0RVE7X          mdrive VALID
443     *
444     *        Function Origin:
445     *
446     *        From Screen:        BACKUP,     Record Number:    3
447     *        Variable:           mdrive
448     *        Called By:          VALID Clause
449     *        Object Type:        Popup
450     *        Snippet Number:     2
451     *
452     *
453     *
454     *Switch to the selected drive
455     FUNCTION _q9r0rve7x      && mdrive VALID
456     #REGION 1
457     PRIVATE newdrive,mready
458
459     *Convert the popup bar number into the matching drive name
460     m.newdrive = drivearray[mdrive]
461
462     IF UPPER(m.newdrive) $ "A:B:"
463        mready = yesno("Please insert disk into drive " + m.newdrive, "rea
=> dy","cancel")
464     ELSE
465        mready = .T.
466     ENDIF
467     IF mready
468        *Go there and reset all the other popups to match
469        SET DEFAULT TO (m.newdrive)
470        DO newpathpop
471        DO newfilepop
472     ELSE
473        mdrive =  m.prevdrive
474        m.newdrive = drivearray[mdrive]
475     ENDIF
476     SHOW GETS
477
478     *
479     *
480     *        _Q9R0RVEMQ          mpath VALID
481     *
482     *        Function Origin:
483     *
484     *        From Screen:        BACKUP,     Record Number:    6
485     *        Variable:           mpath
486     *        Called By:          VALID Clause
487     *        Object Type:        Popup
488     *        Snippet Number:     3
489     *
490     *
491     *
492     FUNCTION _q9r0rvemq      && mpath VALID
493     #REGION 1
494     m.newdefault = pathstring()
495     SET DEFAULT TO (m.newdefault)
496     DO newpathpop
497     DO newfilepop
498     SHOW GETS
499
500     *
501     *
502     *        _Q9R0RVEV4          maction VALID
503     *
504     *        Function Origin:
505     *
506     *        From Screen:        BACKUP,     Record Number:    7
507     *        Variable:           maction
508     *        Called By:          VALID Clause
509     *        Object Type:        Push Button
510     *        Snippet Number:     4
511     *
512     *
513     *
514     FUNCTION _q9r0rvev4      && maction VALID
515     #REGION 1
516     DO CASE
517     CASE maction = 1
518
519        pkzip = .T.
520        IF FILE(mfname)
521           mdel = yesno( mfname + " already exists, overwrite it? ", "Yes"
=> , "Cancel")
522           IF mdel
523              del &mfname
```

```
524        ELSE
525            pkzip = .F.
526        ENDIF
527     ENDIF
528     IF pkzip
529        DO pkzip
530     ENDIF
531  OTHERWISE
532  ENDCASE
533
534  *
535  *
536  *        _09R0RVF7K          mfile VALID
537  *
538  *        Function Origin:
539  *
540  *        From Screen:        BACKUP,     Record Number:    8
541  *        Variable:           mfile
542  *        Called By:          VALID Clause
543  *        Object Type:        List
544  *        Snippet Number:     5
545  *
546  *
547  *
548  FUNCTION _q9r0rvf7k      && mfile VALID
549  #REGION 1
550  mnewfile = filearray[mfile,1]
551  IF "[" $ mnewfile
552     mnewpath = SUBSTR(mnewfile,2,LEN(mnewfile)-2)
553     SET DEFA TO (mnewpath)
554     DO newpathpop
555     DO newfilepop
556     SHOW GETS
557  ELSE
558     mfname = mnewfile
559     SHOW GETS
560  ENDIF
562  *: EOF: BACKUP.AC1
```

```
1   *:*****************************************************************
2   *:
3   *: Procedure file: C:\CAMD\WORK\BIGCHARS.PRG
4   *:
5   *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
6   *:          Author: ROY DOBBINS and HCA LY
7   *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
8   *:  Last modified: 10/08/92      9:52
9   *:
10  *:          Set by: SECHECK.PRG
11  *:
12  *:       Documented 10/14/92 at 10:52                FoxDoc  version 2.10
13  *:*****************************************************************
14      *****************************************************************
15      *
16      *    FUNCTION: BIGCHARS
17      *    PURPOSE: PRINT HEADER
18      *    RETURN: None.
19      *    PARAMETERS:
20      *            startx          x-coordinate to start   (sugg: 6)
21      *            starty          y-coordianate to start  (sugg: 17)
22      *            instring        title to be blown up
23      *            o_t                     waiting time  (sugg: 6)
24      *    REFERENCE:      C:\FOXPROZ\GOODIES\DEMO\PRGS\BIGCHARS.PRG
25      *    SIDE EFFECTS: Not known.
26      *    CREATED BY:  Anh Le             18 JUN 92
27      *    MODIFIED:       AL              23 JUN 92
28      *
29      *****************************************************************
30      PARAMETER startx,starty,instring,o_t
31
32      IF PARAMETERS() = 3
33          o_t = 6
34      ENDIF
35
36      x_coor = startx
37      y_coor = starty
38      instring = UPPER(instring)
39      len_str = LEN(instring)
40
41      DO CASE
42      CASE (len_str = 5)
43          starty = starty
44      CASE (len_str = 3)
45          starty = starty + 9
46      CASE (len_str < 5)
47          starty = starty + 3
48      CASE (len_str = 6)
49          starty = starty - 3
50      CASE (len_str = 7)
51          starty = starty - 6
52      OTHERWISE
53          starty = starty -starty+4
54      ENDCASE
55
56      SET BLINK OFF
57      CLEAR
58
59      IF startx > 20
60          RETURN
61      ENDIF
62
63      curenty = starty
64      curlet = 1
65
66      IF "MONO" $ UPPER(SYS(2006))
67          colorvar = "n/n"
68      ELSE
69          colorvar = "n/b"
70      ENDIF
71      CLEAR
72
73      DO WHILE curlet <= LEN(instring) AND curenty < 89
74          DO CASE
75          CASE SUBSTR(instring,curlet,1) = " "
76              DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
77                  FROM startx,curenty TO startx + 3,curenty + 5 ;
78                  NONE NOSHADOW
79              ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
80              curenty = curenty + 6
81          CASE SUBSTR(instring,curlet,1) = "t"
82              DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
83                  FROM startx,curenty TO startx + 5,curenty + 5 ;
84                  NONE NOSHADOW
85              ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
86              curenty = curenty + 6
87              @ 0,0 SAY "
88              @ 0,3 SAY "" COLOR (colorvar)
89
90              @ 1,0 SAY "
91              @ 1,3 SAY "" COLOR W+/N
92              @ 1,4 SAY "" COLOR (colorvar)
93
94              @ 2,1 SAY "  " COLOR W+/N
95
96              @ 3,0 SAY "
97              @ 3,3 SAY "" COLOR W+/N
98
99              @ 4,0 SAY "    " COLOR (colorvar)
100         CASE SUBSTR(instring,curlet,1) = "s"
101             DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
102                 FROM startx,curenty TO startx + 5,curenty + 5 ;
103                 NONE NOSHADOW
104             ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
105             curenty = curenty + 7
106             @ 1,0 SAY "
107             @ 1,2 SAY "" COLOR W+/N
108             @ 1,3 SAY "
109
110             @ 2,0 SAY "
111             @ 2,2 SAY "" COLOR W+/N
112             @ 2,5 SAY "" COLOR (colorvar)
113
114             @ 3,0 SAY "
115             @ 3,4 SAY "" COLOR W+/N
116
117             @ 4,0 SAY "    " COLOR (colorvar)
118         CASE SUBSTR(instring,curlet,1) = "e"
119             DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
120                 FROM startx,curenty TO startx + 5,curenty + 5 ;
121                 NONE NOSHADOW
122             ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
123             curenty = curenty + 7
124             @ 1,0 SAY "
125             @ 1,2 SAY "" COLOR W+/N
126             @ 1,4 SAY "
127
128             @ 2,0 SAY "    " COLOR W+/N
129
130             @ 3,0 SAY "
131             @ 3,2 SAY "" COLOR W+/N
132             @ 3,5 SAY "" COLOR (colorvar)
```

```
133
134     @ 4,0 SAY "▄▄▄▄" COLOR (colorvar)
135    CASE SUBSTR(instring,curlet,1) = "r"
136     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
137         FROM startx,curenty TO startx + 5,curenty + 5 ;
138         NONE NOSHADOW
139     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
140     curenty = curenty + 6
141     @ 1,0 SAY "▄▄"
142     @ 1,3 SAY "▄▄▄" COLOR W+/N
143     @ 1,4 SAY "▄" COLOR (colorvar)
144
145     @ 2,0 SAY "▄▄ " COLOR W+/N
146     @ 2,3 SAY "▄▄" COLOR (colorvar)
147
148     @ 3,0 SAY "▄▄ " COLOR W+/N
149
150     @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
151    CASE SUBSTR(instring,curlet,1) = "o"
152     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
153         FROM startx,curenty TO startx + 5,curenty + 5 ;
154         NONE NOSHADOW
155     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
156     curenty = curenty + 7
157     @ 1,0 SAY "▄▄"
158     @ 1,2 SAY "▄▄" COLOR W+/N
159     @ 1,4 SAY "▄"
160
161     @ 2,0 SAY "▄▄ ▄▄ " COLOR W+/N
162
163     @ 3,0 SAY "▄▄"
164     @ 3,2 SAY "▄▄ " COLOR W+/N
165
166     @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
167    CASE SUBSTR(instring,curlet,1) = "s"
168     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
169         FROM startx,curenty TO startx + 5,curenty + 6 ;
170         NONE NOSHADOW
171     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
172     curenty = curenty + 8
173     @ 1,0 SAY "▄▄"
174     @ 1,2 SAY "▄▄" COLOR W+/N
175     @ 1,4 SAY "▄"
176
177     @ 2,0 SAY "▄▄"
178     @ 2,2 SAY "▄▄ " COLOR W+/N
179
180     @ 3,0 SAY "▄▄"
181     @ 3,2 SAY "▄▄▄" COLOR W+/N
182
183     @ 4,0 SAY " ▄▄ ▄▄ COLOR (colorvar)
184    CASE SUBSTR(instring,curlet,1) = "i"
185     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
186         FROM startx,curenty TO startx + 5,curenty + 3 ;
187         NONE NOSHADOW
188     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
189     curenty = curenty + 4
190     @ 0,0 SAY "▄▄"
191     @ 0,1 SAY "▄▄" COLOR W+/N
192     @ 0,2 SAY "▄" COLOR (colorvar)
193
194     @ 1,0 SAY "▄▄"
195     @ 1,2 SAY "▄" COLOR (colorvar)
196
197     @ 2,0 SAY "▄▄ " COLOR W+/N
198
199     @ 3,0 SAY "▄▄ " COLOR W+/N
200
201     @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
202    CASE SUBSTR(instring,curlet,1) = "n"
203     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
204         FROM startx,curenty TO startx + 5,curenty + 5 ;
205         NONE NOSHADOW
206     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
207     curenty = curenty + 7
208     @ 1,0 SAY "▄▄"
209     @ 1,2 SAY "▄▄" COLOR W+/N
210     @ 1,4 SAY "▄"
211
212     @ 2,0 SAY "▄▄ ▄▄ " COLOR W+/N
213
214     @ 3,0 SAY "▄▄ ▄▄ " COLOR W+/N
215
216     @ 4,0 SAY " ▄▄ ▄▄ COLOR (colorvar)
217    CASE SUBSTR(instring,curlet,1) = "m"
218     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
219         FROM startx,curenty TO startx + 5,curenty + 8 ;
220         NONE NOSHADOW
221     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
222     curenty = curenty + 10
223     @ 1,0 SAY "▄▄"
224     @ 1,2 SAY "▄▄" COLOR W+/N
225     @ 1,4 SAY "▄"
226     @ 1,5 SAY "▄▄" COLOR W+/N
227     @ 1,7 SAY "▄"
228
229     @ 2,0 SAY "▄▄ ▄▄ ▄▄ " COLOR W+/N
230
231     @ 3,0 SAY "▄▄ ▄▄ ▄▄ " COLOR W+/N
232
233     @ 4,0 SAY " ▄▄ ▄▄ ▄▄ COLOR (colorvar)
234    CASE SUBSTR(instring,curlet,1) = "f"
235     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
236         FROM startx,curenty TO startx + 5,curenty + 6 ;
237         NONE NOSHADOW
238     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
239     curenty = curenty + 7
240     @ 0,0 SAY "▄▄"
241     @ 0,3 SAY "▄▄▄" COLOR W+/N
242     @ 0,5 SAY "▄" COLOR (colorvar)
243
244     @ 1,0 SAY "▄▄"
245     @ 1,3 SAY "▄▄" COLOR W+/N
246     @ 1,4 SAY "▄" COLOR (colorvar)
247
248     @ 2,1 SAY "▄▄ " COLOR W+/N
249
250     @ 3,1 SAY "▄▄ " COLOR W+/N
251
252     @ 4,0 SAY " ▄▄ COLOR (colorvar)
253    CASE SUBSTR(instring,curlet,1) = "l"
254     DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
255         FROM startx,curenty TO startx + 5,curenty + 3 ;
256         NONE NOSHADOW
257     ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
258     curenty = curenty + 4
259     @ 0,0 SAY "▄▄"
260     @ 0,2 SAY "▄" COLOR (colorvar)
261
262     @ 1,0 SAY "▄▄ " COLOR W+/N
263
264     @ 2,0 SAY "▄▄ " COLOR W+/N
```

```
265
266         @ 3,0 SAY "▄▄ " COLOR W+/N
267
268         @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
269     CASE SUBSTR(instring,curlet,1) = "k"
270         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
271             FROM startx,curenty TO startx + 5,curenty + 7 ;
272             NONE NOSHADOW
273         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
274         curenty = curenty + 8
275         @ 0,0 SAY "▄▄"
276         @ 0,2 SAY "▄" COLOR (colorvar)
277
278         @ 1,0 SAY "▄▄▄▄" COLOR W+/N
279         @ 1,5 SAY "▄" COLOR (colorvar)
280
281         @ 2,0 SAY "▄▄▄▄" COLOR W+/N
282         @ 2,4 SAY "▄▄" COLOR (colorvar)
283
284         @ 3,0 SAY "▄▄ " COLOR W+/N
285         @ 3,3 SAY "▄▄"
286
287         @ 4,0 SAY " ▄▄  ▄▄ COLOR (colorvar)
288     CASE SUBSTR(instring,curlet,1) = "u"
289         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
290             FROM startx,curenty TO startx + 5,curenty + 6 ;
291             NONE NOSHADOW
292         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
293         curenty = curenty + 7
294         @ 1,0 SAY "▄▄"
295         @ 1,2 SAY "▄" COLOR (colorvar)
296         @ 1,3 SAY "▄▄"
297         @ 1,5 SAY "▄" COLOR (colorvar)
298
299         @ 2,0 SAY "▄▄ ▄▄ " COLOR W+/N
300
301         @ 3,0 SAY "▄▄▄"
302         @ 3,2 SAY "▄" COLOR W+/N
303         @ 3,3 SAY "▄▄"
304         @ 3,4 SAY "▄ " COLOR W+/N
305
306         @ 4,0 SAY " ▄▄ ▄▄ COLOR (colorvar)
307     CASE SUBSTR(instring,curlet,1) = "b"
308         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
309             FROM startx,curenty TO startx + 5,curenty + 6 ;
310             NONE NOSHADOW
311         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
312         curenty = curenty + 7
313         @ 0,0 SAY "▄▄▄"
314         @ 0,2 SAY "▄" COLOR (colorvar)
315
316         @ 1,0 SAY "▄▄▄▄" COLOR W+/N
317         @ 1,4 SAY "▄"
318
319         @ 2,0 SAY "▄▄ " COLOR W+/N
320         @ 2,3 SAY "▄▄"
321         @ 2,5 SAY "▄" COLOR (colorvar)
322
323         @ 3,0 SAY "▄▄▄▄ " COLOR W+/N
324
325         @ 4,0 SAY " ▄▄▄▄ COLOR (colorvar)
326     CASE SUBSTR(instring,curlet,1) = "c"
327         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
328             FROM startx,curenty TO startx + 5,curenty + 5 ;
329             NONE NOSHADOW
330         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
```



```
331         curenty = curenty + 6
332         @ 1,0 SAY "▄▄"
333         @ 1,2 SAY "▄▄" COLOR W+/N
334         @ 1,4 SAY "▄" COLOR (colorvar)
335
336         @ 2,0 SAY "▄▄ " COLOR W+/N
337         @ 2,3 SAY " ▄▄ COLOR (colorvar)
338
339         @ 3,0 SAY "▄▄"
340         @ 3,2 SAY "▄▄" COLOR W+/N
341         @ 3,4 SAY "▄" COLOR (colorvar)
342
343         @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
344     CASE SUBSTR(instring,curlet,1) = "d"
345         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
346             FROM startx,curenty TO startx +^5,curenty + 5 ;
347             NONE NOSHADOW
348         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
349         curenty = curenty + 7
350         @ 0,0 SAY "   ▄▄"
351         @ 0,5 SAY "▄" COLOR (colorvar)
352
353         @ 1,0 SAY "▄▄"
354         @ 1,2 SAY "▄▄ " COLOR W+/N
355
356         @ 2,0 SAY "▄ " COLOR W+/N
357         @ 2,2 SAY "▄▄ COLOR (colorvar)
358         @ 2,3 SAY "▄▄ " COLOR W+/N
359
360         @ 3,0 SAY "▄▄▄▄"
361         @ 3,5 SAY "▄" COLOR (colorvar)
362
363         @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
364     CASE SUBSTR(instring,curlet,1) = "q"
365         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
366             FROM startx,curenty TO startx + 5,curenty + 5 ;
367             NONE NOSHADOW
368         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
369         curenty = curenty + 7
370         @ 1,0 SAY "▄▄"
371         @ 1,2 SAY "▄▄ COLOR W+/N
372         @ 1,3 SAY "▄▄"
373         @ 1,5 SAY "▄" COLOR (colorvar)
374
375         @ 2,0 SAY "▄ " COLOR W+/N
376         @ 2,2 SAY "▄▄ COLOR (colorvar)
377         @ 2,3 SAY "▄▄ " COLOR W+/N
378
379         @ 3,0 SAY "▄▄▄▄"
380         @ 3,5 SAY "▄" COLOR (colorvar)
381
382         @ 4,0 SAY " ▄▄ COLOR (colorvar)
383         @ 4,3 SAY "▄▄ " COLOR W+/N
384
385         @ 5,0 SAY "   ▄▄ COLOR (colorvar)
386
387     CASE SUBSTR(instring,curlet,1) = "g"
388         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
389             FROM startx,curenty TO startx + 5,curenty + 6 ;
390             NONE NOSHADOW
391         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
392         curenty = curenty + 7
393         @ 1,0 SAY "▄▄"
394         @ 1,2 SAY "▄▄ COLOR W+/N
395         @ 1,3 SAY "▄▄"
396         @ 1,5 SAY "▄" COLOR (colorvar)
```

```
397
398         @ 2,0 SAY "▄▄ " COLOR W+/N
399         @ 2,2 SAY "▄▄▄ COLOR (colorvar)
400         @ 2,3 SAY "▄▄ " COLOR W+/N
401
402         @ 3,0 SAY "▄▄▄▄"
403         @ 3,5 SAY "▄ " COLOR (colorvar)
404
405         @ 4,0 SAY " ▄"
406         @ 4,2 SAY "▄ " COLOR W+/N
407         @ 4,3 SAY "▄▄"
408         @ 4,4 SAY "▄ " COLOR W+/N
409
410         @ 5,0 SAY " ▄▄▄▄ COLOR (colorvar)
411      CASE SUBSTR(instring,curlet,1) = "h"
412         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
413             FROM startx,curenty TO startx + 5,curenty + 5 ;
414             NONE NOSHADOW
415         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
416         curenty = curenty + 7
417         @ 0,0 SAY "▄▄▄ COLOR W+/N
418         @ 0,2 SAY "▄ COLOR (colorvar)
419
420         @ 1,0 SAY "▄▄▄▄▄ COLOR W+/N
421         @ 1,4 SAY "▄"
422
423         @ 2,0 SAY "▄▄ ▄▄ " COLOR W+/N
424
425         @ 3,0 SAY "▄▄ ▄▄ " COLOR W+/N
426
427         @ 4,0 SAY " ▄▄ ▄▄ COLOR (colorvar)
428      CASE SUBSTR(instring,curlet,1) = "j"
429         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
430             FROM startx,curenty TO startx + 5,curenty + 5 ;
431             NONE NOSHADOW
432         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
433         curenty = curenty + 5
434         @ 0,0 SAY " ▄▄"
435         @ 0,2 SAY "▄▄ COLOR W+/N
436         @ 0,3 SAY "▄ COLOR (colorvar)
437
438         @ 1,0 SAY " ▄▄"
439         @ 1,3 SAY "▄ COLOR (colorvar)
440
441         @ 2,1 SAY "▄▄ " COLOR W+/N
442
443         @ 3,1 SAY "▄▄ " COLOR W+/N
444
445         @ 4,0 SAY "▄▄"
446         @ 4,1 SAY "▄▄ " COLOR W+/N
447         @ 4,3 SAY "▄▄ COLOR (colorvar)
448      CASE SUBSTR(instring,curlet,1) = "p"
449         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
450             FROM startx,curenty TO startx + 5,curenty + 6 ;
451             NONE NOSHADOW
452         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
453         curenty = curenty + 7
454         @ 1,0 SAY "▄▄▄▄"
455
456         @ 2,0 SAY "▄▄ " COLOR W+/N
457         @ 2,3 SAY "▄▄"
458         @ 2,5 SAY "▄ COLOR (colorvar)
459
460         @ 3,0 SAY "▄▄▄▄ " COLOR W+/N
461
462         @ 4,0 SAY "▄▄ " COLOR W+/N
463         @ 4,3 SAY "▄▄ COLOR (colorvar)
464
465         @ 5,0 SAY " ▄▄ COLOR (colorvar)
466      CASE SUBSTR(instring,curlet,1) = "w"
467         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
468             FROM startx,curenty TO startx + 5,curenty + 8 ;
469             NONE NOSHADOW
470         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
471         curenty = curenty + 9
472         @ 1,0 SAY "▄▄"
473         @ 1,2 SAY "▄ COLOR (colorvar)
474         @ 1,3 SAY "▄▄"
475         @ 1,4 SAY "▄ COLOR (colorvar)
476         @ 1,5 SAY "▄▄"
477         @ 1,7 SAY "▄ COLOR (colorvar)
478
479         @ 2,0 SAY "▄▄"
480         @ 2,2 SAY "▄▄▄▄ " COLOR W+/N
481
482         @ 3,1 SAY "▄▄ ▄▄ " COLOR W+/N
483
484         @ 4,0 SAY " ▄▄ ▄▄ COLOR (colorvar)
485      CASE SUBSTR(instring,curlet,1) = "x"
486         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
487             FROM startx,curenty TO startx + 5,curenty + 7 ;
488             NONE NOSHADOW
489         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
490         curenty = curenty + 8
491         @ 1,0 SAY "▄▄ ▄▄"
492         @ 1,5 SAY "▄▄ COLOR W+/N
493         @ 1,6 SAY "▄ COLOR (colorvar)
494
495         @ 2,0 SAY " ▄▄"
496         @ 2,4 SAY "▄ COLOR W+/N
497         @ 2,5 SAY "▄▄ COLOR (colorvar)
498
499         @ 3,0 SAY "▄▄ " COLOR W+/N
500         @ 3,3 SAY "▄▄ COLOR (colorvar)
501         @ 3,4 SAY "▄▄ COLOR W+/N
502         @ 3,6 SAY "▄ COLOR (colorvar)
503
504         @ 4,0 SAY " ▄▄ ▄▄ COLOR (colorvar)
505      CASE SUBSTR(instring,curlet,1) = "y"
506         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
507             FROM startx,curenty TO startx + 5,curenty + 6 ;
508             NONE NOSHADOW
509         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
510         curenty = curenty + 7
511         @ 1,0 SAY "▄▄"
512         @ 1,2 SAY "▄ COLOR (colorvar)
513         @ 1,3 SAY "▄▄"
514         @ 1,5 SAY "▄ COLOR (colorvar)
515
516         @ 2,0 SAY "▄▄ ▄▄ " COLOR W+/N
517
518         @ 3,0 SAY " ▄▄"
519         @ 3,4 SAY "▄▄ COLOR W+/N
520
521         @ 4,0 SAY " ▄▄"
522         @ 4,3 SAY "▄▄ COLOR W+/N
523
524         @ 5,0 SAY " ▄▄ COLOR (colorvar)
525      CASE SUBSTR(instring,curlet,1) = "v"
526         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
527             FROM startx,curenty TO startx + 5,curenty + 6 ;
528             NONE NOSHADOW
```

```
    ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
    curenty = curenty + 7
    @ 1,0 SAY "▄▄▄"
    @ 1,2 SAY "▄" COLOR (colorvar)
    @ 1,3 SAY "▄▄▄"
    @ 1,5 SAY "▄" COLOR (colorvar)

    @ 2,0 SAY "▄▄▄"
    @ 2,1 SAY "▄ ▄" COLOR W+/N

    @ 3,0 SAY " ▄▄▄"
    @ 3,3 SAY "▄▄▄" COLOR W+/N

    @ 4,0 SAY "   ▄▄▄" COLOR (colorvar)
CASE SUBSTR(instring,curlet,1) = "/"
    DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
        FROM startx,curenty TO startx + 5,curenty + 4 ;
        NONE NOSHADOW
    ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
    curenty = curenty + 6
    @ 0,0 SAY "   ▄"
    @ 0,4 SAY "▄" COLOR (colorvar)

    @ 1,0 SAY "  ▄"
    @ 1,3 SAY "▄▄" COLOR (colorvar)

    @ 2,0 SAY " ▄"
    @ 2,2 SAY "▄" COLOR (colorvar)

    @ 3,0 SAY "▄▄"
    @ 3,1 SAY "▄▄" COLOR (colorvar)

    @ 4,0 SAY " ▄▄" COLOR (colorvar)
CASE SUBSTR(instring,curlet,1) = "I"
    DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
        FROM startx,curenty TO startx + 5,curenty + 6 ;
        NONE NOSHADOW
    ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
    curenty = curenty + 7
    @ 1,0 SAY "▄▄▄"
    @ 1,1 SAY "▄▄▄▄▄" COLOR W+/N
    @ 1,5 SAY "▄" COLOR (colorvar)

    @ 2,0 SAY "  ▄▄"
    @ 2,3 SAY "▄▄" COLOR W+/N
    @ 2,5 SAY "▄▄▄" COLOR (colorvar)

    @ 3,0 SAY "▄▄▄▄" COLOR W+/N
    @ 3,4 SAY "▄"

    @ 4,0 SAY " ▄▄▄▄▄" COLOR (colorvar)
CASE SUBSTR(instring,curlet,1) = "|"
    startx = startx + 6
    curenty = starty
CASE SUBSTR(instring,curlet,1) = "A"
    DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
        FROM startx,curenty TO startx + 5,curenty + 7 ;
        NONE NOSHADOW
    ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
    curenty = curenty + 9
    @ 0,2 SAY "▄▄▄▄" COLOR W+/N
    @ 0,5 SAY "▄" COLOR (colorvar)

    @ 1,1 SAY "▄▄ ▄▄" COLOR W+/N
    @ 1,6 SAY "▄" COLOR (colorvar)

    @ 2,0 SAY "▄"
    @ 2,1 SAY "▄▄▄▄▄" COLOR W+/N

    @ 3,0 SAY "▄▄ " COLOR W+/N
    @ 3,3 SAY "▄▄▄" COLOR (colorvar)
    @ 3,5 SAY "▄▄ " COLOR W+/N

    @ 4,0 SAY " ▄▄   ▄▄▄" COLOR (colorvar)
CASE SUBSTR(instring,curlet,1) = "B"
    DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
        FROM startx,curenty TO startx + 5,curenty + 6 ;
        NONE NOSHADOW
    ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
    curenty = curenty + 8
    @ 0,0 SAY "▄▄▄▄▄" COLOR W+/N
    @ 0,4 SAY "▄"

    @ 1,0 SAY "▄▄▄" COLOR W+/N
    @ 1,3 SAY "▄▄"
    @ 1,5 SAY "▄▄" COLOR W+/N

    @ 2,0 SAY "▄▄ " COLOR W+/N
    @ 2,3 SAY "▄▄▄" COLOR (colorvar)
    @ 2,4 SAY "▄▄▄" COLOR W+/N
    @ 2,6 SAY "▄" COLOR (colorvar)

    @ 3,0 SAY "▄▄▄" COLOR W+/N
    @ 3,3 SAY "▄▄"
    @ 3,5 SAY "▄▄" COLOR W+/N

    @ 4,0 SAY " ▄▄▄▄▄" COLOR (colorvar)
CASE SUBSTR(instring,curlet,1) = "C"
    DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
        FROM startx,curenty TO startx + 5,curenty + 6 ;
        NONE NOSHADOW
    ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
    curenty = curenty + 9
    @ 0,0 SAY "  ▄▄"
    @ 0,3 SAY "▄▄▄" COLOR W+/N
    @ 0,6 SAY "▄"

    @ 1,0 SAY "▄▄ " COLOR W+/N
    @ 1,3 SAY "▄▄▄" COLOR (colorvar)
    @ 1,4 SAY "  ▄▄"
    @ 1,7 SAY "▄" COLOR (colorvar)

    @ 2,0 SAY "▄▄ " COLOR W+/N
    @ 2,3 SAY "   ▄ "

    @ 3,0 SAY " ▄▄▄▄"
    @ 3,6 SAY "▄▄ " COLOR W+/N

    @ 4,0 SAY "   ▄▄▄▄" COLOR (colorvar)
CASE SUBSTR(instring,curlet,1) = "D"
    DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
        FROM startx,curenty TO startx + 5,curenty + 6 ;
        NONE NOSHADOW
    ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
    curenty = curenty + 8
    @ 0,0 SAY "▄▄▄▄▄" COLOR W+/N
    @ 0,4 SAY "▄"

    @ 1,0 SAY "▄▄ " COLOR W+/N
    @ 1,3 SAY "▄▄"
    @ 1,6 SAY "▄" COLOR (colorvar)
```

```
661         @ 2,0 SAY "▀▄ " COLOR W+/N
662         @ 2,3 SAY "▄▄▄"
663         @ 2,6 SAY "▄▀" COLOR (colorvar)
664
665         @ 3,0 SAY "▄▄▄▄ " COLOR W+/N
666         @ 3,6 SAY "▄▄" COLOR (colorvar)
667
668         @ 4,0 SAY " ▄▄▄▄▄ COLOR (colorvar)
669     CASE SUBSTR(instring,curlet,1) = "E"
670         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
671             FROM startx,curenty TO startx + 5,curenty + 6 ;
672             NONE NOSHADOW
673         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
674         curenty = curenty + 7
675         @ 0,0 SAY "▄▄▄▄▄▄ COLOR W+/N
676         @ 0,5 SAY "▄" COLOR (colorvar)
677
678         @ 1,0 SAY "▄▄▄▄" COLOR W+/N
679         @ 1,3 SAY "▄▄"
680
681         @ 2,0 SAY "▄▄▄ " COLOR W+/N
682         @ 2,3 SAY "▄▄▄▄ COLOR (colorvar)
683
684         @ 3,0 SAY "▄▄▄▄" COLOR W+/N
685         @ 3,3 SAY "▄▄"
686
687         @ 4,0 SAY " ▄▄▄▄▄ COLOR (colorvar)
688     CASE SUBSTR(instring,curlet,1) = "F"
689         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
690             FROM startx,curenty TO startx + 5,curenty + 6 ;
691             NONE NOSHADOW
692         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
693         curenty = curenty + 7
694         @ 0,0 SAY "▄▄▄▄▄▄ COLOR W+/N
695         @ 0,5 SAY "▄" COLOR (colorvar)
696
697         @ 1,0 SAY "▄▄▄▄" COLOR W+/N
698         @ 1,3 SAY "▄"
699
700         @ 2,0 SAY "▄▄▄ " COLOR W+/N
701         @ 2,3 SAY "▄▄▄ COLOR (colorvar)
702
703         @ 3,0 SAY "▄▄▄ " COLOR W+/N
704
705         @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
706     CASE SUBSTR(instring,curlet,1) = "G"
707         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
708             FROM startx,curenty TO startx + 5,curenty + 8 ;
709             NONE NOSHADOW
710         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
711         curenty = curenty + 9
712         @ 0,0 SAY " ▄▄"
713         @ 0,3 SAY "▄▄▄▄" COLOR W+/N
714         @ 0,6 SAY "▄"
715
716         @ 1,0 SAY "▄▄▄ " COLOR W+/N
717         @ 1,3 SAY "▄▄▄ COLOR (colorvar)
718         @ 1,4 SAY " ▄▄"
719         @ 1,7 SAY "▄" COLOR (colorvar)
720
721         @ 2,0 SAY "▄▄▄ " COLOR W+/N
722         @ 2,3 SAY "▄▄▄▄"
723         @ 2,7 SAY "▄" COLOR (colorvar)
724
725         @ 3,0 SAY " ▄▄▄▄"
726         @ 3,6 SAY "▄▄ " COLOR W+/N
727
728         @ 4,0 SAY "  ▄▄▄▄ COLOR (colorvar)
729     CASE SUBSTR(instring,curlet,1) = "H"
730         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
731             FROM startx,curenty TO startx + 5,curenty + 7 ;
732             NONE NOSHADOW
733         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
734         curenty = curenty + 8
735         @ 0,0 SAY "▄▄▄"
736         @ 0,2 SAY "▄▄" COLOR (colorvar)
737         @ 0,4 SAY "▄▄▄"
738         @ 0,6 SAY "▄ " COLOR (colorvar)
739
740         @ 1,0 SAY "▄▄▄▄" COLOR W+/N
741         @ 1,3 SAY "▄▄▄"
742         @ 1,6 SAY "▄" COLOR (colorvar)
743
744         @ 2,0 SAY "▄▄▄ " COLOR W+/N
745         @ 2,3 SAY "▄▄▄ COLOR (colorvar)
746         @ 2,4 SAY "▄▄▄ " COLOR W+/N
747
748         @ 3,0 SAY "▄▄▄ " COLOR W+/N
749         @ 3,3 SAY "▄"
750         @ 3,4 SAY "▄▄▄ " COLOR W+/N
751
752         @ 4,0 SAY " ▄▄  ▄▄▄ COLOR (colorvar)
753     CASE SUBSTR(instring,curlet,1) = "I"
754         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
755             FROM startx,curenty TO startx + 5,curenty + 3 ;
756             NONE NOSHADOW
757         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
758         curenty = curenty + 4
759         @ 0,0 SAY "▄▄▄"
760         @ 0,2 SAY "▄" COLOR (colorvar)
761
762         @ 1,0 SAY "▄▄▄ " COLOR W+/N
763
764         @ 2,0 SAY "▄▄▄ " COLOR W+/N
765
766         @ 3,0 SAY "▄▄▄ " COLOR W+/N
767
768         @ 4,0 SAY " ▄▄▄ COLOR (colorvar)
769     CASE SUBSTR(instring,curlet,1) = "J"
770         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
771             FROM startx,curenty TO startx + 5,curenty + 6 ;
772             NONE NOSHADOW
773         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
774         curenty = curenty + 7
775         @ 0,0 SAY "   ▄▄▄"
776         @ 0,5 SAY "▄" COLOR (colorvar)
777
778         @ 1,3 SAY "▄▄▄ " COLOR W+/N
779
780         @ 2,3 SAY "▄▄▄ " COLOR W+/N
781
782         @ 3,0 SAY "▄▄▄▄"
783         @ 3,4 SAY "▄▄ " COLOR W+/N
784
785         @ 4,0 SAY "  ▄▄▄ COLOR (colorvar)
786     CASE SUBSTR(instring,curlet,1) = "K"
787         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
788             FROM startx,curenty TO startx + 5,curenty + 7 ;
789             NONE NOSHADOW
790         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
791         curenty = curenty + 9
792         @ 0,0 SAY "▄▄▄"
```

```
793         @ 0,2 SAY "▄" COLOR (colorvar)
794         @ 0,3 SAY "▄"
795         @ 0,5 SAY "▄▄▄" COLOR W+/N
796         @ 0,6 SAY "▄" COLOR (colorvar)
797
798         @ 1,0 SAY "▄▄▄▄ " COLOR W+/N
799         @ 1,5 SAY "▄▄▄" COLOR (colorvar)
800
801         @ 2,0 SAY "▄▄▄▄▄" COLOR W+/N
802         @ 2,4 SAY "▄"
803
804         @ 3,0 SAY "▄▄ " COLOR W+/N
805         @ 3,3 SAY "▄▄"
806
807         @ 4,0 SAY " ▄▄   ▄▄▄" COLOR (colorvar)
808     CASE SUBSTR(instring,curlet,1) = "L"
809         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
810             FROM startx,curenty TO startx + 5,curenty + 6 ;
811             NONE NOSHADOW
812         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
813         curenty = curenty + 7
814         @ 0,0 SAY "▄▄"
815         @ 0,2 SAY "▄" COLOR (colorvar)
816
817         @ 1,0 SAY "▄▄ " COLOR W+/N
818
819         @ 2,0 SAY "▄▄ " COLOR W+/N
820
821         @ 3,0 SAY "▄▄▄" COLOR W+/N
822         @ 3,3 SAY "▄▄"
823
824         @ 4,0 SAY " ▄▄▄▄▄" COLOR (colorvar)
825     CASE SUBSTR(instring,curlet,1) = "M"
826         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
827             FROM startx,curenty TO startx + 5,curenty + 10 ;
828             NONE NOSHADOW
829         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
830         curenty = curenty + 11
831         @ 0,0 SAY "▄▄▄   ▄▄▄"
832         @ 0,9 SAY "▄" COLOR (colorvar)
833
834         @ 1,0 SAY "▄▄▄▄" COLOR W+/N
835         @ 1,4 SAY "▄▄▄▄"
836         @ 1,9 SAY "▄" COLOR (colorvar)
837
838
839         @ 2,0 SAY "▄▄ ▄▄ ▄▄ " COLOR W+/N
840
841         @ 3,0 SAY "▄▄ " COLOR W+/N
842         @ 3,3 SAY "▄▄"
843         @ 3,5 SAY "▄▄" COLOR (colorvar)
844         @ 3,7 SAY "▄▄ " COLOR W+/N
845
846         @ 4,0 SAY " ▄▄  ▄  ▄▄▄" COLOR (colorvar)
847     CASE SUBSTR(instring,curlet,1) = "N"
848         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
849             FROM startx,curenty TO startx + 5,curenty + 8 ;
850             NONE NOSHADOW
851         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
852         curenty = curenty + 9
853         @ 0,0 SAY "▄▄▄   ▄▄"
854         @ 0,7  SAY "▄" COLOR (colorvar)
855
856         @ 1,0 SAY "▄▄▄▄" COLOR W+/N
857         @ 1,4 SAY "▄▄▄▄"
858         @ 1,7 SAY "▄▄" COLOR (colorvar)
```

```
859
860         @ 2,0 SAY "▄▄ " COLOR W+/N
861         @ 2,3 SAY "▄▄▄"
862         @ 2,4 SAY "▄▄▄ " COLOR W+/N
863
864         @ 3,0 SAY "▄▄ " COLOR W+/N
865         @ 3,3 SAY " ▄▄"
866         @ 3,7 SAY " " COLOR W+/N
867
868         @ 4,0 SAY " ▄▄   ▄▄▄" COLOR (colorvar)
869
870     CASE SUBSTR(instring,curlet,1) = "O"
871         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
872             FROM startx,curenty TO startx + 7,curenty + 7 ;
87'             NONE NOSHADOW
874         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
875         curenty = curenty + 9
876         @ 0,0 SAY " ▄▄"
877         @ 0,3 SAY "▄▄" COLOR W+/N
878         @ 0,4 SAY "▄▄"
879
880         @ 1,0 SAY "▄▄ " COLOR W+/N
881         @ 1,3 SAY "▄▄" COLOR (colorvar)
882         @ 1,5 SAY "▄▄"
883         @ 1,7 SAY "▄" COLOR (colorvar)
884
885         @ 2,0 SAY "▄▄ " COLOR W+/N
886         @ 2,3 SAY " ▄▄"
887         @ 2,7 SAY "▄" COLOR (colorvar)
888
889         @ 3,0 SAY " ▄▄▄"
890         @ 3,5 SAY "▄▄ " COLOR W+/N
891         @ 3,7 SAY "▄▄" COLOR (colorvar)
892
893         @ 4,0 SAY "   ▄▄▄" COLOR (colorvar)
894     CASE SUBSTR(instring,curlet,1) = "P"
895         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
896             FROM startx,curenty TO startx + 5,curenty + 6 ;
897             NONE NOSHADOW
898         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
899         curenty = curenty + 8
900         @ 0,0 SAY "▄▄▄" COLOR W+/N
901         @ 0,3 SAY "▄▄"
902
903         @ 1,0 SAY "▄▄ " COLOR W+/N
904         @ 1,3 SAY " ▄▄"
905         @ 1,6 SAY "▄" COLOR (colorvar)
906
907         @ 2,0 SAY "▄▄▄▄ " COLOR W+/N
908         @ 2,6 SAY "▄▄" COLOR (colorvar)
909
910         @ 3,0 SAY "▄▄ " COLOR W+/N
911
912         @ 4,0 SAY " ▄▄▄" COLOR (colorvar)
913     CASE SUBSTR(instring,curlet,1) = "Q"
914         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
915             FROM startx,curenty TO startx + 5,curenty + 8 ;
916             NONE NOSHADOW
917         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
918         curenty = curenty + 9
919         @ 0,0 SAY " ▄▄"
920         @ 0,3 SAY "▄▄" COLOR W+/N
921         @ 0,4 SAY "▄▄"
922
923         @ 1,0 SAY "▄▄ " COLOR W+/N
924         @ 1,3 SAY "▄▄" COLOR (colorvar)
```

```
925         @ 1,5 SAY "▄▄"
926         @ 1,7 SAY "▄" COLOR (colorvar)
927
928         @ 2,0 SAY "▄▄ " COLOR W+/N
929         @ 2,3 SAY "▄▄"
930         @ 2,7 SAY "▄" COLOR (colorvar)
931
932         @ 3,0 SAY " ▄▄▄▄"
933         @ 3,6 SAY "▄▄▄" COLOR (colorvar)
934
935         @ 4,0 SAY "  ▄▄▄▄" COLOR (colorvar)
936         @ 4,6 SAY "▄▄"
937         @ 4,7 SAY "▄" COLOR (colorvar)
938     CASE SUBSTR(instring,curlet,1) = "R"
939         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
940             FROM startx,curenty TO startx + 5,curenty + 7 ;
941             NONE NOSHADOW
942         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
943         curenty = curenty + 8
944         @ 0,0 SAY "▄▄▄▄" COLOR W+/N
945         @ 0,3 SAY "▄▄"
946
947         @ 1,0 SAY "▄▄ " COLOR W+/N
948         @ 1,3 SAY "▄▄"
949         @ 1,6 SAY "▄" COLOR (colorvar)
950
951         @ 2,0 SAY "▄▄▄▄▄▄" COLOR W+/N
952         @ 2,6 SAY "▄▄▄" COLOR (colorvar)
953
954         @ 3,0 SAY "▄▄ " COLOR W+/N
955         @ 3,3 SAY "▄▄"
956         @ 3,6 SAY "▄" COLOR (colorvar)
957
958         @ 4,0 SAY " ▄▄  ▄▄" COLOR (colorvar)
959     CASE SUBSTR(instring,curlet,1) = "S"
960         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
961             FROM startx,curenty TO startx + 5,curenty + 7 ;
962             NONE NOSHADOW
963         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
964         curenty = curenty + 8
965         @ 0,0 SAY "▄▄"
966         @ 0,2 SAY "▄▄▄▄" COLOR W+/N
967         @ 0,5 SAY "▄"
968
969         @ 1,0 SAY "▄▄▄▄"
970         @ 1,5 SAY "▄▄▄" COLOR (colorvar)
971
972         @ 2,0 SAY "▄  ▄▄▄"
973         @ 2,6 SAY "▄" COLOR (colorvar)
974
975         @ 3,0 SAY "▄▄▄▄"
976         @ 3,5 SAY "▄▄ " COLOR W+/N
977
978         @ 4,0 SAY "  ▄▄▄▄" COLOR (colorvar)
979     CASE SUBSTR(instring,curlet,1) = "T"
980         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
981             FROM startx,curenty TO startx + 5,curenty + 7 ;
982             NONE NOSHADOW
983         ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
984         curenty = curenty + 8
985         @ 0,0 SAY "▄▄"
986         @ 0,1 SAY "▄▄▄▄▄▄" COLOR W+/N
987         @ 0,6 SAY "▄" COLOR (colorvar)
988
989         @ 1,2 SAY "▄▄ " COLOR W+/N
990
991         @ 2,2 SAY "▄▄ " COLOR W+/N
992
993         @ 3,2 SAY "▄▄ " COLOR W+/N
994
995         @ 4,0 SAY "  ▄▄" COLOR (colorvar)
996     CASE SUBSTR(instring,curlet,1) = "U"
997         DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
998             FROM startx,curenty TO startx + 5,curenty + 6 ;
999             NONE NOSHADOW
1000        ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
1001        curenty = curenty + 8
1002        @ 0,0 SAY "▄▄"
1003        @ 0,2 SAY "▄ " COLOR (colorvar)
1004        @ 0,4 SAY "▄▄"
1005        @ 0,6 SAY "▄ " COLOR (colorvar)
1006
1007        @ 1,0 SAY "▄▄ " COLOR W+/N
1008        @ 1,3 SAY "▄▄"
1009        @ 1,6 SAY "▄" COLOR (colorvar)
1010
1011        @ 2,0 SAY "▄▄ " COLOR W+/N
1012        @ 2,3 SAY "▄"
1013        @ 2,4 SAY "▄▄ " COLOR W+/N
1014
1015        @ 3,0 SAY "▄▄"
1016        @ 3,2 SAY "▄" COLOR W+/N
1017        @ 3,3 SAY "▄"
1018        @ 3,4 SAY "▄▄ " COLOR W+/N
1019
1020        @ 4,0 SAY "  ▄▄▄▄" COLOR (colorvar)
1021    CASE SUBSTR(instring,curlet,1) = "V"
1022        DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
1023            FROM startx,curenty TO startx + 5,curenty + 7 ;
1024            NONE NOSHADOW
1025        ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
1026        curenty = curenty + 9
1027        @ 0,0 SAY "▄▄"
1028        @ 0,2 SAY "▄" COLOR (colorvar)
1029        @ 0,3 SAY "  ▄▄"
1030        @ 0,7 SAY "▄" COLOR (colorvar)
1031
1032        @ 1,0 SAY " ▄▄"
1033        @ 1,3 SAY "▄" COLOR (colorvar)
1034        @ 1,4 SAY "▄▄ " COLOR W+/N
1035        @ 1,7 SAY "▄▄" COLOR (colorvar)
1036
1037        @ 2,0 SAY " ▄▄"
1038        @ 2,3 SAY "▄▄ " COLOR W+/N
1039
1040        @ 3,0 SAY "  ▄▄"
1041        @ 3,4 SAY "▄▄ " COLOR W+/N
1042
1043        @ 4,0 SAY "    ▄▄" COLOR (colorvar)
1044    CASE SUBSTR(instring,curlet,1) = "W"
1045        DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
1046            FROM startx,curenty TO startx + 5,curenty + 10 ;
1047            NONE NOSHADOW
1048        ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
1049        curenty = curenty + 12
1050        @ 0,0 SAY "▄▄"
1051        @ 0,2 SAY "▄ " COLOR (colorvar)
1052        @ 0,4 SAY "▄▄"
1053        @ 0,6 SAY "▄ " COLOR (colorvar)
1054        @ 0,8 SAY "▄▄"
1055        @ 0,10 SAY "▄" COLOR (colorvar)
1056
```

```
1057        @ 1,0 SAY "[illegible]"
1058        @ 1,2 SAY "[illegible]" COLOR W+/N
1059        @ 1,3 SAY "[illegible]"
1060        @ 1,6 SAY "[illegible]" COLOR (colorvar)
1061        @ 1,7 SAY "[illegible]"
1062        @ 1,9 SAY "[illegible]" COLOR W+/N
1063
1064        @ 2,1 SAY "[illegible]" COLOR W+/N
1065
1066        @ 3,2 SAY "[illegible]" COLOR W+/N
1067        @ 3,6 SAY "[illegible]" COLOR W+/N
1068        @ 3,9 SAY "[illegible]" COLOR (colorvar)
1069
1070        @ 4,0 SAY "[illegible]" COLOR (colorvar)
1071     CASE SUBSTR(instring,curlet,1) = "X"
1072        DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
1073            FROM startx,curenty TO startx + 5,curenty + 8 ;
1074            NONE NOSHADOW
1075        ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
1076        curenty = curenty + 9
1077        @ 0,0 SAY "[illegible]"
1078        @ 0,2 SAY "[illegible]" COLOR (colorvar)
1079        @ 0,3 SAY "[illegible]"
1080        @ 0,7 SAY "[illegible]" COLOR (colorvar)
1081
1082        @ 1,0 SAY "[illegible]"
1083        @ 1,5 SAY "[illegible]" COLOR W+/N
1084        @ 1,6 SAY "[illegible]" COLOR (colorvar)
1085
1086        @ 2,0 SAY "[illegible]"
1087        @ 2,3 SAY "[illegible]" COLOR W+/N
1088
1089        @ 3,0 SAY "[illegible]" COLOR W+/N
1090        @ 3,3 SAY "[illegible]" COLOR (colorvar)
1091        @ 3,4 SAY "[illegible]"
1092        @ 3,7 SAY "[illegible]" COLOR (colorvar)
1093
1094        @ 4,0 SAY "[illegible]" COLOR (colorvar)
1095     CASE SUBSTR(instring,curlet,1) = "Y"
1096        DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
1097            FROM startx,curenty TO startx + 5,curenty + 9 ;
1098            NONE NOSHADOW
1099        ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
1100        curenty = curenty + 10
1101        @ 0,0 SAY "[illegible]"
1102        @ 0,2 SAY "[illegible]" COLOR (colorvar)
1103        @ 0,3 SAY "[illegible]"
1104        @ 0,8 SAY "[illegible]" COLOR (colorvar)
1105
1106        @ 1,0 SAY "[illegible]"
1107        @ 1,3 SAY "[illegible]" COLOR (colorvar)
1108        @ 1,4 SAY "[illegible]"
1109        @ 1,7 SAY "[illegible]" COLOR (colorvar)
1110
1111        @ 2,0 SAY "[illegible]"
1112        @ 2,5 SAY "[illegible]" COLOR W+/N
1113        @ 2,7 SAY "[illegible]" COLOR (colorvar)
1114
1115        @ 3,3 SAY "[illegible]" COLOR W+/N
1116
1117        @ 4,0 SAY "[illegible]" COLOR (colorvar)
1118     CASE SUBSTR(instring,curlet,1) = "Z"
1119        DEFINE WINDOW ("letter" + ALLTRIM(STR(curlet))) ;
1120            FROM startx,curenty TO startx + 5,curenty + 8 ;
1121            NONE NOSHADOW
1122        ACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
```

```
1123        curenty = curenty + 8
1124        @ 0,0 SAY "[illegible]"
1125        @ 0,1 SAY "[illegible]" COLOR W+/N
1126        @ 0,6 SAY "[illegible]" COLOR (colorvar)
1127
1128        @ 1,0 SAY "[illegible]"
1129        @ 1,4 SAY "[illegible]" COLOR W+/N
1130        @ 1,6 SAY "[illegible]" COLOR (colorvar)
1131
1132        @ 2,0 SAY "[illegible]"
1133        @ 2,3 SAY "[illegible]" COLOR W+/N
1134
1135        @ 3,0 SAY "[illegible]" COLOR W+/N
1136        @ 3,4 SAY "[illegible]"
1137
1138        @ 4,0 SAY "[illegible]" COLOR (colorvar)
1139     ENDCASE
1140     curlet = curlet + 1
1141  ENDDO
1142
1143  DEFINE WINDOW ("WRN" ) ;
1144      FROM x_coor+startx,y_coor-12 TO x_coor+startx + 11,y_coor + 60 ;
1145      NONE NOSHADOW
1146  ACTIVATE WINDOW ("WRN")
1147  @ 3,12 SAY "(c) Naval Health Research Center, Pat.Pend."
1148
1149  @ 5,0 SAY "#################################################################
=> ##############"
1150  @ 6,0 SAY "If any problem please contact:    D.M. Pearsall (Dee)
=>                                                 "
1151  @ 7,0 SAY "                                  Naval Health Research Ce
=> nter Code 22 "
1152  @ 8,0 SAY "                                  P.O.Box 85122
=>                                                 "
1153  @ 9,0 SAY "                                  San Diego, CA 92186-5122
=>                                                 "
1154  @ 10,0 SAY "                                  (619) 553 - 8401
=>                                                 "
1155  @ 11,0 SAY "#################################################################
=> ##############"
1156
1157  WAIT "" TIMEOUT o_t
1158  IF NOT EMPTY(instring)
1159     curlet = 1
1160     FOR i = 1 TO len_str
1161        DEACTIVATE WINDOW ("letter" + ALLTRIM(STR(curlet)))
1162        RELEASE WINDOW ("letter" + ALLTRIM(STR(curlet)))
1163        curlet = curlet + 1
1164     NEXT
1165  ENDIF
1166  DEACTIVATE WINDOW "wrn"
1167  RELEASE WINDOW "wrn"
1168  CLEAR
1169  *: EOF: BIGCHARS.ACT
```

```
1     *
2     *
3     *        10/07/92              CAMD.MPR                 11:51:56
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENMENU.
15    *
16    *
17
18
19    *
20    *
21    *                    Menu Definition
22    *
23    *
24    *
25
26    SET SYSMENU TO
27
28    SET SYSMENU AUTOMATIC
29
30    DEFINE PAD _q9p0pfko5 OF _msysmenu PROMPT "\<System" COLOR SCHEME 3 ;
31        KEY alt+s, ""
32    DEFINE PAD _q9p0pfkpp OF _msysmenu PROMPT "\<File" COLOR SCHEME 3 ;
33        KEY alt+F, ""
34    DEFINE PAD _q9p0pfkqc OF _msysmenu PROMPT "\<Edit" COLOR SCHEME 3 ;
35        KEY alt+E, ""
36    DEFINE PAD _q9p0pfkr1 OF _msysmenu PROMPT "\<Diagnosis" COLOR SCHEME
=> 3 ;
37        KEY alt+D, ""
38    DEFINE PAD _q9p0pfkro OF _msysmenu PROMPT "\<Window" COLOR SCHEME 3 ;
39        KEY alt+W, ""
40    ON PAD _q9p0pfko5 OF _msysmenu ACTIVATE POPUP _csystem
41    ON PAD _q9p0pfkpp OF _msysmenu ACTIVATE POPUP _cfile
42    ON PAD _q9p0pfkqc OF _msysmenu ACTIVATE POPUP _cedit
43    ON PAD _q9p0pfkr1 OF _msysmenu ACTIVATE POPUP diagnosis
44    ON PAD _q9p0pfkro OF _msysmenu ACTIVATE POPUP _cwindow
45
46    DEFINE POPUP _csystem MARGIN RELATIVE SHADOW COLOR SCHEME 4
47    DEFINE BAR _mst_help OF _csystem PROMPT "\<Help..." ;
48        KEY f1, "F1"
49    DEFINE BAR 2 OF _csystem PROMPT "\-"
50    DEFINE BAR 3 OF _csystem PROMPT "Change \<Password" ;
51        SKIP FOR NOT EMPTY(WONTOP())
52    DEFINE BAR 4 OF _csystem PROMPT "\<Backup" ;
53        SKIP FOR NOT EMPTY(WONTOP())
54    DEFINE BAR 5 OF _csystem PROMPT "\<Restore" ;
55        SKIP FOR NOT EMPTY(WONTOP())
56    DEFINE BAR 6 OF _csystem PROMPT "\-"
57    DEFINE BAR _mst_filer OF _csystem PROMPT "\<Filer"
58    DEFINE BAR _mst_calcu OF _csystem PROMPT "\<Calculator"
59    DEFINE BAR _mst_diary OF _csystem PROMPT "Calendar/\<Diary"
60    DEFINE BAR _mst_puzzl OF _csystem PROMPT "Pu\<zzle"
61    ON SELECTION BAR 3 OF _csystem ;
62        DO _q9p0pfl0y ;
63        IN LOCFILE("CAMD" ,"MPX;MPR|FXP;PRG" ,"Where is CAMD?")
64    ON SELECTION BAR 4 OF _csystem DO _mhit WITH "backup"
65    ON SELECTION BAR 5 OF _csystem DO _mhit WITH "restore"
66
67    DEFINE POPUP _cfile MARGIN RELATIVE SHADOW COLOR SCHEME 4
68    DEFINE BAR 1 OF _cfile PROMPT "\<Reindex" ;
69        SKIP FOR NOT EMPTY(WONTOP())
70    DEFINE BAR 2 OF _cfile PROMPT "\-"
71    DEFINE BAR 3 OF _cfile PROMPT "\<Quit"
72    ON SELECTION BAR 1 OF _cfile DO REINDEX
73    ON SELECTION BAR 3 OF _cfile DO _quit
74
75    DEFINE POPUP _cedit MARGIN RELATIVE SHADOW COLOR SCHEME 4
76    DEFINE BAR _med_undo OF _cedit PROMPT "\<Undo" ;
77        KEY ctrl+u, "^U"
78    DEFINE BAR _med_redo OF _cedit PROMPT "\<Redo" ;
79        KEY ctrl+R, "^R"
80    DEFINE BAR _med_sp100 OF _cedit PROMPT "\-"
81    DEFINE BAR _med_cut OF _cedit PROMPT "Cu\<t" ;
82        KEY ctrl+X, "^X"
83    DEFINE BAR _med_copy OF _cedit PROMPT "\<Copy" ;
84        KEY ctrl+C, "^C"
85    DEFINE BAR _med_paste OF _cedit PROMPT "\<Paste" ;
86        KEY ctrl+v, "^V"
87    DEFINE BAR _med_clear OF _cedit PROMPT "Clear"
88    DEFINE BAR _med_sp200 OF _cedit PROMPT "\-"
89    DEFINE BAR _med_slcta OF _cedit PROMPT "Select \<All" ;
90        KEY ctrl+A, "^A"
91    DEFINE BAR _med_sp300 OF _cedit PROMPT "\-"
92    DEFINE BAR _med_goto OF _cedit PROMPT "Goto \<Line..."
93    DEFINE BAR _med_find OF _cedit PROMPT "\<Find..." ;
94        KEY ctrl+F, "^F"
95    DEFINE BAR _med_finda OF _cedit PROMPT "Find A\<gain" ;
96        KEY ctrl+G, "^G"
97    DEFINE BAR _med_repl OF _cedit PROMPT "R\<eplace And Find Again" ;
98        KEY ctrl+E, "^E"
99    DEFINE BAR _med_repla OF _cedit PROMPT "Replace All"
100   DEFINE BAR _med_sp400 OF _cedit PROMPT "\-"
101   DEFINE BAR _med_pref OF _cedit PROMPT "Prefere\<nces..."
102
103   DEFINE POPUP diagnosis MARGIN RELATIVE SHADOW COLOR SCHEME 4
104   DEFINE BAR 1 OF diagnosis PROMPT "\<Register Patient..." ;
105       SKIP FOR enc_open AND mencstate != "N"
106   DEFINE BAR 2 OF diagnosis PROMPT "\<Encounter..."
107   DEFINE BAR 3 OF diagnosis PROMPT "\<SF600 Report"
108   DEFINE BAR 4 OF diagnosis PROMPT "\-"
109   DEFINE BAR 5 OF diagnosis PROMPT "\<Disease/Treatment Lookup"
110   DEFINE BAR 6 OF diagnosis PROMPT "\-"
111   DEFINE BAR 7 OF diagnosis PROMPT "Encounter \<Training Mode"
112   ON SELECTION BAR 1 OF diagnosis DO _mhit WITH "m_pat"
113   ON SELECTION BAR 2 OF diagnosis DO _mhit WITH "m_newenc"
114   ON SELECTION BAR 3 OF diagnosis DO _mhit WITH "m_sf600"
115   ON SELECTION BAR 5 OF diagnosis DO _mhit WITH "m_lookup"
116   ON SELECTION BAR 7 OF diagnosis ;
117       DO _q9p0pflr8 ;
118       IN LOCFILE("CAMD" ,"MPX;MPR|FXP;PRG" ,"Where is CAMD?")
119
120   DEFINE POPUP _cwindow MARGIN RELATIVE SHADOW COLOR SCHEME 4
121   DEFINE BAR _mwi_move OF _cwindow PROMPT "\<Move" ;
122       KEY ctrl+f7, "^F7"
123   DEFINE BAR _mwi_size OF _cwindow PROMPT "\<Size" ;
124       KEY ctrl+f8, "^F8"
125   DEFINE BAR _mwi_zoom OF _cwindow PROMPT "\<Zoom 1" ;
126       KEY ctrl+f10, "^F10"
127   DEFINE BAR _mwi_min OF _cwindow PROMPT "Z\<oom 1" ;
128       KEY ctrl+f9, "^F9"
129   DEFINE BAR _mwi_rotat OF _cwindow PROMPT "\<Cycle" ;
130       KEY ctrl+f1, "^F1"
131   DEFINE BAR 6 OF _cwindow PROMPT "Co\<lor..."
```

```
132     DEFINE BAR _mwi_sp200 OF _cwindow PROMPT "\-"
133     DEFINE BAR _mwi_cmd OF _cwindow PROMPT "Command" ;
134         KEY ctrl+f2, "^F2"
135     DEFINE BAR _mwi_debug OF _cwindow PROMPT "\<Debug" ;
136         SKIP FOR !is_debug
137     DEFINE BAR _mwi_trace OF _cwindow PROMPT "\<Trace" ;
138         SKIP FOR !is_debug
139     DEFINE BAR _mwi_view OF _cwindow PROMPT "\<View"
140     ON SELECTION BAR 6 OF _cwindow DO colorset.spr
141
142
143     *
144     *
145     *                    Cleanup Code & Procedures
146     *
147     *
148     *
149
150     PROCEDURE mhit
151     PARAMETER PROG
152     *
153     *    If an application READ is already active,
154     *    execute CLEAR READ.  This causes the active
155     *    application READ to terminate as soon as
156     *    the menu processing routine is complete and
157     *    also requests that the foundation READ be teminated.
158     *    The active application READ terminates first
159     *    and control passes back to the foundation READ.
160     *    Since it has also been CLEARed, it executes
161     *    its VALID routine.  This VALID routine, noting
162     *    that the PUBLIC variable 'tobedone' has been
163     *    assigned a value, launches the new application
164     *    which has been requested.
165     *
166     IF RDLEVEL() > 1
167        tobedone = PROG
168        CLEAR READ
169     ELSE
170        *
171        * If no other application is running,
172        * just start the requested application
173        *
174        DO (PROG)
175     ENDIF
176
177
178
179     PROCEDURE mexit
180     dropdead = .T.
181     CLEAR READ ALL
182
183
184     *
185     *
186     *    _Q9P0PFLOY  ON SELECTION BAR 3 OF POPUP _csystem
187     *
188     *    Procedure Origin:
189     *
190     *    From Menu:  CAMD.MPR,            Record:    7
191     *    Called By:  ON SELECTION BAR 3 OF POPUP _csystem
192     *    Prompt:     Change Password
193     *    Snippet:    1
194     *
195     *
196     *
197     PROCEDURE _q9p0pfl0y
198     =pushhelp("CHANGE PASSWORD")
199     DO pwdedt
200     =pophelp()
201
202
203     *
204     *
205     *    _Q9P0PFLR8  ON SELECTION BAR 7 OF POPUP diagnosis
206     *
207     *    Procedure Origin:
208     *
209     *    From Menu:  CAMD.MPR,            Record:   47
210     *    Called By:  ON SELECTION BAR 7 OF POPUP diagnosis
211     *    Prompt:     Encounter Training Mode
212     *    Snippet:    2
213     *
214     *
215     *
216     PROCEDURE _q9p0pflr8
217     DO camdtr.mpr
218     DO _mhit WITH "m_training"
219     DO camd.mpr
221     *: EOF: CAMD.AC2
```

```
 1      *
 2      *
 3      *        09/28/92              CAMDTR.MPR                14:22:23
 4      *
 5      *
 6      *
 7      *        Author's Name
 8      *
 9      *        Copyright (c) 1992 Company Name
10      *        Address
11      *        City,     Zip
12      *
13      *        Description:
14      *        This program was automatically generated by GENMENU.
15      *
16      *
17
18
19      *
20      *
21      *                          Menu Definition
22      *
23      *
24      *
25
26      SET SYSMENU TO
27
28      SET SYSMENU AUTOMATIC
29
30      DEFINE PAD _q9g0utlky OF _msysmenu PROMPT "\<System" COLOR SCHEME 3 ;
31          KEY alt+S, ""
32      DEFINE PAD _q9g0utllc OF _msysmenu PROMPT "\<File" COLOR SCHEME 3 ;
33          KEY alt+F, ""
34      DEFINE PAD _q9g0utlm7 OF _msysmenu PROMPT "\<Edit" COLOR SCHEME 3 ;
35          KEY alt+E, ""
36      DEFINE PAD _q9g0utlmt OF _msysmenu PROMPT "\<Diagnosis" COLOR SCHEME
=> 3 ;
37          KEY alt+D, ""
38      DEFINE PAD _q9g0utlnc OF _msysmenu PROMPT "\<Window" COLOR SCHEME 3 ;
39          KEY alt+W, ""
40      DEFINE PAD _q9g0utloc OF _msysmenu PROMPT "\TRAINING MODE" COLOR SCHE
=> ME 3
41      ON PAD _q9g0utlky OF _msysmenu ACTIVATE POPUP _csystem
42      ON PAD _q9g0utllo OF _msysmenu ACTIVATE POPUP _cfile
43      ON PAD _q9g0utlm7 OF _msysmenu ACTIVATE POPUP _cedit
44      ON PAD _q9g0utlmt OF _msysmenu ACTIVATE POPUP diagnosis
45      ON PAD _q9g0utlnc OF _msysmenu ACTIVATE POPUP _cwindow
46
47      DEFINE POPUP _csystem MARGIN RELATIVE SHADOW COLOR SCHEME 4
48      DEFINE BAR _mst_help OF _csystem PROMPT "\<Help..." ;
49          KEY f1, "F1"
50      DEFINE BAR 2 OF _csystem PROMPT "\-"
51      DEFINE BAR 3 OF _csystem PROMPT "Change \<Password" ;
52          SKIP FOR NOT EMPTY(WONTOP())
53      DEFINE BAR 4 OF _csystem PROMPT "\-Backup" ;
54          SKIP FOR NOT EMPTY(WONTOP())
55      DEFINE BAR 5 OF _csystem PROMPT "\<Restore" ;
56          SKIP FOR NOT EMPTY(WONTOP())
57      DEFINE BAR 6 OF _csystem PROMPT "\-"
58      DEFINE BAR _mst_filer OF _csystem PROMPT "\<Filer"
59      DEFINE BAR _mst_calcu OF _csystem PROMPT "\<Calculator"
60      DEFINE BAR _mst_diary OF _csystem PROMPT "Calendar/\<Diary"
61      DEFINE BAR _mst_puzzl OF _csystem PROMPT "Pu\<zzle"
62      ON SELECTION BAR 3 OF _csystem DO pwdedt
63      ON SELECTION BAR 4 OF _csystem DO _mhit WITH "backup"
64      ON SELECTION BAR 5 OF _csystem DO _mhit WITH "restore"
65
66      DEFINE POPUP _cfile MARGIN RELATIVE SHADOW COLOR SCHEME 4
67      DEFINE BAR 1 OF _cfile PROMPT "\<Reindex" ;
68          SKIP FOR NOT EMPTY(WONTOP())
69      DEFINE BAR 2 OF _cfile PROMPT "\-"
70      DEFINE BAR 3 OF _cfile PROMPT "\<Quit"
71      ON SELECTION BAR 1 OF _cfile DO REINDEX
72      ON SELECTION BAR 3 OF _cfile DO _quit
73
74      DEFINE POPUP _cedit MARGIN RELATIVE SHADOW COLOR SCHEME 4
75      DEFINE BAR _med_undo OF _cedit PROMPT "\<Undo" ;
76          KEY ctrl+u, "^U"
77      DEFINE BAR _med_redo OF _cedit PROMPT "\<Redo" ;
78          KEY ctrl+R, "^R"
79      DEFINE BAR _med_sp100 OF _cedit PROMPT "\-"
80      DEFINE BAR _med_cut OF _cedit PROMPT "Cu\<t" ;
81          KEY ctrl+X, "^X"
82      DEFINE BAR _med_copy OF _cedit PROMPT "\<Copy" ;
83          KEY ctrl+C, "^C"
84      DEFINE BAR _med_paste OF _cedit PROMPT "\<Paste" ;
85          KEY ctrl+v, "^V"
86      DEFINE BAR _med_clear OF _cedit PROMPT "Clear"
87      DEFINE BAR _med_sp200 OF _cedit PROMPT "\-"
88      DEFINE BAR _med_slcta OF _cedit PROMPT "Select \<All" ;
89          KEY ctrl+A, "^A"
90      DEFINE BAR _med_sp300 OF _cedit PROMPT "\-"
91      DEFINE BAR _med_goto OF _cedit PROMPT "Goto \<Line..."
92      DEFINE BAR _med_find OF _cedit PROMPT "\<Find..." ;
93          KEY ctrl+F, "^F"
94      DEFINE BAR _med_finda OF _cedit PROMPT "Find A\<gain" ;
95          KEY ctrl+G, "^G"
96      DEFINE BAR _med_repl OF _cedit PROMPT "R\<eplace And Find Again" ;
97          KEY ctrl+E, "^E"
98      DEFINE BAR _med_repla OF _cedit PROMPT "Replace All"
99      DEFINE BAR _med_sp400 OF _cedit PROMPT "\-"
100     DEFINE BAR _med_pref OF _cedit PROMPT "Prefere\<nces..."
101
102     DEFINE POPUP diagnosis MARGIN RELATIVE SHADOW COLOR SCHEME 4
103     DEFINE BAR 1 OF diagnosis PROMPT "\<Register Patient..." ;
104         SKIP FOR enc_open AND mencstate != "N"
105     DEFINE BAR 2 OF diagnosis PROMPT "\<Encounter..."
106     DEFINE BAR 3 OF diagnosis PROMPT "\<SF600 Report"
107     DEFINE BAR 4 OF diagnosis PROMPT "\-"
108     DEFINE BAR 5 OF diagnosis PROMPT "\<Disease/Treatment Lookup"
109     DEFINE BAR 6 OF diagnosis PROMPT "\-"
110     DEFINE BAR 7 OF diagnosis PROMPT "Encounter \<Training Mode"
111     ON SELECTION BAR 1 OF diagnosis DO _mhit WITH "m_pat"
112     ON SELECTION BAR 2 OF diagnosis DO _mhit WITH "m_newenc"
113     ON SELECTION BAR 3 OF diagnosis DO _mhit WITH "m_sf600"
114     ON SELECTION BAR 5 OF diagnosis DO _mhit WITH "m_lookup"
115     ON SELECTION BAR 7 OF diagnosis DO _mhit WITH "m_training"
116
117     DEFINE POPUP _cwindow MARGIN RELATIVE SHADOW COLOR SCHEME 4
118     DEFINE BAR _mwi_move OF _cwindow PROMPT "\<Move" ;
119         KEY ctrl+f7, "^F7"
120     DEFINE BAR _mwi_size OF _cwindow PROMPT "\<Size" ;
121         KEY ctrl+f8, "^F8"
122     DEFINE BAR _mwi_zoom OF _cwindow PROMPT "\<Zoom 1" ;
123         KEY ctrl+f10, "^F10"
124     DEFINE BAR _mwi_min OF _cwindow PROMPT "Z\<oom 1" ;
125         KEY ctrl+f9, "^F9"
126     DEFINE BAR _mwi_rotat OF _cwindow PROMPT "\<Cycle" ;
127         KEY ctrl+f1, "^F1"
128     DEFINE BAR _mwi_color OF _cwindow PROMPT "Co\<lor..."
129     DEFINE BAR _mwi_sp200 OF _cwindow PROMPT "\-"
130     DEFINE BAR _mwi_cmd OF _cwindow PROMPT "Command" ;
```

```
131         KEY ctrl+f2, "^F2"
132     DEFINE BAR _mwi_debug OF _cwindow PROMPT "\<Debug" ;
133         SKIP FOR !is_debug
134     DEFINE BAR _mwi_trace OF _cwindow PROMPT "\<Trace" ;
135         SKIP FOR !is_debug
136     DEFINE BAR _mwi_view OF _cwindow PROMPT "\<View"
137
138
139     *
140     *
141     *                    Cleanup Code & Procedures
142     *
143     *
144     *
145
146     PROCEDURE mhit
147     PARAMETER PROG
148     *
149     *    If an application READ is already active,
150     *    execute CLEAR READ.  This causes the active
151     *    application READ to terminate as soon as
152     *    the menu processing routine is complete and
153     *    also requests that the foundation READ be teminated.
154     *    The active application READ terminates first
155     *    and control passes back to the foundation READ.
156     *    Since it has also been CLEARed, it executes
157     *    its VALID routine.  This VALID routine, noting
158     *    that the PUBLIC variable 'tobedone' has been
159     *    assigned a value, launches the new application
160     *    which has been requested.
161     *
162     IF RDLEVEL() > 1
163        tobedone = PROG
164        CLEAR READ
165     ELSE
166        *
167        * If no other application is running,
168        * just start the requested application
169        *
170        DO (PROG)
171     ENDIF
172
173
174
175     PROCEDURE mexit
176     dropdead = .T.
177     CLEAR READ ALL
179     *: EOF: CAMDTR.AC2
```

```
1     *
2     *
3     *        10/07/92            COLORSET.SPR                11:52:12
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                    Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_color")
39       DEFINE WINDOW w_color ;
40           FROM INT((SROW()-14)/2),INT((SCOL()-32)/2) ;
41           TO INT((SROW()-14)/2)+13,INT((SCOL()-32)/2)+31 ;
42           NOFLOAT ;
43           NOCLOSE ;
44           SHADOW ;
45           DOUBLE ;
46           COLOR SCHEME 5
47    ENDIF
48
49
50    *
51    *
52    *                COLORSET Setup Code - SECTION 2
53    *
54    *
55    *
56
57    #REGION 1
58    IF !USED("colorset")
59       SELECT 0
60       USE colorset INDEX colorset
61    ELSE
62       SELECT colorset
63    ENDIF
64    COPY TO ARRAY colorlist FIELD color_set
65
66
```

```
67    *
68    *
69    *                COLORSET Screen Layout
70    *
71    *
72    *
73
74    #REGION 1
75    IF WVISIBLE("w_color")
76       ACTIVATE WINDOW w_color SAME
77    ELSE
78       ACTIVATE WINDOW w_color NOSHOW
79    ENDIF
80    @ 0,1 GET m.coloridx ;
81        PICTURE "@&N" ;
82        FROM colorlist ;
83        SIZE 10,27 ;
84        DEFAULT 1 ;
85        COLOR SCHEME 6
86    @ 11,5 GET m.action ;
87        PICTURE "@*HT \<Load;\<Cancel" ;
88        SIZE 1,8,3 ;
89        DEFAULT 1 ;
90        VALID _q9p0pfx2h()
91
92    IF NOT WVISIBLE("w_color")
93       ACTIVATE WINDOW w_color
94    ENDIF
95
96    READ CYCLE
97
98    RELEASE WINDOW w_color
99
100   #REGION 0
101   IF m.talkstat = "ON"
102      SET TALK ON
103   ENDIF
104   IF m.compstat = "ON"
105      SET COMPATIBLE ON
106   ENDIF
107
108
109   *
110   *
111   *        _q9p0pfx2h          m.action VALID
112   *
113   *        Function Origin:
114   *
115   *        From Screen:        COLORSET,     Record Number:    3
116   *        Variable:           m.action
117   *        Called By:          VALID Clause
118   *        Object Type:        Push Button
119   *        Snippet Number:     1
120   *
121   *
122   *
123   FUNCTION _q9p0pfx2h      && m.action VALID
124   #REGION 1
125   IF m.action = 1
126      m.colorset = TRIM(colorlist[m.coloridx])
127      IF m.colorset $ "COLOR"
128         m.colorset = "DEFAULT"
129      ENDIF
130      SET COLOR SET TO (m.colorset)
131   ENDIF
133   *: EOF: COLORSET.AC1
```

```
1   *:*****************************************************************
2   *:
3   *: Procedure file: C:\CAMD\WORK\DEMO.PRG
4   *:
5   *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
6   *:          Author: ROY DOBBINS and HOA LY
7   *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
8   *:  Last modified: 10/05/92      16:36
9   *:
10  *:  Procs & Fncts: SYSINIT()
11  *:                : INIENCOUNT()
12  *:                : INISYMPTOM()
13  *:                : INIDIAGNOS()
14  *:                : BATCHTEST()
15  *:                : SYSCLOSE()
16  *:                : M_RESUME
17  *:                : M_REGPAT
18  *:                : M_PAT
19  *:                : M_ENC
20  *:                : M_COMPL
21  *:                : M_DIAG
22  *:                : M_ASSIST
23  *:                : M_HANDP
24  *:                : M_TREAT
25  *:                : M_REDOQ
26  *:                : ENCOUNT
27  *:                : M_ENCCLO
28  *:                : M_ENCSAV
29  *:                : M_ENCVALID
30  *:                : M_ENCGET
31  *:                : M_ENCNEW
32  *:                : EDSEXAGE
33  *:                : M_ENCRULE
34  *:                : LODRULE
35  *:                : TESTRULES
36  *:                : EVALRULES
37  *:                : OBJTYPE
38  *:                : EVALCOND
39  *:                : UNDORULES
40  *:                : UNDOFACTOR
41  *:                : UNDOQUAL
42  *:                : ASKDIAG
43  *:                : GETDIAGAREA
44  *:                : GETPATIENT
45  *:                : GETENCNT
46  *:                : LODENC
47  *:                : M_CHANGE
48  *:                : M_UNDO
49  *:                : M_UNDORULE
50  *:                : MTUNDO
51  *:                : MAKEDIAG
52  *:                : GETCURPAT
53  *:                : SETCURPAT
54  *:                : M_PATLIST
55  *:                : M_PATGET
56  *:                : M_PATPUT
57  *:                : M_PATADD
58  *:                : AGEAT
59  *:                : ADDDIAGNOS
60  *:                : ADDHISTORY
61  *:                : ADDSYMPTOM
62  *:                : DELDIAGNOS
63  *:                : RESETDIAG
64  *:                : DELSYMPTOM
65  *:                : INCR
66  *:                : DECR
67  *:                : MULFACTOR
68  *:                : SUMFACTOR
69  *:                : NORMFACTOR
70  *:                : GETFACTOR
71  *:                : SETFACTOR
72  *:                : DEFFACTOR
73  *:                : FACTORID
74  *:                : ADDFACTOR
75  *:                : LOOKUPQ
76  *:                : LOOKUPNAM
77  *:                : ASK
78  *:                : ISQ
79  *:                : VALQ
80  *:                : EVALQ
81  *:                : EDITSYMP
82  *:                : SETQ
83  *:                : DEFQ
84  *:                : GETQ
85  *:                : ERASEQ
86  *:                : RESTOREQ
87  *:                : PUSHQ
88  *:                : CHANGEQ
89  *:                : POPQ
90  *:                : LASTQ
91  *:                : QID
92  *:                : ADDQ
93  *:                : STRCOND
94  *:                : STRDIAG
95  *:                : STRNATURAL
96  *:                : ENC2STR
97  *:                : ENC2BIT
98  *:                : INVERTCOND
99  *:                : WPAT
100 *:                : WPCLEAN
101 *:                : M_SF600
102 *:                : STREPORT
103 *:                : SFSYMP
104 *:                : EDITTEXT
105 *:                : SFDIAG
106 *:                : SFTREAT
107 *:                : M_NEWENC
108 *:                : M_TRAINING
109 *:                : BACKUP
110 *:                : RESTORE
111 *:                : M_LOOKUP
112 *:                : M_DATA
113 *:
114 *:          Calls: SECNECK.PRG
115 *:                 : SYSINIT()          (function  in DEMO.PRG)
116 *:                 : USERINIT()         (function  in U.PRG)
117 *:                 : POPUPSHOW()        (function  in U.PRG)
118 *:                 : INIENCOUNT()       (function  in DEMO.PRG)
119 *:                 : INISYMPTOM()       (function  in DEMO.PRG)
120 *:                 : INIDIAGNOS()       (function  in DEMO.PRG)
121 *:                 : POPUPHIDE()        (function  in U.PRG)
122 *:                 : OPENFILES()        (function  in U.PRG)
123 *:                 : BATCHTEST()        (function  in DEMO.PRG)
124 *:                 : MAINMENU()         (function  in U.PRG)
125 *:                 : USERCLOSE()        (function  in U.PRG)
126 *:                 : SYSCLOSE()         (function  in DEMO.PRG)
127 *:
128 *:      Documented 10/14/92 at 10:49                FoxDoc  version 2.10
129 *:*****************************************************************
130     * demo.prg
131     *-----------------------------------------------------
132     *
```

```
133     * Medical Practice Support System
134     * $Revision: 1.1 $   $Date: 92/05/04 11:10:24 $
135     * $Author: RoyWDobbins $
136     * 4/21/92 rwd01  popup warning when assist run without all
137     *                 symptoms entered by user
138     * 4/27/92 rwd03  batch test mode
139     * 6/1/92  rwd05  handle empty ruleset
140     * 6/2/92  rwd07  performance tuning of assist
141     * 6/9/92  hll01  delete variable "startdir"
142     * 6/10/92 hll02  add procedures and functions for patient intergrate
=> with SAMS
143     * 6/10/92 hll03  add procedures and functions for doing sf600 report
144     * 6/18/92 hll04  Update patient info (sex and age) to working memory
145     * 6/18/92 hll05  Comment the computer diagnosis out from sf600 report
146     * 6/18/92 hll06  add mreport to PUBLIC variable for geting the sf600
=> report
147     *-----------------------------------------------------
148     PARAMETERS testdir, testfile, logfile                 && rwd03
149
150     * These parameters are only required for testing.   && rwd03
151     * By default, the system runs in interactive mode   && rwd03
152     * and none of these parameters are then required:   && rwd03
153
154     * testdir is the name of the test directory         && rwd03
155     *   This directory is the location of the patient   && rwd03
156     *   encounter transaction files used for testing.   && rwd03
157
158     * testfile is the name of the test driver file      && rwd03
159     *   This file lists the set of test cases to be run && rwd03
160
161     * logfile is the name of the log file               && rwd03
162     *   This file records a summary of the test results && rwd03
163     PUBLIC mtalk,mproc                          && hll01 store the init
164     IF SET("TALK") = "ON"
165        SET TALK OFF
166        mtalk = SET("TALK")
167     ENDIF
168     uservalid = secheck()                       && hll12 security check
169     IF !uservalid
170        RETURN
171     ENDIF
172     =sysinit()                  && system initialisation
173     =userinit()                 && init user interface subsystem && rwd
=> 03
174
175     * delay expected here ...
176     PRIVATE wpopup
177     wpopup = popupshow("Initializing...")
178     =iniencount()               && initialize encounter record
179     =inisymptom()               && initialize list of symptoms
180     =inidiagnos()               && initialize list of diagnoses
181     =popuphide(wpopup)
182
183     *-----------------------------------------------------
184     *
185     * main application menu
186     *
187     *-----------------------------------------------------
188     * system state vector for knowledge base processing
189     PRIVATE statevecf           && factors
190     PRIVATE statevecq           && qualifiers
191     PRIVATE fire                && active rules
192     PUBLIC  agenda              && indicates agenda tested   && rwd07
193     PRIVATE ruleaction          && action clause        && rwd07
194     * to be re-dimensioned
195     DIMENSION statevecf[10], statevecq[10], fire[10], ruleaction[10]   &&
=>  rwd07
196
197     mprovider = IIF(mprovider > 0, mprovider, 1)      && accept any logon
198     IF mprovider > 0
199        mstate = s_main              && user logged on
200        IF !EMPTY(testfile)                                  && rwd03
201           * open encounter test files                       && rwd03
202           =openfiles(testdir)                               && rwd03
203           logfile = IIF( !EMPTY(logfile), logfile, "")      && rwd03
204           =batchtest(testdir, testfile, logfile)            && rwd03
205        ELSE                                                 && rwd03
206           * open encounter files                            && rwd03
207           *=openfiles()                                     && rwd03, &&hll1
=> 0
208           =mainmenu()                                       && rwd03
209        ENDIF                                                && rwd03
210        event = e_ok
211     ELSE
212        mstate = s_shutdown          && logon failed
213     ENDIF
214     =userclose()                    && close user interface subsystem
215     =sysclose()                     && close system
216     RETURN
217
218     *
219     *      *** Main menu service procedures ***
220     *
221
222     *-----------------------------------------------------
223     *
224     * menu service procedure - resume encounter screen
225     *
226     *-----------------------------------------------------
247  *!*****************************************************************
248  *!
249  *!      Procedure: M_RESUME
250  *!
251  *!      Called by: M_ENC              (procedure in DEMO.PRG)
252  *!               : MYHANDLER          (procedure in U.PRG)
253  *!
254  *!          Calls: STRCOND            (procedure in DEMO.PRG)
255  *!               : STRDIAG            (procedure in DEMO.PRG)
256  *!               : GETEXAREA()        (function  in U.PRG)
257  *!               : DISEASENAM()       (function  in U.PRG)
258  *!               : AGEAT              (procedure in DEMO.PRG)
259  *!               : EDSEXAGE           (procedure in DEMO.PRG)
260  *!               : ENCNTTITLE()       (function  in U.PRG)
261  *!               : M_ENCSAV           (procedure in DEMO.PRG)
262  *!               : M_ENCCLO           (procedure in DEMO.PRG)
263  *!               : M_ENCVALID         (procedure in DEMO.PRG)
264  *!               : WCONFIRM()         (function  in U.PRG)
265  *!
266  *!*****************************************************************
247     PROCEDURE m_resume
248
249     PRIVATE mevent
250     PRIVATE mdiagname
251     PRIVATE N
252     PRIVATE mcond
253     PRIVATE mqid
254     PRIVATE nc
255     PRIVATE mscreenname
256     PRIVATE mprdiagnum
257     PRIVATE mstatevname
258     DIMENSION mdiagname[ alen(mdiag) ]
259     DIMENSION mcond[ alen(mcondition) ]
```

```
260      DIMENSION mqid[ alen(mcondition) ]
261
262      event     = e_ok                && reset event flag
263
264      * set up working storage for the encounter screen
265      mdiagname = ""                  && differential diagnosis array
266      mcond     = ""                  && signs and symptoms array
267      mqid      = ""                  && signs and symptom ids
268      nc        = strcond(mcond, mqid, mcondition, msympval)
269      N         = strdiag(mdiagname, mdiag, mconf, cthreshold, "", is_conf)
270      mareaname = getexarea(marea)
271      mprdiagnam = diseasenam(mdiagnosis)
272      mstatename = mencstate
273
274      * edit the screen
275      mage       = ageat(mdateenc, mdatebirth)  && patient's age in years
276      =edsexage(mage, msex)
277      mevent     = encnttitle(msocsec, mlastname, mfirstname, msex, mage,;
278          mdateenc, mtimeenc, mareaname, mprovider,;
279          mprdiagnam, mdiagname, mcond, mtreatment, mstatename)
280
281      * close the screen
282
283      * respond to one of the following pushbutton events
284      DO CASE
285      CASE mevent = e_save
286          * SAVE button
287          =m_encsav()                  && save encounter
288          =m_encclo()                  && close encounter
289      CASE mevent = e_esc
290          * ESC button                 && abort encounter
291          IF m_encvalid()              && valid encounter, confirm...     &
=> & rwd07
292              IF wconfirm("DO you want to save the encounter first", ;
293                      "PATIENT ENCOUNTER") == e_ok                        && rwd
=> 07
294                  =m_encsav()              && save encounter               &
=> & rwd07
295              ENDIF                                                       &
=> & rwd07
296          ENDIF                                                           &
=> & rwd07
297          =m_encclo()                  && close encounter
298      CASE mevent = e_print
299          * PRINT button
300          =m_encclo()                  && close encounter
301      ENDCASE
302      RETURN
303
304      *-----------------------------------------------------
305      *
306      * menu service procedure - register patient
307      *
308      *-----------------------------------------------------
324  *!*****************************************************************
325  *!
326  *!      Procedure: M_REGPAT
327  *!
328  *!      Called by: MYHANDLER          (procedure in U.PRG)
329  *!
330  *!          Calls: GETCURPAT          (procedure in DEMO.PRG)
331  *!               : PATID()            (function  in U.PRG)
332  *!               : M_PATPUT           (procedure in DEMO.PRG)
333  *!               : M_PATADD           (procedure in DEMO.PRG)
334  *!               : M_ENCVALID         (procedure in DEMO.PRG)
335  *!               : SETCURPAT          (procedure in DEMO.PRG)
```



```
336  *!               : AGEAT              (procedure in DEMO.PRG)
337  *!
338  *!*****************************************************************
324      PROCEDURE m_regpat
325      PRIVATE allow_new
326      PRIVATE mevent
327      PRIVATE mpid, mssn, mlast, mfirst, mbirth  msx, mrate
328
329      allow_new = is_newpat              && true if new patients allowed
330
331      * get the current patient details
332      mpid   = 0
333      mssn   = ""
334      mlast  = ""
335      mfirst = ""
336      mbirth = CTOD("  /  /  ")                 *
337      msx    = "M"
338      mrate  = ""
339      =getcurpat(@mpid,@mssn,@mlast,@mfirst,@mbirth,@msx,@mrate)
340
341      * edit patient details on screen
342      mevent = patid(@mpid,@mssn,@mlast,@mfirst,@mbirth,@msx,@mrate)
343      IF mevent != e_esc
344          DO CASE
345          CASE mevent = e_save
346              * update existing patient record
347              mevent = m_patput(mpid, mssn, mlast, mfirst, mbirth, msx, mrate
=> )
348          CASE mevent = e_init
349              * register new patient record
350              mevent = m_patadd(mpid, mssn, mlast, mfirst, mbirth, msx, mrate
=> )
351          ENDCASE
352          IF mevent != e_esc
353              * select patient only if encounter not already open
354              IF .NOT. ( enc_open .AND. m_encvalid() )       && rwd07
355                  * set current patient in global memory
356                  =setcurpat(mpid,mssn,mlast,mfirst,mbirth,msx,mrate)
357                  * set details for current encounter
358                  mage  = ageat(mdateenc, mbirth)  && patient's age in years
359              ENDIF
360          ENDIF
361      ENDIF
362      RETURN
363
364      *-----------------------------------------------------
365      *
366      * menu service procedure - register patient
367      *
368      *-----------------------------------------------------
379  *!*****************************************************************
380  *!
381  *!      Procedure: M_PAT
382  *!
383  *!          Calls: OPENFILES()        (function  in U.PRG)
384  *!               : KEYLOG.PRG
385  *!               : WPAT               (procedure in DEMO.PRG)
386  *!               : FLUSHFILES()       (function  in U.PRG)
387  *!
388  *!*****************************************************************
379      PROCEDURE m_pat
380      =openfiles()
381      =keylog("Register Patient")
382      =wpat()                        && resume patient screen
383      =flushfiles()
384      RETURN
```

```
385
386     *-------------------------------------------------------
387     *
388     * menu service procedure - open encounter
389     *
390     *-------------------------------------------------------
403  *!*****************************************************************
404  *!
405  *!      Procedure: M_ENC
406  *!
407  *!      Called by: M_NEWENC          (procedure in DEMO.PRG)
408  *!               : M_TRAINING        (procedure in DEMO.PRG)
409  *!
410  *!          Calls: ENCOUNT           (procedure in DEMO.PRG)
411  *!               : M_RESUME          (procedure in DEMO.PRG)
412  *!               : POPMESSAGE()      (function  in U.PRG)
413  *!
414  *!*****************************************************************
403     PROCEDURE m_enc
404     * only allowed if encounter not open
405     IF !enc_open
406        event     = e_ok                 && reset event flag  && rwd07
407        IF encount() > 0
408           =m_resume()                    && resume encounter screen
409        ENDIF
410     ELSE
411        =popmessage("Encounter already open")
412     ENDIF
413     RETURN
414
415     *-------------------------------------------------------
416     *
417     * menu service procedure - select patient's complaint
418     *
419     *-------------------------------------------------------
431  *!*****************************************************************
432  *!
433  *!      Procedure: M_COMPL
434  *!
435  *!      Called by: _EDITCOMPL        (procedure in U.PRG)
436  *!
437  *!          Calls: GETDIAGAREA       (procedure in DEMO.PRG)
438  *!               : M_ENCRULE         (procedure in DEMO.PRG)
439  *!               : POPMESSAGE()      (function  in U.PRG)
440  *!
441  *!*****************************************************************
431     PROCEDURE m_compl
432     PRIVATE newarea
433     * only allowed if encounter not already open
434     IF mencstate = "N"
435        * must select patient first                    && rwd07
436        IF !EMPTY(mpatientid)                          && rwd07
437           newarea    = getdiagarea()
438           IF !EMPTY(newarea)
439              marea = newarea
440              =m_encrule()          && get applicable rulebase
441           ENDIF
442        ELSE                                           && rwd07
443           =popmessage("First select a patient")       && rwd07
444        ENDIF                                          && rwd07
445     ELSE
446        =popmessage("Pending encounter open")
447     ENDIF
448     RETURN
449
450     *-------------------------------------------------------
451     *
452     * menu service procedure - enter care provider's diagnosis
453     *
454     *-------------------------------------------------------
467  *!*****************************************************************
468  *!
469  *!      Procedure: M_DIAG
470  *!
471  *!      Called by: _EDITDIAG         (procedure in U.PRG)
472  *!
473  *!          Calls: ASKDIAG           (procedure in DEMO.PRG)
474  *!               : DISEASENAM()      (function  in U.PRG)
475  *!               : DISEASEID()       (function  in U.PRG)
476  *!               : POPMESSAGE()      (function  in U.PRG)
477  *!
478  *!*****************************************************************
467     PROCEDURE m_diag
468     PRIVATE newdiag
469     * only allowed after complaint has been entered     && rwd07
470     IF !EMPTY(marea)                                    && rwd07
471        * map id to external name
472        newdiag = askdiag( diseasenam( mdiagnosis ) )
473        IF !EMPTY( newdiag )
474           * map name back to id
475           mdiagnosis = diseaseid( newdiag )
476        ENDIF
477     ELSE                                                && rwd07
478        =popmessage("Complaint must be entered first")   && rwd07
479        CUROBJ = OBJNUM(mckcompl)
480        SHOW GETS
481     ENDIF                                               && rwd07
482     RETURN
483
484     *-------------------------------------------------------
485     *
486     * menu service procedure - provide diagnostic assistance
487     *
488     *-------------------------------------------------------
505  *!*****************************************************************
506  *!
507  *!      Procedure: M_ASSIST
508  *!
509  *!      Called by: BATCHTEST()       (function  in DEMO.PRG)
510  *!               : _EDITASSIS        (procedure in U.PRG)
511  *!
512  *!          Calls: POPUPSHOW()       (function  in U.PRG)
513  *!               : RESETDIAG         (procedure in DEMO.PRG)
514  *!               : TESTRULES         (procedure in DEMO.PRG)
515  *!               : EVALRULES         (procedure in DEMO.PRG)
516  *!               : MAKEDIAG          (procedure in DEMO.PRG)
517  *!               : POPMESSAGE()      (function  in U.PRG)
518  *!               : POPUPHIDE()       (function  in U.PRG)
519  *!
520  *!*****************************************************************
505     PROCEDURE m_assist
506     * only allowed after complaint has been entered
507     IF !EMPTY(marea)
508        * delay expected here ...
509        PRIVATE wpopup
510        wpopup = popupshow("Calculating...")                 && rwd07
511        event = e_ok
512        =resetdiag()            && reset diagnosis
513        =testrules(marea)       && test/evaluate data conditions
514        =evalrules(marea)       && inference on hist/phys data
515        IF event == e_ok                                     && rwd01
516           =makediag()             && make diagnosis         && rwd01
```

```
517         ELSE                                                    && rwd01
518             =popmessage("Insufficient data for diagnosis")      && rwd01
519             =resetdiag()                                        && rwd01
520         ENDIF                                                   && rwd01
521         =popuphide(wpopup)                                      && rwd07
522     ELSE
523         =popmessage("Complaint must be entered first")
524         =RELEASE("w_assist")
525          CUROBJ = OBJNUM(mckcompl)
526         SHOW GETS
527     ENDIF
528     RETURN
529
530     *-----------------------------------------------------
531     *
532     * menu service procedure - enter signs and symptoms
533     *
534     *-----------------------------------------------------
547 *!*****************************************************************
548 *!
549 *!      Procedure: M_HANDP
550 *!
551 *!      Called by: _EDITHANDP          (procedure in U.PRG)
552 *!
553 *!          Calls: POPUPSHOW()         (function  in U.PRG)
554 *!               : TESTRULES           (procedure in DEMO.PRG)
555 *!               : POPUPHIDE()         (function  in U.PRG)
556 *!               : POPMESSAGE()        (function  in U.PRG)
557 *!
558 *!*****************************************************************
547     PROCEDURE m_handp
548     * only allowed after complaint has been entered
549     IF !EMPTY(marea)
550         * gather signs and symptoms data
551         * delay expected here ...
552         PRIVATE wpopup
553         wpopup = popupshow("Checking...")                       && rwd07
554         event = e_ok
555         =testrules(marea)            && test/evaluate data conditions
556         =popuphide(wpopup)                                      && rwd07
557     ELSE
558         =popmessage("Complaint must be entered first")
559         =RELEASE("w_handp")
560          CUROBJ = OBJNUM(mckcompl)
561         SHOW GETS
562     ENDIF
563     RETURN
564
565     *-----------------------------------------------------
566     *
567     * menu service procedure - provide treatment recommendation
568     *
569     *-----------------------------------------------------
579 *!*****************************************************************
580 *!
581 *!      Procedure: M_TREAT
582 *!
583 *!      Called by: _EDITTREAT          (procedure in U.PRG)
584 *!
585 *!          Calls: POPMESSAGE()        (function  in U.PRG)
586 *!
587 *!*****************************************************************
579     PROCEDURE m_treat
580     PARAMETERS newtreat
581     * only allowed after diagnosis has been entered
582     IF !EMPTY(mdiagnosis)
583         mtreatment = newtreat
584          CUROBJ = OBJNUM(me_treat)
585     ELSE
586         =popmessage("Diagnosis must be entered first")
587         =RELEASE("w_treat")
588          CUROBJ = OBJNUM(mckdiag)
589     ENDIF
590     RETURN
591
592     *-----------------------------------------------------
593     *
594     * menu service procedure - redo qualifier
595     *
596     *-----------------------------------------------------
609 *!*****************************************************************
610 *!
611 *!      Procedure: M_REDOQ
612 *!
613 *!      Called by: _EDITCOMD           (procedure in U.PRG)
614 *!
615 *!          Calls: GETQ                (procedure in DEMO.PRG)
616 *!               : ERASEQ              (procedure in DEMO.PRG)
617 *!               : ASK                 (procedure in DEMO.PRG)
618 *!               : RESTOREQ            (procedure in DEMO.PRG)
619 *!
620 *!*****************************************************************
609     FUNCTION m_redoq
610     PARAMETERS mitem
611     * This procedure is triggered by the user selecting an item
612     * in the symptom list.  Wants to edit the value
613     * mitem is the selected item
614     * get its id from the mqid array stored when the screen is
615     * opened (see PROC m_resume)
616     PRIVATE mid, xstr, mresult, mprev
617     mresult = .f.
618     event = e_ok
619     IF mitem > 0
620         mid =  mqid[mitem]                  && object identifier
621         xstr = ""                           && object value
622         mprev = getq(mid)                   && get previous value
623         =eraseq(mid)                        && erase previous value
624         mresult = ask(mid, @xstr)           && re-ask object value
625         IF !mresult
626             =restoreq(mid, mprev)           && aborted, restore value
627         ENDIF
628     ENDIF
629     RETURN mresult
630
631     *-----------------------------------------------------
632     *
633     * select patient encounter
634     *
635     *-----------------------------------------------------
649 *!*****************************************************************
650 *!
651 *!      Procedure: ENCOUNT
652 *!
653 *!      Called by: M_ENC               (procedure in DEMO.PRG)
654 *!
655 *!          Calls: GETENCNT            (procedure in DEMO.PRG)
656 *!               : PATSSN()            (function  in U.PRG)
657 *!               : M_PATGET            (procedure in DEMO.PRG)
658 *!               : M_ENCGET            (procedure in DEMO.PRG)
659 *!               : M_ENCNEW            (procedure in DEMO.PRG)
660 *!
661 *!*****************************************************************
```

```
649     FUNCTION encount
650     PRIVATE mprevpat
651     mprevpat = mpatientid           && previous patient selected
652     mencountid = getencnt(@mpatientid, mdateenc)
653     DO CASE
654     CASE mencountid > 0
655         * pending encounter
656         IF mpatientid != mprevpat
657             * selected different patient
658             msocsec = patssn(mpatientid)
659             * get patient record
660             =m_patget(@mpatientid,msocsec,@mlastname,@mfirstname,@mdatebirt
=> h,@msex,@mraterank)
661         ENDIF
662         =m_encget()
663     CASE mencountid = 0
664         * new encounter
665         =m_encnew()
666     OTHERWISE
667         * aborted encounter
668         mencountid = 0
669     ENDCASE
670     RETURN mencountid
671
672
673     *-----------------------------------------------------
674     *
675     * encounter event procedures
676     *
677     *-----------------------------------------------------
678
686 *!*****************************************************************
687 *!
688 *!      Procedure: M_ENCCLO
689 *!
690 *!      Called by: M_RESUME            (procedure in DEMO.PRG)
691 *!
692 *!*****************************************************************
686     FUNCTION m_enccio
687     * close encounter without saving
688     enc_open = .F.
689     mencstate = " "
690     RETURN .T.
691
708 *!*****************************************************************
709 *!
710 *!      Procedure: M_ENCSAV
711 *!
712 *!      Called by: BATCHTEST()         (function  in DEMO.PRG)
713 *!               : M_RESUME            (procedure in DEMO.PRG)
714 *!               :
715 *!          Calls: POPMESSAGE()        (function  in U.PRG)
716 *!               : M_ENCVALID          (procedure in DEMO.PRG)
717 *!               : ADDENCNT()          (function  in U.PRG)
718 *!               : NEWENCNT()          (function  in U.PRG)
719 *!               : PUTENCNT()          (function  in U.PRG)
720 *!               : PUTSYMPTOM()        (function  in U.PRG)
721 *!               : PUTDIAGNOS()        (function  in U.PRG)
722 *!
723 *!*****************************************************************
708     FUNCTION m_encsav
709     * save encounter
710     IF mencstate = "C"
711         =popmessage("Encounter is closed")
712 <-------RETURN .F.
713     ENDIF
714     IF .NOT. ( enc_open .AND. m_encvalid() )
715         =popmessage("Encounter is empty, not saved")
716 <-------RETURN .F.
717     ENDIF
718     IF mencstate = "N"
719         * save new encounter
720         mencstate = "P"             && becomes pending
721         =addencnt()
722         =newencnt(mencountid, mpatientid, mdateenc, mtimeenc, ;
723             mprovider, mstation, mdept, marea, mdiagnosis, mtreatment, ;
724             mencstate)
725     ELSE
726         * save pending encounter
727         =putencnt(mencountid, mpatientid, mdateenc, mtimeenc,;
728             mprovider, mstation, mdept, marea, mdiagnosis, mtreatment, ;
729             mencstate)
730     ENDIF
731     * save symptoms/computer's diagnosis for new/pending encounter
732     =putsymptom(mencountid, mcondition, msympval)
733     =putdiagnos(mencountid, mdiag,mconf)
734     RETURN .T.
735
745 *!*****************************************************************
746 *!
747 *!      Procedure: M_ENCVALID
748 *!
749 *!      Called by: M_RESUME            (procedure in DEMO.PRG)
750 *!               : M_REGPAT            (procedure in DEMO.PRG)
751 *!               : M_ENCSAV            (procedure in DEMO.PRG)
752 *!
753 *!*****************************************************************
745     FUNCTION m_encvalid
746     * test validity of encounter
747     * encounter must have the following minimum attributes:
748     IF EMPTY(mencountid).OR. EMPTY(mpatientid).OR. EMPTY(marea)
749 <-------RETURN .F.      && encounter is missing some vital attributes
750     ENDIF
751     RETURN .T.          && encounter is valid
752
768 *!*****************************************************************
769 *!
770 *!      Procedure: M_ENCGET
771 *!
772 *!      Called by: BATCHTEST()         (function  in DEMO.PRG)
773 *!               : ENCOUNT             (procedure in DEMO.PRG)
774 *!
775 *!          Calls: POPUPSHOW()         (function  in U.PRG)
776 *!               : LODENC              (procedure in DEMO.PRG)
777 *!               : M_ENCRULE           (procedure in DEMO.PRG)
778 *!               : AGEAT               (procedure in DEMO.PRG)
779 *!               : EDSEXAGE            (procedure in DEMO.PRG)
780 *!               : POPUPHIDE()         (function  in U.PRG)
781 *!
782 *!*****************************************************************
768     FUNCTION m_encget
769     * resume previous encounter
770     * delay expected here ...
771     PRIVATE wpopup
772     wpopup = popupshow("Loading...")                    && rud07
773     enc_open = .T.
774     =lodenc(mencountid)
775     =m_encrule()                    && load rulebase
776     mage    = ageat(mdateenc, mdatebirth)  && patient's age in years
777     =edsexage(mage, msex)
=>                          && hll04
778     =popuphide(wpopup)                                  && rud07
```

```
780
794  *|*****************************************************************
795  *|
796  *|      Procedure: M_ENCNEW
797  *|
798  *|      Called by: ENCOUNT            (procedure in DEMO.PRG)
799  *|
800  *|          Calls: NEXTENCNT()        (function  in U.PRG)
801  *|               : AGEAT             (procedure in DEMO.PRG)
802  *|               : DELSYMPTOM        (procedure in DEMO.PRG)
803  *|               : DELDIAGNOS        (procedure in DEMO.PRG)
804  *|               : EDSEXAGE          (procedure in DEMO.PRG)
805  *|
806  *|*****************************************************************
794      FUNCTION m_encnew
795      * start a new blank encounter
796      enc_open = .T.
797      mencstate = "N"
798      mencountid = nextencnt()        && identifier for this encounter
799      mdateenc = DATE()               && today's date
800      mtimeenc = TIME()               && current time is start of encounter
801      marea      = 0                  && empty area
802      mdiagnosis = 0                  && empty provider's diagnosis
803      mtreatment = ""                 && empty treatment description
804      mage       = ageat(mdateenc, mdatebirth)  && patient's age in years
805      * clear the list of symptoms/diagnoses for new encounter
806      =delsymptom()
807      =deldiagnos()
808      =edsexage(mage,msex)
809      RETURN .T.
810
811      *-------------------------------------------------- -
812      * edit basic patient history details to working memory
813      * &&hll04
814      *-------------------------------------------------
826  *|*****************************************************************
827  *|
828  *|      Procedure: EDSEXAGE
829  *|
830  *|      Called by: M_RESUME           (procedure in DEMO.PRG)
831  *|               : M_ENCGET          (procedure in DEMO.PRG)
832  *|               : M_ENCNEW          (procedure in DEMO.PRG)
833  *|
834  *|          Calls: ADDHISTORY         (procedure in DEMO.PRG)
835  *|
836  *|*****************************************************************
826      FUNCTION edsexage
827      PARAMETER mage, msex
828      PRIVATE mstr, mval
829      DIMENSION mstr[2], mval[2]
830      mstr[1] = "AGE"
831      mstr[2] = "SEX"
832      mval[1] = mage
833      mval[2] = msex
834      =addhistory(mstr, mval)
835      RETURN .T.
836
848  *|*****************************************************************
849  *|
850  *|      Procedure: M_ENCRULE
851  *|
852  *|      Called by: BATCHTEST()        (function  in DEMO.PRG)
853  *|               : M_COMPL           (procedure in DEMO.PRG)
854  *|               : M_ENCGET          (procedure in DEMO.PRG)
855  *|
```



```
857  *|
858  *|*****************************************************************
848      FUNCTION m_encrule
849      * load rulebase
850      =lodrule(marea)
851      RETURN .T.
852
853
854      * *** Expert system inference engine ***
855
856      *-------------------------------------------------
857      *
858      * load a rulebase
859      *
860      *-------------------------------------------------
870  *|*****************************************************************
871  *|
872  *|      Procedure: LODRULE
873  *|
874  *|      Called by: M_ENCRULE          (procedure in DEMO.PRG)
875  *|
876  *|          Calls: GETRULEBASE()      (function  in U.PRG)
877  *|
878  *|*****************************************************************
870      FUNCTION lodrule
871      PARAMETERS marea
872      rulebase = getrulebase(marea, @cthreshold, @cthreshpro, @cthreshlik,
=> @nrules)
873      IF nrules > 0                                          && rwd05
874         * DIMENSION statevecf[nrules], statevecq[nrules], fire[nrules]  &&
=>  rwd07
875         DIMENSION fire[nrules]                              && rwd07
876         DIMENSION ruleaction[nrules]                        && rwd07
877      ENDIF                                                  && rwd05
878      statevecq = ""                  && state of qualifiers
879      statevecf = ""                  && state of confidence factors
880      fire = .F.                      && rule agenda
881      ruleaction = 0                  && rule clause
882      agenda = .F.                    && agenda empty        && rwd07
883      mrule = 1                       && initial rule selection
884      RETURN .T.
885
886      *-------------------------------------------------
887      *
888      * test a rule set
889      *
890      *-------------------------------------------------
901  *|*****************************************************************
902  *|
903  *|      Procedure: TESTRULES
904  *|
905  *|      Called by: M_ASSIST           (procedure in DEMO.PRG)
906  *|               : M_MANOP           (procedure in DEMO.PRG)
907  *|
908  *|          Calls: EVALCOND           (procedure in DEMO.PRG)
909  *|
910  *|*****************************************************************
901      FUNCTION testrules
902
903      PARAMETERS marea
904
905      PRIVATE mresult
906      PRIVATE msel
907      msel = SELECT()                         && save current database sel
=> ection
```

```
908
909     IF !agenda                                  && test for agenda complete
=>  && rwd07
910         SELECT rule                             && primary index is salie
=> nce
911         GOTO TOP                                && set filters
912
913         DO WHILE !EOF()                         && process the rulebase
914            * test rules in salience (priority) order
915            mrule = rule                         && select a rule
916            *IF is_rule                                  && rwd07
917            *   * display rule for debugging              && rwd07
918            *   *=viewrule(mrule)                         && rwd07
919            *ENDIF                                       && rwd07
920            SELECT premise                       && premise clauses for th
=> is rule
921
922            * test premise clauses of rule
923            * logical AND of all clauses
924            mresult = .T.                        && result of clause evalu
=> ation
925            SEEK rule->premise                           && rwd07
926            DO WHILE clause = rule->premise
927               IF !evalcond( OBJECT, ALLTRIM( STR(id) ), op, VAL )      &&
=> rwd07
928                  mresult = .F.                  && premise is false
929               EXIT
930               ENDIF
931               SKIP
932            ENDDO
933
934            IF event = e_esc
935               * abort flag has been set
936            EXIT
937            ELSE
938               fire[mrule]  =  mresult            && add rule to agenda
939               ruleaction[mrule] = rule->action   && keep rule clause
=> && rwd07
940            ENDIF
941            SELECT rule                          && select another rule
942            SKIP
943         ENDDO
944         IF event != e_esc                               && rwd07
945            agenda = .T.                                 && rwd07
946         ENDIF                                           && rwd07
947      ENDIF
948      SELECT (msel)
949      RETURN event
950
951      *-----------------------------------------------------
952      *
953      * evaluate rule agenda
954      *
955      *-----------------------------------------------------
965   *!*****************************************************************
966   *!
967   *!      Procedure: EVALRULES
968   *!
969   *!      Called by: M_ASSIST          (procedure in DEMO.PRG)
970   *!
971   *!          Calls: EVALCOND          (procedure in DEMO.PRG)
972   *!
973   *!*****************************************************************
965      FUNCTION evalrules
966
967      PARAMETERS mares
```

```
968
969      PRIVATE msel
970      msel = SELECT()                             && save current database sel
=> ection
971      PRIVATE maxrules                                        && rwd07
972      maxrules = ALEN(fire)                                   && rwd07
973      *SELECT rule                                            && rwd07
974      *SET ORDER TO 2                             && index on rule id  && rwd0
=> 7
975      SELECT action                               && execute rule actions  &&
=>  rwd07
976
977     FOR mrule = 1 TO maxrules                    && process the rule agenda
=> && rwd07
978        IF fire[mrule]                            && rule is eligible
979           *SEEK mrule                                          && rwd07
980           *SELECT action                         && execute rule actions  &&
=> rwd07
981           SEEK ruleaction[mrule]                                   && rwd07
982        DO WHILE clause = ruleaction[mrule]
983           =evalcond( OBJECT, ALLTRIM( STR(id) ),  op, VAL )     && rwd
=> 07
984           SKIP
985        ENDDO
986
987           *SELECT rule                                         && rwd07
988        IF event = e_esc
989           EXIT                                   && abort flag has been set
990        ENDIF
991     ENDIF
992     NEXT
993
994     *SELECT RULE                                            && rwd07
995     *SET ORDER TO 1                                         && rwd07
996     SELECT (msel)
997     RETURN event
998
999     *-----------------------------------------------------
1000    *
1001    * return object's data type
1002    *
1003    *-----------------------------------------------------
1009  *!*****************************************************************
1010  *!
1011  *!      Procedure: OBJTYPE
1012  *!
1013  *!*****************************************************************
1009     FUNCTION objtype
1010
1011     PARAMETERS mobject
1012     * !NOTE!
1013     * only dict "S" objects currently need a type
1014     * other objects, simply return a null type
1015     * Assumes object clause ==> dict database relation is active
1016     RETURN  IIF(mobject = "S", dict->datatype, "")
1017
1018     *-----------------------------------------------------
1019     *
1020     * evaluate an object clause
1021     *
1022     *-----------------------------------------------------
1036  *!*****************************************************************
1037  *!
1038  *!      Procedure: EVALCOND
1039  *!
1040  *!      Called by: TESTRULES         (procedure in DEMO.PRG)
```

```
1041  *!              : EVALRULES          (procedure in DEMO.PRG)
1042  *!              : INVERTCOND         (procedure in DEMO.PRG)
1043  *!
1044  *!         Calls: MULFACTOR          (procedure in DEMO.PRG)
1045  *!              : SETQ               (procedure in DEMO.PRG)
1046  *!              : ASK                (procedure in DEMO.PRG)
1047  *!
1048  *!*****************************************************************
1036      FUNCTION evalcond
1037
1038          PARAMETERS mobject, mid, mop, mval
1039          * mobject     object type descriptor
1040          * mid         object identifier
1041          * mop         operation code
1042          * mval        constant value
1043
1044          * Evaluate a clause of the form:
1045          *  <objectid> <op> <val>
1046          * where:
1047          *  <objectid> is the (composite) object identifier, consisting of
1048          *             object + id.  object is "S" or "D"
1049          *  <op>       is the operator:
1050          *                relational:   <  >  <=  >=  !=  ==
1051          *                assignment:   =
1052          *  <val>      constant value
1053
1054          * ! NOTE!
1055          * ! This version handles only two object types (symptom, disease)
1056          * ! An enhanced version should handle any object type
1057          * ! and consult the appropriate segment of working memory.
1058          * ! Assumes that symptoms/diseases have already been loaded into w
 => orking
1059          * ! memory prior to calling.
1060          * ! Requests value from user for uninstantiated symptom values.
1061
1062          PRIVATE mresult
1063          mresult = .F.                    && default boolean result
1064          IF mop = "= "
1065             * assignment operator
1066             DO CASE
1067             CASE mobject = "D"
1068                * disease type: apply confidence factor computation
1069                * !NOTE!
1070                * ! This version does only Bayesian (multiplicative)
1071                * ! confidence factors.  An enhanced version should do more
1072                * ! (eg fuzzy, additive, etc).  The method of
1073                * ! computation should be set as a parameter:
1074                * ! DO CASE
1075                * ! CASE cfmethod = "BAYES"
1076                =mulfactor( mid, VAL(mval) )          && rwd07
1077                * ! CASE cfmethod = "ADDITIVE"
1078                * ! =incr(mid, mval)
1079                * ! CASE cfmethod = "FUZZY"
1080                * ! =fuzfactor(mid, mval)
1081                * ! ENDCASE
1082             CASE mobject = "S"
1083                * symptom type: set qualifier value
1084                =setq( mid, TRIM(mval) )                && rwd07
1085             ENDCASE
1086          ELSE
1087             * relational operator
1088             PRIVATE xstr
1089             PRIVATE N                                  && rwd03
1090             PRIVATE mitem                              && rwd03
1091             PRIVATE mexact                             && rwd03
1092             PRIVATE mtype        &&  object data type  && rwd07
```



```
1093             DO CASE
1094             CASE mobject = "D"
1095                * disease object:
1096                * !NOTE! right now I don't know what to do with this!
1097                * It is not currently required in any of the knowledge
1098                * bases supported, but in general could be needed.
1099             CASE mobject = "S"
1100                * symptom object:
1101                xstr = ""                          && object value
1102                mresult = ask(mid, @xstr)          && get object value
1103                IF mresult
1104                   * object has a defined value
1105                   mexact = SET("EXACT")                          && rwd03
1106                   SET EXACT ON                                   && rwd03
1107                   mval = TRIM(mval)                              && rwd07
1108                   * Assumes clause ==> dict relation is set      && rwd07
1109                   mtype = dict->datatype                         && rwd07
1110                   DO CASE
1111                   CASE mtype = "N"                               && rwd03
1112                      mresult = EVALUATE( xstr + mop + mval )     && rwd03
1113                   CASE mtype = "L"  .OR. mtype = "E"             && rwd03
1114                      * Enumerated type or logical types          && rwd03
1115                      * quote operands
1116                      xstr = '"' + xstr + '"'
1117                      mval = '"' + mval + '"'
1118                      mresult = EVALUATE( xstr + mop + mval )     && rwd03
1119                   CASE mtype = "M"                               && rwd03
1120                      * Multiple choice type                      && rwd03
1121                      mresult = mval $ xstr                       && rwd03
1122                      IF mresult                                  && rwd03
1123                         * check exact match                      && rwd03
1124                         * extract names from qualifier           && rwd03
1125                         mresult = .F.                            && rwd03
1126                         DO WHILE LEN(xstr) > 0                   && rwd03
1127                            * find end of name item
1128                            N = AT(mconcat, xstr) - 1
1129                            IF N > 0                              && rwd03
1130                               * get the item
1131                               mitem   = LEFT(xstr, N)
1132                            ELSE                                  && rwd03
1133                               * take whole string
1134                               mitem = xstr
1135                            ENDIF                                 && rwd03
1136                            IF mitem = mval                       && rwd03
1137                               mresult = .T.                      && rwd03
1138                            EXIT                && exact match
1139                            ENDIF                                 && rwd03
1140                            IF N <= 0                             && rwd03
1141                            EXIT                && no exact match
1142                            ENDIF                                 && rwd03
1143                            * skip to next item
1144                            xstr = SUBSTR(xstr, N + 2 )
1145                         ENDDO                                    && rwd03
1146                      ENDIF
1147                   ENDCASE
1148                   IF mexact = "OFF"                              && rwd03
1149                      SET EXACT OFF                               && rwd03
1150                   ENDIF                                          && rwd03
1151                ENDIF
1152             ENDCASE
1153          ENDIF
1154          RETURN mresult
1155
1156       *-----------------------------------------------------------
1157       *
1158       * undo rules
```

```
1159        *
1160        *-------------------------------------------------------
1171  *!*****************************************************************
1172  *!
1173  *!      Procedure: UNDORULES
1174  *!
1175  *!      Called by: M_UNDORULE          (procedure in DEMO.PRG)
1176  *!
1177  *!          Calls: UNDOFACTOR          (procedure in DEMO.PRG)
1178  *!               : UNDOQUAL            (procedure in DEMO.PRG)
1179  *!
1180  *!*****************************************************************
1171      FUNCTION undorules
1172        * undo actions of rules that have fired
1173        * use the string "fire" to find previously active rules
1174        * fire consists of a string of 'T's and 'F's
1175        * backtrack through all rules
1176        PRIVATE i
1177        i = ALEN(fire)
1178        DO WHILE i > 0
1179          * rule is position in string
1180          IF fire[i]
1181            * this rule was active, undo possible actions
1182            =undofactor(statevecf[i])        && undo confidence factors
1183            =undoqual(statevecq[i])          && undo qualifiers
1184            fire[i] = .F.                    && remove from active agenda
1185          ENDIF
1186          i = i - 1
1187        ENDDO
1188        RETURN  .T.
1189
1190        *-------------------------------------------------------
1191        *
1192        * Undo factors from state vector
1193        *
1194        *-------------------------------------------------------
1204  *!*****************************************************************
1205  *!
1206  *!      Procedure: UNDOFACTOR
1207  *!
1208  *!      Called by: UNDORULES           (procedure in DEMO.PRG)
1209  *!
1210  *!          Calls: FACTORID            (procedure in DEMO. PG)
1211  *!
1212  *!*****************************************************************
1204      FUNCTION undofactor
1205
1206        PARAMETERS statestr
1207
1208        * this restores the system state to what it was BEFORE the
1209        * rule fired, that is a complete "roll-back"
1210        PRIVATE i, eolpos, arrowpos, mstr, mval
1211
1212        DO WHILE LEN(statestr) > 0
1213          * extract an item of the form:-    name <arrow> val <eol>
1214          arrowpos  =  AT(arrow, statestr)    && position of delimiters
1215          eolpos    =  AT(eol, statestr)
1216          IF arrowpos = 0
1217            EXIT
1218          ENDIF
1219
1220          * extract left hand side and right hand side
1221          mstr = LEFT(statestr,  arrowpos - 1)
1222          mval = SUBSTR(statestr,   arrowpos + LEN(arrow),   eolpos - (ar
 => rowpos + LEN(arrow)))
1223
1224          i = factorid(mstr)                    && lookup the name
1225          IF mval = uval
1226            * undefined value, delete element
1227            mconf[i] = 0
1228          ELSE
1229            * restore previous value
1230            mconf[i] = VAL(mval)
1231          ENDIF
1232          * remove last item processed
1233          statestr = IIF(eolpos < LEN(statestr), SUBSTR(statestr, eolpos+
 => 1), "")
1234        ENDDO
1235        RETURN .T.
1236
1237        *-------------------------------------------------------
1238        *
1239        * Undo qualifiers from working memory
1240        *
1241        *-------------------------------------------------------
1252  *!*****************************************************************
1253  *!
1254  *!      Procedure: UNDOQUAL
1255  *!
1256  *!      Called by: UNDORULES           (procedure in DEMO.PRG)
1257  *!               : POPQ                (procedure in DEMO.PRG)
1258  *!
1259  *!          Calls: QID                 (procedure in DEMO.PRG)
1260  *!
1261  *!*****************************************************************
1252      FUNCTION undoqual
1253        PARAMETERS statestr
1254        * this restores the system state to what it was BEFORE the
1255        * rule fired, that is a complete "roll-back"
1256        PRIVATE i, eolpos, arrowpos, mstr, mval
1257
1258        DO WHILE LEN(statestr) > 0
1259          * extract an item of the form:-    name <arrow> val <eol>
1260          arrowpos  =  AT(arrow, statestr)    && position of delimiters
1261          eolpos    =  AT(eol, statestr)
1262          IF arrowpos = 0
1263            EXIT
1264          ENDIF
1265
1266          * extract left hand side and right hand side
1267          mstr = LEFT(statestr,  arrowpos - 1)
1268          mval = SUBSTR(statestr,   arrowpos + LEN(arrow),   eolpos - (ar
 => rowpos + LEN(arrow)))
1269
1270          i = qid(mstr)                         && lookup the name
1271          IF mval = uval
1272            * undefined value, delete element
1273            msympval[i] = ""
1274          ELSE
1275            * restore previous value
1276            msympval[i] = mval
1277          ENDIF
1278          * remove last item processed
1279          statestr = IIF(eolpos < LEN(statestr), SUBSTR(statestr, eolpos+
 => 1), "")
1280        ENDDO
1281        RETURN .T.
1282
1283        *-------------------------------------------------------
1284        *
1285        * ask providers diagnosis
1286        *
```

```
1287        *-------------------------------------------------------
1299     *!*****************************************************************
1300     *!
1301     *!      Procedure: ASKDIAG
1302     *!
1303     *!      Called by: M_DIAG              (procedure in DEMO.PRG)
1304     *!
1305     *!          Calls: DISEASECNT()        (function  in U.PRG)
1306     *!               : DISNAMES()          (function  in U.PRG)
1307     *!               : GETDISLIST()        (function  in U PRG)
1308     *!
1309     *!*****************************************************************
1299        FUNCTION ask_diag
1300           PARAMETERS mdiagnosis
1301
1302           PRIVATE names, N, ndis, mchoice, i
1303           N = diseasecnt()
1304           mchoice = ""
1305           IF N > 0
1306              DIMENSION names[n]
1307              ndis = disnames(names,mares)                  && create list of
 => names
1308              i = IIF( ndis > 0, getdislist(names), 0 )
1309              IF i > 0
1310                 * extract disease from chosen entry
1311                 mchoice = names[i]
1312              ENDIF
1313           ENDIF
1314           RETURN  mchoice
1315
1316           *-------------------------------------------------------
1317           *
1318           * select diagnostic area
1319           *
1320           *-------------------------------------------------------
1333     *!*****************************************************************
1334     *!
1335     *!      Procedure: GETDIAGAREA
1336     *!
1337     *!      Called by: M_COMPL             (procedure in DEMO.PRG)
1338     *!               : M_DATA              (procedure in DEMO.PRG)
1339     *!
1340     *!          Calls: AREACOUNT()         (function  in U.PRG)
1341     *!               : AREANAMES()         (function  in U.PRG)
1342     *!               : GETAREALIST()       (function  in U.PRG)
1343     *!
1344     *!*****************************************************************
1333        FUNCTION getdiagarea
1334
1335           * get list of diagnostic area choices
1336           PRIVATE names, N
1337           PRIVATE i
1338           N = areacount()
1339           DIMENSION areas[n], names[n]
1340           =areanames(areas, names)                      && create list of names
1341           i =  getarealist(names)                       && select diagnostic ar
 => ea
1342           RETURN IIF(i > 0, areas[i], 0)                && id prev/new area
1343
1344           *-------------------------------------------------------
1345           *
1346           * get patient from file
1347           *
1348           *-------------------------------------------------------
1360     *!*****************************************************************
1361     *!
```



```
1362     *!      Procedure: GETPATIENT
1363     *!
1364     *!      Called by: M_PATLIST           (procedure in DEMO.PRG)
1365     *!
1366     *!          Calls: PATCOUNT()          (function  in U.PRG)
1367     *!               : PATNAMES()          (function  in U.PRG)
1368     *!               : GETPATLIST()        (function  in U.PRG)
1369     *!
1370     *!*****************************************************************
1360        FUNCTION getpatient
1361           PARAMETERS mkey
1362
1363           * mkey is the search key:
1364           * if it is string numeric  ==> social security number
1365           *                otherwise  ==> lastname
1366
1367           PRIVATE DELIM
1368           PRIVATE names, N, npats, mchoice, i
1369           N = patcount()
1370           IF N = 0
1371              mchoice = ""
1372           ELSE
1373              DIMENSION names[n]
1374              DELIM = "."                                   && menu delimiter
1375              npats = patnames(names, DELIM, mkey)          && create list of names
1376
1377              * Get users choice from menu of names:
1378              *   >0   selected patient ssn
1379              *  -1    user aborted
1380              i = IIF(npats > 0, getpatlist(mkey, names), 0)
1381              IF i > 0
1382                 * extract socsec number from chosen entry
1383                 mchoice = SUBSTR(names[i], 1, AT(DELIM, names[i]) - 1)
1384                 IF mchoice = "**********"
1385                    mchoice = ""
1386                 ENDIF
1387              ELSE
1388                 mchoice = ""
1389              ENDIF
1390           ENDIF
1391           RETURN  mchoice                               && selected patient or ""(a
 => bort)
1392
1393           *-------------------------------------------------------
1394           *
1395           * get previous patient encounter for processing/review
1396           *
1397           *-------------------------------------------------------
1410     *!*****************************************************************
1411     *!
1412     *!      Procedure: GETENCNT
1413     *!
1414     *!      Called by: ENCOUNT             (procedure in DEMO.PRG)
1415     *!
1416     *!          Calls: ENCCOUNT()          (function  in U.PRG)
1417     *!               : ENCNAMES()          (function  in U.PRG)
1418     *!               : GETENCLIST()        (function  in U.PRG)
1419     *!               : PATSSN()            (function  in U.PRG)
1420     *!
1421     *!*****************************************************************
1410        FUNCTION getencnt
1411           PARAMETERS mpatientid, mdate
1412
1413           PRIVATE N
1414           PRIVATE names
1415           PRIVATE mencntid
```

```
1416        mencntid = 0
1417        PRIVATE names, ids, patients, N, mchoice
1418        N = enccount()
1419        IF N > 0                          && number of encounters
1420           * user selects an encounter from list
1421           DIMENSION names[n], ids[n], patients[n]
1422
1423           N = encnames(names, ids, patients, mpatientid, mdate)        &&
 => create list of names
1424           IF N > 0
1425              DIMENSION names[n], ids[n], patients[n]
1426              mchoice = getenclist(names, IIF(TYPE("mpatientid") = "N", pa
 => tssn(mpatientid),""))
1427              IF mchoice > 0
1428                 mencntid = ids[mchoice]
1429                 mpatientid = patients[mchoice]
1430              ELSE
1431                 mencntid = mchoice       && 0 =>new, -1=>cancel
1432              ENDIF
1433           ENDIF
1434        ENDIF
1435        RETURN   mencntid
1436
1437        *------------------------------ --------------------
1438        *
1439        * initialize encounter details
1440        *
1441        *-------------------------------------------------------
1451     *!*****************************************************************
1452     *!
1453     *!       Function: INIENCOUNT()
1454     *!
1455     *!      Called by: DEMO.PRG
1456     *!
1457     *!          Calls: GETSTATION()       (function  in U.PRG)
1458     *!
1459     *!*****************************************************************
1451        FUNCTION iniencount
1452
1453        * global variables describing the current encounter
1454
1455        * details about the encounter
1456        PUBLIC mencountid                && unique identifier for the encount
 => er
1457        PUBLIC mdateenc                  && date of the encounter
1458        PUBLIC mtimeenc                  && time of the encounter (starting t
 => ime)
1459        PUBLIC mstation                  && name of duty station for the enco
 => unter
1460        PUBLIC mencstate                 && encounter state
1461        PUBLIC mreport                   && encounter sf600 report   &&hll06
1462
1463        * details about the patient
1464        PUBLIC mpatientid                && patient identifier code
1465        PUBLIC msocsec                   && social security number
1466        PUBLIC mlastname                 && patient's last name
1467        PUBLIC mfirstname                && patient's first name
1468        PUBLIC mmiddlename                   && patient's middle initial
1469        PUBLIC mdatebirth                && date of birth
1470        PUBLIC msex                      && sex
1471        PUBLIC mraterank                 && current rank or pay scale identif
 => ier
1472        PUBLIC mdept                     && patient's department affiliation
1473        PUBLIC mage                      && patient's age
1474
1475        * details about the diagnosis
1476        PUBLIC marea                     && the general disease area
1477        PUBLIC mdiagnosis                && medical provider's diagnosis (dis
 => ease name)
1478        PUBLIC mcomplaint                && patients complaint (reason for vi
 => sit)
1479        PUBLIC mtreatment                && treatment description
1480
1481        * initialize a null encounter record
1482        mencountid = 0
1483        mpatientid = 0
1484        mlastname  = ""
1485        mfirstname = ""
1486        mmiddlename= ""
1487        msocsec    = ""
1488        mdatebirth = CTOD("  /  /  ")
1489        msex       = "M"                 && are we sexist or what?
1490        mraterank  = ""
1491        mdateenc   = DATE()
1492        mtimeenc   = TIME()
1493        marea      = 0
1494        mdiagnosis = 0
1495        mcomplaint = ""
1496        mtreatment = ""
1497        mage       = ""
1498        mencstate  = "N"
1499
1500        * get station details
1501
1502        =getstation(@mstation, @mdept)
1503
1504        RETURN .T.
1505
1506        *-------------------------------------------------------
1507        *
1508        * load patient encounter record
1509        *
1510        *-------------------------------------------------------
1525     *!*****************************************************************
1526     *!
1527     *!      Procedure: LODENC
1528     *!
1529     *!      Called by: M_ENCGET          (procedure in DEMO.PRG)
1530     *!               : STENCOUNT         (procedure in U.PRG)
1531     *!
1532     *!          Calls: ENCGETINFO()      (function  in U.PRG)
1533     *!               : DELDIAGNOS        (procedure in DEMO.PRG)
1534     *!               : GETDIAGNOS()      (function  in U.PRG)
1535     *!               : DELSYMPTON        (procedure in DEMO.PRG)
1536     *!               : GETSYMPTON()      (function  in U.PRG)
1537     *!
1538     *!*****************************************************************
1525        FUNCTION lodenc
1526        PARAMETERS mencountid
1527
1528        * get details from (selected) encounter database record
1529        =encgetinfo(mencountid,mpatientid,@mdateenc,@mtimeenc,@mprovider,;
1530           @mstation,@mdept,@marea,@mdiagnosis,@mtreatment,@mencstate,@mr
 => epor)
1531
1532        * load list of computer's diagnoses for this encounter
1533        =deldiagnos()                    && erase current diagnoses
1534        =getdiagnos(mencountid)          && load from data base
1535
1536        * load the list of symptoms for this encounter
1537        =delsymptom()                    && erase all symptoms
1538        =getsymptom(mencountid)          && load from data base
```

```
1539  <------RETURN  mencountid
1540
1541        *-------------------------------------------------------
1542        *
1543        * Diagnostic assistance
1544        *
1545        *-------------------------------------------------------
1556  *!*****************************************************************
1557  *!
1558  *!      Procedure: M_CHANGE
1559  *!
1560  *!          Calls: CHANGEQ            (procedure in DEMO.PRG)
1561  *!               : POPQ               (procedure in DEMO.PRG)
1562  *!               : M_UNDORULE         (procedure in DEMO.PRG)
1563  *!               : POPMESSAGE()       (function  in U.PRG)
1564  *!
1565  *!*****************************************************************
1556      FUNCTION m_change
1557         event = e_redo      && redo current question
1558         IF  mqual > 1
1559            * change data, only if stack not empty
1560            * present choice of data to change
1561            mq = changeq(mqual-1)
1562            IF mq > 0
1563               * user made a selection
1564               =popq(mq)           && pop questions from stack
1565               event = e_goback
1566               =m_undorule()
1567            ENDIF
1568         ELSE
1569            =popmessage("This is first question")
1570         ENDIF
1571  <------RETURN .T.
1572
1582  *!*****************************************************************
1583  *!
1584  *!      Procedure: M_UNDO
1585  *!
1586  *!          Calls: LASTQ              (procedure in DEMO.PRG)
1587  *!               : POPQ               (procedure in DEMO.PRG)
1588  *!               : M_UNDORULE         (procedure in DEMO.PRG)
1589  *!
1590  *!*****************************************************************
1582      FUNCTION m_undo
1583         * go back (remove most recent question from stack)
1584         mq = lastq()             && index of last question
1585         =popq(mq)                && pop previous question from stack
1586         =m_undorule()
1587  <------RETURN .T.
1588
1599  *!*****************************************************************
1600  *!
1601  *!      Procedure: M_UNDORULE
1602  *!
1603  *!      Called by: M_CHANGE           (procedure in DEMO.PRG)
1604  *!               : M_UNDO             (procedure in DEMO.PRG)
1605  *!
1606  *!          Calls: UNDORULES          (procedure in DEMO.PRG)
1607  *!
1608  *!*****************************************************************
1599      FUNCTION m_undorule
1600         =undorules()             && restore data from rule state vector
1601         statevecq = ""           && reset state of qualifiers
1602         statevecf =  ""          && state of confidence factors
1603         fire = .F.               && reset rules fired
1604         mrule = 1                && begin at first rule
```

```
1605  <------RETURN .T.
1606
1614  *!*****************************************************************
1615  *!
1616  *!      Procedure: MYUNDO
1617  *!
1618  *!          Calls: UNDOMENU()         (function  in U.PRG)
1619  *!
1620  *!*****************************************************************
1614      FUNCTION myundo
1615         IF event = e_esc
1616            * user hit abort key during data entry
1617            event = undomenu()        && popup menu
1618         ENDIF
1619  <------RETURN event
1620
1621
1622        *-------------------------------------------------------
1623        *
1624        * make a diagnosis
1625        *
1626        *-------------------------------------------------------
1636  *!*****************************************************************
1637  *!
1638  *!      Procedure: MAKEDIAG
1639  *!
1640  *!      Called by: M_ASSIST           (procedure in DEMO.PRG)
1641  *!
1642  *!          Calls: NORMFACTOR         (procedure in DEMO.PRG)
1643  *!
1644  *!*****************************************************************
1636      FUNCTION makediag
1637         * !NOTE!
1638         * ! This version does only Bayesian (multiplicative)
1639         * ! confidence factors.  An enhanced version should do more
1640         * ! (eg fuzzy, additive, etc).  The method of
1641         * ! computation should be set as a parameter:
1642         * ! DO CASE
1643         * ! CASE cfmethod = "BAYES"
1644         * normalize probabilities already calculated
1645         =normfactor()
1646         * ! CASE cfmethod = "ADDITIVE"
1647         * ! CASE cfmethod = "FUZZY"
1648         * ! ENDCASE
1649  <------RETURN .T.
1650
1651        *-------------------------------------------------------
1652        *
1653        * get current patient info
1654        *
1655        *-------------------------------------------------------
1663  *!*****************************************************************
1664  *!
1665  *!      Procedure: GETCURPAT
1666  *!
1667  *!      Called by: M_REGPAT           (procedure in DEMO.PRG)
1668  *!
1669  *!*****************************************************************
1663      FUNCTION getcurpat
1664         PARAMETERS mpid, mssn, mlast, mfirst, mbirth, msx, mrate
1665
1666         * load patient details from global memory
1667         mpid   = mpatientid
1668         mssn   = msocsec
1669         mlast  = mlastname
1670         mfirst = mfirstname
```

```
1671        mbirth = mdatebirth
1672        msx    = msex
1673        mrate  = mraterank
1674   <----RETURN mpid
1675
1676        *-------------------------------------------------------
1677        *
1678        * set current patient info
1679        *
1680        *-------------------------------------------------------
1688   *|*****************************************************************
1689   *|
1690   *|      Procedure: SETCURPAT
1691   *|
1692   *|      Called by: M_REGPAT            (procedure in DEMO.PRG)
1693   *|
1694   *|*****************************************************************
1688      FUNCTION setcurpat
1689        PARAMETERS mpid, mssn, mlast, mfirst, mbirth, msx, mrate
1690
1691        * save patient details in global memory
1692        mpatientid  = mpid
1693        msocsec     = mssn
1694        mlastname   = mlast
1695        mfirstname  = mfirst
1696        mdatebirth  = mbirth
1697        msex        = msx
1698        mraterank   = mrate
1699   <----RETURN mpatientid
1700
1701        * patient screen event procedures
1702
1703        * Select patient from browse list
1714   *|*****************************************************************
1715   *|
1716   *|      Procedure: M_PATLIST
1717   *|
1718   *|      Called by: _Q700ZD65J()        (function  in W4.SPR)
1719   *|
1720   *|          Calls: GETPATIENT          (procedure in DEMO.PRG)
1721   *|               : M_PATGET            (procedure in DEMO.PRG)
1722   *|
1723   *|*****************************************************************
1714      PROCEDURE m_patlist
1715        PRIVATE newssn
1716        newssn = getpatient()
1717        IF !EMPTY(newssn)
1718           msocsec = newssn
1719           =m_patget(@mpatientid,msocsec,@mlastname,@mfirstname,@mdatebirt
 => h,@msex,@mraterank)
1720        ENDIF
1721   <----RETURN
1722
1723        * GET Existing patient record from database
1736   *|*****************************************************************
1737   *|
1738   *|      Procedure: M_PATGET
1739   *|
1740   *|      Called by: ENCOUNT             (procedure in DEMO.PRG)
1741   *|               : M_PATLIST          (procedure in DEMO.PRG)
1742   *|               : STENCOUNT          (procedure in U.PRG)
1743   *|
1744   *|          Calls: PATGETINFO()        (function  in U.PRG)
1745   *|               : ERRMESSAGE()       (function  in U.PRG)
1746   *|
1747   *|*****************************************************************
```

```
1736      FUNCTION m_patget
1737        PARAMETERS mpid, mssn, mlast, mfirst, mbirth, msx, mrate
1738        PRIVATE mevent
1739        mevent = e_ok
1740        IF !patgetinfo(@mpid,mssn,@mlast,@mfirst,@mbirth,@msx,@mrate)
1741           * not found:- this should never happen!
1742           =errmessage("Patient record not found [" + mssn + "]")
1743           mevent = e_esc
1744        ENDIF
1745   <----RETURN mevent
1746
1747        * PUT updated patient record to database
1759   *|*****************************************************************
1760   *|
1761   *|      Procedure: M_PATPUT
1762   *|
1763   *|      Called by: M_REGPAT            (procedure in DEMO.PRG)
1764   *|
1765   *|          Calls: POPMESSAGE()        (function  in U.PRG)
1766   *|               : ISVALIDSSN()       (function  in U.PRG)
1767   *|               : PATPUTINFO()       (function  in U.PRG)
1768   *|
1769   *|*****************************************************************
1759      FUNCTION m_patput
1760        PARAMETERS mpid, mssn, mlast, mfirst, mbirth, msx, mrate
1761        PRIVATE mevent
1762        mevent = e_ok
1763        IF !!IF(TYPE("allow_new") = "U", .F., allow_new)
1764           * new patient not authorized!
1765           =popmessage("New patient not allowed", 5)
1766           mevent = e_esc
1767        ELSE
1768           * ensure patient exists
1769           IF !isvalidssn(mssn)
1770              =popmessage("Patient not on file [" + mssn + "]")
1771              mevent = e_esc
1772           ELSE
1773              * update record for patient
1774              =patputinfo(mpid,mssn,mlast,mfirst,mbirth,msx,mrate)
1775           ENDIF
1776        ENDIF
1777   <----RETURN mevent
1778
1779        * PUT New patient record
1792   *|*****************************************************************
1793   *|
1794   *|      Procedure: M_PATADD
1795   *|
1796   *|      Called by: M_REGPAT            (procedure in DEMO.PRG)
1797   *|
1798   *|          Calls: POPMESSAGE()        (function  in U.PRG)
1799   *|               : ISVALIDSSN()       (function  in U.PRG)
1800   *|               : ADDPAT()           (function  in U.PRG)
1801   *|               : NEWPATIENT()       (function  in U.PRG)
1802   *|
1803   *|*****************************************************************
1792      FUNCTION m_patadd
1793        PARAMETERS mpid, mssn, mlast, mfirst, mbirth, msx, mrate
1794        PRIVATE mevent
1795        mevent = e_ok
1796        IF !!IF(TYPE("allow_new") = "U", .F., allow_new)
1797           * new patient not authorized!
1798           =popmessage("New patient not allowed", 5)
1799           mevent = e_esc
1800        ELSE
1801           * ensure patient unique
```

```
1802        IF isvalidssn(mssn)
1803           =popmessage("Patient already on file [" + mssn + "]")
1804           mevent = e_esc
1805        ELSE
1806           * create record for new patient
1807           mpid = addpat()                    && add patient id
1808           =newpatient(mpid,mssn,mlast,mfirst,mbirth,msx,mrate)
1809        ENDIF
1810     ENDIF
1811     RETURN mevent
1812
1813     *-------------------------------------------------------
1814     *
1815     * compute patient's age on given date
1816     *
1817     *-------------------------------------------------------
1829  *!*****************************************************************
1830  *!
1831  *!      Procedure: AGEAT
1832  *!
1833  *!      Called by: M_RESUME          (procedure in DEMO.PRG)
1834  *!               : M_REGPAT          (procedure in DEMO.PRG)
1835  *!               : M_ENCGET          (procedure in DEMO.PRG)
1836  *!               : M_ENCNEW          (procedure in DEMO.PRG)
1837  *!               : WPAT              (procedure in DEMO.PRG)
1838  *!
1839  *!*****************************************************************
1829     FUNCTIUN ageat
1830        PARAMETERS mdate, mdatebirth
1831
1832        PRIVATE mage
1833        mage = YEAR(mdate) - YEAR(mdatebirth)
1834        IF MONTH(mdate) < MONTH(mdatebirth)
1835           mage = mage - 1
1836        ELSE
1837           IF MONTH(mdate) = MONTH(mdatebirth) .AND.;
1838                 DAY(mdate) < DAY(mdatebirth)
1839              mage = mage - 1
1840           ENDIF
1841        ENDIF
1842        IF mage < 16 OR mage > 79
1843           RETURN ""
1844        ENDIF
1845     RETURN  LTRIM(STR(mage))
1846
1847     *-------------------------------------------------------
1848     *
1849     *  add a diagnosis data element to the memory array
1850     *
1851     *-------------------------------------------------------
1861  *!*****************************************************************
1862  *!
1863  *!      Procedure: ADDDIAGNOS
1864  *!
1865  *!      Called by: GETDIAGNOS()      (function  in U.PRG)
1866  *!
1867  *!          Calls: DEFFACTOR         (procedure in DEMO.PRG)
1868  *!
1869  *!*****************************************************************
1861     FUNCTION adddiagnos
1862        PARAMETERS dname, dval
1863
1864        PRIVATE mv, mn
1865        mn = ALLTRIM(dname)
1866        mv = dval
1867     RETURN  deffactor(mn, mv)          && define data element in worki
```



```
=> ng memory
1868
1869     *-------------------------------------------------------
1870     *
1871     *  add history data elements to the memory array
1872     *
1873     *-------------------------------------------------------
1885  *!*****************************************************************
1886  *!
1887  *!      Procedure: ADDHISTORY
1888  *!
1889  *!      Called by: EDSEXAGE          (procedure in DEMO.PRG)
1890  *!
1891  *!          Calls: DICTQNAME()       (function  in U.PRG)
1892  *!               : ERASEQ            (procedure in DEMO.PRG)
1893  *!               : ADDSYMPTOM        (procedure in DEMO.PRG)
1894  *!
1895  *!*****************************************************************
1885     FUNCTION addhistory
1886        PARAMETERS histname, histval
1887        EXTERNAL ARRAY histname, histval                    && rwd07
1888
1889        PRIVATE i, mid, mstr
1890
1891        FOR i = 1 TO ALEN(histname)
1892           mid = dictqname(histname[i])
1893           mstr = histval[i]
1894           IF !EMPTY(mid)
1895              =eraseq(mid)
1896              IF !EMPTY(mstr)
1897                 =addsymptom(mid, mstr)    && hll04
1898              ENDIF
=>  && hll04
1899           ENDIF
1900        NEXT
1901
1902     RETURN .T.
1903
1904     *-------------------------------------------------------
1905     *
1906     *  add data element to the memory array
1907     *
1908     *-------------------------------------------------------
1919  *!*****************************************************************
1920  *!
1921  *!      Procedure: ADDSYMPTOM
1922  *!
1923  *!      Called by: ADDHISTORY        (procedure in DEMO.PRG)
1924  *!               : GETSYMPTOM()      (function  in U.PRG)
1925  *!
1926  *!          Calls: DEFQ              (procedure in DEMO.PRG)
1927  *!
1928  *!*****************************************************************
1919     FUNCTION addsymptom
1920        PARAMETERS mname, mval
1921
1922        PRIVATE mv, mn
1923        mn = ALLTRIM(mname)
1924        mv = ALLTRIM(mval)
1925        =defq(mn, mv)                    && define data element in working
=>  memory
1926     RETURN mv
1927
1928     *-------------------------------------------------------
1929     *
1930     * delete all diagnoses from memory
```

```
1931      *
1932      *-------------------------------------------------------
1945 *!*****************************************************************
1946 *!
1947 *!      Procedure: DELDIAGNOS
1948 *!
1949 *!      Called by: INIDIAGNOS()        (function  in DEMO.PRG)
1950 *!               : BATCHTEST()         (function  in DEMO.PRG)
1951 *!               : M_ENCNEW            (procedure in DEMO.PRG)
1952 *!               : LODENC              (procedure in DEMO.PRG)
1953 *!
1954 *!          Calls: RESETDIAG           (procedure in DEMO.PRG)
1955 *!
1956 *!*****************************************************************
1945      FUNCTION deldiagnos
1946         mdiag = ""                 && erase names
1947         =resetdiag()               && erase values
1948         nextdiag = 1               && next available slot in list
1949 <-------RETURN ""
1950
1959 *!*****************************************************************
1960 *!
1961 *!      Procedure: RESETDIAG
1962 *!
1963 *!      Called by: M_ASSIST            (procedure in DEMO.PRG)
1964 *!               : DELDIAGNOS          (procedure in DEMO.PRG)
1965 *!
1966 *!*****************************************************************
1959      FUNCTION resetdiag
1960         mconf= 0                   && reset diagnosis values
1961 <-------RETURN ""
1962
1963         *-------------------------------------------------------
1964         *
1965         * delete all symptoms from memory
1966         *
1967         *-------------------------------------------------------
1978 *!*****************************************************************
1979 *!
1980 *!      Procedure: DELSYMPTOM
1981 *!
1982 *!      Called by: INISYMPTOM()        (function  in DEMO.PRG)
1983 *!               : BATCHTEST()         (function  in DEMO.PRG)
1984 *!               : M_ENCNEW            (procedure in DEMO.PRG)
1985 *!               : LODENC              (procedure in DEMO.PRG)
1986 *!
1987 *!*****************************************************************
1978      FUNCTION delsymptom
1979         mcondition = ""            && erase names
1980         msympval   = ""            && erase values
1981         qualstack  = ""            && erase qualifier stack
1982         redoval    = ""            && redo stack for qualifiers
1983         mqual      = 1             && next slot in qualifier stack
1984         nextqual   = 1             && next slot in list of conditions
1985 <-------RETURN nextqual
1986
1987         *-------------------------------------------------------
1988         *
1989         * Increment confidence factor
1990         *
1991         *-------------------------------------------------------
2002 *!*****************************************************************
2003 *!
2004 *!      Procedure: INCR
2005 *!
2006 *!      Called by: DECR                (procedure in DEMO.PRG)
2007 *!
2008 *!          Calls: GETFACTOR           (procedure in DEMO.PRG)
2009 *!               : SETFACTOR           (procedure in DEMO.PRG)
2010 *!
2011 *!*****************************************************************
2002      FUNCTION incr
2003         PARAMETERS mstr, mincr
2004         PRIVATE N
2005         N = getfactor(mstr)
2006 <-------RETURN setfactor(mstr, N + mincr)
2007
2008         *-------------------------------------------------------
2009         *
2010         * Decrement confidence factor
2011         *
2012         *-------------------------------------------------------
2020 *!*****************************************************************
2021 *!
2022 *!      Procedure: DECR
2023 *!
2024 *!          Calls: INCR                (procedure in DEMO.PRG)
2025 *!
2026 *!*****************************************************************
2020      FUNCTION decr
2021         PARAMETERS mstr, mincr
2022 <-------RETURN incr(mstr, -mincr)
2023
2024         *-------------------------------------------------------
2025         *
2026         * Multiply confidence factor
2027         *
2028         *-------------------------------------------------------
2038 *!*****************************************************************
2039 *!
2040 *!      Procedure: MULFACTOR
2041 *!
2042 *!      Called by: EVALCOND            (procedure in DEMO.PRG)
2043 *!
2044 *!          Calls: FACTORID            (procedure in DEMO.PRG)
2045 *!
2046 *!*****************************************************************
2038      FUNCTION mulfactor
2039
2040         PARAMETERS mstr, mval
2041
2042         PRIVATE N
2043         PRIVATE i                                          && rud07
2044         i  =  factorid(mstr)     && search for name of factor && rud07
2045         N  =  mconf[i]                                     && rud07
2046         N  =  IIF( N i= 0,  N * mval,  mval )
2047         mconf[i] = N                                       && rud07
2048 <-------RETURN N                                           && rud07
2049
2050         *-------------------------------------------------------
2051         *
2052         * Sum confidence factors
2053         *
2054         *-------------------------------------------------------
2062 *!*****************************************************************
2063 *!
2064 *!      Procedure: SUMFACTOR
2065 *!
2066 *!      Called by: NORMFACTOR          (procedure in DEMO.PRG)
2067 *!
2068 *!*****************************************************************
2062      FUNCTION sumfactor
```

```
2063
2064          PRIVATE N
2065          PRIVATE i
2066          N = 0
2067
2068          FOR i = 1  TO  nextdiag - 1
2069             N  =  N + mconf[i]
2070          NEXT
2071
2072          RETURN N
2073
2074          *-----------------------------------------------------
2075          *
2076          * Normalize confidence factors
2077          *
2078          *-----------------------------------------------------
2088 *!*****************************************************************
2089 *!
2090 *!      Procedure: NORMFACTOR
2091 *!
2092 *!      Called by: MAKEDIAG            (procedure in DEMO.PRG)
2093 *!
2094 *!          Calls: SUMFACTOR           (procedure in DEMO.PRG)
2095 *!
2096 *!*****************************************************************
2088          FUNCTION normfactor
2089
2090          PRIVATE N
2091          PRIVATE i
2092          N = sumfactor()
2093
2094          FOR i = 1  TO  nextdiag - 1
2095             mconf[i]  =  mconf[i] * 100 / N
2096          NEXT
2097
2098          RETURN N
2099
2100          *-----------------------------------------------------
2101          *
2102          * Get confidence factor
2103          *
2104          *-----------------------------------------------------
2114 *!*****************************************************************
2115 *!
2116 *!      Procedure: GETFACTOR
2117 *!
2118 *!      Called by: INCR                (procedure in DEMO.PRG)
2119 *!
2120 *!          Calls: FACTORID            (procedure in DEMO.PRG)
2121 *!
2122 *!*****************************************************************
2114          FUNCTION getfactor
2115             PARAMETERS mstr
2116             PRIVATE i, N
2117             i = factorid(mstr)        && search for name of factor
2118             N = IIF(TYPE("mconf[i]")="U", 0, mconf[i])
2119             RETURN N
2120
2121             *-----------------------------------------------------
2122             *
2123             * Set confidence factor
2124             *
2125             *-----------------------------------------------------
2135 *!*****************************************************************
2136 *!
2137 *!      Procedure: SETFACTOR
```

```
2138 *!
2139 *!      Called by: INCR                (procedure in DEMO.PRG)
2140 *!
2141 *!          Calls: DEFFACTOR           (procedure in DEMO.PRG)
2142 *!
2143 *!*****************************************************************
2135          FUNCTION setfactor
2136             PARAMETERS mstr, mval
2137             *=addfactor(mstr)         && add factor to state vector
 => && rwd07
2138             RETURN deffactor(mstr, mval)
2139
2140             *-----------------------------------------------------
2141             *
2142             * Define Confidence Factor
2143             *
2144             *-----------------------------------------------------
2155 *!*****************************************************************
2156 *!
2157 *!      Procedure: DEFFACTOR
2158 *!
2159 *!      Called by: ADDDIAGNOS          (procedure in DEMO.PRG)
2160 *!               : SETFACTOR           (procedure in DEMO.PRG)
2161 *!
2162 *!          Calls: FACTORID            (procedure in DEMO.PRG)
2163 *!
2164 *!*****************************************************************
2155          FUNCTION deffactor
2156             PARAMETERS mstr, mval
2157             PRIVATE i
2158             i = factorid(mstr)        && search for name of factor
2159             mconf[i] = mval
2160             RETURN mconf[i]
2161
2162             *-----------------------------------------------------
2163             *
2164             * Identify diagnostic factor
2165             *
2166             *-----------------------------------------------------
2180 *!*****************************************************************
2181 *!
2182 *!      Procedure: FACTORID
2183 *!
2184 *!      Called by: UNDOFACTOR          (procedure in DEMO.PRG)
2185 *!               : MULFACTOR           (procedure in DEMO.PRG)
2186 *!               : GETFACTOR           (procedure in DEMO.PRG)
2187 *!               : DEFFACTOR           (procedure in DEMO.PRG)
2188 *!               : ADDFACTOR           (procedure in DEMO.PRG)
2189 *!
2190 *!          Calls: ERRMESSAGE()        (function  in U.PRG)
2191 *!
2192 *!*****************************************************************
2180          FUNCTION factorid
2181             PARAMETERS mstr
2182             PRIVATE i
2183             PRIVATE mexact
2184             mexact = SET("EXACT")
2185             SET EXACT ON
2186             i = ASCAN(mdiag,mstr)     && search for name of factor
2187             IF i = 0
2188                * factor does not exist yet, define its name
2189                IF nextdiag >= ndiseases
2190                   * no more free slots
2191                   =errmessage("*** Warning: factor overflow ***")
2192                   i = 1
2193                ELSE
```

```
2194                 i = nextdiag          && next available position
2195                 =AINS(mdiag, i)
2196                 =AINS(mconf, i)
2197                 mdiag[i] = mstr
2198                 mconf[i] = 0
2199                 nextdiag = nextdiag + 1
2200              ENDIF
2201           ENDIF
2202           IF mexact = "OFF"
2203              SET EXACT OFF
2204           ENDIF
2205           RETURN i
2206
2207           *-----------------------------------------------------
2208           *
2209           * Add factor to state vector
2210           *
2211           *-----------------------------------------------------
2219 *!*****************************************************************
2220 *!
2221 *!      Procedure: ADDFACTOR
2222 *!
2223 *!          Calls: FACTORID          (procedure in DEMO.PRG)
2224 *!
2225 *!*****************************************************************
2219           FUNCTION addfactor
2220              PARAMETERS mstr
2221              PRIVATE i, savef
2222              i = factorid(mstr)       && get identifier
2223              savef = mstr + arrow + IIF( mconf[i] = 0,  uval,  STR(mco
 => nf[i]) )
2224              IF  TYPE("statevecf[mrule]") = "U"
2225                 statevecf[mrule] = ""
2226              ENDIF
2227              statevecf[mrule] = statevecf[mrule] + savef + eol
2228              RETURN .T.
2229
2230              *-----------------------------------------------------
2231              *
2232              * initialize array of diagnoses
2233              *
2234              *-----------------------------------------------------
2245 *!*****************************************************************
2246 *!
2247 *!      Function: INIDIAGNOS()
2248 *!
2249 *!      Called by: DEMO.PRG
2250 *!
2251 *!          Calls: DISEASECNT()      (function  in U.PRG)
2252 *!                 : DELDIAGNOS      (procedure in DEMO.PRG)
2253 *!
2254 *!*****************************************************************
2245           FUNCTION inidiagnos
2246              PUBLIC  ndiseases                    && maximum pos
 => sible diseases
2247              ndiseases = diseasecnt()             && number of n
 => ames in dictionary
2248
2249              * create array of diagnoses for current encounter
2250              PUBLIC mdiag[ndiseases]              && diagnoses
 => names
2251              PUBLIC mconf[ndiseases]              && confidence
 =>  factors
2252              PUBLIC nextdiag                      && next slot
 => in array to fill
2253              =deldiagnos()
2254           RETURN ndiseases
2255
2256           *-----------------------------------------------------
2257           *
2258           * initialize array of symptoms
2259           *
2260           *-----------------------------------------------------
2271 *!*****************************************************************
2272 *!
2273 *!      Function: INISYMPTOM()
2274 *!
2275 *!      Called by: DEMO.PRG
2276 *!
2277 *!          Calls: DICTCOUNT()       (function  in U.PRG)
2278 *!                 : DELSYMPTOM      (procedure in DEMO.PRG)
2279 *!
2280 *!*****************************************************************
2271           FUNCTION inisymptom
2272              PUBLIC nconditions                   && maximum possi
 => ble conditions
2273              nconditions= dictcount()             && number of nam
 => es in dictionary
2274
2275              * create array of symptoms/conditions for current encount
 => er
2276              PUBLIC mcondition[nconditions]       && list of names
 =>  and ...
2277              PUBLIC msympval[nconditions]         && corresponding
 =>  values
2278              PUBLIC qualstack[nconditions]        && stack of qual
 => ifiers asked
2279              PUBLIC qualval[nconditions]          && corresponding
 =>  values ...
2280              PUBLIC redoval[nconditions]          && qualifier red
 => o values
2281              PUBLIC mqual                         && qualifier cur
 => rently being asked
2282              PUBLIC nextqual                      && position to i
 => nsert next qualifier
2283              =delsymptom()
2284              RETURN nconditions
2285
2286              *-----------------------------------------------------
2287              *
2288              * Lookup Qualifier
2289              *
2290              *-----------------------------------------------------
2301 *!*****************************************************************
2302 *!
2303 *!      Procedure: LOOKUPQ
2304 *!
2305 *!      Called by: CHANGEQ       (procedure in DEMO.PRG)
2306 *!                 : LASTQ       (procedure in DEMO.PRG)
2307 *!
2308 *!          Calls: DICTQ()       (function  in U.PRG)
2309 *!
2310 *!*****************************************************************
2301           FUNCTION lookupq
2302              PARAMETERS mid
2303              PRIVATE mq, mtype, maskable
2304              * look up identifier in dictionary
2305              * return associated question text
2306              mq = ""
2307              maskable = .f.
2308              mtype = dictq(mid, @maskable, @mq)
2309              IF !EMPTY(mtype)
```

```
2310                IF mtype = "X" .OR. lmaskable
2311                   * prepend meta symbol
2312                   mq = "$" + mq
2313                ENDIF
2314             ENDIF
2315          <------------RETURN mq
2316
2317
2327       *!*****************************************************************
2328       *!
2329       *!      Procedure: LOOKUPNAM
2330       *!
2331       *!      Called by: STRCOND            (procedure in DEMO.PRG)
2332       *!
2333       *!          Calls: DICTQ()            (function  in U.PRG)
2334       *!
2335       *!*****************************************************************
2327             FUNCTION lookupnam
2328                PARAMETERS mid
2329                PRIVATE mq, mtype, maskable, mname
2330                * look up identifier in dictionary
2331                * return associated name
2332                mq = ""
2333                mname = ""
2334                maskable = .F.
2335                mtype = dictq(mid, @maskable, @mq, @mname)
2336             IF !EMPTY(mtype)
2337                IF mtype = "X" .OR. lmaskable
2338                   * prepend meta symbol
2339                   mname = "$" + mname
2340                ENDIF
2341             ENDIF
2342          <------------RETURN mname
2343
2344                *-----------------------------------------------------
2345                *
2346                * ask user for value of qualifier
2347                *
2348                *-----------------------------------------------------
2359       *!*****************************************************************
2360       *!
2361       *!      Procedure: ASK
2362       *!
2363       *!      Called by: M_REDOQ            (procedure in DEMO.PRG)
2364       *!               : EVALCOND           (procedure in DEMO.PRG)
2365       *!
2366       *!          Calls: EVALQ              (procedure in DEMO.PRG)
2367       *!*****************************************************************
2359             FUNCTION ask
2360                PARAMETERS mstr, mq
2361                * mstr is name of string qualifier
2362                * mq is previous/new value of qualifier
2363                * forces the question to be asked and the qualifier to be
2364                * instantiated.
2365                PRIVATE mresult
2366             IF event = e_esc
2367                * Note! the flag, event is set to e_esc by user hittin
 => g
2368                * the abort key during data entry, but a rule may be i
 => n the
2369                * middle of a complex expression evaluation. Therefore
 => , this
2370                * code prevents any more questions until the end of th
 => e
2371                * rule.
```

```
2372                mresult = .F.
2373             ELSE
2374                mq = evalq(mstr)                    && get value of q
 => ualifier
2375                * ensure value is actually valid:
2376                mresult  =  IIF(event = e_esc, .F., .T.)
2377             ENDIF
2378          <------------RETURN mresult
2379
2380             *-----------------------------------------------------
2381             *
2382             * test qualifier mstr for occurrence of mval
2383             *
2384             *-----------------------------------------------------
2392       *!*****************************************************************
2393       *!
2394       *!      Procedure: ISQ
2395       *!
2396       *!          Calls: EVALQ              (procedure in DEMO.PRG)
2397       *!
2398       *!*****************************************************************
2392             FUNCTION isq
2393                PARAMETERS mstr, mval
2394                * mstr is name of string qualifier
2395                * mval is string value to test
2396                PRIVATE mq, mresult
2397                PRIVATE   N, mitem
2398             IF event = e_esc
2399                * Note! the flag, event is set to e_esc by user hittin
 => g
2400                * the abort key during data entry, but a rule may be i
 => n the
2401                * middle of a complex expression evaluation. Therefore
 => , this
2402                * code prevents any more questions until the end of th
 => e
2403                * rule.
2404                mresult = .F.
2405             ELSE
2406                mq = evalq(mstr)                    && get value of q
 => ualifier
2407                mresult = mval $ mq                 && test occurrenc
 => e
2408             ENDIF
2409             IF mresult
2410                * partial match found, look for exact item
2411                mresult = .F.
2412                * extract names from qualifier
2413                DO WHILE LEN(mq) > 0
2414                   N = AT(mconcat, mq) - 1          && find end of name
 => item
2415                   IF N > 0
2416                      mitem   = LEFT(mq,N)          && get the item
2417                   ELSE
2418                      mitem = mq                    && take whole string
2419                   ENDIF
2420                   IF mitem = mval
2421                      mresult = .T.
2422                   EXIT                             && exact match
2423                   ENDIF
2424                   IF N <= 0
2425                   EXIT                             && no exact match fo
 => und
2426                   ENDIF
2427                   mq      = SUBSTR(mq,N+2)         && skip to next item
2428                ENDDO
```

```
2429          ENDIF
2430          RETURN mresult
2431
2432          *-----------------------------------------------------
2433          *
2434          * Get numeric value of qualifier
2435          *
2436          *-----------------------------------------------------
2444  *!*****************************************************************
2445  *!
2446  *!      Procedure: VALQ
2447  *!
2448  *!          Calls: EVALQ              (procedure in DEMO.PRG)
2449  *!
2450  *!*****************************************************************
2444          FUNCTION valq
2445             PARAMETERS mstr
2446             RETURN  VAL(evalq(mstr))
2447
2448          *-----------------------------------------------------
2449          *
2450          * Evaluate Qualifier
2451          *
2452          *-----------------------------------------------------
2467  *!*****************************************************************
2468  *!
2469  *!      Procedure: EVALQ
2470  *!
2471  *!      Called by: ASK                (procedure in DEMO.PRG)
2472  *!               : ISQ                (procedure in DEMO.PRG)
2473  *!               : VALQ               (procedure in DEMO.PRG)
2474  *!
2475  *!          Calls: QID                (procedure in DEMO.PRG)
2476  *!               : EDITSYMP           (procedure in DEMO.PRG)
2477  *!               : DEFQ               (procedure in DEMO.PRG)
2478  *!               : PUSHQ              (procedure in DEMO.PRG)
2479  *!
2480  *!*****************************************************************
2467          FUNCTION evalq
2468             PARAMETERS mstr
2469             PRIVATE mval
2470             PRIVATE i, need
2471             i = qid(mstr)        && get identifier for qualifier
2472             need = .F.
2473             IF EMPTY(msympval[i])
2474                * qualifier does not have value yet, get value from us
 => er
2475                need = .T.
2476             ENDIF
2477             IF need
2478                mval = redoval[i]            && previously entered va
 => lue if any
2479                event = editsymp(mstr, @mval)
2480                IF event = e_ok
2481                   =defq(mstr, mval)         && define value only if
 => edit was ok
2482                   =pushq(mstr, mval)        && push new value onto q
 => uestion stack
2483                ELSE
2484                   mval = ""
2485                ENDIF
2486             ELSE
2487                * qualifier has a value
2488                mval = msympval[i]
2489                =pushq(mstr, mval)           && push value in case it
 => s not there
```

```
2490          ENDIF
2491          RETURN mval
2492
2493          *-----------------------------------------------------
2494          *
2495          * edit symptom values on screen
2496          *
2497          *-----------------------------------------------------
2514  *!*****************************************************************
2515  *!
2516  *!      Procedure: EDITSYMP
2517  *!
2518  *!      Called by: EVALQ              (procedure in DEMO.PRG)
2519  *!
2520  *!          Calls: DICTQ()            (function  in U.PRG)
2521  *!               : ERRMESSAGE()       (function  in U.PRG)
2522  *!               : DICTQV()           (function  in U.PRG)
2523  *!               : EDNUM()            (function  in U.PRG)
2524  *!               : EDBOOL()           (function  in U.PRG)
2525  *!               : DICTECOUNT()       (function  in U.PRG)
2526  *!               : DICTQE()           (function  in U.PRG)
2527  *!               : EDCHECK()          (function  in U.PRG)
2528  *!
2529  *!*****************************************************************
2514          FUNCTION editsymp
2515
2516             PARAMETERS  dictid, matrval
2517
2518             * dictid   is dictionary id of the symptom
2519             * matrval    is initial value of symptom, returned value
2520
2521             PRIVATE mevent                        && returned event c
 => ode
2522             mevent = e_ok
2523
2524             * assumes symptom dictionary file(s) opened by caller
2525             * lookup item in symptom dictionary
2526             PRIVATE mname, msymptom, mtype, mprompt, maskable
2527             mname    = ""
2528             msymptom = ""
2529             mtype    = ""
2530             mprompt  = ""
2531             maskable = .F.
2532             mtype = dictq(dictid, @maskable, @msymptom, @mname, @mpro
 => mpt)
2533             IF EMPTY(mtype)
2534                * serious database error encountered!
2535                * should always return a valid type here!
2536                =errmessage("Missing [" + dictid + "]")
2537                matrval = ""
2538                mevent = e_esc
2539             ELSE
2540                IF !maskable
2541                   * cannot ask user for value
2542                   * supply default
2543                   matrval = ""
2544                   mevent = e_ok
2545                ELSE
2546                   DO CASE
2547                   CASE mtype = "N"
2548                      *
2549                      * numeric data type
2550                      *
2551                      PRIVATE  mhi, mlo, mwidth, mdec, munits
2552                      mhi    = 0
2553                      mlo    = 0
```

```
2554                    mwidth = 0
2555                    mdec   = 0
2556                    munits = ""
2557                  IF dictqv(dictid, @mhi, @mlo, @mwidth, @mdec, @m
 => units)
2558                      mwidth   = MAX(mwidth, 9)          && set wi
 => dth for data entry
2559                      mevent = ednum(dictid, mname, msymptom, mprom
 => pt, mhi, mlo, mwidth, mdec, munits, @mstrval)
2560                  ELSE
2561                      =errmessage("Missing val [" + dictid + "]")
2562                  ENDIF
2563                CASE mtype = "L"
2564                    *
2565                    * logical values
2566                    *
2567                    mevent  = edbool(dictid, mname, msymptom, mpromp
 => t, @mstrval)
2568                    mstrval = IIF(mstrval = "DON'T KNOW", " ", LEFT(
 => mstrval,1))
2569
2570                CASE mtype = "M"
2571                    *
2572                    * multiple choice values
2573                    *
2574                    PRIVATE sympnames
2575                    PRIVATE check
2576                    PRIVATE mtype
2577                    PRIVATE msize
2578                    PRIVATE i
2579                    msize = dictecount(dictid)
2580                  IF msize > 0
2581                      DIMENSION sympnames[msize]          && list
 => of item names
2582                      DIMENSION check[msize]              && items
 =>  that have been checked
2583                      DIMENSION mtype[msize]              && + is
 => inclusive, - is exclusive
2584                      =dictqe(dictid, sympnames, mtype)
2585
2586                    FOR i = 1 TO msize
2587                        * check if current value contains this opt
 => ion
2588                        check[i] = AT(sympnames[i], mstrval) > 0
2589                    NEXT
2590                        mevent  = edcheck(dictid, mname, msymptom,
 =>  mprompt, sympnames, mtype, check, @mstrval)
2591                    ELSE
2592                        =errmessage("Missing enum [" + dictid + "]
 => ")
**** Expected NEXT--Found ENDIF
2593                  ENDIF
2594                CASE mtype = "E"
2595                    *
2596                    * enumerated choice values
2597                    *
2598                    PRIVATE sympnames
2599                    PRIVATE check
2600                    PRIVATE mtype
2601                    PRIVATE msize
2602                    PRIVATE i
2603                    msize = dictecount(dictid)
2604                  IF msize > 0
2605                      DIMENSION sympnames[msize]          && li
 => st of item names
2606                      DIMENSION check[msize]              && it
```



```
 => ems that have been checked
2607                      DIMENSION mtype[msize]              && +
 => is inclusive, - is exclusive
2608                      =dictqe(dictid, sympnames, mtype)
2609                    FOR i = 1 TO msize
2610                        * check if current value contains this
 => option
2611                        check[i] = AT(sympnames[i], mstrval) >
 => 0
2612                    NEXT
2613                        mevent  = edcheck(dictid, mname, msympt
 => om, mprompt, sympnames, mtype, check, @mstrval)
2614                    ELSE
2615                        =errmessage("Missing enum [" + dictid +
 =>  "]")
**** Expected NEXT--Found ENDIF
2616                  ENDIF
**** Expected ENDIF--Found ENDCASE
2617                ENDCASE
2618              ENDIF
**** Expected ENDCASE--Found ENDIF
2619            ENDIF
2620            RETURN mevent
2621
2622            *-----------------------------------------------
 => ------
2623            *
2624            * Set Qualifier
2625            *
2626            *-----------------------------------------------
 => ------
2636            *!*****************************************************************
2637            *!
2638            *!      Procedure: SETQ
2639            *!
2640            *!      Called by: EVALCOND          (procedure in DEMO.PRG)
2641            *!
2642            *!          Calls: DEFQ              (procedure in DEMO.PRG)
2643            *!
2644            *!*****************************************************************
2636            FUNCTION setq
2637              PARAMETERS mstr, mval
2638              *=addq(mstr)                  && add qualifier to st
 => ate vector && rwd07
2639            RETURN  defq(mstr, mval)    && define the qualifier
 =>  value
2640
2641            *-----------------------------------------------
 => ------
2642            *
2643            * Define Qualifier
2644            *
2645            *-----------------------------------------------
 => ------
2657            *!*****************************************************************
2658            *!
2659            *!      Procedure: DEFQ
2660            *!
2661            *!      Called by: ADDSYMPTOM        (procedure in DEMO.PRG)
2662            *!               : EVALQ             (procedure in DEMO.PRG)
2663            *!               : SETQ              (procedure in DEMO.PRG)
2664            *!
2665            *!          Calls: QID               (procedure in DEMO.PRG)
2666            *!
2667            *!*****************************************************************
2657            FUNCTION defq
```

```
2658                PARAMETERS mstr, mval
2659                PRIVATE newval
2660                PRIVATE i
2661                IF event <> e_esc
2662                   * Note!  the flag, event is set to e_esc by user
 => hitting
2663                   * the abort key during data entry, but a rule ma
 => y be in the
2664                   * middle of a complex expression evaluation. The
 => refore, this
2665                   * code prevents any instantiations until the end
 => of the
2666                   * rule.
2667                   i = qid(mstr)        && get identifier for qualif
 => ier
2668                   IF EMPTY(msympval[i])
2669                      * qualifier undefined, define new value
2670                      msympval[i] = mval
2671                   ELSE
2672                      IF LEN(mval) > 0 .AND. .NOT. mval $ msympval[
 => i]
2673                         * qualifier does not already contain this
 => value
2674                         msympval[i] = msympval[i] + mconcat + mval
2675                      ENDIF
2676                   ENDIF
2677                   newval = msympval[i]
2678                   redoval[i] = newval              && save on redo
 => stack
2679                ELSE
2680                   * suppress instantiation, return null value
2681                   newval = ""
2682                ENDIF
2683 <-------------RETURN newval
2684
2685                *-----------------------------------------------
 => ------
2686                *
2687                * Get Qualifier
2688                *
2689                *-----------------------------------------------
 => ------
2701 *!*****************************************************************
2702 *!
2703 *!      Procedure: GETQ
2704 *!
2705 *!      Called by: M_REDOQ            (procedure in DEMO.PRG)
2706 *!               : STRNATURAL         (procedure in DEMO.PRG)
2707 *!               : SFSYMP             (procedure in DEMO.PRG)
2708 *!
2709 *!          Calls: QID                (procedure in DEMO.PRG)
2710 *!
2711 *!*****************************************************************
2701                FUNCTION getq
2702                   PARAMETERS mstr
2703                   i = qid(mstr)        && get identifier for qualifier
2704 <-------------RETURN  IIF( i > 0,  msympval[i], "" )    && get va
 => lue
2705
2706                *-----------------------------------------------
 => ------
2707                *
2708                * Erase Qualifier
2709                *
2710                *-----------------------------------------------
 => ------
```

```
2721 *!*****************************************************************
2722 *!
2723 *!      Procedure: ERASEQ
2724 *!
2725 *!      Called by: M_REDOQ            (procedure in DEMO.PRG)
2726 *!               : ADDHISTORY         (procedure in DEMO.PRG)
2727 *!
2728 *!          Calls: QID                (procedure in DEMO.PRG)
2729 *!
2730 *!*****************************************************************
2721                FUNCTION eraseq
2722                   PARAMETERS mstr
2723                   PRIVATE i
2724                   i = qid(mstr)        && get identifier for qualifier
2725                   IF i > 0
2726                      msympval[i] = ""
2727                   ENDIF
2728 <-------------RETURN ""
2729
2730                   *-----------------------------------------------
 => ------
2731                   *
2732                   * Restore Qualifier
2733                   *
2734                   *-----------------------------------------------
 => ------
2744 *!*****************************************************************
2745 *!
2746 *!      Procedure: RESTOREQ
2747 *!
2748 *!      Called by: M_REDOQ            (procedure in DEMO.PRG)
2749 *!
2750 *!          Calls: QID                (procedure in DEMO.PRG)
2751 *!
2752 *!*****************************************************************
2744                FUNCTION restoreq
2745                   PARAMETERS mstr, mval
2746                   i = qid(mstr)        && get identifier for qualifier
2747                   IF i > 0
2748                      msympval[i] = mval
2749                   ENDIF
2750 <-------------RETURN mval
2751
2752                   *-----------------------------------------------
 => ------
2753                   *
2754                   * Push Qualifier onto question stack
2755                   *
2756                   *-----------------------------------------------
 => ------
2764 *!*****************************************************************
2765 *!
2766 *!      Procedure: PUSHQ
2767 *!
2768 *!      Called by: EVALQ              (procedure in DEMO.PRG)
2769 *!
2770 *!*****************************************************************
2764                FUNCTION pushq
2765                   PARAMETERS mstr, mval
2766                   * ensure name unique
2767                   PRIVATE mexact
2768                   mexact = SET("EXACT")
2769                   SET EXACT ON
2770                   IF IIF(mqual>1, ASCAN(qualstack, mstr,1,mqual-1)=0,
 => .T.)
2771                      qualstack[mqual] = mstr
```

```
2772                 qualval[mqual] = mval
2773                 mqual = mqual + 1
2774              ENDIF
2775              IF mexact = "OFF"
2776                 SET EXACT OFF
2777              ENDIF
2778              RETURN mqual
2779
2780           *.. ---------------------------------------------
=> ------
2781           *
2782           * change data for rerun
2783           *
2784           *---------------------------------------------
=> ------
2795  *!*****************************************************************
2796  *!
2797  *!      Procedure: CHANGEQ
2798  *!
2799  *!      Called by: M_CHANGE          (procedure in DEMO.PRG)
2800  *!
2801  *!          Calls: LOOKUPQ           (procedure in DEMO.PRG)
2802  *!               : CHANGELIST()      (function  in U.PRG)
2803  *!
2804  *!*****************************************************************
2795           FUNCTION changeq
2796              PARAMETERS N
2797
2798              PRIVATE  mq, i, j, qnames, qindex
2799              PRIVATE  mid, mval, qlabel
2800              DIMENSION qnames[n], qindex[n]
2801
2802              * collect list of qualifier names from stack
2803              j = 0                          && number of actual names
=> collected
2804              FOR i = 1 TO N
2805                 * translate qualifier id to dictionary name
2806                 mid  =  qualstack[i]
2807                 mval = qualval[i]
2808                 qlabel = lookupq(mid)
2809                 IF LEFT(qlabel,1) != "$"
2810                    * skip metavariables (have leading $ characte
=> r)
2811                    * enter name in list with its index
2812                    j = j + 1
2813                    qnames[j] = qlabel +  arrow + mval + eol
2814                    qindex[j] = i
2815                 ENDIF
2816              NEXT
2817
2818              * user selects item from list
2819              mq = changelist(qnames, N)
2820              IF mq > 0
2821                 * get actual position in qualifier stack
2822                 mq = qindex[mq]
2823              ENDIF
2824              RETURN mq
2825
2826              *---------------------------------------------
=> ---------
2827              *
2828              * Pop Qualifier from question stack
2829              *
2830              *---------------------------------------------
=> ---------
2841  *!*****************************************************************
```



```
2842  *!
2843  *!      Procedure: POPQ
2844  *!
2845  *!      Called by: M_CHANGE          (procedure in DEMO.PRG)
2846  *!               : M_UNDO            (procedure in DEMO.PRG)
2847  *!
2848  *!          Calls: UNDOQUAL          (procedure in DEMO.PRG)
2849  *!
2850  *!*****************************************************************
2841           FUNCTION popq
2842              PARAMETERS N
2843              PRIVATE s
2844
2845              DO WHILE mqual > N  .AND. mqual > 1
2846                 * pop questions from stack
2847                 mqual = mqual - 1
2848                 s = qualstack[mqual]              && get quali
=> fier
2849                 qualstack[mqual] = ""             && erase sta
=> ck item
2850                 =undoqual(s + arrow + uval + eol) && undefine
=>  the qualifier
2851              ENDDO
2852              RETURN mqual
2853
2854              *---------------------------------------------
=> ---------
2855              *
2856              * Find last question on qualifier stack
2857              *
2858              *---------------------------------------------
=> --------
2868  *!*****************************************************************
2869  *!
2870  *!      Procedure: LASTQ
2871  *!
2872  *!      Called by: M_UNDO            (procedure in DEMO.PRG)
2873  *!
2874  *!          Calls: LOOKUPQ           (procedure in DEMO.PRG)
2875  *!
2876  *!*****************************************************************
2868           FUNCTION lastq
2869              PRIVATE i
2870              i = mqual - 1
2871              DO WHILE i > 0
2872                 IF LEFT(lookupq(qualstack[i]), 1) != "$"
2873                    EXIT                          && not a meta-variabl
=> e
2874                 ENDIF
2875                 i = i - 1
2876              ENDDO
2877              RETURN i
2878
2879              *---------------------------------------------
=> ---------
2880              *
2881              * Identify Qualifier
2882              *
2883              *---------------------------------------------
=> ---------
2884              * if qualifier does not exist, instantiate it
2900  *!*****************************************************************
2901  *!
2902  *!      Procedure: QID
2903  *!
2904  *!      Called by: UNDOQUAL          (procedure in DEMO.PRG)
```

```
2905  *|                : EVALQ              (procedure in DEMO.PRG)
2906  *|                : DEFQ               (procedure in DEMO.PRG)
2907  *|                : GETQ               (procedure in DEMO.PRG)
2908  *|                : ERASEQ             (procedure in DEMO.PRG)
2909  *|                : RESTOREQ           (procedure in DEMO.PRG)
2910  *|                : ADDQ               (procedure in DEMO.PRG)
2911  *|
2912  *|          Calls: ERRMESSAGE()        (function  in U.PRG)
2913  *|
2914  *|*****************************************************************
2900                     FUNCTION qid
2901                        PARAMETERS mstr
2902                        PRIVATE i
2903                        PRIVATE mexact
2904                        mexact = SET("EXACT")
2905                        SET EXACT ON
2906                        i = ASCAN(mcondition, mstr)       && search for
 => name of qualifier
2907                        IF i = 0
2908                           * qualifier does not exist yet, define its na
 => me
2909                           IF nextqual >= nconditions
2910                              * no more free slots in memory
2911                              =errmessage("**** Warning: qualifier overfl
 => ow ****")
2912                              i = 1
2913                           ELSE
2914                              i = nextqual              && next available
 => position
2915                              =AINS(mcondition, i)
2916                              =AINS(msympval, i)
2917                              mcondition[i] = mstr
2918                              msympval[i] = ""
2919                              nextqual = nextqual + 1
2920                           ENDIF
2921                        ENDIF
2922                        IF mexact ="OFF"
2923                           SET EXACT OFF
2924                        ENDIF
2925                        RETURN i
2926
2927                        *-----------------------------------------
 => ---------
2928                        *
2929                        * Add Qualifier to state vector
2930                        *
2931                        *-----------------------------------------
 => ---------
2939  *|*****************************************************************
2940  *|
2941  *|      Procedure: ADDQ
2942  *|
2943  *|          Calls: QID                (procedure in DEMO.PRG)
2944  *|
2945  *|*****************************************************************
2939                     FUNCTION addq
2940                        PARAMETERS mstr
2941                        PRIVATE i, saveq
2942                        IF mrule > 0
2943                           * only if there is an active rule
2944                           i = qid(mstr)        && get identifier for qua
 => lifier
2945                           saveq = mstr + arrow + IIF(EMPTY(msympval[i])
 => , uval, msympval[i])
2946                           IF  TYPE("statevecq[mrule]") = "U"
2947                              statevecq[mrule] = ""
2948                           ENDIF
2949                           statevecq[mrule] = statevecq[mrule] + saveq +
 => eol
2950                        ENDIF
2951                        RETURN .T.
2952
2953                        *-----------------------------------------
 => ---------
2954                        *
2955                        * get conditions to string form
2956                        *
2957                        *-----------------------------------------
 => ---------
2969  *|*****************************************************************
2970  *|
2971  *|      Procedure: STRCOND
2972  *|
2973  *|      Called by: M_RESUME          (procedure in DEMO.PRG)
2974  *|               : _EDITCOND         (procedure in U.PRG)
2975  *|               : _EDITHANDP        (procedure in U.PRG)
2976  *|
2977  *|          Calls: LOOKUPNAM         (procedure in DEMO.PRG)
2978  *|
2979  *|*****************************************************************
2969                     FUNCTION strcond
2970                        PARAMETERS mcond, mqid, mname, mval, DELIM
2971
2972                        EXTERNAL ARRAY mname
 =>     && rw007
2973
2974                        PRIVATE i, j, N, DELIM
2975                        PRIVATE mstr
2976
2977                        IF PARAMETERS() = 4
2978                           DELIM = " "
2979                        ENDIF
2980                        N = nextqual - 1                  && number of act
 => ive conditions
2981
2982                        j = 0
2983                        FOR i = 1 TO N
2984                           mcond[i] = ""
2985                           IF TYPE("mval[i]") != "U"
2986                              IF LEN(mval[i]) > 0
2987                                 * translate qualifier id to dictionary
 => name
2988                                 mstr = lookupnam(mname[i])
2989                                 IF LEFT(mstr,1) != "$"
2990                                    j = j + 1
2991                                    mcond[j] = mstr + DELIM + mval[i]
2992                                    mqid[j] = mname[i]
2993                                 ENDIF
2994                              ENDIF
2995                           ENDIF
2996                        NEXT
2997
2998                        RETURN j
2999
3000                        *-----------------------------------------
 => -----------
3001                        *
3002                        * get diagnostic session results
3003                        *
3004                        *-----------------------------------------
 => -----------
3015  *|*****************************************************************
```

```
3016  *|
3017  *|      Procedure: STRDIAG
3018  *|
3019  *|      Called by: M_RESUME          (procedure in DEMO.PRG)
3020  *|               : _EDITASSIS        (procedure in U.PRG)
3021  *|
3022  *|          Calls: DISEASENAM()      (function  in U.PRG)
3023  *|
3024  *|*****************************************************************
3015                        FUNCTION strdiag
3016
3017                          PARAMETERS msorted, mdiag, mconf, threshold,
 => limit, showconf
3018
3019                          EXTERNAL ARRAY msorted                    && r
 => wd07
3020
3021                          PRIVATE  i, DELIM, K, inserted, msorted, xnam
 => e, N
3022                          DIMENSION mrank( nextdiag )          && ranked
 => factors
3023                          inserted = 0                         &&  count
 => factors inserted
3024                          DELIM = " "
3025                          N = nextdiag - 1                     && number
 => of active factors
3026
3027                        FOR i = 1 TO N
3028                           =AINS(mrank, i)                     && create
 => rank slot
3029                          IF mconf[i] >= threshold .AND. ;
3030                                 IIF(TYPE("limit") <> "N", .T., mcon
 => f[i] < limit)
3031                            * factor is within specified range
3032                            * insert in ranked list
3033                            K = 1
3034
3035                            DO WHILE K <= inserted
3036                              * find slot in ranked list of factor
 => s
3037                              IF mconf[i] >= mconf[mrank[k]]
3038                                * insert factor
3039                                =AINS(mrank, K)
3040                                EXIT
3041                              ENDIF
3042                              K = K + 1
3043                            ENDDO
3044
3045                            mrank[k] = i
3046                            inserted = inserted + 1
3047                          ENDIF
3048                        NEXT
3049
3050                          * output ranked list as string
3051
3052                        FOR i = 1 TO inserted
3053                            K = mrank[i]
3054                            msorted[i] = diseasenam( VAL( mdiag[k]
 => ) ) + ;
3055                                 IIF( PARAMETERS() = 6, ;
3056                                 DELIM  +  LTRIM( STR( mconf[ k ], 6
 => , 1 ) ), ;
3057                                 "" )
3058                          NEXT
3059                        RETURN inserted
3060
```



```
3061                                    *-----------------------------------
 => ----------------
3062                                    *
3063                                    * report session in natural language fo
 => rm
3064                                    *
3065                                    *-----------------------------------
 => ----------------
3085  *|*****************************************************************
3086  *|
3087  *|      Procedure: STRNATURAL
3088  *|
3089  *|          Calls: GETQ                (procedure in DEMO.PRG)
3090  *|               : DICTREPORT()        (function  in U.PRG)
3091  *|               : SPLIT()             (function  in U.PRG)
3092  *|
3093  *|           Uses: PHRASE.DBF
3094  *|               : SENTENCE.DBF
3095  *|               : SECTION.DBF
3096  *|               : REPORT.DBF
3097  *|
3098  *|        Indexes: PHRASE.IDX
3099  *|               : SENTENCE.IDX
3100  *|               : SECTION.IDX
3101  *|               : REPORT.IDX
3102  *|
3103  *|*****************************************************************
3085                                    FUNCTION strnatural
3086                                       PARAMETERS marea
3087
3088                                       PRIVATE mstr, xstr, ystr, zstr
3089                                       PRIVATE mid
3090                                       PRIVATE msection, msentence, mbegin
3091                                       PRIVATE msel
3092                                       msel = SELECT()
 =>   && save curr  t database selection
3093                                       SELECT 0
3094                                       USE phrase INDEX phrase
3095                                       SELECT 0
3096                                       USE sentence INDEX sentence
3097                                       SET RELATION TO id INTO dict,id INTO ph
 => rase
3098                                       SELECT 0
3099                                       USE SECTION INDEX SECTION
3100                                       SET RELATION TO sentence INTO sentence
3101                                       SELECT 0
3102                                       USE REPORT INDEX REPORT
3103                                       SET RELATION TO SECTION INTO SECTION
3104                                       SET FILTER TO area = marea
3105                                       GOTO TOP
3106                                       mstr = ""
3107
3108                                       DO WHILE !EOF()
3109                                          msection = SECTION
3110                                          SELECT SECTION
3111                                          mbegin = .T.
3112                                          ystr = ""
3113
3114                                          DO WHILE SECTION = report->section
3115                                             * test controlling condition
3116                                             DO CASE
3117                                             CASE OBJECT = "S"
3118                                                xstr = getq( ALLTRIM( STR( id
 => ) )
3119                                                IF !EMPTY( xstr ) .AND. LEN(xs
 => tr) = 1 .AND. UPPER(xstr) = "S"
```

```
3120                                                    * filter logically false co
 => ndition
3121                                                      xstr = ""
3122                                                    ENDIF
3123                                                  OTHERWISE
3124                                                    * default is unconditional pro
 => cessing of sentence
3125                                                    xstr = "Y"
3126                                                  ENDCASE
3127                                                  IF !EMPTY( xstr )
3128                                                    * valid condition
3129                                                    * process sentence phrase
3130                                                    msentence = sentence
3131                                                    SELECT sentence
3132
3133                                                    DO WHILE sentence = section->s
 => entence
3134                                                      xstr = ""
 =>          && object value
3135                                                      DO CASE
3136                                                      CASE OBJECT = "S"
3137                                                        * symptom object:
3138                                                        mid = ALLTRIM( STR( id )
 =>  )       && object identifier
3139                                                        DO CASE
3140                                                        CASE attribute = "VAL"
3141                                                          * value attribute
3142                                                          xstr = getq( mid )
 =>          && fetch value
3143                                                          IF !EMPTY( xstr )
3144                                                            * object has a def
 => ined value
3145                                                            DO CASE
3146                                                            CASE dict->datatyp
 => e = "L"
3147                                                              * logical types
3148                                                              xstr = dictrepo
 => rt(mid, xstr)
3149                                                              xstr = IIF( LOW
 => ER( xstr ) = "yes", "", xstr )
3150                                                            CASE  dict->dataty
 => pe = "E"  .OR.  dict->datatype = "A"
3151                                                              * enumerated ty
 => pes
3152                                                              * concatenate r
 => eport strings
3153                                                              zstr = xstr
3154                                                              xstr = ""
3155                                                            DO WHILE !EMPTY
 => ( zstr )
3156                                                              xstr  =  IIF
 => ( EMPTY(xstr), "", xstr + " and " )
3157                                                                    +  dictr
 => eport(mid, split( @zstr ) )
3158                                                            ENDDO
3159                                                            CASE  dict->dataty
 => pe = "N"
3160                                                              * Numeric types
3161                                                            ENDCASE
3162                                                          ENDIF
3163                                                        CASE attribute = "NAM"
3164                                                          * name attribute
3165                                                          xstr = dict->name
3166                                                        ENDCASE
3167                                                      CASE OBJECT = "T"
3168                                                        * text string
3169                                                        xstr = phrase->phrase
3170                                                      CASE OBJECT = "P"
3171                                                        * punctuation delimiter
3172                                                        IF LEFT( attribute, 1 )
 => = "\"
3173                                                          xstr = SUBSTR( attribu
 => te, 1)
3174                                                          DO CASE
3175                                                          CASE xstr = "t"
3176                                                            xstr = CHR(8)
3177                                                          CASE xstr = "n"
3178                                                            xstr = CHR(13) + C
 => HR(10)
3179                                                          CASE xstr = "l"
3180                                                            xstr = CHR(12)
3181                                                          ENDCASE
3182                                                        ELSE
3183                                                          xstr = attribute
3184                                                        ENDIF
3185                                                      ENDCASE
3186                                                      xstr  =  LOWER( TRIM(xstr)
 => `
3187                                                      IF mbegin
3188                                                        xstr  =  UPPER( LEFT(xst
 => r,1) ) + SUBSTR(xstr, 2)
3189                                                        mbegin = .F.
3190                                                      ENDIF
3191                                                      IF !EMPTY( xstr )
3192                                                        IF !EMPTY( ystr )
3193                                                          IF ! ( xstr  $  "," +
 =>  CHR(13) + CHR(8) + CHR(12) )
3194                                                            xstr = " " + xstr
3195                                                          ENDIF
3196                                                        ENDIF
3197                                                        ystr  =  ystr  +  xstr
3198                                                      ENDIF
3199                                                      SKIP
3200                                                    ENDDO
3201                                                  ENDIF
3202                                                  SELECT SECTION
3203                                                  SKIP
3204                                                ENDDO
3205
3206                                              * ending section
3207                                              IF !EMPTY(ystr)
3208                                                * add punctuation
3209                                                IF ! ( RIGHT( ystr, 1 ) $ ".:;,"
 => )
3210                                                  ystr = ystr + "."
3211                                                ENDIF
3212                                                mstr = IIF( EMPTY(mstr), "", mstr
 =>  + " " )  +  ystr
3213                                              ENDIF
3214                                              SELECT REPORT
3215                                              SKIP
3216                                            ENDDO
3217
3218                                          USE
3219                                          SELECT phrase
3220                                          USE
3221                                          SELECT SECTION
3222                                          USE
3223                                          SELECT sentence
3224                                          USE
3225                                          SELECT (msel)
3226                                          RETURN mstr
```

```
3227
3228
3229                                    *-----------------------------------
 => ----------------
3230                                    *
3231                                    * batch test system
3232                                    *
3233                                    *-----------------------------------
 => ----------------
3251  *!*****************************************************************
3252  *!
3253  *!        Function: BATCHTEST()
3254  *!
3255  *!       Called by: DEMO.PRG
3256  *!
3257  *!           Calls: DELSYMPTOM          (procedure in DEMO.PRG)
3258  *!                : DELDIAGNOS          (procedure in DEMO.PRG)
3259  *!                : M_ENCRULE           (procedure in DEMO.PRG)
3260  *!                : ENC2STR             (procedure in DEMO.PRG)
3261  *!                : M_ENCGET            (procedure in DEMO.PRG)
3262  *!                : M_ASSIST            (procedure in DEMO.PRG)
3263  *!                : M_ENCSAV            (procedure in DEMO.PRG)
3264  *!                : ENC2BIT             (procedure in DEMO.PRG)
3265  *!                : DISEASENAM()        (function  in U.PRG)
3266  *!
3267  *!*****************************************************************
3251                                    FUNCTION batchtest
 =>    && rwd03
3252                                    PARAMETERS testing, testfile, logfile
3253
3254                                    * This test function replaces the main
 => menu
3255                                    PRIVATE i, j, K
3256                                    PRIVATE szcasedata, szoutcome, szdiag,
 => szconf
3257
3258                                    * use test files
3259                                    IF RIGHT(testing, 1) != "\"
3260                                       testing = testing + "\"
3261                                    ENDIF
3262                                    SELECT 0
3263                                    USE (testing + testfile)
 =>    && open test file
3264
3265                                    * list results to screen or file
3266                                    @ 0,0 CLEAR
3267                                    IF !EMPTY(logfile)
3268                                       * list to log file
3269                                       SET PRINTER TO (testing + logfile)
3270                                       SET PRINTER ON
3271                                    ENDIF
3272
3273                                    ?? mtitle
3274                                    ?  mversion
3275                                    ?  "Naval Health Research Center"
3276                                    ?  "Test " + mepssdata + "\" + testing
 => + testfile + "    " + TIME() + "  "
3277                                    ?? DATE()
3278                                    ?
3279                                    ?  "Case  Outcome"
3280
3281                                    j = 0
 =>    && count completed test cases
3282                                    K = 0
 =>    && count total test cases
3283
```

```
3284                                    DO WHILE !EOF()
3285                                       * Get next case from test file:
3286                                       event = e_ok
 =>    && reset event flag
3287                                       mencountid = caseid
 =>    && get id of case
3288                                       IF hasdata
3289                                          * using case data directly from t
 => est file
3290                                          marea = area
 =>    && get complaint area
3291                                          mpatientid = 9999
 =>    && fake patient for test
3292                                          mprovider = 9999
 =>    && fake provider for test
3293                                          mencstate = "N"
 =>    && start a new encounter
3294                                          enc_open = .T.
3295                                          mdateenc = DATE()
 =>    && today's date
3296                                          mtimeenc = TIME()
 =>    && current time is start of encounter
3297                                          mdiagnosis = 0
 =>    && empty provider's diagnosis
3298                                          mtreatment = ""
 =>    && empty treatment description
3299                                          =delsymptom()
 =>    && clear the list of symptoms
3300                                          =deldiagnos()
 =>    && diagnoses for new encounter
3301                                          =m_encrule()
 =>    && load rulebase
3302                                          =enc2str(marea, ALLTRIM( casedata
 => ) )  && translate the case data
3303                                       ELSE
3304                                          * loading from encounter file
3305                                          =m_encget()
 =>    && load the encounter
3306                                       ENDIF
3307
3308                                       =m_assist()
 =>    && calc diagnosis
3309                                       szoutcome = ""
3310                                       IF event = e_ok
3311                                          =m_encsav()
 =>    && save the encounter
3312
3313                                          * print diagnostic result
3314                                          ? ALLTRIM( STR( mencountid ) ) +
 => " "
3315                                          szcasedata = enc2bit(fire)
3316                                          ?? '"' + szcasedata + '"'
3317
3318                                          ?? ' "'
3319                                          FOR  i = 1  TO  ALEN( mdiag )
3320                                             IF  EMPTY( mconf[i] )
3321                                                EXIT
3322                                             ENDIF
3323                                             szdiag = diseasenam( VAL( mdia
 => g[i] )
3324                                             szconf = STR( mconf[i], 4, 1 )
3325                                             ?? IIF( !EMPTY(szoutcome), ","
 => , "") + LEFT( szdiag, 3 ) + "=" +  szconf
3326                                             szoutcome = szoutcome + IIF( !
 => EMPTY(szoutcome), ",","") + szdiag + "=" + szconf
3327                                             NEXT
```

```
3328                                    ?? '"'
3329                                    j = j + 1
3330                                ELSE
3331                                    ? ALLTRIM( STR( mencountid ) )
3332                                    ?? "  *** Incomplete ****
**** Expected NEXT--Found ENDIF
3333                                ENDIF
3334                                IF lhasdata
3335                                    REPLACE hasdata WITH .T., case
 => data WITH szcasedata
3336                                ENDIF
3337                                REPLACE outcome WITH szoutcome
3338                                K = K + 1
3339                                SKIP
**** Expected ENDIF--Found ENDDO
3340                            ENDDO
3341
3342                            ?
3343                            ?  "Total cases read: " + ALLTRIM( S
 => TR( K ) )
3344                            ?  "Total completed:  " + ALLTRIM( S
 => TR(   ) )
3345
3346                            IF !EMPTY(logfile)
3347                                SET PRINTER OFF
3348                                SET PRINTER TO
3349                                SET CONSOLE ON
3350                            ENDIF
3351
3352                            ? "Hit any key to continue..."
3353                            =INKEY(0)
3354
3355                            RETURN .T.
3356
3357
3358                            *--------------------------------
 => -------------------
3359                            *
3360                            * translate encounter
3361                            * bit to string
3362                            *
3363                            *--------------------------------
 => -------------------
3373  *!*****************************************************************
3374  *!
3375  *!      Procedure: ENC2STR
3376  *!
3377  *!      Called by: BATCHTEST()        (function  in DEMO.PRG)
3378  *!
3379  *!          Calls: INVERTCOND         (procedure in DEMO.PRG)
3380  *!
3381  *!*****************************************************************
3373                            FUNCTION enc2str
 =>         && rwd03
3374
3375                            PARAMETERS marea, mbit
3376                            * mbit is a bit string representing
 => logical true/false state
3377                            * of the set of symptoms
3378
3379                            PRIVATE mresult
3380                            PRIVATE msel
3381                            msel  = SELECT()
 =>     && save current database selection
3382                            SELECT rule
3383                            SET ORDER TO 2
 =>     && index on rule id
3384
3385                            FOR mrule = 1 TO LEN(mbit)
 =>     && process the symptom string
3386                                IF SUBSTR(mbit, mrule, 1) = "1"
3387                                    SEEK mrule
 =>     && rule is eligible
3388                                    SELECT premise
 =>     && get corresponding rule clauses
3389                                    SEEK rule->premise
 =>                 && rwd07
3390                                    IF clause = rule->premise
3391                                        * set applicable symptom in
 =>  premise clause
3392                                        =invertcond( OBJECT, ALLTRI
 => M( STR(id) ),  op, VAL )    && rwd07
3393                                    ENDIF
3394                                    SELECT rule
3395                                    IF event = e_esc
3396                                        EXIT
 =>     && abort flag has been set
3397                                    ENDIF
3398                                ENDIF
3399                            NEXT
3400
3401                            SELECT rule
3402                            SET ORDER TO 1
3403                            SELECT (msel)
3404                            RETURN event
3405
3406
3407                            *--------------------------------
 => -------------------
3408                            *
3409                            * translate encounter
3410                            * string to bit
3411                            *
3412                            *--------------------------------
 => -------------------
3420  *!*****************************************************************
3421  *!
3422  *!      Procedure: ENC2BIT
3423  *!
3424  *!      Called by: BATCHTEST()        (function  in DEMO.PRG)
3425  *!
3426  *!*****************************************************************
3420                            FUNCTION enc2bit
 =>         && rwd03
3421
3422                            PARAMETERS fire
3423                            * fire is the rule agenda; each e
 => lement represents state of
3424                            * the corresponding rule
3425                            * mresult is a bit string represe
 => nting logical true/false state
3426                            * of the set of symptoms, one for
 =>  each rule
3427
3428                            PRIVATE mresult
3429                            PRIVATE i
3430                            mresult = ""
3431
3432                            FOR i = 1 TO ALEN(fire)
3433                                mresult  =  mresult  +  IIF( f
 => ire[i], "1", "0" )
3434                            NEXT
```

```
3435
3436  <------------------------------------RETURN mresult
3437
3438
3439
3440                                         *---------------------------
 => -------------------------
3441                                         *
3442                                         * invert an object clause
3443                                         *
3444                                         *---------------------------
 => -------------------------
3454  *|*****************************************************************
3455  *|
3456  *|      Procedure: INVERTCOND
3457  *|
3458  *|      Called by: ENC2STR            (procedure in DEMO.PRG)
3459  *|
3460  *|          Calls: EVALCOND           (procedure in DEMO.PRG)
3461  *|
3462  *|*****************************************************************
3454                                       FUNCTION invertcond
 =>                && rwd03
3455                                       PARAMETERS mobject, mid, mop,
 => mval                 && rwd07
3456                                       * mobject      object type desc
 => riptor
3457                                       * mid          object identifie
 => r
3458                                       * mop          operation code
3459                                       * mval         constant value
3460
3461                                       * Invert a clause of the form:
3462                                       *  <objectid> <op> <val>
3463
3464                                       * See also evalcond()
3465                                       * Inversion means assigning a
 => value to the object such that
3466                                       * the above relational express
 => ion is true.  Force object to
3467                                       * a value that satisfies the r
 => elation expression with op one of:
3468                                       * >=  <=  >  <  ==  !=
3469                                       * this is mostly easy... only
 => need to handle the exceptions:
3470                                       * numeric inequality  <  >  !=
3471                                       * logical negation    !=
3472
3473                                       IF mop != "= "
3474                                          * relational operator  >=
 => <=  >  <  ==  !=
3475                                         DO CASE
3476                                         CASE mobject = "S"
3477                                            * symptom object:
3478                                            * Assumes clause ==> dic
 => t relation is set          && rwd07
3479                                            mtype = dict->datatype
 =>                        && rwd07
3480                                         DO CASE
3481                                         CASE mtype = "N"
3482                                            * numeric comparisons
 =>  > < !=
3483                                            nval = VAL(mval)
3484                                            DO CASE
3485                                            CASE mop = "> "
3486                                               * force value > by
```

```
 =>  adding a small delta
3487                                            nval = nval  +  II
 => F( nval > 0,  0.1,  -0.1 )
3488                                            CASE mop = "< "  .OR.
 =>   mop = "!="
3489                                               * force value < or
 =>  != by subtracting a small delta
3490                                            nval = nval  -  II
 => F( nval > 0,  0.1,  -0.1 )
3491                                            ENDCASE
3492                                              mval = ALLTRIM( STR(
 => nval, 9, 2 ) )
3493                                            CASE mtype = "L"
3494                                               * logical equality or
 =>  negation
3495                                            *  mval = IIF( mop = "==
 => ", mval, IIF(mval = "Y", "N", "Y") )
3496                                            ENDCASE
3497                                         ENDCASE
3498                                      ENDIF
3499
3500                                      * set the value:
3501  <------------------------------------RETURN evalcond(mobject, mid,
 => "= ", mval)
3502
3503                                      *---------------------------
 => -------------------------
3504                                      *
3505                                      * init system
3506                                      *
3507                                      *---------------------------
 => -------------------------
3518  *|*****************************************************************
3519  *|
3520  *|       Function: SYSINIT()
3521  *|
3522  *|      Called by: DEMO.PRG
3523  *|
3524  *|          Calls: SETPATH           (procedure in U.PRG)
3525  *|                : GETOPTS          (procedure in U.PRG)
3526  *|
3527  *|*****************************************************************
3518                                       FUNCTION sysinit
3519
3520                                       * global system variables
3521
3522                                       PUBLIC mpath, mproc
 =>    && previous procedure and path
3523                                       PUBLIC event
 =>    && system state transition events
3524                                       PUBLIC e_shutdown, e_esc, e_in
 => it, e_ok, e_change, e_redo
3525                                       PUBLIC e_goback, e_next, e_pre
 => vfld, e_nextfld, e_save, e_past
3526                                       PUBLIC e_any, e_help, e_print
3527                                       PUBLIC mstate
 =>    &&  system states
3528                                       PUBLIC s_shutdown, s_init, s_m
 => ain, s_diag, s_disease
3529                                       PUBLIC s_encnt, s_open, s_save
 => , s_close, s_new
3530                                       PUBLIC s_treatmnt, s_review, s
 => _pending
3531                                       PUBLIC mconcat
3532                                       PUBLIC mprovider
3533                                       PUBLIC cthreshold
```

```
=>    && Threshold for possible diagnosis (confidence factors)
3534                                                    PUBLIC cthreshpro
=>    && Threshold for probable diagnosis
3535                                                    PUBLIC cthreshlik
=>    && Threshold for most likely diagnosis
3536                                                    PUBLIC cfmethod
=>    && Confidence factor method
3537                                                    PUBLIC rulebase
=>    && name of the current rulebase
3538                                                    PUBLIC mrule
=>    && current rule being interpreted
3539                                                    PUBLIC nrules
=>    && number of rules in current rulebase
3540                                                    PUBLIC msignon,mstart,mfinish
=>    && names of applicable info screens
3541                                                    PUBLIC arrow, eol, uval
3542                                                    PUBLIC is_debug, is_conf, is_r
=> ule, is_novice, is_edit
3543                                                    PUBLIC mprivateact, mdisclaim
=>  && get warning text
3544                                                    PUBLIC is_newpat
=>    && allow new patient registration
3545                                                    PUBLIC enc_open
=>    && patient encounter open/closed state
3546                                                    PUBLIC e_quit
3547                                                    PUBLIC mtrain
3548                                                    e_quit = .F.
3549                                                    e_shutdown = -1
3550                                                    e_esc      = -1
3551                                                    e_init     = 0
3552                                                    e_ok       = 11
3553                                                    e_change   = 12
3554                                                    e_redo     = 13
3555                                                    e_goback   = 18
3556                                                    e_next     = 24
3557                                                    e_prevfld  = 25
3558                                                    e_nextfld  = 30
3559                                                    e_save     = 31
3560                                                    e_past     = 32
3561                                                    e_any      = 33
3562                                                    e_help     = 34
3563                                                    e_print    = 35
3564                                                    event = e_init
=>    && initial event
3565                                                    s_shutdown = -1
3566                                                    s_init     = 0
3567                                                    s_main     = 2
3568                                                    s_diag     = 3
3569                                                    s_encnt    = 4
3570                                                    s_open     = 5
3571                                                    s_save     = 6
3572                                                    s_close    = 7
3573                                                    s_new      = 8
3574                                                    s_pending  = 9
3575                                                    s_disease  = 18
3576                                                    s_treatmnt = 19
3577                                                    s_review   = 20
3578
3579                                                    mtrain = .F.
=>                && initial the training mode
3580                                                    mstate = s_init
=>    && initial system state
3581                                                    enc_open   = .F.
=>    && initial encounter state
3582                                                    mconcat    = "|"
=>    && string concat symbol
```

```
3583                                                    cthreshold =  0
=>    && diagnostic confidence threshold
3584                                                    cfmethod   = ""
=>    && confidence factor method
3585                                                    rulebase   = ""
=>    && default rulebase name
3586                                                    msignon    = "SIGNON"
=>    && default signon screen
3587                                                    mstart     = "START"
=>    && default rulebase start screen
3588                                                    mfinish    = "END"
=>    && default rulebase end screen
3589                                                    arrow      = "="
=>    && qualifier assignment symbol
3590                                                    eol        = ";"
=>    && qualifier end of line symbol
3591                                                    uval       = "%"
=>    && undefined qualifier symbol
3592                                                    is_debug   = .F.
=>    && debug mode
3593                                                    is_newpat  = .T.
=>    && allow new patient registration
3594                                                    is_conf    = .F.
=>    && display confidence factors
3595                                                    is_rule    = .F.
=>    && display active rules
3596                                                    mprovider  = 0
=>    && medical provider user identification
3597
3598                                                    * set path for database files
3599                                                    mproc = SET("PROC")
3600                                                    SET PROCEDURE TO u.prg
3601                                                    mpath = SET("PATH")
3602                                                    IF !EMPTY(GETENV("CAMD"))
3603                                                       DO setpath WITH GETENV("CAM
=> D")
3604                                                    ENDIF
3605                                                    =getopts(@is_debug, @is_conf,
=> @is_rule, @is_novice, @is_edit, @mprivateact, @mdisclaim)
3606                                                    RETURN ""
3607
3608
3609                                                    *-----------------------------
=> ----------------------------------------
3610                                                    * System close
3611                                                    * Programmer : RWD
3612                                                    * Modify by NLL: - Not close t
=> he database (no database open at this time
3613                                                    *                - Set the pat
=> h back to original
3614                                                    *                - Set the pro
=> cedure back to original
3615                                                    *-----------------------------
=> ----------------------------------------
3616
3624                                                  *!*****************************************************************
3625                                                  *!
3626                                                  *!       Function: SYSCLOSE()
3627                                                  *!
3628                                                  *!      Called by: DEMO.PRG
3629                                                  *!
3630                                                  *!*****************************************************************
3624                                                    FUNCTION sysclose
3625
3626                                                    SET PATH TO (mpath)
3627                                                    SET PROCEDURE TO (mproc)
```

```
3628                                        CLEAR READ ALL
3629                                        CLEAR ALL
3630  <-------------------------------------RETURN ""
3631
3632                                        *----------------------------
 => ------------------------------------
3633                                        * Interface patient CAMD with
 => SNAP/SARS
3634                                        * Programmer: HLL
3635                                        *----------------------------
 => ------------------------------------
3652  *!*****************************************************************
3653  *!
3654  *!      Procedure: WPAT
3655  *!
3656  *!      Called by: M_PAT              (procedure in DEMO.PRG)
3657  *!               : M_NEWENC           (procedure in DEMO.PRG)
3658  *!               : M_TRAINING         (procedure in DEMO.PRG)
3659  *!
3660  *!          Calls: PUSHHELP           (procedure in U.PRG)
3661  *!               : WP1.PRG
3662  *!               : DP.PRG
3663  *!               : AGEAT              (procedure in DEMO.PRG)
3664  *!               : WPCLEAN            (procedure in DEMO.PRG)
3665  *!               : POPHELP            (procedure in U.PRG)
3666  *!
3667  *!*****************************************************************
3652                                      PROCEDURE wpat
3653
3654                                        PUBLIC u
3655                                        u = "^"
3656                                        mudf = SET("udfparms")
3657                                        SET UDFPARMS TO VALUE
3658                                        =pushhelp("REGISTER PATIENT")
3659                                        mdata = wp1()
3660                                        IF !EMPTY(mudf)
3661                                           SET UDFPARMS TO &mudf
3662                                        ENDIF
3663                                        msocsec  =  dp(mdata,u,1)
3664                                        mlastname = dp(mdata,u,2)
3665                                        mfirstname = dp(mdata,u,3)
3666                                        mmidlename = dp(mdata,u,4)
3667                                        mdatebirth = CTOD(dp(mdata,u,5
 => ))
3668                                        msex = dp(mdata,u,6)
3669                                        mraterank = dp(mdata,u,7)
3670                                        mpatientid = VAL(dp(mdata,u,8)
 => )
3671                                        mdate = DATE()
3672                                        mage = ageat(mdate,mdatebirth)
3673                                        DO wpclean
3674                                        =pophelp()
3675  <-------------------------------------RETURN
3676
3677
3678                                        ******************************
 => ------------------------------------
3679                                        * Close the patient file and r
 => eturn the environment
3680                                        * Programmer: HLL
3681                                        ******************************
 => ------------------------------------
3689  *!*****************************************************************
3690  *!
3691  *!      Procedure: WPCLEAN
3692  *!
```

```
3693  *!      Called by: WPAT               (procedure in DEMO.PRG)
3694  *!
3695  *!*****************************************************************
3689                                      PROCEDURE wpclean
3690
3691                                        IF TYPE("fsams") != "U"
3692                                           IF !EMPTY(fsams)
3693                                              SELECT (fsams)
3694                                              USE
3695                                           ENDIF
3696                                        ENDIF
3697                                        IF USED("wpat1")
3698                                           SELECT wpat1
3699                                           USE
3700                                        ENDIF
3701                                        IF USED("wpat2")
3702                                           SELECT wpat2
3703                                           USE
3704                                        ENDIF
3705
3706  <-------------------------------------RETURN
3707
3708                                        ******************************
 => ------------------------------------
3709                                        * print sf600 report
3710                                        * Programmer: HLL
3711                                        ******************************
 => ------------------------------------
3722  *!*****************************************************************
3723  *!
3724  *!      Procedure: M_SF600
3725  *!
3726  *!          Calls: OPENFILES()        (function  in U.PRG)
3727  *!               : SFENCOUNT          (procedure in U.PRG)
3728  *!               : SFREPORT           (procedure in DEMO.PRG)
3729  *!               : FLUSHFILES()       (function  in U.PRG)
3730  *!
3731  *!*****************************************************************
3722                                      PROCEDURE m_sf600
3723                                        IF !enc_open
 =>                                          && encounter has not open
3724                                           =openfiles()
3725                                           IF sfencount() > 0
3726                                              mtext = sfreport(marea)
3727                                           ENDIF
3728                                           =flushfiles()
3729                                        ELSE
3730                                           mtext = sfreport(marea)
3731                                        ENDIF
3732  <-------------------------------------RETURN
3733
3734                                        ******************************
 => ------------------------------------
3735                                        * Get the sf600 report base on
 =>  the encounter select
3736                                        * Programmer: HLL
3737                                        ******************************
 => ------------------------------------
3753  *!*****************************************************************
3754  *!
3755  *!      Procedure: SFREPORT
3756  *!
3757  *!      Called by: M_SF600            (procedure in DEMO.PRG)
3758  *!
3759  *!          Calls: YESNO.PRG
3760  *!               : POPUPSHOW()        (function  in U.PRG)
```

```
3761  *!                : SFSYMP              (procedure in DEMO.PRG)
3762  *!                : POPUPHIDE()         (function  in U.PRG)
3763  *!                : SFDIAG              (procedure in DEMO.PRG)
3764  *!                : SFTREAT             (procedure in DEMO.PRG)
3765  *!                : EDITTEXT            (procedure in DEMO.PRG)
3766  *!
3767  *!*****************************************************************
3753                                          FUNCTION sfreport
3754                                             PARAMETER marea
3755                                             mtext = ""
3756                                             mnogenerate = .F.
3757                                             IF !EMPTY(mreport)
3758                                                mnogenerate = yesno("SF600
 => report exist, Do you want:", "RESUME", "RE-GENERATE")
3759                                             ENDIF
3760                                             IF !mnogenerate
3761                                                winpop = popupshow("Generat
 => e SF600 report....")
3762                                                newline = CHR(13) + CHR(10)
3763                                                me_area = marea
3764                                                mtext = sfsymp(me_area)
3765                                                =popuphide(winpop)
3766                                                maccess = sfdiag()
3767                                                mtreat = sftreat()
3768                                                IF !EMPTY(mtext)
3769                                                   mtext = IIF(!EMPTY(macce
 => ss),mtext+maccess,mtext)
3770                                                   mtext = IIF(!EMPTY(mtrea
 => t),mtext+mtreat,mtext)
3771                                                ENDIF
3772                                             ELSE
3773                                                mtext = mreport
3774                                             ENDIF
3775                                             IF !EMPTY(mtext)
3776                                                =edittext(mtext)
3777                                             ENDIF
3778                                             mreport = mtext
3779  <-------------------------------------    RETURN
3780                                             ******************************
 => ************************************
3781                                             * Symptom Report session in na
 => tural language form
3782                                             * Programmer: HLL
3783                                             ******************************
 => ************************************
3804  *!*****************************************************************
3805  *!
3806  *!      Procedure: SFSYMP
3807  *!
3808  *!      Called by: SFREPORT           (procedure in DEMO.PRG)
3809  *!
3810  *!          Calls: CASECONVERT        (procedure in U.PRG)
3811  *!                 : GETQ             (procedure in DEMO.PRG)
3812  *!                 : DICTREPORT()     (function  in U.PRG)
3813  *!                 : DP.PRG
3814  *!
3815  *!           Uses: SF600TEX.DBF
3816  *!                 : SECTION.DBF
3817  *!                 : REPORT.DBF
3818  *!
3819  *!        Indexes: SF600TEX.IDX
3820  *!                 : SECTION.IDX
3821  *!                 : REPORT.IDX
3822  *!
3823  *!*****************************************************************
3804                                          FUNCTION sfsymp
```

```
3805                                          PARAMETERS marea
3806
3807                                          PRIVATE reportstr, senstr, sec
 => str, xstr, zstr
3808                                          PRIVATE mid
3809                                          PRIVATE msection, msentence, m
 => begin
3810                                          PRIVATE msel
3811
3812                                          msel = SELECT()
3813                                          SELECT 0
3814                                          USE sf600tex INDEX sf600tex
3815                                          SET RELATION TO id INTO dict
3816                                          SELECT 0
3817                                          USE SECTION INDEX SECTION
3818                                          SET RELATION TO sentence INTO
 => sf600tex
3819                                          SELECT 0
3820                                          USE REPORT INDEX REPORT
3821                                          SET RELATION TO SECTION INTO S
 => ECTION
3822                                          SET FILTER TO area = marea
3823                                          GOTO TOP
3824                                          DO CASE
3825                                          CASE msex = "F"
3826                                             msexw = "Female"
3827                                             musex = "She"
3828                                             mupsex = "Her"
3829                                             mlsex = caseconvert(musex,"
 => L")
3830                                             mlpsex = caseconvert(mupsex
 => ,"L")
3831                                          OTHERWISE
3832                                             msexw = "Male"
3833                                             musex = "He"
3834                                             mupsex = "His"
3835                                             mlsex = caseconvert(musex,
 => "L")
3836                                             mlpsex = caseconvert(mupsex
 => , "L")
3837                                          ENDCASE
3838                                          reportstr = ""
3839                                          mp = .F.
3840                                          DO WHILE !EOF()
 =>                  &&report
3841                                             msection = SECTION
3842                                             SELECT SECTION
3843                                             mbegin = .T.
3844                                             secstr = ""
3845                                             DO WHILE SECTION = report->
 => section
3846                                                msentence = sentence
3847                                                SELECT sf600tex
3848                                                prior = 0
3849                                                senstr = ""
3850                                                DO WHILE sentence = sect
 => ion->sentence
3851                                                   xstr = ""
 =>                  && object value
3852                                                   DO CASE
3853                                                   CASE OBJECT = "S"
 =>                  && symptom object:
3854                                                      mid = ALLTRIM( STR
 => ( id ) )         && object identifier
3855                                                      xstr = getq( mid )
 =>                  && fetch value
```

```
3856                                                                    IF !EMPTY( xstr )
3857                                                                      DO CASE
 =>                    && object has a defined value
3858                                                                      CASE dict->data
 => type = "L"
3859                                                                        xstr = dictr
 => eport(mid, xstr)   && logical types or enumerate
3860                                                                      CASE  dict->dat
 => atype = "E"  .OR.  dict->datatype = "N"
3861                                                                        zstr = xstr
 =>                    && enumerated types
3862                                                                        xstr = ""
 =>                    && concatenate report strings
3863                                                                        i = 0
3864                                                                        mp = .F.
3865                                                                        DO WHILE !EM
 => PTY( zstr )
3866                                                                          xstr1 = d
 => p(zstr,"|" 1)
3867                                                                          zstr = dp
 => (zstr," ",2,99)
3868                                                                          phrase =
 => IIF(EMPTY(xstr1),"",dictreport(mid, xstr1))
3869                                                                          IF !EMPTY
 => (phrase)
3870                                                                            i = i
 => + 1
3871                                                                            IF i > 1
3872                                                                              cond =
 =>  IIF(EMPTY(zstr) ," and ", ")
3873                                                                              mp = .
 => T.
3874                                                                            ELSE
3875                                                                              cond =
 =>  ""
3876                                                                            ENDIF
3877                                                                            xstr
 =>  =  IIF( EMPTY(xstr), "", xstr) + cond + phrase
3878                                                                          ENDIF
3879                                                                          ENDDO
3880                                                                      CASE  dic
 => t->datatype = "N"
3881                                                                      IF VAL(xs
 => tr) > 0
3882                                                                        mp = .
 => T.
3883                                                                      ENDIF
3884                                                                        *Numer
 => ic types
3885                                                                      ENDCAS
 => E
3886                                                                      ELSE
3887                                                                        xstr =
 =>  ALLTRIM(DEFAULT)
3888                                                                      ENDIF
3889                                                                        starts
 => tr = ALLTRIM(IIF(prior = 1, noprevsen, starttext))
3890                                                                        endstr
 =>  = ALLTRIM(ENDTEXT)
3891                                                                      IF !EMPTY
 => (xstr)
3892                                                                      IF xstr !
 => = "|"
3893                                                                        xstr =
 =>  ALLTRIM(startstr + " " + xstr + " " + endstr)
3894                                                                      ENDIF
```

```
3895                                                                        xstr =
 =>  STRTRAN(xstr, "SUPSEX", mupsex)
3896                                                                        xstr =
 =>  STRTRAN(xstr, "SUSEX", musex)
3897                                                                        xstr =
 =>  STRTRAN(xstr, "SLSEX", mlsex)
3898                                                                        xstr =
 =>  STRTRAN(xstr, "SLPSEX", mlpsex)
3899                                                                        xstr =
 =>  STRTRAN(xstr, "|", CHR(13) + CHR(10))
3900                                                                      IF mp
3901                                                                        xstr =
 =>  STRTRAN(xstr,"{s}")
3902                                                                        xstr =
 =>  STRTRAN(xstr,")")
3903                                                                        xstr =
 =>  STRTRAN(xstr,"(")
3904                                                                      ELSE
3905                                                                        xstr =
 =>  STRTRAN(xstr,"{s}")
3906                                                                        xstr =
 =>  STRTRAN(xstr,"[")
3907                                                                        xstr =
 =>  STRTRAN(xstr,"]")
3908                                                                      ENDIF
3909                                                                        prior
 => = 0
3910                                                                      IF mbegin
3911                                                                        xstr =
 =>  caseconvert(xstr,"U")
3912                                                                        mbegin
 =>  = .F.
3913                                                                      ENDIF
3914                                                                      ELSE
3915                                                                      IF !mbegi
 => n AND prior = 0
3916                                                                        prior
 =>  = 0
3917                                                                      ELSE
3918                                                                        prior
 => = 1
3919                                                                      ENDIF
3920                                                                      ENDIF
3921                                                                      CASE OBJE
 => CT = "T"
3922                                                                        senstr
 =>  = ALLTRIM(starttext)          &&text string
3923                                                                      ENDCAS
 => E
3924                                                                      IF !EMPTY
 => (xstr)
3925                                                                      IF !(xstr
 =>  $ (CHR(13) + CHR(10)))
3926                                                                        xstr =
 =>   " " + xstr
3927                                                                      ENDIF
3928                                                                        senstr
 =>  = IIF(EMPTY(senstr), xstr, senstr + xstr)
3929                                                                      ENDIF
3930                                                                        SKIP
3931                                                                      ENDDO
3932                                                                      IF !EMPTY
 => (senstr)                          && ending section
3933                                                                        secstr
 =>  = IIF(EMPTY(secstr), "", secstr) + senstr
3934                                                                      ENDIF
```

```
3935                                                                    SELECT
 => SECTION
3936                                                                     SKIP
3937                                                                    └ENDDO
3938                                                                  ┌IF !EMPTY
 => (secstr)
3939                                                                  ┌IF ! ( RI
 => GNT( secstr, ' ) $ ".:;,"  )
3940                                                                    secstr
 => = secstr + "."
3941                                                                    └ENDIF
3942                                                                    report
 => str = IIF(EMPTY(reportstr), secstr, reportstr + "." + secstr)
3943                                                                    └ENDIF
3944                                                                    SELECT
 => REPORT
3945                                                                  ┌IF !EOF()
3946                                                                     SKIP
3947                                                                    └ENDIF
3948                                                                    └ENDDO
3949                                                                    USE
3950                                                                    SELECT
 => SECTION
3951                                                                    USE
3952                                                                    SELECT
 => sf600tex
3953                                                                    USE
3954                                                                    SELECT
 => (msel)
3955  <──────────────────────────────────────────────────────────────RETURN
 => reportstr
3956
3957
3958                                                                    *-----
 => ------------------------------------------------------------------
3959                                                                    * edit
 => memo text
3960                                                                    * Prog
 => rammer: HLL
3961                                                                    *-----
 => ------------------------------------------------------------------
3974  *|*****************************************************************
3975  *|
3976  *|      Procedure: EDITTEXT
3977  *|
3978  *|      Called by: SFREPORT          (procedure in DEMO.PRG)
3979  *|
3980  *|          Calls: PUSHHELP          (procedure in U.PRG)
3981  *|               : WP6.SPR
3982  *|               : POPHELP          (procedure in U.PRG)
3983  *|               : WP7.SPR
3984  *|
3985  *|*****************************************************************
3974                                                                  FUNCTION
 => edittext
3975                                                                    PARAME
 => TER mtext
3976                                                                    mprevs
 => tr = mtext
3977                                                                    mstr =
 => mtext
3978                                                                    mbutto
 => ns = 0
3979                                                                    =pushh
 => elp("SF600 REPORT EDITING WINDOW")
3980                                                                    DO wp6
```

```
 => .spr
3981                                                                    =pophe
 => lp()
3982                                                                    mtext
 => = mstr
3983                                                                  ┌DO CASE
3984                                                                  ├CASE mbut
 => tons =
3985                                                                    =pushh
 => elp("PRINT SF600 REPORT")
3986                                                                    DO wp7
 => .spr
3987                                                                    =pophe
 => lp()
3988                                                                  ├CASE mbut
 => tons = 2
3989                                                                    PRIVAT
 => E msel
3990                                                                    msel =
 => SELECT()                  && save current database selection
3991                                                                    SELECT
 => (fencountr)
3992                                                                    SET OR
 => DER TO 3                  && key is encounter id
3993                                                                    * look
 => up encounter record
3994                                                                    SEEK m
 => encountid
3995                                                                  ┌IF FOUND(
 => )
3996                                                                    REPLAC
 => E REPORT WITH ↳text
3997                                                                    └ENDIF
3998                                                                    SELECT
 => (msel)
3999                                                                  ├OTHERWISE
4000                                                                    mtext
 => = mprevstr
4001                                                                    *Cance
 => l
4002                                                                    └ENDCAS
 => E
4003
4004  <──────────────────────────────────────────────────────────────RETURN
 => mtext
4005
4006                                                                    *-----
 => ------------------------------------------------------------------
4007                                                                    * Repo
 => rt diagnoses
4008                                                                    * Prog
 => rammer: HLL
4009                                                                    *-----
 => ------------------------------------------------------------------
4019  *|*****************************************************************
4020  *|
4021  *|      Procedure: SFDIAG
4022  *|
4023  *|      Called by: SFREPORT          (procedure in DEMO.PRG)
4024  *|
4025  *|          Calls: DISEASENAME()     (function  in ?)
4026  *|
4027  *|*****************************************************************
4019                                                                  FUNCTION
 => sfdiag
4020                                                                    PRIVAT
```

```
 => E mstr
4021                                                                msize
 => = ALEN(mdiag)
4022                                                                DIMENS
 => ION mdiagname[msize]
4023                                                                mstr =
 =>  ""
4024                                                                newlin
 => e = CHR(13)+CHR(10)
4025                                                                ┌IF !EMPTY
 => (mdiagnosis)
4026                                                                mtemp
 => = diseasename(mdiagnosis)
4027                                                                mstr =
 =>  newline + newline + "Impressions:  " + TRIM( PROPER(mtemp))
4028                                                                └ENDIF
4029                                                                *IF al
 => en(mdiag)> 0                                          && HLL05
4030                                                                *
 => =getrulebase(marea, @cthreshold, @cthreshpro, @cthreshlik)
4031                                                                *
 => =strdiag(mdiagname, mdiag, mconf, cthreshold, "", is_conf)
4032                                                                *
 => mstr = mstr + newline + "Computer's suggested diagnoses: "
4033                                                                *
 => FOR i = 1 to alen(mdiagname)
4034                                                                *
 =>        IF type("mdiagname[i]")="C"
4035                                                                *
 =>                mstr =  mstr + newline + trim(proper(mdiagname[i]))
4036                                                                *
 =>        ELSE
4037                                                                *
 =>                EXIT
4038                                                                *
 =>        ENDIF
4039                                                                *
 => ENDFOR
4040                                                                *ENDIF
4041  <──────────────────────────────────────────────────────────────RETURN
 =>  mstr
4042
4043                                                                *-----
 => ------------------------------------------------------------
4044                                                                * Repo
 => rt treatment
4045                                                                * Prog
 => rammer: HLL
4046                                                                *-----
 => ------------------------------------------------------------
4054  *!*****************************************************************
4055  *!
4056  *!      Procedure: SFTREAT
4057  *!
4058  *!      Called by: SFREPORT          (procedure in DEMO.PRG)
4059  *!
4060  *!*****************************************************************
4054                                                                FUNCTION
 => sftreat
4055
4056                                                                PRIVAT
 => E mstr
4057                                                                mstr =
 =>  ""
4058                                                                newlin
 => e = CHR(13) + CHR(10)
```

```
4059                                                                ┌IF !EMPTY
 => (mtreatment)
4060                                                                mstr =
 =>  newline + newline + "Plan:"
4061                                                                mstr =
 =>  mstr + newline + mtreatment
4062                                                                └ENDIF
4063  <──────────────────────────────────────────────────────────────RETURN
 =>  mstr
4064                                                                *-----
 => ------------------------------------------------------------
4065                                                                * Call
 =>  encounter
4066                                                                * Prog
 => rammer: HLL
4067                                                                *-----
 => ------------------------------------------------------------
4079  *!*****************************************************************
4080  *!
4081  *!      Procedure: M_NEWENC
4082  *!
4083  *!          Calls: OPENFILES()        (function  in U.PRG)
4084  *!               : WPAT               (procedure in DEMO.PRG)
4085  *!               : M_ENC              (procedure in DEMO.PRG)
4086  *!               : KEYLOG.PRG
4087  *!               : FLUSHFILES()       (function  in U.PRG)
4088  *!
4089  *!*****************************************************************
4079                                                                FUNCTION
 => m_newenc
4080                                                                =openf
 => iles()
4081                                                                *IF em
 => pty(mpatientid)
4082                                                                DO wpa
 => t
4083                                                                *ENDIF
4084                                                                ┌IF !EMPTY
 => (mpatientid)
4085                                                                =m_enc
 => ()
4086                                                                =keylo
 => g("Encounter")
4087                                                                └ENDIF
4088                                                                =flush
 => files()
4089  <──────────────────────────────────────────────────────────────RETURN
4090                                                                *-----
 => ------------------------------------------------------------
4091                                                                * Call
 =>  Training
4092                                                                * Prog
 => rammer: HLL
4093                                                                *-----
 => ------------------------------------------------------------
4105  *!*****************************************************************
4106  *!
4107  *!      Procedure: M_TRAINING
4108  *!
4109  *!          Calls: OPENFILES()        (function  in U.PRG)
4110  *!               : WPAT               (procedure in DEMO.PRG)
4111  *!               : KEYLOG.PRG
4112  *!               : M_ENC              (procedure in DEMO.PRG)
4113  *!               : FLUSHFILES()       (function  in U.PRG)
4114  *!
4115  *!*****************************************************************
```

```
4105                                                                    FUNCTION
 => m_training
4106                                                                      PUBLIC
 =>  mtrain
4107                                                                      PRIVAT
 => E startdir
4108
4109                                                                      mtrain
 =>  = .F.
4110                                                                      startd
 => ir = CURDIR()
4111                                                                    ┌IF !EMPTY
 => (GETENV("CAMD"))
4112                                                                      mdirda
 => ta = FULLPATH(GETENV("CAMD"))
4113                                                                      SET DE
 => FA TO (mdirdata + "\training")
4114                                                                    ├ELSE
4115                                                                      SET DE
 => FA TO training
4116                                                                    └ENDIF
4117                                                                      =openf
 => iles()
4118                                                                      msamsd
 => ata = .F.
4119                                                                      DO wpa
 => t
4120                                                                    ┌IF !EMPTY
 => (mpatientid)
4121                                                                      =keylo
 => g("Encounter Training Mode")
4122                                                                      =m_enc
 => ()
4123                                                                    └ENDIF
4124                                                                      =flush
 => files()
4125                                                                      SET DE
 => FA TO (startdir)
4126                                                                      mtrain
 =>  = .F.
4127  <──────────────────────────────────────────────────────────────────RETURN
4128
4129                                                                      *-----
 => ------------------------------------------------------------------
4130                                                                      * Back
 => up data files
4131                                                                      * Prog
 => rammer: HLL
4132                                                                      *-----
 => ------------------------------------------------------------------
4143  *!*****************************************************************
4144  *!
4145  *!      Procedure: BACKUP
4146  *!
4147  *!          Calls: PUSHHELP          (procedure in U.PRG)
4148  *!               : BACKUP.SPR
4149  *!               : POPHELP           (procedure in U.PRG)
4150  *!               : KEYLOG.PRG
4151  *!
4152  *!*****************************************************************
4143                                                                    PROCEDURE
 =>  backup
4144                                                                      PUBLIC
 =>  mscreen
4145                                                                      PRIVAT
 => E MESSAGE
```

```
4146                                                                      SAVE S
 => CREEN TO mscreen
4147                                                                      MESSAG
 => E = "Backup CAMD database"
4148                                                                      =pushh
 => elp("BACKUP")
4149                                                                      DO bac
 => kup.spr WITH MESSAGE
4150                                                                      =pophe
 => lp()
4151                                                                      RESTOR
 => E SCREEN FROM mscreen
4152                                                                      =keylo
 => g("Backup")
4153  <──────────────────────────────────────────────────────────────────RETURN
4154
4155                                                                      *-----
 => ------------------------------------------------------------------
4156                                                                      * Rest
 => ore data files
4157                                                                      * Prog
 => rammer: HLL
4158                                                                      *-----
 => ------------------------------------------------------------------
4169  *!*****************************************************************
4170  *!
4171  *!      Procedure: RESTORE
4172  *!
4173  *!          Calls: PUSHHELP          (procedure in U.PRG)
4174  *!               : RESTORE.SPR
4175  *!               : POPHELP           (procedure in U.PRG)
4176  *!               : KEYLOG.PRG
4177  *!
4178  *!*****************************************************************
4169                                                                    PROCEDURE
 =>  RESTORE
4170                                                                      PUBLIC
 =>  mscreen
4171                                                                      PRIVAT
 => E MESSAGE
4172                                                                      SAVE S
 => CREEN TO mscreen
4173                                                                      MESSAG
 => E = "Restore Computer Assisted Medical Diagnose System database"
4174                                                                      =pushh
 => elp("RESTORE")
4175                                                                      DO res
 => tore.spr WITH MESSAGE
4176                                                                      =pophe
 => lp()
4177                                                                      RESTOR
 => E SCREEN FROM mscreen
4178                                                                      =keylo
 => g("Restore")
4179  <──────────────────────────────────────────────────────────────────RETURN
4180                                                                      *-----
 => ------------------------------------------------------------------
4181                                                                      * Dise
 => ase/Treatment Lookup
4182                                                                      * Prog
 => rammer: HLL
4183                                                                      *-----
 => ------------------------------------------------------------------
4194  *!*****************************************************************
4195  *!
4196  *!      Procedure: M_LOOKUP
```

```
4197  *|
4198  *|          Calls: DISEASECNT()       (function  in U.PRG)
4199  *|                : DISNAMES()        (function  in U.PRG)
4200  *|                : GETDISLOOKUPLIST  (procedure in U.PRG)
4201  *|                : KEYLOG.PRG
4202  *|
4203  *|*****************************************************************
4194                                                                PROCEDURE
 => m_lookup
4195                                                                  PRIVAT
 => E mdiag
4196                                                                  PRIVAT
 => E names, N, ndis, mchoice, i
4197                                                                  N = di
 => seasecnt()
4198                                                                  mchoic
 => e = ""
4199                                                                 ┌IF N > 0
4200                                                                 │DIMENS
 => ION names[n]
4201                                                                  ndis =
 =>  disnames(names)                 && create list of names
4202                                                                  = IIF(
 =>  ndis > 0, getdislookuplist(names), 0 )
4203                                                                  = keyl
 => og("Disease/Treatment Reference")
4204                                                                 └ENDIF
4205  <──────────────────────────────────────────────────────────────RETURN
4206                                                                  *-----
 => ------------------------------------------------------------
4207                                                                  * Prin
 => tout the data form
4208                                                                  * Prog
 => rammer: HLL
4209                                                                  *-----
 => ------------------------------------------------------------
4219  *|*****************************************************************
4220  *|
4221  *|      Procedure: M_DATA
4222  *|
4223  *|          Calls: GETDIAGAREA        (procedure in DEMO.PRG)
4224  *|                : QUESTION.PRG
4225  *|                : KEYLOG.PRG
4226  *|
4227  *|*****************************************************************
4219                                                                PROCEDURE
 => m_data
4220                                                                  PRIVAT
 => E newarea
4221                                                                  neware
 => a = getdiagarea()
4222                                                                 ┌IF !EMPTY
 => (newarea)
4223                                                                  = ques
 => tion(newarea)
4224                                                                 └ENDIF
4225                                                                  =keylo
 => g("Print Data Entry Form")
4226  <──────────────────────────────────────────────────────────────RETURN
4228                                                                  *: EOF
 => : DEMO.ACT
```

```
 1  *:*****************************************************************
 2  *:
 3  *: Procedure file: C:\CAMD\WORK\DP.PRG
 4  *:
 5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
 6  *:          Author: ROY DOBBINS and HOA LY
 7  *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
 8  *:  Last modified: 08/04/92     15:10
 9  *:
10  *:          Set by: WP2.PRG
11  *:                : WPAT                (procedure in DEMO.PRG)
12  *:                : SFSYMP              (procedure in DEMO.PRG)
13  *:                : WPGETPATLIST        (procedure in U.PRG)
14  *:                : SEARCH              (procedure in U.PRG)
15  *:                : MORESEARCH          (procedure in U.PRG)
16  *:                : GETPATDATA          (procedure in U.PRG)
17  *:                : SAMSDATA            (procedure in U.PRG)
18  *:                : SAVREC              (procedure in U.PRG)
19  *:                : SFGETPAT            (procedure in U.PRG)
20  *:                : SFGETPATLIST        (procedure in U.PRG)
21  *:                : FILEFIND            (procedure in U.PRG)
22  *:                : RESULT              (procedure in WPATS.PRG)
23  *:                : PATEDIT             (procedure in WPATS.PRG)
24  *:
25  *:       Documented 10/14/92 at 10:52                FoxDoc  version 2.10
26  *:*****************************************************************
27        * ==================================================================
=> ==================
28        * Date: 08/04/92  10:01:15
29        * Program Name: Question.prg
30        * Author's Name: Hoa Le Ly
31        *
32        * Copyright (c) 1992 Company Name: NHRC
33        * Departament: Code 22
34        * San Diego, CA 92138 - 5122
35        * Description: This program emulate $PIECE of MUMPS function.  Which
=> return the
36        *    the portion of string which is bounded by the characters in deli
=> miter. If both
37        *    expr and expr2 are present, the value returned includes all char
=> acters from
38        *    the expr1-1th occurrence of delimiter, up to but not including
=> the expr2th
39        *    occurence of delimiter.  If expr2 is not present. it is assumed
=> to have the same
40        *    value as expr.  If expr1 is not present. Then its value is assu
=> me to be 1
41        * SYNTAX: DP(string, delimiter[,expr[,expr2]])
42        * PARAMETER: string: Character expression which character extract fro
=> m
43        *                 delimiter: Character to delimiter
44        *                     expr: start of number occurrence of delimiter
45        *                     expr2: number of occurrence delimiter
46        * eg: string = "last, first^age^date of birth"
47        *    ?dp(string, "^",1)            ===>  last, first
48        *    ?dp(string,"^",1,2)           ===>  last, first^age
49        * ==================================================================
=> ================
50
51        PARAMETER ms,mm,mp,mp2
52        PRIVATE ALL
53        DO CASE
54        CASE PARAMETER() < 2
55  <-----RETURN ""
56        CASE PARAMETER() = 2
57           mp = 1
58           mp2 = -1
59        CASE PARAMETER() = 3
60           IF TYPE("mp") != "N"
61  <-------RETURN ""
62           ELSE
63              IF mp < 1
64  <----------RETURN ""
65              ENDIF
66           ENDIF
67           mp2 = -1
68        CASE PARAMETER() = 4
69           IF TYPE("mp") != "N"
70  <-------RETURN ""
71           ENDIF
72           IF TYPE("mp2") = "N"
73              mp2 = IIF(mp >= mp2, -1, mp2)
74           ELSE
75              mp2 = -1
76           ENDIF
77        ENDCASE
78        IF TYPE("ms") != "C"
79           ms = STR(ms)
80        ENDIF
81        moccurs = OCCURS(mm,ms)
82        IF moccurs = 0                                    && Return ms if
=>  mp=1&(ms'[mm)
83           mstr = IIF(mp = 1, ms, "")                 && Return null if mp>1&
=> (ms'[mm)
84  <-----RETURN mstr
85        ENDIF
86        mbegin = IIF(mp = 1, 1, (AT(mm,ms,(mp-1))+1))
87        IF mp2 = -1
88           mend = AT(mm,ms,mp) - 1
89           IF mend < 0
90              IF mbegin > 1
91                 mend = LEN(ms)
92              ELSE
93  <----------RETURN ""
94              ENDIF
95           ENDIF
96        ELSE
97           mend = AT(mm,ms,mp2) - 1
98           mend = IIF(mend<0,LEN(ms),mend)
99        ENDIF
100       mstr = SUBSTR(ms, mbegin, (mend - mbegin +1))
101       RETURN mstr
103       *: EOF: DP.ACT
```

```
 1  *:*****************************************************************
 2  *:
 3  *: Procedure file: C:\CAMD\WORK\ERRMSG.PRG
 4  *:
 5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
 6  *:          Author: ROY DOBBINS and HOA LY
 7  *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
 8  *:  Last modified: 07/30/92      16:40
 9  *:
10  *:          Set by: GETPWD.PRG
11  *:                : BROWSEPAT          (procedure in U.PRG)
12  *:                : ERRNAMD            (procedure in U.PRG)
13  *:                : FILEMISSING        (procedure in SECHECK.PRG)
14  *:
15  *:       Documented 10/14/92 at 10:52                FoxDoc  version 2.10
16  *:*****************************************************************
17      * Open a message window
18      *
19      * ERRMSG( <expC>[, <expN>] )
20      *
21      PARAMETERS errmsg, timelimit
22      PRIVATE ALL
23
24      * If no errmessage is sent then quit
25      IF PARAMETERS()=0
26      RETURN
27      ENDIF
28
29      * m.TimeLimit is used in a WAIT TIMEOUT command
30      * to control the amount of time ERRMSG display's its message
31      * If no time limit is received set time to 0 wait forever
32
33      IF PARAMETERS()=1 OR EMPTY(m.timelimit)
34          m.timelimit=0
35      ENDIF
36
37      IF TYPE("m.TimeLimit")!="N"
38          m.timelimit=IIF(TYPE("m.TimeLimit")="C",INT(VAL(m.timelimit)),0)
39      ENDIF
40
41      m.errmsg=IIF(TYPE("m.errmsg")!="C","",m.errmsg)
42
43      IF PARAMETERS()=2 AND EMPTY(m.errmsg) AND m.timelimit>0
44      RETURN
45      ENDIF
46
47      * set talk off
48      IF SET('TALK') = 'ON'   && TALK handled as a special case.
49          SET TALK OFF            && Turn TALK OFF
50          savetalk = 'ON'         && TALK was ON, save the setting
51      ELSE    && TALK is OFF
52          savetalk = 'OFF'        && TALK was OFF, save the setting
53      ENDIF
54
55      * Get the length of the message to size window
56      m.len=LEN(m.errmsg)
57
58      * Set miminum length for the Press any key
59      IF m.timelimit = 0
60          m.len=IIF(m.len<28,28,m.len)
61      ENDIF
62
63      IF m.len >70
64          m.errmsg=SUBSTR(m.errmsg,1,70)
65          m.len=70
66      ENDIF
67
68      *Find begining/ending of the window
69      m.begin=40-(INT(m.len/2)+1)
70      m.end=40+(INT(m.len/2)+1)
71
72      * Remember the current window status
73      m.oldwindow =IIF( WOUTPUT() = errmsg, "" , WOUTPUT() )
74
75      * Create the window
76      DEFINE WINDOW errmsg FROM 20,m.begin TO 23,m.end COLOR SCHEME 7
77      ACTIVATE WINDOW errmsg
78
79      * Print the message centered in the window
80
81      CLEAR
82      @ 0, ( WCOL() - LEN( m.errmsg ) )/2 SAY m.errmsg
83
84      IF m.timelimit = 0
85          m.presskey = "Press any key to continue"
86          @ 1, ( WCOL() - LEN( m.presskey ) )/2 SAY m.presskey
87      ENDIF
88
89      * Wait for the number of seconds in m.Timeout
90      * A value of 0 will wait forever
91      WAIT "" TIMEOUT m.timelimit
92
93      * Close Window
94      RELEASE WINDOW errmsg
95
96      * If there was no output window originally
97      IF EMPTY( m.oldwindow )
98
99          * Send future output back to the screen
100         ACTIVATE SCREEN
101
102     ELSE
103
104         * Return output to the original window
105         ACTIVATE WINDOW ( m.oldwindow )
106     ENDIF
107
108     SET TALK &savetalk    && Restore original TALK setting
109     RETURN
111     *: EOF: ERRMSG.ACT
```

```
 1  *:*********************************************************************
 2  *:
 3  *: Procedure file: C:\CAMD\WORK\GETPWD.PRG
 4  *:
 5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
 6  *:          Author: ROY DOBBINS and HOA LY
 7  *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
 8  *:   Last modified: 09/08/92     13:09
 9  *:
10  *:          Set by: SECHECK.PRG
11  *:
12  *:           Calls: ERRMSG.PRG
13  *:
14  *:       Documented 10/14/92 at 10:52                FoxDoc  version 2.10
15  *:*********************************************************************
16      *---------------------------------------------------------------------
=> ---
17      * Validation the password.  Is boolean returnable
18      *---------------------------------------------------------------------
=>
19      PARAMETER mpass
20      PRIVATE ALL
21      mflag = .T.
22      IF EMPTY(mpass)
23         mpass = ""
24      ENDIF
25      mesc = SET("ESCAPE")
26      mtalk = SET("TALK")
27      SET ESCAPE OFF
28      SET TALK OFF
29      DEFINE WINDOW get_pwd FROM 10,16 TO 16,63 DOUBLE FLOAT NOCLOSE GROW Z
=> OOM
30      ACTIVATE WINDOW get_pwd
31      temp_pass = SPACE (8)
32      @ 2,7 SAY "ENTER PASSWORD:"
33      @ 2,33 GET temp_pass
34      CLEAR GETS
35      @ 2,32 SAY SPACE(1)
36      tpasswrd=[]
37      tcol = 33
38      msuccess = .F.
39      mtry = 0
40      DO WHILE mflag
41         key1 = 0
42         DO WHILE key1 = 0
43            key1 = INKEY()
44         ENDDO
45         IF ( (LASTKEY() >= -9 .AND. LASTKEY() <= -1) OR (LASTKEY() = 16) )
=>    && Ignore F2 to F10 and DEL key
46            key1 = 0
47            LOOP
48         ENDIF
49         IF LASTKEY() = 28   && F1
50            *HELP PASSWORD
51            key1 = 0
52            LOOP
53         ENDIF
54         IF ( (LASTKEY() = 19) OR (LASTKEY() = 127) OR (LASTKEY() = 7 ))
=> && back arrow, back space, or del key
55            IF tcol = 33
56               key1 = 0
57               LOOP
58            ELSE
59               IF tcol > 33
60                  tcol = tcol - 1
61                  @ 2,tcol SAY [ ] COLOR SCHEME 10
62                  @ 2,tcol SAY ""
63            ELSE
64               @ 2,32 SAY SPACE(1)
65            ENDIF
66            tpasswrd = SUBSTR(tpasswrd,1,LEN(tpasswrd) - 1)
67            LOOP
68         ENDIF
69      ENDIF
70      IF ( (LASTKEY() = 27) OR (LASTKEY() = 5) )   && ESC
71         mflag = .F.
72         EXIT
73      ENDIF
74      IF (key1 = 13)
75         IF UPPER(tpasswrd) == UPPER(mpass)
76            msuccess = .T.
77            mflag = .F.
78         ELSE
79            msuccess = .F.
80            mtry = mtry + 1
81            IF mtry > 2
82               mflag = .F.
83               =errmsg("Invalid User, Try next time !!!!!!",1)
84            ELSE
85               =errmsg("Invalid Password, Try again !!",1)
86               @ 2, 33 GET temp_pass
87               CLEAR GETS
88               @ 2, 32 SAY SPACE(1)
89               tpasswrd = []
90               tcol = 32
91            ENDIF
92         ENDIF
93      ELSE
94         tpasswrd = tpasswrd + CHR(key1)
95         @ 2,tcol SAY [*] COLOR SCHEME 10
96      ENDIF
97      tcol = tcol + 1
98   ENDDO
99   DEACTIVATE WINDOW get_pwd
100  RELEASE WINDOW get_pwd
101  SET ESCAPE &mesc
102  SET TALK &mtalk
103  RETURN msuccess
105  *: EOF: GETPWD.ACT
```

```
1  *:*********************************************************************
2  *:
3  *: Procedure file: C:\CAMD\WORK\KEYLOG.PRG
4  *:
5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
6  *:          Author: ROY DOBBINS and HOA LY
7  *:      Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
8  *:  Last modified: 08/03/92     15:46
9  *:
10 *:          Set by: M_PAT              (procedure in DEMO.PRG)
11 *:                : M_NEWENC           (procedure in DEMO.PRG)
12 *:                : M_TRAINING         (procedure in DEMO.PRG)
13 *:                : BACKUP             (procedure in DEMO.PRG)
14 *:                : RESTORE            (procedure in DEMO.PRG)
15 *:                : M_LOOKUP           (procedure in DEMO.PRG)
16 *:                : M_DATA             (procedure in DEMO.PRG)
17 *:
18 *:           Calls: FILEFIND           (procedure in U.PRG)
19 *:
20 *:            Uses: KEYLOG.DBF
21 *:
22 *:         Indexes: KEY.IDX
23 *:                : KEYDATE.IDX
24 *:
25 *:      Documented 10/14/92 at 10:52              FoxDoc  version 2.10
26 *:*********************************************************************
27     * keylog.prg
28     *-----------------------------------------------------
29     *
30     * Computer Assist Medical Diagnosis System
31     *
32     * Revision: 1.0   Date: 92/08/03 11:11:10
33     * Author: Hoa L. LY
34     *-----------------------------------------------------
35     *
36     * This procedure store every action to keylog.dbf.
37     * Keylog.dbf has index on action and date
38     *
39     *-----------------------------------------------------
40     PARAMETER maction
41     PRIVATE mffind, msel
42     msel = SELECT()
43     mffind = filefind('keylog.dbf')
44     IF !mffind
45         RETURN .F.
46     ENDIF
47     IF PARAMETER() > 0
48         SELECT 0
49         USE keylog AGAIN INDEX KEY, keydate
50         APPEND BLANK
51         REPLACE action WITH maction
52         REPLACE DATE  WITH DATE()
53         REPLACE TIME WITH TIME()
54         USE
55     ENDIF
56     SELECT (msel)
57     RETURN
58     *: EOF: KEYLOG.ACT
```

```
 1  *:*********************************************************************
 2  *:
 3  *: Procedure file: C:\CAMD\WORK\PWDEDT.PRG
 4  *:
 5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
 6  *:          Author: ROY DOBBINS and HOA LY
 7  *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
 8  *:  Last modified: 07/29/92      9:25
 9  *:
10  *:          Set by: _09POPFLOY          (procedure in CAMD.MPR)
11  *:                : CAMDTR.MPR
12  *:
13  *:            Uses: SYSTEM.DBF
14  *:
15  *:       Documented 10/14/92 at 10:52                FoxDoc  version 2.10
16  *:*********************************************************************
17      *-----------------------------------------------------------------
=> ---
18      * Validation the password.  Is boolean returnable
19      *
=>
20
21      PRIVATE ALL
22      IF SET("TALK") == "ON"
23         mtalk = "ON"
24      ELSE
25         mtalk = "OFF"
26      ENDIF
27      SET TALK OFF
28      startdir=FULLPATH(SYS(2003))
29      mepssdata = FULLPATH(GETENV("CAMD"))          && get environment string
30      moldpath = SET("PATH")                        && store current path
31      mpath = startdir + ";" + mepssdata            && set path for database
=> files
32      SET PATH TO (mpath)
33      SELECT 0
34      USE SYSTEM AGAIN
35      mflag = .T.
36      mesc = SET("ESCAPE")
37      SET ESCAPE OFF
38      DEFINE WINDOW get_pwd FROM 10,1' TO 16,63 DOUBLE FLOAT NOCLOSE GROW Z
=> OOM COLOR SCHEME 5
39      ACTIVATE WINDOW get_pwd
40      temp_pass = SPACE (8)
41      @ 2,7 SAY "ENTER NEW PASSWORD:"
42      @ 2,33 GET temp_pass
43      CLEAR GETS
44      @ 2,32 SAY SPACE(1)
45      tpasswrd=[]
46      tcol = 33
47      msuccess = .F.
48      mtry = 0
49      DO WHILE mflag
50         key1 = 0
51         DO WHILE key1 = 0
52            key1 = INKEY()
53         ENDDO
54         IF ( (LASTKEY() >= -9 .AND. LASTKEY() <= -1) OR (LASTKEY() = 16) )
=>    && ignore F2 to F10 and DEL key
55            key1 = 0
56            LOOP
57         ENDIF
58         IF LASTKEY() = 28   && F1
59            *HELP PASSWORD
60            key1 = 0
61            LOOP
62         ENDIF
63         IF ( (LASTKEY() = 19) OR (LASTKEY() = 127) )   && back arrow or ba
=> ck space
64            IF tcol = 33
65               key1 = 0
66               LOOP
67            ELSE
68               tcol = tcol - 1
69               @ 2,tcol SAY [ ] COLOR SCHEME 10
70               IF tcol > 33
71                  @ 2,(tcol-1) SAY [ ] COLOR SCHEME 10
72               ELSE
73                  @ 2,32 SAY SPACE(1)
74               ENDIF
75               tpasswrd = SUBSTR(tpasswrd,1,LEN(tpasswrd) - 1)
76               LOOP
77            ENDIF
78         ENDIF
79         IF ( (LASTKEY() = 27) OR (LASTKEY() = 5) )   && ESC
80            mflag = .F.
81            EXIT
82         ENDIF
83         IF (key1 = 13)
84            IF !EMPTY(tpasswrd) AND LEN(tpasswrd) < 10
85               mpass = UPPER(tpasswrd)
86               REPLACE password WITH mpass
87               mflag = .F.
88            ENDIF
89         ELSE
90            tpasswrd = tpasswrd + CHR(key1)
91            @ 2,tcol SAY [*] COLOR SCHEME 10
92         ENDIF
93         tcol = tcol + 1
94      ENDDO
95      DEACTIVATE WINDOW get_pwd
96      RELEASE WINDOW get_pwd
97      USE
98      SET ESCAPE &mesc
99      SET TALK &mtalk
100     RETURN msuccess
102     *: EOF: PWDEDT.ACT
```

```
 1   *
 2   *
 3   *        07/20/92              Q2.SPR                10:01:15
 4   *
 5   *
 6   *
 7   *        Author's Name
 8   *
 9   *        Copyright (c) 1992 Company Name
10   *        Address
11   *        City,     Zip
12   *
13   *        Description:
14   *        This program was automatically generated by GENSCRN.
15   *        Return variable: Name of the file
16   *

17
18   #REGION 1
19   PARAMETERS mfilename
20   IF PARAMETERS() = 0
21      mfilename = ""
22   ENDIF
23
24   #REGION 0
25   REGIONAL m.currarea, m.talkstat, m.compstat
26
27   IF SET("TALK") = "ON"
28      SET TALK OFF
29      m.talkstat = "ON"
30   ELSE
31      m.talkstat = "OFF"
32   ENDIF
33   m.compstat = SET("COMPATIBLE")
34   SET COMPATIBLE FOXPLUS
35
36   *
37   *
38   *        Window definitions
39   *
40   *
41   *

42
43   IF NOT WEXIST("w_prn")
44      DEFINE WINDOW w_prn ;
45            FROM INT((SROW()-13)/2),INT((SCOL()-43)/2) ;
46            TO INT((SROW()-13)/2)+12,INT((SCOL()-43)/2)+42 ;
47            TITLE "Print Question list" ;
48            FLOAT ;
49            NOCLOSE ;
50            SHADOW ;
51            DOUBLE ;
52            COLOR SCHEME 5
53   ENDIF
54
55
56   *
57   *
58   *        W2 Setup Code - SECTION 2
59   *
60   *
61   *

62
63   #REGION 1
64   PRIVATE mchoice, mprintfile, mfilename, i
65   mchoice = 1
66   mprintfile = mfilename
```



```
67
68   *
69   *
70   *        W2 Screen Layout
71   *
72   *
73   *

74
75   #REGION 1
76   IF WVISIBLE("w_prn")
77      ACTIVATE WINDOW w_prn SAME
78   ELSE
79      ACTIVATE WINDOW w_prn NOSHOW
80   ENDIF
81   @ 0,0 TO 3,40
82   @ 5,0 TO 8,40
83   @ 6,1 SAY "Filename:"
84   @ 6,11 GET mprintfile ;
85       SIZE 1,29 ;
86       DEFAULT " " ;
87       VALID _q7i0lh8ot()
88   @ 10,9 GET mbuttons ;
89       PICTURE "@*HT OK;\?Cancel" ;
90       SIZE 1,8,5 ;
91       DEFAULT 1 ;
92       VALID _q7i0lh8wd()
93   @ 1,24 GET mchoice ;
94       PICTURE "@*RVN File;Printer" ;
95       SIZE 1,11,0 ;
96       DEFAULT 1 ;
97       VALID  q7i0lh94q()
98   @ 1,1 SAY "Print Encounter to:"
99
100  IF NOT WVISIBLE("w_prn")
101     ACTIVATE WINDOW w_prn
102  ENDIF
103
104  READ CYCLE MODAL
105
106  RELEASE WINDOW w_prn
107
108  #REGION 0
109  IF m.talkstat = "ON"
110     SET TALK ON
111  ENDIF
112  IF m.compstat = "ON"
113     SET COMPATIBLE ON
114  ENDIF
115
116  RETURN mprintfile
117
118  *
119  *
120  *        _Q7I0LH8OT            mprintfile VALID
121  *
122  *        Function Origin:
123  *
124  *        From Screen:          W2,     Record Number:    5
125  *        Variable:             mprintfile
126  *        Called By:            VALID Clause
127  *        Object Type:          Field
128  *        Snippet Number:       1
129  *
130  *
131  *

132  FUNCTION _q7i0lh8ot      && mprintfile VALID
```

```
133     #REGION 1
134     mfilename = mprintfile
135     RETURN .T.
136
137     *
138     *
139     *          _Q7IOLH8WD            mbuttons VALID
140     *
141     *          Function Origin:
142     *
143     *          From Screen:          W2,     Record Number:    6
144     *          Variable:             mbuttons
145     *          Called By:            VALID Clause
146     *          Object Type:          Push Button
147     *          Snippet Number:       2
148     *
149     *
150     *
151     FUNCTION _q7i0lh8wd      && mbuttons VALID
152    ┌IF mbuttons = 2
153    │   mprintfile = ""
154    ├ELSE
155    │   mprintfile = mfilename
156    └ENDIF
157     RETURN
158
159
160     *
161     *
162     *          _Q7IOLH94Q            mchoice VALID
163     *
164     *          Function Origin:
165     *
166     *          From Screen:          W2,     Record Number:    7
167     *          Variable:             mchoice
168     *          Called By:            VALID Clause
169     *          Object Type:          Radio Button
170     *          Snippet Number:       3
171     *
172     *
173     *
174     FUNCTION _q7i0lh94q      && mchoice VALID
175     #REGION 1
176    ┌IF mchoice = 1
177    │   mprintfile = mfilename
178    │    CUROBJ = OBJNUM(mprintfile)
179    ├ELSE
180    │   mprintfile = "PRINT"
181    └ENDIF
182     SHOW GETS
183     mfilename = mprintfile
184     RETURN .T.
186     *: EOF: Q2.ACT
```

```
 1  *:*****************************************************************
 2  *:
 3  *: Procedure file: C:\CAMD\WORK\QUESTION.PRG
 4  *:
 5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
 6  *:          Author: ROY DOBBINS and HOA LY
 7  *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
 8  *:  Last modified: 08/04/92     14:48
 9  *:
10  *:  Procs & Fncts: PRINTQUES()
11  *:                : WRITE
12  *:                : OPENFILE
13  *:
14  *:          Set by: M_DATA              (procedure in DEMO.PRG)
15  *:
16  *:           Calls: Q2.PRG
17  *:                : PRINTQUES()         (function  in QUESTION.PRG)
18  *:
19  *:       Documented 10/14/92 at 10:52                FoxDoc  version 2.10
20  *:*****************************************************************
21      * ==================================================================
=> ==================
22      * Date: 08/04/92  10:01:15
23      * Program Name: Question.prg
24      * Author's Name: Hoa Le Ly
25      *
26      * Copyright (c) 1992 Company Name: NHRC
27      * Departament: Code 22
28      * San Diego, CA 92138 - 5122
29      * Description: This program print out the list of question for each
30      *              area input
31      * PARAMETER: area id (numeric type)
32      * ==================================================================
=> ================
33
34      PARAMETER marea
35      PUBLIC mprintfile
36
37      mprintfile = ""
38      mbuttons = 0
39      mudf = SET("UDFPARMS")
40      SET UDFPARMS TO REFERENCE
41      mprintfile = q2(mprintfile)
42
43     ┌IF mbuttons = 2
44  <──────RETURN
45     └ENDIF
46      SET UDFPARMS TO &mudf
47
48      =printques(mprintfile)
49
50      CLOSE DATABASE
51      RETURN
52      *-----------------------------------------------------
53      *
54      * write the question and it's alternate of reponse
55      *
56      *-----------------------------------------------------
66  *!*****************************************************************
67  *!
68  *!      Procedure: WRITE
69  *!
70  *!      Called by: PRINTQUES()        (function  in QUESTION.PRG)
71  *!
72  *!          Calls: OPENFILE           (procedure in QUESTION.PRG)
73  *!
```

```
74  *!*****************************************************************
66      FUNCTION write
67      PRIVATE msel
68
69      =openfile()
70
71      SELECT area
72      SEEK (marea)
73     ┌IF FOUND()
74     │   ? PADC(UPPER(name),80)
75     │   ?
76     └ENDIF
77      SELECT area
78      SET RELATION TO area INTO rule
79      SELECT dict
80      SET RELATION TO id INTO enum, id INTO dictval
81      SELECT rule
82      SET FILTER TO area = marea                && select rule segment
83      SET RELATION TO premise INTO premise, action INTO action
84
85      GO TOP
86      pid = 0
87
88     ┌DO WHILE !EOF()
89     │   SELECT premise
90     │  ┌IF premise.id <> pid
91     │  │   mid = premise.id
92     │  │   SELECT dict
93     │  │  ┌IF dict.askable
94     │  │  │   ?
95     │  │  │   ? ALLTRIM(question)
96     │  │  │  ┌DO CASE
97     │  │  │  ├CASE dict.datatype != "N"
98     │  │  │  │   SELECT enum
99     │  │  │  │  ┌DO WHILE id = mid
100    │  │  │  │  │   ? SPACE(10) + ALLTRIM(enumerate)
101    │  │  │  │  │   SKIP
102    │  │  │  │  └ENDDO
103    │  │  │  ├CASE dict.datatype = "N"
104    │  │  │  │   SELECT dictval
105    │  │  │  │   ? SPACE(10) + "Range" + SPACE(1) + ALLTRIM(STR(lo)) + " t
=> o " + ALLTRIM(STR(hi)) + SPACE(1) + units
106    │  │  │  └ENDCASE
107    │  │  └ENDIF
108    │  │   pid = mid
109    │  └ENDIF
110    │   SELECT rule
111    │   SKIP
112    └ENDDO
113     RETURN
114
115     *-----------------------------------------------------
116     *
117     * open rule file
118     *
119     *-----------------------------------------------------
145 *!*****************************************************************
146 *!
147 *!      Procedure: OPENFILE
148 *!
149 *!      Called by: WRITE              (procedure in QUESTION.PRG)
150 *!
151 *!           Uses: AREA.DBF
152 *!               : DICT.DBF
153 *!               : VAL.DBF                  Alias: DICTVAL
154 *!               : ENUM.DBF
```

```
155  *!                 : PREMISE.DBF
156  *!                 : ACTION.DBF
157  *!                 : RULE.DBF
158  *!
159  *!          Indexes: AREA.IDX
160  *!                 : DICTID.IDX
161  *!                 : DICTNAME.IDX
162  *!                 : VAL.IDX
163  *!                 : ENUM.IDX
164  *!                 : PREMISE.IDX
165  *!                 : ACTION.IDX
166  *!                 : SALIENCE.IDX
167  *!                 : RULE.IDX
168  *!
169  *!*********************************************************************
145      FUNCTION openfile
146
147      SELECT 0
148      USE area AGAIN INDEX area
149      SELECT 0
150      USE dict AGAIN INDEX dictid,dictname
151      SELECT 0
152      USE VAL AGAIN INDEX VAL ALIAS dictval
153      SELECT 0
154      USE enum AGAIN INDEX enum
155      SELECT 0
156      USE premise INDEX premise                && list of rule premises
157      SET RELATION TO id INTO dict             && dictionary object associa
=> ted
158      SELECT 0
159      USE action INDEX action                  && list of rule actions
160      SELECT 0
161      USE rule INDEX salience,rule             && list of rules
162      RETURN .T.
163
164      *-----------------------------------------------------
165      *
166      * print
167      *
168      *-----------------------------------------------------
169
179  *!*********************************************************************
180  *!
181  *!        Function: PRINTQUES()
182  *!
183  *!       Called by: QUESTION.PRG
184  *!
185  *!           Calls: WRITE              (procedure in QUESTION.PRG)
186  *!
187  *!*********************************************************************
179      FUNCTION printques
180      PARAMETER mprintfile
181      mprintfile = ALLTRIM( mprintfile )
182      IF !EMPTY( mprintfile )
183         * list to printer or file
184         IF mprintfile  != "PRINT"
185            * list to file
186            SET PRINTER TO (mprintfile)
187         ENDIF
188         SET CONSOLE OFF
189         SET PRINTER ON
190         = write()
191         IF mprintfile = "PRINT"
192            EJECT
193         ENDIF
194         SET PRINTER OFF
195         SET PRINTER TO
196         SET CONSOLE ON
197      ENDIF
198      RETURN .T.
199
201      *: EOF: QUESTION.ACT
```

```
 1   *
 2   *
 3   *        10/09/92            RESTORE.SPR                14:10:03
 4   *
 5   *
 6   *
 7   *        Author's Name
 8   *
 9   *        Copyright (c) 1992 Company Name
10   *        Address
11   *        City,     Zip
12   *
13   *        Description:
14   *        This program was automatically generated by GENSCRN.
15   *
16   *
17
18
19   *
20   *
21   *                RESTORE Setup Code - SECTION 1
22   *
23   *
24   *
25
26   #REGION 1
27   PARAMETERS MESSAGE, wildcard, filename
28   PRIVATE mfanme, mpath, olddrive, predrive, oldpath, maction, mdrive,
=> mfiles
29   DO CASE
30   CASE PARAMETERS() = 0
31       m.message = ""
32       m.wildcard = "*.zip"
33       mfname = ""
34   CASE PARAMETERS() = 1 OR EMPTY(m.wildcard)
35       m.wildcard = "*.zip"
36       mfname = ""
37   CASE PARAMETERS() = 2 OR EMPTY(m.filename)
38       mfname = ""
39   OTHERWISE
40       mfname = filename
41   ENDCASE
42
43
44   #REGION 0
45   REGIONAL m.currarea, m.talkstat, m.compstat
46
47   IF SET("TALK") = "ON"
48       SET TALK OFF
49       m.talkstat = "ON"
50   ELSE
51       m.talkstat = "OFF"
52   ENDIF
53   m.compstat = SET("COMPATIBLE")
54   SET COMPATIBLE FOXPLUS
55
56   *
57   *
58   *                Window definitions
59   *
60   *
61   *
62
63   IF NOT WEXIST("w_restore")
64       DEFINE WINDOW w_restore ;
65           FROM INT((SROW()-18)/2),INT((SCOL()-61)/2) ;
66           TO INT((SROW()-18)/2)+17,INT((SCOL()-61)/2)+60 ;
67           NOFLOAT ;
68           NOCLOSE ;
69           SHADOW ;
70           DOUBLE ;
71           COLOR SCHEME 5
72   ENDIF
73
74
75   *
76   *
77   *                RESTORE Setup Code - SECTION 2
78   *
79   *
80   *
81
82   #REGION 1
83   m.olddrive = SET("DEFAULT")
84   m.prevdrive = m.olddrive
85   m.oldpath = CURDIR()
86   maction = 1
87
88   DECLARE drivearray[1,1]
89   mdrive = 1
90   DO newdrivepop
91
92   DECLARE patharray[1,1]
93   mpath = 1
94   DO newpathpop
95
96   DECLARE filearray[1,1]
97   mfiles = 1
98   DO newfilepop
99
100
101  =closefiles()
102
103
104  *
105  *
106  *                RESTORE Screen Layout
107  *
108  *
109  *
110
111  #REGION 1
112  IF WVISIBLE("w_restore")
113      ACTIVATE WINDOW w_restore SAME
114  ELSE
115      ACTIVATE WINDOW w_restore NOSHOW
116  ENDIF
117  @ 2,40 GET mdrive ;
118      PICTURE "@^" ;
119      FROM drivearray ;
120      SIZE 3,18 ;
121      DEFAULT 1 ;
122      WHEN _q9r0ud7na() ;
123      VALID _q9r0ud7u5() ;
124      COLOR SCHEME 5, 6
125  @ 6,28 SAY "Directory:"
126  @ 0,0 SAY MESSAGE ;
127      SIZE 1,40
128  @ 5,40 GET mpath ;
129      PICTURE "@^" ;
130      FROM patharray ;
131      SIZE 3,18 ;
```

```
132          DEFAULT 1 ;
133          VALID  q9r0ud888() ;
134          COLOR SCHEME 5, 6
135      @ 9,45 GET maction ;
136          PICTURE "@*VT \i\<Restore;\<Cancel" ;
137          SIZE 1,9,1 ;
138          DEFAULT 1 ;
139          VALID  _q9r0ud8g7()
140      @ 2,1 GET mfile ;
141          PICTURE "@&N" ;
142          FROM filearray ;
143          SIZE 11,26 ;
144          DEFAULT 1 ;
145          VALID  q9r0ud8p2() ;
146          COLOR SCHEME 6
147      @ 14,19 GET mfname ;
148          SIZE 1,39 ;
149          DEFAULT " " ;
150          VALID  _q9r0ud8zo()
151      @ 14,0 SAY "Restore file name: "
152      @ 3,28 SAY "From drive: "
153
154      IF NOT WVISIBLE("w_restore")
155          ACTIVATE WINDOW w_restore
156      ENDIF
157
158      READ CYCLE MODAL
159
160      RELEASE WINDOW w_restore
161
162      #REGION 0
163      IF m.talkstat = "ON"
164          SET TALK ON
165      ENDIF
166      IF m.compstat = "ON"
167          SET COMPATIBLE ON
168      ENDIF
169
170
171      *
172      *
173      *                    RESTORE Cleanup Code
174      *
175      *
176      *
177
178      #REGION 1
179      SET DEFAULT TO (m.olddrive + m.oldpath)
180      RETURN
181
182      ************************************************************
183
184      *********************
185      PROCEDURE newdrivepop
186      *********************
187      * Create a popup with each of the legal drive names
188      * The popup will be based on an array of the drive names
189      PRIVATE olddefault, olderror, drivename, ERROR
190
191      * We will change drives, so remember where we started
192      m.olddefault = SET( "DEFAULT" )
193
194      * To test for legal drives this program SET DEFAULT TO each drive
195      * If no error occurs the drive name will be added to an array
196
197      * Create the error trap
198      m.olderror = ON( "ERROR" )
199      m.error = .F.
200      ON ERROR m.error = .T.
201
202      * Create the array of legal drive names
203      DECLARE drivearray[ 1 ]
204      drivearray[ 1 ] = ""
205
206      * Loop from A to Z
207      m.drivename = "A"
208      FOR i = 1 TO 26
209
210          * Try to switch to the drive
211          SET DEFAULT TO (m.drivename)
212
213          * If it worked
214          IF NOT m.error
215
216              * Add the name to the last element in the array
217              drivearray[ ALEN( DriveArray ) ] = m.drivename + ":"
218
219              * Add another element at the end of the array
220              DECLARE drivearray[ ALEN( DriveArray ) + 1 ]
221
222          ENDIF
223
224          * Change to the next letter in the alphabet
225          m.drivename = CHR( ASC( m.drivename ) + 1 )
226
227          * Reset the error trap
228          m.error = .F.
229      NEXT
230
231      * If the first element is empty, no drives were found
232      IF EMPTY( drivearray[ 1 ] )
233          SHOW GET m.drive disabled
234      ELSE
235          * Drives were found so cut off the last empty element
236          * added to the array in the loop
237          DECLARE drivearray[ ALEN( DriveArray ) - 1 ]
238      ENDIF
239
240      * Reset the error trap and return to home
241      ON ERROR &olderror
242      SET DEFAULT TO (m.olddefault)
243
244      * Initialize the popup variable to the element in the
245      * drive array which contains the current drive
246      mdrive = ASCAN( drivearray, m.olddefault )
247      RETURN
248
249      *********************
250      PROCEDURE newpathpop
251      *********************
252      * Create a popup with one bar for each subdirectory
253      * in the current path
254      PRIVATE dirstring, dircount
255
256      * Base the popup on an array with one element for
257      * each subdirectory in the current path
258      *
259      * Start the array with the current drive
260      DECLARE patharray[ 1 ]
261      patharray[ 1 ] = SET( "DEFAULT" )
262
263      * Get the current path string
```

```
264     m.dirstring = CURDIR()
265
266     * If we are not in the root directory
267     IF LEN( m.dirstring ) > 1
268
269        * Start parsing the path string for each directory name
270        m.dircount = 1
271
272        * Continue as long as there is a pair of slashes
273        * Surrounding the next part of the string
274        DO WHILE AT( "\", m.dirstring, m.dircount ) <> 0 AND ;
275                 AT( "\", m.dirstring, m.dircount + 1 ) <> 0
276
277           * Add another element at the end of the array
278           DECLARE patharray[ m.DirCount + 1 ]
279
280           * Cut out the next set of characters between the slashes
281           m.firstchar = AT( "\", m.dirstring, m.dircount ) + 1
282
283           m.pathlength = AT( "\", m.dirstring, m.dircount + 1 )  ;
284               - m.firstchar
285
286           * And add them to the next array element
287           patharray[ m.DirCount + 1 ] = SUBSTR( m.dirstring, ;
288               m.firstchar, m.pathlength )
289
290           * Look for the next directory name in the path string
291           m.dircount = m.dircount + 1
292
293        ENDDO
294     ENDIF
295
296     * Point the popup variable at the last element in the array
297     mpath = ALEN( patharray )
298     RETURN
299
300     ********************
301     PROCEDURE newfilepop
302     ********************
303
304     * Create a popup with all of the file names and its subdirection in t
=> he current directory
305     * This will be based on an array as well
306
307     * Create an array with all the files matching the current wild card
308     * ADIR() creates an array with 5 columns.
309
310     PRIVATE startsort
311     * Fill an array with directory names only
312     =ADIR( dirarray, "", "D")
313     SIZE = ALEN(dirarray,1)
314     IF dirarray[ 1, 1 ] = "."
315        =ADEL(dirarray,1)
316        SIZE = SIZE - 1
317        DECLARE dirarray[size,5]
318     ELSE
319        * We must be in the root directory "\"
320        * Add one more row to the directory array
321        SIZE = SIZE + 1
322        DECLARE dirarray[ size, 5 ]
323
324        * Push all the rows down by one
325        =AINS( dirarray, 1 )
326
327        * Fill in the first directory name in the array
328        * a bug in the popups makes it refuse to display a
329        * prompt of "\", use "\\" to make one \ appear
330        * even then the bar will automatically be non-selectable
331        * which in this case is fine
332        dirarray[ 1,1 ] = "\\"
333
334        * Start the sort after the root name
335     ENDIF
336
337     IF ALEN( dirarray, 1 ) > 2
338
339        * Sort the array starting at the starting row
340        =ASORT( dirarray, AELEMENT( dirarray, 2, 1 ) )
341     ENDIF
342     FOR i = 1 TO ALEN(dirarray,1)
343        dirarray[i,1] = "[" + dirarray[i,1] + "]"
344     ENDFOR
345
346     IF ADIR(filearray, m.wildcard) = 0
347        SIZE = ALEN(dirarray,1)
348        DECLARE filearray[size,5]
349        =ACOPY(dirarray, filearray)
350     ELSE
351        stop = ALEN(dirarray,1)
352        FOR i = 1 TO stop
353           SIZE = ALEN(filearray,1)
354           DECLARE filearray[size + 1,5]
355           =AINS(filearray,1)
356           filearray[1,1] = dirarray(ALEN[dirarray,1],1)
357           IF ALEN(dirarray,1) > 1
358              DECLARE dirarray[alen(dirarray,1)-1,5]
359           ENDIF
360        ENDFOR
361     ENDIF
362     mfile = 1
363     RETURN
364
365     ********************
366     FUNCTION pathstring
367     ********************
368     * Convert an array of subdirectories, such as PathArray,
369     * into a legal path string for use with SET DEFAULT
370     * Use the current value of the path popup as the
371     * ending point on the path
372     PRIVATE ppath
373
374     * Start with the drive name
375     m.ppath = patharray[ 1 ]
376
377     * If the path popup is pointing to a subdirectory
378     IF mpath > 1
379
380        * Add all the subdirectories to the path string
381        FOR i = 2 TO mpath
382           m.ppath = m.ppath + "\" + patharray[ i ]
383
384        NEXT
385     ENDIF
386
387     * End the path with one last slash for good luck
388     m.ppath = m.ppath + "\"
389
390     RETURN m.ppath
391
392     ********************
393     PROCEDURE pkunzip
394     ********************
```

```
395
396     PRIVATE CURDIR, datadir, msel
397     IF !EMPTY(GETENV("CAMDUTIL"))
398        DO setpath WITH GETENV("CAMDUTIL")
399     ENDIF
400     CURDIR = FULLPATH(CURDIR())
401     IF !EMPTY(GETENV("CAMD"))
402        datadir = FULLPATH(GETENV("CAMD"))
403     ELSE
404        datadir = CURDIR()
405     ENDIF
406     desdir = pathstring() + mfname
407     SET DEFA TO (datadir)
408     msel = SELECT()
409     SELECT 0
410     USE backup AGAIN
411     COPY TO t0000000.txt TYPE SDF
412     IF !EMPTY(GETENV("CAMDUTIL"))
413        pkzipcom = "!" + GETENV("CAMDUTIL") + "\PKUNZIP -O &desdir @t00000
=> 00.txt"
414     ELSE
415        pkzipcom = "!PKUNZIP -O &desdir @t0000000.txt"
416     ENDIF
417     &pkzipcom
418     USE
419     DELETE FILE t0000000.txt
420     SET DEFA TO (CURDIR)
421
422     RETURN
423
424     *
425     *
426     *        _Q9R0UD7NA          mdrive WHEN
427     *
428     *        Function Origin:
429     *
430     *        From Screen:        RESTORE,     Record Number:    2
431     *        Variable:           mdrive
432     *        Called By:          WHEN Clause
433     *        Object Type:        Popup
434     *        Snippet Number:     1
435     *
436     *
437     *
438     FUNCTION _q9r0ud7na      && mdrive WHEN
439     #REGION 1
440     m.prevdrive = mdrive
441
442     *
443     *
444     *        _Q9R0UQ7U5          mdrive VALID
445     *
446     *        Function Origin:
447     *
448     *        From Screen:        RESTORE,     Record Number:    2
449     *        Variable:           mdrive
450     *        Called By:          VALID Clause
451     *        Object Type:        Popup
452     *        Snippet Number:     2
453     *
454     *
455     *
456     *Switch to the selected drive
457     FUNCTION _q9r0ud7u5      && mdrive VALID
458     #REGION 1
459     PRIVATE newdrive,mready
```

```
460
461     *Convert the popup bar number into the matching drive name
462     m.newdrive = drivearray[mdrive]
463
464     IF UPPER(m.newdrive) $ "A:B:"
465        mready = yesno("Please insert disk into drive " + m.newdrive, "rea
=> dy","cancel")
466     ELSE
467        mready = .T.
468     ENDIF
469     IF mready
470        *Go there and reset all the other popups to match
471        SET DEFAULT TO (m.newdrive)
472        DO newpathpop
473        DO newfilepop
474     ELSE
475        mdrive =  m.prevdrive
476        m.newdrive = drivearray[mdrive]
477     ENDIF
478     SHOW GETS
479
480     *
481     *
482     *        _Q9R0UD888          mpath VALID
483     *
484     *        Function Origin:
485     *
486     *        From Screen:        RESTORE,     Record Number:    5
487     *        Variable:           mpath
488     *        Called By:          VALID Clause
489     *        Object Type:        Popup
490     *        Snippet Number:     3
491     *
492     *
493     *
494     FUNCTION _q9r0ud888      && mpath VALID
495     #REGION 1
496     m.newdefault = pathstring()
497     SET DEFAULT TO (m.newdefault)
498     DO newpathpop
499     DO newfilepop
500     SHOW GETS
501
502     *
503     *
504     *        _Q9R0UD8G7          maction VALID
505     *
506     *        Function Origin:
507     *
508     *        From Screen:        RESTORE,     Record Number:    6
509     *        Variable:           maction
510     *        Called By:          VALID Clause
511     *        Object Type:        Push Button
512     *        Snippet Number:     4
513     *
514     *
515     *
516     FUNCTION _q9r0ud8g7      && maction VALID
517     #REGION 1
518     DO CASE
519     CASE maction = 1
520        IF FILE(mfname)
521           DO pkunzip
522        ENDIF
523     OTHERWISE
524     ENDCASE
```

```
525
526     *
527     *
528     *         _Q9R0UD8P2          mfile VALID
529     *
530     *         Function Origin:
531     *
532     *         From Screen:        RESTORE,     Record Number:    7
533     *         Variable:           mfile
534     *         Called By:          VALID Clause
535     *         Object Type:        List
536     *         Snippet Number:     5
537     *
538     *
539     *
540     FUNCTION _q9r0ud8p2      && mfile VALID
541     #REGION 1
542     mnewfile = filearray[mfile,1]
543    ┌IF "(" $ mnewfile
544    │   mnewpath = SUBSTR(mnewfile,2,LEN(mnewfile)-2)
545    │   SET DEFA TO (mnewpath)
546    │   DO newpathpop
547    │   DO newfilepop
548    │   SHOW GETS
549    ├ELSE
550    │   mfname = mnewfile
551    │   SHOW GETS
552    └ENDIF
553
554     *
555     *
556     *         _Q9R0UD8Z0          mfname VALID
557     *
558     *         Function Origin:
559     *
560     *         From Screen:        RESTORE,     Record Number:    8
561     *         Variable:           mfname
562     *         Called By:          VALID Clause
563     *         Object Type:        Field
564     *         Snippet Number:     6
565     *
566     *
567     *
568     FUNCTION _q9r0ud8zo      && mfname VALID
569     #REGION 1
570    ┌IF !EMPTY(mfname)
571    │ ┌IF !FILE(mfname)
572    │ │   mfname = ""
573    │ └ENDIF
574    └ENDIF
576     *: EOF: RESTORE.AC1
```

```
 1  *:*****************************************************************
 2  *:
 3  *: Procedure file: C:\CAMD\WORK\SECHECK.PRG
 4  *:
 5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
 6  *:          Author: ROY DOBBINS and HOA LY
 7  *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
 8  *:  Last modified: 09/03/92     12:01
 9  *:
10  *:  Procs & Fncts: GETSYSCODE()
11  *:                : SYSWARN()
12  *:                : SYSCLAIM()
13  *:                : FILEMISSING
14  *:
15  *:          Set by: DEMO.PRG
16  *:
17  *:           Calls: GETSYSCODE()       (function  in SECHECK.PRG)
18  *:                : SYSWARN()          (function  in SECHECK.PRG)
19  *:                : GETPWD.PRG
20  *:                : BIGCHARS.PRG
21  *:                : SYSCLAIM()         (function  in SECHECK.PRG)
22  *:
23  *:       Documented 10/14/92 at 10:52              FoxDoc  version 2.10
24  *:*****************************************************************
25      *-----------------------------------------------------------------
=> ----------------
26      * Security check for user
27      *-----------------------------------------------------------------
=> ----------------
28      PUBLIC msyswarning, mprivateact, mdisclaim, mpassword
29      PRIVATE moldpath, mpath
30      IF SET("TALK")="ON"
31         SET TALK OFF
32         mtalk = "ON"
33      ELSE
34         mtalk = "OFF"
35      ENDIF
36
37      startdir=FULLPATH(SYS(2003))
38      mepssdata = FULLPATH(GETENV("CAMD"))        && get environment string
39      moldpath = SET("PATH")                      && store current path
40      mpath = startdir + ";" + mepssdata          && set path for database
=> files
41      SET PATH TO (mpath)
42      =getsyscode(@msyswarning, @mprivateact, @mdisclaim, @mpassword)
43      msyswin=syswarn(msyswarning)
44      muservalid = getpwd(mpassword)
45      RELEASE WINDOW &msyswin
46      IF muservalid
47         =bigchars(6,17,"CAMD",6)
48         =sysclaim(mdisclaim)
49      ENDIF
50      SET PATH TO (moldpath)                      && store back previous pa
=> th
51      SET TALK &mtalk
52      RETURN muservalid
53
54      *-----------------------------------------------------------------
=> ----------------
55      * Print warning text to screen
56      *-----------------------------------------------------------------
=> ----------------
57
65  *!*****************************************************************
66  *!
67  *!       Function: SYSWARN()
68  *!
69  *!       Called by: SECHECK.PRG
70  *!
71  *!*****************************************************************
65      FUNCTION syswarn
66      PARAMETER msyswar
67      IF PARAMETER() = 0
68         RETURN
69      ENDIF
70      mtext = msyswar
71      DEFINE WINDOW syswarn FROM 1,0 TO 6,76 NOFLOAT NOCLOSE COLOR SCHEME 7
72      ACTIVATE WINDOW syswarn SAME
73      @ 0,1 SAY mtext ;
74         SIZE 4,73
75      RETURN "syswarn"
76
77      *-----------------------------------------------------------------
=> ----------------
78      * Get system security text and password
79      *-----------------------------------------------------------------
=> ----------------
91  *!*****************************************************************
92  *!
93  *!       Function: GETSYSCODE()
94  *!
95  *!       Called by: SECHECK.PRG
96  *!
97  *!           Calls: FILEMISSING        (procedure in SECHECK.PRG)
98  *!
99  *!            Uses: SYSTEM.DBF
100 *!
101 *!*****************************************************************
91      FUNCTION getsyscode
92      PARAMETER mwarn, mprivate, mdis, mpass
93      msel = SELECT()
94      IF NOT USED("SYSTEM")
95         IF NOT FILE("system.dbf")
96            ON ERROR DO filemissing
97            = LOCFILE("SYSTEM","DBF","Locate File SYSTEM.DBF")
98         ENDIF
99         ON ERROR
100        SELECT 0
101        USE SYSTEM
102     ELSE
103        SELECT SYSTEM
104     ENDIF
105     mwarn     =  syswarning   && system warnign text
106     mprivate  =  privacyact   && system privatecy act text
107     mdis      =  sysdiscl     && system disclaiming text
108     mpass     =  ALLTRIM(password)     && system password
109     USE
110     SELECT (msel)
111     RETURN .T.
112     *-----------------------------------------------------------------
=> ----------------
113     * Error for filemissing
114     *-----------------------------------------------------------------
=> ----------------
124 *!*****************************************************************
125 *!
126 *!       Procedure: FILEMISSING
127 *!
128 *!       Called by: GETSYSCODE()       (function  in SECHECK.PRG)
129 *!
130 *!           Calls: ERRMSG.PRG
131 *!
```

```
132  *!*********************************************************************
124      PROCEDURE filemissing
125      =errmsg("File missing !!!!!!")
126      ON ERROR
127      CANCEL
128      RETURN
129
130      *-------------------------------------------------------------------
=> ----------------
131      * System disclaim
132      *-------------------------------------------------------------------
=> ----------------
140  *!*********************************************************************
141  *!
142  *!        Function: SYSCLAIM()
143  *!
144  *!       Called by: SECHECK.PRG
145  *!
146  *!*********************************************************************
140      FUNCTION sysclaim
141      PARAMETER m.txt
142      IF NOT WEXIST("w_claim")
143         DEFINE WINDOW w_claim ;
144             FROM INT((SROW()-22)/2),INT((SCOL()-79)/2) ;
145             TO INT((SROW()-22)/2)+21,INT((SCOL()-79)/2)+78 ;
146             NOFLOAT ;
147             NOCLOSE ;
148             SHADOW ;
149             DOUBLE ;
150             COLOR SCHEME 1
151      ENDIF
152      IF WVISIBLE("w_claim")
153         ACTIVATE WINDOW w_claim SAME
154      ELSE
155         ACTIVATE WINDOW w_claim NOSHOW
156      ENDIF
157      @ 0,0 EDIT m.txt ;
158          PICTURE "@T" ;
159          SIZE 18,77,0 ;
160          DEFAULT " " ;
161          NOMODIFY;
162          SCROLL
163
164      ACTIVATE WINDOW w_claim
165      WAIT
166      CLEAR GETS
167      RELEASE WINDOW w_claim
168      RETURN
170      *: EOF: SECHECK.ACT
```

```
1   *:*****************************************************************************
2   *:
3   *: Procedure file: C:\CAMD\WORK\U.PRG
4   *:
5   *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
6   *:          Author: ROY DOBBINS and HOA LY
7   *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
8   *:  Last modified: 10/13/92     10:21
9   *:
10  *:  Procs & Fncts: USERINIT()
11  *:                : POPUPSHOW()
12  *:                : POPUPHIDE()
13  *:                : OPENFILES()
14  *:                : MAINMENU()
15  *:                : USERCLOSE()
16  *:                : GETEXAREA()
17  *:                : DISEASENAM()
18  *:                : ENCNTTITLE()
19  *:                : WCONFIRM()
20  *:                : PATID()
21  *:                : FLUSHFILES()
22  *:                : POPMESSAGE()
23  *:                : DISEASEID()
24  *:                : PATSSN()
25  *:                : ADDENCNT()
26  *:                : NEWENCNT()
27  *:                : PUTENCNT()
28  *:                : PUTSYMPTOM()
29  *:                : PUTDIAGNOS()
30  *:                : NEXTENCNT()
31  *:                : GETRULEBASE()
32  *:                : DISEASECNT()
33  *:                : DISNAMES()
34  *:                : GETDISLIST()
35  *:                : AREACOUNT()
36  *:                : AREANAMES()
37  *:                : GETAREALIST()
38  *:                : PATCOUNT()
39  *:                : PATNAMES()
40  *:                : GETPATLIST()
41  *:                : ENCCOUNT()
42  *:                : ENCNAMES()
43  *:                : GETENCLIST()
44  *:                : GETSTATION()
45  *:                : ENCGETINFO()
46  *:                : GETDIAGNOS()
47  *:                : GETSYMPTOM()
48  *:                : UNDOMENU()
49  *:                : PATGETINFO()
50  *:                : ERRMESSAGE()
51  *:                : ISVALIDSSN()
52  *:                : PATPUTINFO()
53  *:                : ADDPAT()
54  *:                : NEWPATIENT()
55  *:                : DICTONAME()
56  *:                : DICTCOUNT()
57  *:                : DICTQ()
58  *:                : DICTQV()
59  *:                : EDNUM()
60  *:                : EDBOOL()
61  *:                : DICTECOUNT()
62  *:                : DICTQE()
63  *:                : EDCHECK()
64  *:                : CHANGELIST()
65  *:                : DICTREPORT()
66  *:                : SPLIT()
67  *:                : WCHOICE
68  *:                : GETDISLOOKUPLIST
69  *:                : EDLIST
70  *:                : _EDCHECK
71  *:                : SETCHECK
72  *:                : _EVALID
73  *:                : _EDITCOND
74  *:                : _EDITVDIAG
75  *:                : _EDITCOMPL
76  *:                : _EDITDIAG
77  *:                : _EDITHANDP
78  *:                : _EDITASSIS
79  *:                : _EDITTREAT
80  *:                : RELEASE
81  *:                : _PRINTENC
82  *:                : _STOPREAD
83  *:                : MYHANDLER
84  *:                : _MHIT
85  *:                : _QUIT
86  *:                : _PVALID
87  *:                : PUSHHELP
88  *:                : POPHELP
89  *:                : SETHELP
90  *:                : OLMR
91  *:                : INITHELP
92  *:                : GETCOLORS
93  *:                : OPENDIAG
94  *:                : OPENDICT
95  *:                : CLOSEDICT
96  *:                : DICTORD
97  *:                : DICTENUM
98  *:                : DICTQID
99  *:                : OPENENCNT
100 *:                : FLUSHENCNT
101 *:                : ENCID
102 *:                : DISEASEGET
103 *:                : DISEASEPUT
104 *:                : CLOSEFILES
105 *:                : OPENPAT
106 *:                : FLUSHPAT
107 *:                : DELPATIENT
108 *:                : ISVALIDPAT
109 *:                : STRNAME
110 *:                : OPENRULE
111 *:                : CLOSERULE
112 *:                : GETRULE
113 *:                : OPENSYMP
114 *:                : GETOPTS
115 *:                : NEXTPAT
116 *:                : NENCNT
117 *:                : NEXTUSER
118 *:                : ADDUSER
119 *:                : BROWSEPAT
120 *:                : MPGETPATLIST
121 *:                : SEARCH
122 *:                : MORESEARCH
123 *:                : GETCAMDPAT
124 *:                : GETSAMSPAT
125 *:                : GETPATDATA
126 *:                : GETSAMS
127 *:                : SAMSDATA
128 *:                : GETPAT
129 *:                : GETPATTEMP
130 *:                : PATDATA
131 *:                : COMBINED
132 *:                : SAVREC
```

```
133  *:                  : OPENSAMS
134  *:                  : SECONVERT
135  *:                  : SFENCOUNT
136  *:                  : SFGETENC
137  *:                  : SFENCLIST
138  *:                  : SFGETPAT
139  *:                  : SFENCNAMES
140  *:                  : SFGETPATLIST
141  *:                  : SFPRINT
142  *:                  : SETPATH
143  *:                  : ERRHAND
144  *:                  : FILEFIND
145  *:                  : REINDEX
146  *:
147  *:      Documented 10/14/92 at 10:50              FoxDoc  version 2.10
148  *:*****************************************************************
149      * u.prg
150      *-----------------------------------------------------
151      *
152      * Medical Practice Support System
153      * $Revision: 1.1 $   $Date: 92/05/04 11:11:10 $
154      * $Author: RoyWDobbins $
155      * 4/24/92 rwd02  context sensitive help settings
156      * 4/27/92 rwd03  batch test mode
157      * 5/14/92 rwd04  simple report format
158      * 6/1/92  rwd06  on-line medical reference process
159      * 6/2/92  rwd07  performance tuning of assist
160      * 6/9/92  hll01  delete variable "startdir"
161      * 6/10/92 hll02  add procedures and functions for patient intergrate
=> with SAMS
162      * 6/10/92 hll03  add procedures and functions for doing sf600 report
163      * 6/18/92 hll04  Update patient info (sex and age) to working memory
164      * 6/18/92 hll05  Comment the computer diagnosis out from sf600 report
165      * 6/18/92 hll06  add mreport to PUBLIC variable for geting the sf600
=> report
166      *-----------------------------------------------------
167
168      * user interface utility layer
169
170      *-----------------------------------------------------
171      *
172      * list box
173      *
174      *-----------------------------------------------------
186  *!*****************************************************************
187  *!
188  *!      Procedure: WCHOICE
189  *!
190  *!      Called by: CHANGELIST()       (function  in U.PRG)
191  *!
192  *!          Calls: PUSHHELP           (procedure in U.PRG)
193  *!               : WT4.SPR
194  *!               : POPHELP            (procedure in U.PRG)
195  *!
196  *!*****************************************************************
186      FUNCTION wchoice
187      PARAMETERS TOP, LEFT, BOTTOM, RIGHT, mpad, mcolor, mtitle, mhelp, mch
=> oice
188
189      PRIVATE mbuttons
190      =pushhelp( mhelp )
191      mchoice = IIF(TYPE("mchoice") = "N", mchoice, 1)      && initial menu
=>  choice
192
193      * get user's selection
194      mbuttons = 0
```

```
195      DO w14.spr
=>                                  && hll01
196      DO CASE
197      CASE mbuttons = 2
198          mchoice = 0
199      ENDCASE
200      =pophelp()
201      RETURN mchoice
202
203
204      *-----------------------------------------------------
205      *
206      * confirm popup
207      *
208      *-----------------------------------------------------
220  *!*****************************************************************
221  *!
222  *!      Function: WCONFIRM()
223  *!
224  *!      Called by: M_RESUME           (procedure in DEMO.PRG)
225  *!
226  *!          Calls: PUSHHELP           (procedure in U.PRG)
227  *!               : W10.SPR
228  *!               : POPHELP            (procedure in U.PRG)
229  *!
230  *!*****************************************************************
220      FUNCTION wconfirm                                      && rwd07
221      PARAMETERS m.prompt, mhelp
222      PRIVATE mbuttons, mchoice
223      =pushhelp( mhelp )
224
225      * get user's selection
226      mbuttons = 0
227      mchoice = e_ok
228      DO w10.spr                                             &&hll01
229      DO CASE
230      CASE mbuttons = 1
231          mchoice = e_ok
232      CASE mbuttons = 2
233          mchoice = e_esc
234      ENDCASE
235      =pophelp()
236
237      RETURN mchoice
238
239      * end user interface utility layer
240
241      * user interface functions
242
243      *-----------------------------------------------------
244      *
245      * init user interface system
246      *
247      *-----------------------------------------------------
261  *!*****************************************************************
262  *!
263  *!      Function: USERINIT()
264  *!
265  *!      Called by: DEMO.PRG
266  *!
267  *!          Calls: INITHELP           (procedure in U.PRG)
268  *!               : PUSHHELP           (procedure in U.PRG)
269  *!               : GETCOLORS          (procedure in U.PRG)
270  *!
271  *!    Other Files: CAMDUSER.DBF
272  *!
```

```
273  *|*****************************************************************
261      FUNCTION userinit
262      PARAMETERS notime
263      PUBLIC msafety, mexact, mescp, mproc, mpath
264      PUBLIC mudfparms, mbell, mscore, mdelete
265
266      mdelete = SET("DELETE")
267      msafety = SET("SAFETY")                                  && hll0
=> 2
268      mexact = SET("EXACT")
=>  && hll02
269      mescp = SET("ESCAPE")
=>  && hll02
270      mpath = SET("PATH")
=>  && hll02
271      mudfparms = SET("UDFPARMS")                              && hll0
=> 2
272      mscore = SET("SCOREBOARD")                               && hll0
=> 2
273      mbell = SET("BELL OFF")
=>  && hll02
274
275      SET RESOURCE TO camduser.dbf
276      SET SAFETY OFF
277      SET UDFPARMS TO REFERENCE
278      SET SCOREBOARD OFF
279      SET BELL OFF
280      SET ESCAPE ON                    && user can terminate read with Esca
=> pe key
281      SET DELETE ON
282      *-----------------------------------------------------
283      *
284      * global user interface variables
285      *
286      *-----------------------------------------------------
287      PUBLIC mtitle, mversion, help_code, help_id, help_level, help_stack
288      DIMENSION help_stack[30]            && stack of help contexts
289      PUBLIC singlebox
290
291      * hypertext identifiers for on-line help
292      PUBLIC hyper_id
293      DIMENSION hyper_id[30]              && allow several entries defining
=>  focus
294
295      mtitle     =  "Computer Assisted Medical Diagnostic System"
296      mversion   =  "Version Release 1.0 "
297      help_code  =  0
298      help_level =  0
299      =inithelp()
300      =pushhelp( "CAMD" )
301      hyper_id   =  ""
302      singlebox  =  CHR(218) + CHR(196) + CHR(191) + CHR(179) +;
303          CHR(217) + CHR(196) + CHR(192) + CHR(179)
304
305      * default screen color set
306      PUBLIC c_deflt,c_disease,c_txt,c_err,c_area,c_pat
307      PUBLIC c_main,c_rev,c_treat,c_popup,c_rule,c_note
308      PUBLIC c_info,c_diag,c_help,c_prompt,c_getrev
309      PUBLIC c_encount,c_logon
310
311      * load optional color set
312      PRIVATE colorset
313      colorset = GETENV("COLORSET")          && get environment string
314      IF !EMPTY(colorset)
315          =getcolors(colorset)
316      ENDIF
```

```
317      @ 3,1 SAY PADC( mtitle, 78 )
318      @ 5,1 SAY PADC( mversion, 78 )
319      @ 7,1 SAY PADC( "by", 78 )
320      @ 9,1 SAY PADC( "Naval Health Research Center", 78 )
321      @ 22,1 SAY PADC( "Press the F1 key for help", 78 )        && rwd07
322      @ 1,0 TO 23,79 PANEL
323      RETURN .T.
324      *-----------------------------------------------------
325      *
326      * Set user return parameter.
327      *
328      *-----------------------------------------------------
336  *|*****************************************************************
337  *|
338  *|       Function: USERCLOSE()
339  *|
340  *|      Called by: DEMO.PRG
341  *|
342  *|*****************************************************************
336      FUNCTION userclose
337      RELEASE WINDOWS ALL
338      @ 0,0 CLEAR
339      IF !EMPTY(mproc)
340          SET PROCEDURE TO &mproc        && user/database interface
341      ENDIF
342      SET SAFETY &msafety                                       && hll02
343      SET PATH TO &mpath                                        && hll02
344      SET UDFPARMS TO &mudfparms                        && hll02
345      SET SCOREBOARD &mscore                                    && hll02
346      SET BELL &mbell
347      SET ESCAPE &mescp                    && user can terminate read with
=> Escape key
348      SET DELETE &mdelete                                       && hll02
349      *SET COLOR TO                                                  && hll;
=> For standalone version
350      *SET HELP TO                         && rwd02 && hll; for standalone v
=> ersion
351      RETURN ""
352
353      *-----------------------------------------------------
354      *
355      * change data for rerun
356      *
357      *-----------------------------------------------------
367  *|*****************************************************************
368  *|
369  *|       Function: CHANGELIST()
370  *|
371  *|      Called by: CHANGEQ            (procedure in DEMO.PRG)
372  *|
373  *|          Calls: WCHOICE            (procedure in U.PRG)
374  *|
375  *|*****************************************************************
367      FUNCTION changelist
368      PARAMETERS qnames, mchoice
369
370      * present choice of data to change
371      RETURN wchoice(3,1,21,78,qnames,c_popup,"Select Data to change","CHAN
=> GE",mchoice)
372
373      *-----------------------------------------------------
374      *
375      * select disease from list
376      *
377      *-----------------------------------------------------
391  *|*****************************************************************
```

```
392  *!
393  *!        Function: GETDISLIST()
394  *!
395  *!       Called by: ASKDIAG            (procedure in DEMO.PRG)
396  *!
397  *!           Calls: PUSHHELP           (procedure in U.PRG)
398  *!                : W7.SPR
399  *!                : POPHELP            (procedure in U.PRG)
400  *!                : DISEASEGET         (procedure in U.PRG)
401  *!                : W8.PRG
402  *!
403  *!*****************************************************************
391      FUNCTION getdislist
392      PARAMETERS mlist
393      * mlist is list of disease names
394      EXTERNAL ARRAY mlist                                  && rwd07
395
396      PRIVATE mchoice
397      PRIVATE mhelp
398
399      IF ALEN(mlist) > 0
400         * display menu of names in window
401         PRIVATE i
402         PRIVATE mtitle
403         mtitle = "Diseases on file"
404         PRIVATE mbuttons
405         DO WHILE .T.
406            mchoice = 0
407            mbuttons = 0
408            =pushhelp("DISEASES ON FILE")                  && rwd02
409            DO w7.spr
=>                  && hll01
410            =pophelp()                                     && rwd02
411            DO CASE
412            CASE mbuttons = 2
413               * VIEW button
414               IF mchoice > 0
415                  mhelp = diseaseget( mlist[mchoice] )
416                  =pushhelp("DISEASE")                     && rwd02
417                  DO w8
=>             && hll01
418                  =pophelp()                               && rwd02
419               ENDIF
420            LOOP
421            CASE mbuttons = 3
422               * CANCEL button
423               mchoice = 0
424            ENDCASE
425            EXIT
426         ENDDO
427      ENDIF
428      RETURN mchoice
429
430      *-------------------------------------------------------
431      *
432      * select disease from list
433      *
434      *-------------------------------------------------------
448  *!*****************************************************************
449  *!
450  *!       Procedure: GETDISLOOKUPLIST
451  *!
452  *!       Called by: M_LOOKUP           (procedure in DEMO.PRG)
453  *!
454  *!           Calls: PUSHHELP           (procedure in U.PRG)
455  *!                : W7A.SPR
456  *!                : POPHELP            (procedure in U.PRG)
457  *!                : DISEASEGET         (procedure in U.PRG)
458  *!                : W8.PRG
459  *!
460  *!*****************************************************************
448      FUNCTION getdislookuplist
449      PARAMETERS mlist
450      * mlist is list of disease names
451      EXTERNAL ARRAY mlist                                  && rwd07
452
453      PRIVATE mchoice
454      PRIVATE mhelp
455
456      IF ALEN(mlist) > 0
457         * display menu of names in window
458         PRIVATE i
459         PRIVATE mtitle
460         mtitle = "Diseases / Treatment Reference"
461         PRIVATE mbuttons
462         DO WHILE .T.
463            mchoice = 0
464            mbuttons = 0
465            =pushhelp("DISEASES/TREATMENT REFERENCE")                    &
=> & rwd02
466            DO w7a.spr
=>                  && hll01
467            =pophelp()                                     && rwd02
468            DO CASE
469            CASE mbuttons = 1
470               * VIEW button
471               IF mchoice > 0
472                  mhelp = diseaseget( mlist[mchoice] )
473                  =pushhelp("DISEASE DESCRIPTION WINDOW")
=>             && rwd02
474                  DO w8
=>             && hll01
475                  =pophelp()                               && rwd02
476               ENDIF
477            LOOP
478            CASE mbuttons = 2
479               * CANCEL button
480               mchoice = 0
481            ENDCASE
482            EXIT
483         ENDDO
484      ENDIF
485      RETURN mchoice
486
487      *-------------------------------------------------------
488      *
489      * edit dialog boxes
490      *
491      *-------------------------------------------------------
492      *-------------------------------------------------------
493      *
494      * edit number
495      *
496      *-------------------------------------------------------
508  *!*****************************************************************
509  *!
510  *!        Function: EDNUM()
511  *!
512  *!       Called by: EDITSYMP           (procedure in DEMO.PRG)
513  *!
514  *!           Calls: PUSHHELP           (procedure in U.PRG)
515  *!                : W6.SPR
```

```
516  *!                : POPHELP            (procedure in U.PRG)
517  *!
518  *!*********************************************************************
508      FUNCTION ednum
509      PARAMETERS mid, mname, mquestion, mextra, mhi, mlo, mwidth, mdec, mun
=> its, mstrval
510      * mid        dictionary id of item
511      * mname      name of item
512      * mquestion  user question text
513      * mextra     additional user prompt text
514      * mhi        hi limit advisory for value
515      * mlo        lo limit advisory
516      * mwidth     field width
517      * mdec       number of decimals
518      * munits     engineering units text
519      * mstrval    current value
520
521      * save help context
522      PRIVATE oldhyp[alen(hyper_id)]
523      =ACOPY(hyper_id, oldhyp)
524      hyper_id   = ""
525      =pushhelp( mname )              && help identifier of item
526      hyper_id[1] =  mname            && entry into hypertext help system
527
528      PRIVATE mprompt
529      mprompt = mquestion
530      PRIVATE mevent
531      PRIVATE mbuttons
532      mevent = e_ok
533      mbuttons = 0
534      PRIVATE mval
535      mval = VAL(mstrval)
536      DO w6.spr                                          && hll01
537      DO CASE
538      CASE mbuttons = 1
539          mevent = e_ok
540      CASE mbuttons = 2
541          mevent = e_esc
542      ENDCASE
543      mstrval = LTRIM( STR( mval, mwidth, mdec ) )
544      =pophelp()                      && restore help context
545      =ACOPY(oldhyp, hyper_id)
546      RETURN mevent
547
548      *-------------------------------------------------------
549      *
550      * edit Boolean
551      *
552      *-------------------------------------------------------
564  *!*********************************************************************
565  *!
566  *!       Function: EDBOOL()
567  *!
568  *!      Called by: EDITSYMP          (procedure in DEMO.PRG)
569  *!
570  *!          Calls: PUSHHELP          (procedure in U.PRG)
571  *!               : W6C.SPR
572  *!               : POPHELP            (procedure in U.PRG)
573  *!
574  *!*********************************************************************
564      FUNCTION edbool
565      PARAMETERS id, mname, mquestion, mextra, mstrval
566      * id         dictionary id of item
567      * mname      name of item
568      * mquestion  user question text
569      * mextra     additional user prompt text
```

```
570      * mstrval    current concated string value
571
572      * save help context
573      PRIVATE oldhyp[alen(hyper_id)]
574      =ACOPY(hyper_id, oldhyp)
575      hyper_id   = ""
576      =pushhelp( mname )              && help identifier of item
577      hyper_id[1] =  mname            && entry into hypertext help system
578
579      PRIVATE mevent
580      PRIVATE mprompt
581      mprompt = mquestion
582      PRIVATE mchoice
583      PRIVATE mbuttons
584      mchoice = IIF(mstrval = "Y", 1, 2)
585      mbuttons = 0
586      mevent = e_ok
587
588      DO w6c.spr                                         && hll01
589      DO CASE
590      CASE mbuttons = 1
591          mevent  = e_ok
592      CASE mbuttons = 2
593          mevent = e_esc
594      ENDCASE
595      IF mevent != e_esc
596          mstrval  = IIF( mchoice = 1,  "YES",  "NO" )
597      ENDIF
598
599      =pophelp()                      && restore help context
600      =ACOPY(oldhyp, hyper_id)
601      RETURN mevent
602
603      *-------------------------------------------------------
604      *
605      * edit single choice checkbox
606      *
607      *-------------------------------------------------------
617  *!*********************************************************************
618  *!
619  *!      Procedure: EDLIST
620  *!
621  *!          Calls: PUSHHELP          (procedure in U.PRG)
622  *!               : W68.SPR
623  *!               : POPHELP            (procedure in U.PRG)
624  *!
625  *!*********************************************************************
617      FUNCTION edlist
618      PARAMETERS id, mname, mquestion, mextra, choices, mstrval
619      * id         dictionary id of item
620      * mname      name of item
621      * mquestion  user question text
622      * mextra     additional user prompt text
623      * choices    enumerated choices
624      * mstrval    current concated string value
625
626      EXTERNAL ARRAY choices                             && rwd07
627
628      * save help context
629      PRIVATE oldhyp[alen(hyper_id)]
630      =ACOPY(hyper_id, oldhyp)
631      hyper_id   = ""
632      =pushhelp( mname )              && help identifier of item
633      hyper_id[1] =  mname            && entry into hypertext help system
634
635      * add hypertext entries to focus
```

```
636      PRIVATE i
637      FOR i = 1 TO ALEN(choices)
638        IF i < ALEN(hyper_id)
639           hyper_id[i + 1]  = LTRIM(choices[i])
640        ENDIF
641      NEXT
642
643        PRIVATE mevent
644        PRIVATE mprompt
645        mprompt = mquestion
646        PRIVATE mchoice
647        PRIVATE mbuttons
648        mchoice = 0
649        mbuttons = 0
650        mevent = e_ok
651
652        * user selects entry from list box
653        DO w6b.spr
=>         && hl101
654        DO CASE
655        CASE mbuttons = 1
656           mevent  = e ok
657        CASE mbuttons = 2
658           mevent = e_esc
659        ENDCASE
660        IF mevent != e_esc
661           IF mchoice > 0
662              mstrval = choices[mchoice]
663           ENDIF
664        ENDIF
665
666        =pophelp()                          && restore help context
667        =ACOPY(oldhyp, hyper_id)
668        RETURN mevent
669
670
671        *-----------------------------------------------------
672        *
673        * edit multiple choice checkbox
674        *
675        *-----------------------------------------------------
688     *!*****************************************************************
689     *!
690     *!       Function: EDCHECK()
691     *!
692     *!      Called by: EDITSYMP          (procedure in DEMO.PRG)
693     *!
694     *!          Calls: PUSHHELP          (procedure in U.PRG)
695     *!               : SETCHECK          (procedure in U.PRG)
696     *!               : W6A.PRG
697     *!               : POPHELP           (procedure in U.PRG)
698     *!
699     *!*****************************************************************
688      FUNCTION edcheck
689         PARAMETERS id, mname, mquestion, mextra, choices, inclusive, check
=>, mstrval
690         * id         dictionary id of item
691         * mname      name of item
692         * mquestion  user question text
693         * mextra     additional user prompt text
694         * choices    enumerated choices
695         * inclusive  inclusive/exclusive selection attributes
696         * check      item checked flags
697         * mstrval    current concated string value
698
699         EXTERNAL ARRAY choices, check                          && r
```

```
=> wd07
700         * save help context
701         PRIVATE oldhyp[alen(hyper_id)]
702         =ACOPY(hyper_id, oldhyp)
703         hyper_id   = ""
704         =pushhelp( mname )              && help identifier of item
705         hyper_id[1] =  mname            && entry into hypertext help system
706
707         * add hypertext entries to focus
708         PRIVATE i
709         FOR i = 1 TO ALEN(choices)
710           IF i < ALEN(hyper_id)
711              hyper_id[i + 1]  = LTRIM(choices[i])
712           ENDIF
713         NEXT
714
715           PRIVATE N
716           PRIVATE msize
717           PRIVATE mevent
718           PRIVATE MROW, MCOL, mrows, mcols
719           PRIVATE mprompt
720           PRIVATE mchoice
721           PRIVATE mbuttons
722           mchoice = 0
723           mbuttons = 0
724           mevent = e_ok
725           mprompt = mquestion
726
727           * validate checkmarks
728           msize = ALEN(check)
729           FOR  N = 1  TO  msize
730              =setcheck(N, check, inclusive)
731           NEXT
732
733              * user selects entry from list box
734              DO w6a
=>  && hl101
735           DO CASE
736           CASE mbuttons = 1
737              mevent  = e_ok
738           CASE mbuttons = 2
739              mevent = e_esc
740           ENDCASE
741
742           IF mevent <> e_esc
743              * concatenate all checked items
744              N = 0                       && count checked items
745              mstrval = ""                && default return value is null
746              FOR  i = 1  TO  msize
747                 IF check[i]
748                    * concat another item
749                    mstrval = mstrval + IIF( N > 0, mconcat, "" ) + cho
=> ices[i]
750                    N = N + 1
751                 ENDIF
752              NEXT
**** Expected NEXT--Found ENDIF
753           ENDIF
754              =pophelp()                          && restore help context
755              =ACOPY(oldhyp, hyper_id)
756              RETURN mevent
757
758
768     *!*****************************************************************
769     *!
770     *!      Procedure: _EDCHECK
```

```
771  *!
772  *!        Called by: W6A.PRG
773  *!
774  *!            Calls: SETCHECK            (procedure in U.PRG)
775  *!
776  *!*********************************************************************
768                FUNCTION  edcheck
769                   PRIVATE  i, j
770                   * KLUDGE ALERT!!!
771                   * this function uses the FOXPRO function varread() to get
=>  the
772                   * name of the current READ object.
773                   * This is assumed to be of the form "CHECK(1)", where CHE
=> CK is
774                   * the name of the check box array.  The check box index i
=> s extracted
775                   * by searching for the first left parenthesis:
776                   j = VAL( SUBSTR( VARREAD(), AT( "(", VARREAD() ) + 1 ) )
777
778                FOR i = 1 TO ALEN(check)
779                   * set checks relative to the selected item
780                   IF j = i
781                      =setcheck(i, check, inclusive)
782                   ENDIF
783                NEXT
784
785                   SHOW GETS
786  <------------------RETURN .T.
787
788                   *-----------------------------------------------
=> ---
789                   *
790                   * set and validate checkmarks
791                   *
792                   *-----------------------------------------------
=> ---
801  *!*********************************************************************
802  *!
803  *!        Procedure: SETCHECK
804  *!
805  *!        Called by: EDCHECK()          (function  in U.PRG)
806  *!                 : _EDCHECK           (procedure in U.PRG)
807  *!
808  *!*********************************************************************
801                FUNCTION setcheck
802                   PARAMETERS N, check, inclusive
803                   PRIVATE K
804                   EXTERNAL ARRAY check, inclusive                    &
=> & rwd07
805
806                   * if checked item is exclusive (-):
807                   *    only one item can be checked  uncheck all others
808                   * if checked item is inclusive (+):
809                   *    multiple (inclusive) items can be checked uncheck
=>  all exclusives
810                   IF check[n]
811                      PRIVATE msize
812                      msize = ALEN(check)
813                      IF inclusive[n] = "-"
814                         * exclusive item uncheck all others
815                         FOR K = 1 TO msize
816                            IF K <> N
817                               check[k] = .F.
818                            ENDIF
819                         NEXT
820                         ELSE
```

```
821                         * inclusive item uncheck all exclusive items
822                         FOR K = 1 TO msize
823                            IF inclusive[k] = "-"
824                               check[k] = .F.
825                            ENDIF
826                         NEXT
**** Expected NEXT--Found ENDIF
827                      ENDIF
**** Expected NEXT--Found ENDIF
828                   ENDIF
829  <------------------RETURN N
830
831                   *-----------------------------------------------
=> ---------
832                   *
833                   * select a previous encounter from list
834                   *
835                   *-----------------------------------------------
=> ---------
847  *!*********************************************************************
848  *!
849  *!        Function: GETENCLIST()
850  *!
851  *!        Called by: GETENCNT          (procedure in DEMO.PRG)
852  *!
853  *!            Calls: PUSHHELP          (procedure in U.PRG)
854  *!                 : W11.SPR
855  *!                 : POPHELP            (procedure in U.PRG)
856  *!
857  *!*********************************************************************
847                FUNCTION getenclist
848                   PARAMETERS mlist, mkey
849
850                   EXTERNAL ARRAY mlist
851                   PRIVATE mchoice
852                   mchoice = 0
853                   IF ALEN(mlist) > 0
854                      * display menu of names in window
855                      PRIVATE i
856                      PRIVATE mtitle
857                      mtitle = "Past Encounters on file" + IIF(TYPE
=> ("mkey")="C", " [" + ALLTRIM(mkey) + "] ","")
858                      PRIVATE mbuttons
859                      mchoice = 0
860                      mbuttons = 0
861                      =pushhelp("PENDING ENCOUNTERS")                 &
=> & rwd02
862                      DO w11.spr
=>                  && hl01
863                      =pophelp()                                      &
=> & rwd02
864                      DO CASE
865                      CASE mbuttons = 1
866                         * RESUME button
867                         IF mchoice = 0
868                            * nothing to resume, so cancel
869                            mchoice = -1
870                         ENDIF
871                      CASE mbuttons = 2
872                         * NEW button
873                         mchoice = 0
874                      CASE mbuttons = 3
875                         * CANCEL button
876                         mchoice = -1
877                      OTHERWISE
878                         mchoice = -1
```

```
879                 └ENDCASE
880                 └ENDIF
881 <------------------RETURN mchoice
882
883                 *-----------------------------------------
=> ---------
884                 *
885                 * encounter title screen
886                 *
887                 *-----------------------------------------
=> ---------
899 *!*****************************************************************
900 *!
901 *!      Function: ENCNTTITLE()
902 *!
903 *!     Called by: M_RESUME          (procedure in DEMO.PRG)
904 *!
905 *!         Calls: PUSHHELP          (procedure in U.PRG)
906 *!              : W12.SPR
907 *!              : POPHELP           (procedure in U.PRG)
908 *!
909 *!*****************************************************************
899                 FUNCTION encnttitle
900                    PARAMETERS me_ssn, me_lastnam, me_firstna, me_se
=> x, me_age,;
901                        me_dateenc, me_timeenc, me_area,;
902                        me_providr, me_diagnos, me_diag, me_cond, me
=> _treat, me_state
903
904                    EXTERNAL ARRAY me_diag, me_cond
=>       && rwd07
905                    * !NOTE! choose unique parameter names so that a
=> synchronous
906                    * event procedures can get to the "real" (global
=> ) variables
907                    * that they need to interact with.  eg an event
=> that needs
908                    * the global 'marea' would get the parameter 'ma
=> rea' instead
909                    * hence this is renamed me_area.
910
911                    PRIVATE mbuttons, mevent
912                    PRIVATE quitting, saving
913                    PRIVATE mckhandp, mckassist, mcktreat, mckcompl,
=>  mckdiag
914                    mckhandp  = 0
915                    mckassist = 0
916                    mcktreat  = 0
917                    mckcompl  = 0
918                    mckdiag   = 0
919                    mbuttons  = 0
920                    mevent = e_ok
921                    quitting = .F.
922                    saving = .T.
923                    =pushhelp("PATIENT ENCOUNTER")
=>  && rwd02
924                    DO w12.spr
=>                  && hl101
925                    =pophelp()
=>  && rwd02
926 <------------------RETURN mevent
927
928                    * Push button controls
940 *!*****************************************************************
941 *!
942 *!      Procedure: _EVALID
943 *!
944 *!     Called by: W12.SPR
945 *!
946 *!         Calls: PUSHHELP          (procedure in U.PRG)
947 *!              : W2.SPR
948 *!              : POPHELP           (procedure in U.PRG)
949 *!
950 *!*****************************************************************
940                 FUNCTION _evalid
941                    ┌DO CASE
942                    ├CASE mbuttons = 1
943                    │   mprintfile = ""
944                    │   mevent = e_print
945                    │   =pushhelp("PRINT ENCOUNTER")
=>  && rwd02
946                    │   DO w2.spr
=>                  && hl101
947                    │   =pophelp()
=>  && rwd02
948                    │   CLEAR READ
949                    ├CASE mbuttons = 2
950                    │   quitting = .T.
951                    │   mevent = e_save
952                    │   CLEAR READ
953                    ├CASE mbuttons = 3
954                    │   quitting = .T.
955                    │   saving = .F.
956                    │   mevent = e_esc
957                    │   CLEAR READ
958                    └ENDCASE
959                    SHOW GETS
960 <------------------RETURN
961
962                    * *** Encounter screen field events ***
963
964                    * edit condition field item
976 *!*****************************************************************
977 *!
978 *!      Procedure: _EDITCOND
979 *!
980 *!     Called by: W12.SPR
981 *!              : W12A.SPR
982 *!
983 *!         Calls: M_REDOQ           (procedure in DEMO.PRG)
984 *!              : STRCOND           (procedure in DEMO.PRG)
985 *!
986 *!*****************************************************************
976                 FUNCTION  _editcond
977                    PARAMETERS mitem
978                    DO m_redoq IN demo.prg WITH mitem
=>                                  && hl101
979                    mevent = e_ok
980                    saving = .T.
981                    =strcond(mcond, mqid, mcondition, msympval)
=>       && rwd07
982                    SHOW GETS
=>       && rwd07
983 <------------------RETURN .T.
984
985                    * edit diagnostic assist field item
999 *!*****************************************************************
1000 *!
1001 *!      Procedure: _EDITVDIAG
1002 *!
1003 *!     Called by: W12.SPR
1004 *!              : W12B.SPR
```

```
1005  *!
1006  *!          Calls: DISEASEGET          (procedure in U.PRG)
1007  *!               : PUSHHELP            (procedure in U.PRG)
1008  *!               : W8.PRG
1009  *!               : POPHELP             (procedure in U.PRG)
1010  *!
1011  *!*****************************************************************
 999                      FUNCTION  _editvdiag
1000                         PARAMETERS mitem
1001                         PRIVATE mhelp
1002                       IF mitem > 0
1003                          mhelp = diseaseget( me_diag[mitem] )
1004                          =pushhelp("DISEASE DESCRIPTION WINDOW")
 =>                  && rwd02
1005                          DO w8
 =>                  && hll01
1006                          =pophelp()                                  &
 => & rwd02
1007                          saving = .T.
1008                       ENDIF
1009                       mevent = e_ok
1010                       SHOW GETS                                     &
 => & rwd07
1011  <                    RETURN .T.
1012
1013
1014                       * edit complaint field
1025  *!*****************************************************************
1026  *!
1027  *!      Procedure: _EDITCOMPL
1028  *!
1029  *!      Called by: W12.SPR
1030  *!
1031  *!          Calls: M_COMPL             (procedure in DEMO.PRG)
1032  *!               : GETEXAREA()         (function  in U.PRG)
1033  *!
1034  *!*****************************************************************
1025                      FUNCTION  _editcompl
1026                         PRIVATE newarea
1027                         DO m_compl IN demo.prg
1028                       IF EMPTY(marea)
1029  <                    RETURN .T.
1030                       ENDIF
1031                         newarea = getexarea( marea )
1032                       IF me_area != newarea
1033                          me_area = newarea
1034                          CUROBJ = OBJNUM(mckcompl)
1035                          SHOW GETS
1036                          saving = .T.
1037                       ENDIF
1038  <                    RETURN .T.
1039
1040                       * edit provider's diagnosis field
1051  *!*****************************************************************
1052  *!
1053  *!      Procedure: _EDITDIAG
1054  *!
1055  *!      Called by: W12.SPR
1056  *!
1057  *!          Calls: M_DIAG              (procedure in DEMO.PRG)
1058  *!               : DISEASENAM()        (function  in U.PRG)
1059  *!
1060  *!*****************************************************************
1051                      FUNCTION  _editdiag
1052                         PRIVATE newdiag
1053                         DO m_diag IN demo.prg
```

```
 =>                  && hll01
1054                       IF EMPTY(mdiagnosis)
1055  <                    RETURN .T.
1056                       ENDIF
1057                         newdiag = diseasenam( mdiagnosis )
1058                       IF me_diagnos != newdiag
1059                          me_diagnos = newdiag
1060                          CUROBJ = OBJNUM(mckdiag)
1061                          SHOW GETS
1062                          saving = .T.
1063                       ENDIF
1064  <                    RETURN .T.
1065
1066                       * edit history and physical list field
1078  *!*****************************************************************
1079  *!
1080  *!      Procedure: _EDITHANDP
1081  *!
1082  *!      Called by: W12.SPR
1083  *!               : W12A.SPR
1084  *!
1085  *!          Calls: M_HANDP             (procedure in DEMO.PRG)
1086  *!               : STRCOND             (procedure in DEMO.PRG)
1087  *!
1088  *!*****************************************************************
1078                      FUNCTION _edithandp
1079                         DO m_handp IN demo.prg
 =>                  && hll01
1080                         =strcond(mcond, mqid, mcondition, msympval)
 =>          && rwd07
1081                         SHOW GETS
 =>          && rwd07
1082                         saving = .T.
1083  <                    RETURN .T.
1084
1085                       * edit diagnostic assistance list field
1097  *!*****************************************************************
1098  *!
1099  *!      Procedure: _EDITASSIS
1100  *!
1101  *!      Called by: W12.SPR
1102  *!               : W12B.SPR
1103  *!
1104  *!          Calls: M_ASSIST            (procedure in DEMO.PRG)
1105  *!               : STRDIAG             (procedure in DEMO.PRG)
1106  *!
1107  *!*****************************************************************
1097                      FUNCTION _editassis
1098                         DO m_assist IN demo.prg
 =>                  && hll01
1099                         =strdiag(mdiagname, mdiag, mconf, cthreshold, ""
 =>, is_conf)  && rwd07
1100                         SHOW GETS
 =>          && rwd07
1101                         saving = .T.
1102  <                    RETURN .T.
1103
1104                       * edit treatment field
1114  *!*****************************************************************
1115  *!
1116  *!      Procedure: _EDITTREAT
1117  *!
1118  *!      Called by: W12.SPR
1119  *!
1120  *!          Calls: M_TREAT             (procedure in DEMO.PRG)
1121  *!
```

```
1122 *!*****************************************************************
1114                    FUNCTION _edittreat
1115                       DO m_treat IN demo.prg WITH me_treat
 =>                                    && hll01
1116                       * CUROBJ = objnum(mcktreat)
1117                       SHOW GETS
1118                       saving = .T.
1119                       RETURN .T.
1120
1121                       * release windows
1127 *!*****************************************************************
1128 *!
1129 *!      Procedure: RELEASE
1130 *!
1131 *!*****************************************************************
1127                    FUNCTION RELEASE
1128                       PARAMETER mwins
1129                       IF PARAMETER() > 0
1130                          IF WONTOP(mwins)
1131                             CLEAR READ
1132                             DEACTIVATE WINDOW &mwins
1133                             RELEASE WINDOW &mwins
1134                          ENDIF
1135                       ENDIF
1136                       RETURN
1137
1138                       * print encounter
1146 *!*****************************************************************
1147 *!
1148 *!      Procedure: _PRINTENC
1149 *!
1150 *!      Called by: _Q7IOLH8WD()       (function  in W2.SPR)
1151 *!
1152 *!*****************************************************************
1146                    FUNCTION _printenc
1147                       PARAMETERS mprintfile
1148                       mprintfile = ALLTRIM( mprintfile )
1149                       IF !EMPTY( mprintfile )
1150                          * list to printer or file
1151                          IF mprintfile  != "PRINT"
1152                             * list to file
1153                             SET PRINTER TO (mprintfile)
1154                          ENDIF
1155                          SET CONSOLE OFF
1156                          SET PRINTER ON
1157
1158                          ?? "Encounter:          "
1159                          ?? DTOC(me_dateenc)
1160                          ?? " "
1161                          ?? me_timeenc
1162                          ?? " Care Provider:  "
1163                          ?? ALLTRIM( STR( me_providr ) )
1164
1165                          ?  "Patient:            "
1166                          ?? "SSN: "
1167                          ?? me_ssn
1168                          ?? " "
1169                          ?? TRIM(me_lastnam)
1170                          ?? ", "
1171                          ?? TRIM(me_firstna)
1172                          ?? "  "
1173                          ?? "Sex: "
1174                          ?? IIF(me_sex="F", "FeMale", "Male")
1175                          ?? "  Age: "
1176                          ?? me_age
1177
1178                          ?  "Complaint:          "
1179                          ?? TRIM(me_area)
1180
1181                          ?  "Signs and Symptoms:"                && rwd
 => 04
1182
1183                          FOR i = 1 TO ALEN(me_cond)              && rwd
 => 04
1184                             IF !EMPTY( me_cond[i] )              && rwd
 => 04
1185                                ?  me_cond[i]                     && rwd
 => 04
1186                             ENDIF                               && rwd
 => 04
1187                          NEXT                                    &&
 => rwd04
1188
1189                          ?  "Impression:         "
1190                          ?? TRIM( PROPER( me_diagnos ) )
1191
1192                          ?  "Computer's suggested diagnoses:"
1193
1194                          FOR i = 1 TO ALEN(me_diag)
1195                             IF !EMPTY( me_diag[i] )
1196                                ?  TRIM( PROPER( me_diag[ i ] ) )
1197                             ENDIF
1198                          NEXT
1199
1200                          ?  "Plan:               "
1201                          ?  me_treat
1202
1203                          IF mprintfile = "PRINT"
1204                             EJECT
1205                          ENDIF
1206                          SET PRINTER OFF
1207                          SET PRINTER TO
1208                          SET CONSOLE ON
**** Expected NEXT--Found ENDIF
1209                       ENDIF
1210                       RETURN .T.
1211
1212
1213                       * main window event
1222 *!*****************************************************************
1223 *!
1224 *!      Procedure: _STOPREAD
1225 *!
1226 *!      Called by: W12.SPR
1227 *!                 : W4.SPR
1228 *!
1229 *!*****************************************************************
1222                    FUNCTION _stopread
1223                       PARAMETERS mwindow
1224                       IF .NOT. WVISIBLE(mwindow)
1225                          WAIT
1226                          quitting = .T.
1227                       ENDIF
1228                       RETURN quitting .OR. .NOT. WREAD()
1229
1230                       *-------------------------------------
 => -------------
1231                       *
1232                       * display error message window
1233                       *
1234                       *-------------------------------------
 => -------------
```

```
1245  *|*****************************************************************
1246  *|
1247  *|       Function: ERRMESSAGE()
1248  *|
1249  *|      Called by: M_PATGET          (procedure in DEMO.PRG)
1250  *|               : FACTORID         (procedure in DEMO.PRG)
1251  *|               : EDITSYMP         (procedure in DEMO.PRG)
1252  *|               : QID              (procedure in DEMO.PRG)
1253  *|
1254  *|*****************************************************************
1245  | | | | | | | | | | FUNCTION errmessage
1246  | | | | | | | | | | PARAMETERS  errstr
1247  <-------------------------RETURN .T.
1248  | | | | | | | | | |
1249  | | | | | | | | | | *-------------------------------------
=> --------------
1250  | | | | | | | | | | *
1251  | | | | | | | | | | * display warning message window
1252  | | | | | | | | | | *
1253  | | | | | | | | | | *-------------------------------------
=> --------------
1281  *|*****************************************************************
1282  *|
1283  *|       Function: POPMESSAGE()
1284  *|
1285  *|      Called by: M_ENC            (procedure in DEMO.PRG)
1286  *|               : M_COMPL          (procedure in DEMO.PRG)
1287  *|               : M_DIAG           (procedure in DEMO.PRG)
1288  *|               : M_ASSIST         (procedure in DEMO.PRG)
1289  *|               : M_HANDP          (procedure in DEMO.PRG)
1290  *|               : M_TREAT          (procedure in DEMO.PRG)
1291  *|               : M_ENCSAV         (procedure in DEMO.PRG)
1292  *|               : M_CHANGE         (procedure in DEMO.PRG)
1293  *|               : M_PATPUT         (procedure in DEMO.PRG)
1294  *|               : M_PATADD         (procedure in DEMO.PRG)
1295  *|               : QUIT             (procedure in U.PRG)
1296  *|               : SEARCH           (procedure in U.PRG)
1297  *|               : MORESEARCH       (procedure in U.PRG)
1298  *|               : SFENCOUNT        (procedure in U.PRG)
1299  *|               : SFGETENC         (procedure in U.PRG)
1300  *|               : _Q8DOTWCZ4()     (function  in WP2.PRG)
1301  *|               : _Q8DOTWDV1()     (function  in WP2.PRG)
1302  *|
1303  *|           Calls: PUSHHELP         (procedure in U.PRG)
1304  *|               : W9.SPR
1305  *|               : POPHELP          (procedure in U.PRG)
1306  *|
1307  *|*****************************************************************
1281  | | | | | | | | | | FUNCTION popmessage
1282  | | | | | | | | | | PARAMETERS  errstr
1283  | | | | | | | | | | PRIVATE mbuttons
1284  | | | | | | | | | | mbuttons = 0
1285  | | | | | | | | | | =pushhelp("WARNING")
=>  && rwd02
1286  | | | | | | | | | | DO w9.spr
=>                && hl01
1287  | | | | | | | | | | =pophelp()
=>  && rwd02
1288  <-------------------------RETURN .T.
1289  | | | | | | | | | |
1290  | | | | | | | | | |
1291  | | | | | | | | | | *-------------------------------------
=> --------------
1292  | | | | | | | | | | *
1293  | | | | | | | | | | * display popup notice
1294  | | | | | | | | | | *
1295  | | | | | | | | | | *-------------------------------------
=> --------------
1309  *|*****************************************************************
1310  *|
1311  *|       Function: POPUPSHOW()
1312  *|
1313  *|      Called by: DEMO.PRG
1314  *|               : OPENFILES()      (function  in U.PRG)
1315  *|               : M_ASSIST         (procedure in DEMO.PRG)
1316  *|               : M_HANDP          (procedure in DEMO.PRG)
1317  *|               : M_ENCGET         (procedure in DEMO.PRG)
1318  *|               : SFREPORT         (procedure in DEMO.PRG)
1319  *|               : REINDEX          (procedure in U.PRG)
1320  *|
1321  *|*****************************************************************
1309  | | | | | | | | | | FUNCTION popupshow
=> && rwd07
1310  | | | | | | | | | | PARAMETERS errstr
1311  | | | | | | | | | |
1312  | | | | | | | | | | IF NOT WEXIST("w_popnote")
1313  | | | | | | | | | | DEFINE WINDOW w_popnote ;
1314  | | | | | | | | | | FROM INT((SROW()-8)/2),INT((SCOL()-
=> 36)/2) ;
1315  | | | | | | | | | | TO INT((SROW()-8)/2)+7,INT((SCOL()-
=> 36)/2)+35 ;
1316  | | | | | | | | | | TITLE "One moment" ;
1317  | | | | | | | | | | FLOAT ;
1318  | | | | | | | | | | CLOSE ;
1319  | | | | | | | | | | SHADOW ;
1320  | | | | | | | | | | DOUBLE ;
1321  | | | | | | | | | | COLOR SCHEME 1
1322  | | | | | | | | | | ENDIF
1323  | | | | | | | | | | IF WVISIBLE("w_popnote")
1324  | | | | | | | | | | ACTIVATE WINDOW w_popnote SAME
1325  | | | | | | | | | | ELSE
1326  | | | | | | | | | | ACTIVATE WINDOW w_popnote NOSHOW
1327  | | | | | | | | | | ENDIF
1328  | | | | | | | | | | @ 1,1 SAY errstr SIZE 3,31
1329  | | | | | | | | | |
1330  | | | | | | | | | | IF NOT WVISIBLE("w_popnote")
1331  | | | | | | | | | | ACTIVATE WINDOW w_popnote
1332  | | | | | | | | | | ENDIF
1333  <-------------------------RETURN ""
1334  | | | | | | | | | |
1335  | | | | | | | | | |
1351  *|*****************************************************************
1352  *|
1353  *|       Function: POPUPHIDE()
1354  *|
1355  *|      Called by: DEMO.PRG
1356  *|               : OPENFILES()      (function  in U.PRG)
1357  *|               : M_ASSIST         (procedure in DEMO.PRG)
1358  *|               : M_HANDP          (procedure in DEMO.PRG)
1359  *|               : M_ENCGET         (procedure in DEMO.PRG)
1360  *|               : SFREPORT         (procedure in DEMO.PRG)
1361  *|               : SEARCH           (procedure in U.PRG)
1362  *|               : MORESEARCH       (procedure in U.PRG)
1363  *|               : REINDEX          (procedure in U.PRG)
1364  *|
1365  *|*****************************************************************
1351  | | | | | | | | | | FUNCTION popuphide
1352  | | | | | | | | | | PARAMETERS W
1353  | | | | | | | | | | RELEASE WINDOW w_popnote
1354  <-------------------------RETURN W
1355  | | | | | | | | | |
1356  | | | | | | | | | |
```

```
1357                                    *-------------------------------------
 => --------------
1358                                    *
1359                                    * get diagnostic area from list
1360                                    *
1361                                    *-------------------------------------
 => --------------
1373  *|*****************************************************************
1374  *|
1375  *|       Function: GETAREALIST()
1376  *|
1377  *|      Called by: GETDIAGAREA        (procedure in DEMO.PRG)
1378  *|
1379  *|          Calls: PUSHHELP           (procedure in U.PRG)
1380  *|               : W5.SPR
1381  *|               : POPHELP           (procedure in U.PRG)
1382  *|
1383  *|*****************************************************************
1373                                    FUNCTION getarealist
1374                                       PARAMETERS mlist
1375
1376                                       EXTERNAL ARRAY mlist
1377                                       * open a info window
1378                                       PRIVATE mtitle
1379                                       mtitle = "Select Complaint"
1380
1381                                       PRIVATE N                          &&
 => number of names in file
1382                                       N = ALEN(mlist)
1383
1384                                       * select from list
1385                                       PRIVATE mchoice, mbuttons
1386                                       mchoice = 0
1387                                       mbuttons = 0
1388                                       =pushhelp("COMPLAINTS")
 =>      && rwd02
1389                                       DO w5.spr
 =>                && hll01
1390                                       =pophelp()
 =>      && rwd02
1391                                       DO CASE
1392                                       CASE mbuttons = 2
1393                                          mchoice = -1
1394                                       ENDCASE
1395  <------------------------------------RETURN mchoice
1396
1397                                       *-------------------------------------
 => --------------
1398                                       *
1399                                       * get patient from list
1400                                       *
1401                                       *-------------------------------------
 => --------------
1413  *|*****************************************************************
1414  *|
1415  *|       Function: GETPATLIST()
1416  *|
1417  *|      Called by: GETPATIENT         (procedure in DEMO.PRG)
1418  *|
1419  *|          Calls: PUSHHELP           (procedure in U.PRG)
1420  *|               : W3.SPR
1421  *|               : POPHELP           (procedure in U.PRG)
1422  *|
1423  *|*****************************************************************
1413                                    FUNCTION getpatlist
1414                                       PARAMETERS mkey, mlist
1415                                       * mkey is the search key
1416                                       * mlist is list of social security numbers
 => , names
1417
1418                                       EXTERNAL ARRAY mlist
1419
1420                                       PRIVATE mchoice
1421                                       IF ALEN(mlist) > 0
1422                                          * display menu of names in window
1423                                          PRIVATE i
1424                                          PRIVATE mtitle
1425                                          mtitle = "Patients on file" + IIF(TYPE(
 => "mkey")="C",," [" + ALLTRIM(mkey) + "] ","")
1426                                          PRIVATE mbuttons
1427                                          mchoice = 0
1428                                          mbuttons = 0
1429                                          =pushhelp("PATIENTS ON FILE")
 =>      && rwd02
1430                                          DO w3.spr
 =>                && hll01
1431                                          =pophelp()
 =>      && rwd02
1432                                          DO CASE
1433                                          CASE mbuttons = 2
1434                                             mchoice = 0
1435                                          ENDCASE
1436                                       ENDIF
1437  <------------------------------------RETURN mchoice
1438
1439                                       *-------------------------------------
 => --------------
1440                                       *
1441                                       * main menu screen
1442                                       *
1443                                       *-------------------------------------
 => --------------
1454  *|*****************************************************************
1455  *|
1456  *|       Function: MAINMENU()
1457  *|
1458  *|      Called by: DEMO.PRG
1459  *|
1460  *|          Calls: CAMD.MPR
1461  *|               : MYHANDLER         (procedure in U.PRG)
1462  *|
1463  *|*****************************************************************
1454                                    FUNCTION mainmenu
1455                                       PUBLIC dropdead, tobedone
1456                                       dropdead = .F.
1457                                       tobedone=""
1458                                       PUSH MENU _msysmenu
1459                                       DO camd.mpr
 =>                && hll01
1460                                       READ VALID myhandler()
1461                                       POP MENU _msysmenu
1462  <------------------------------------RETURN .T.
1463
1474  *|*****************************************************************
1475  *|
1476  *|      Procedure: MYHANDLER
1477  *|
1478  *|      Called by: MAINMENU()        (function  in U.PRG)
1479  *|
1480  *|          Calls: M_RESUME          (procedure in DEMO.PRG)
1481  *|               : M_REGPAT         (procedure in DEMO.PRG)
1482  *|
```

```
1483 *!*****************************************************************
1474                                FUNCTION myhandler
1475                                   PRIVATE m.temp
1476                                ┌IF dropdead
1477 <──────────────────────────────RETURN .T.
1478                                └ENDIF
1479                                ┌IF LEN(tobedone) > 0
1480                                │   m.temp = tobedone
1481                                │   tobedone = ""
1482                                │   DO (m.temp) IN demo.prg
1483 <──────────────────────────────RETURN .F.
1484                                └ENDIF
1485                                ┌DO CASE
1486                                ├CASE WONTOP("w_encount")
1487                                │   DO m_resume IN demo.prg
 =>                                │      && hll01
1488                                ├CASE WONTOP("w_patedit")
1489                                │   DO m_regpat IN demo.prg
 =>                                │      && hll01
1490                                └ENDCASE
1491 <──────────────────────────────RETURN .F.
1492
1502 *!*****************************************************************
1503 *!
1504 *!      Procedure: _MHIT
1505 *!
1506 *!      Called by: CAMD.MPR
1507 *!               : _Q9P0PFLR8          (procedure in CAMD.MPR)
1508 *!               : CAMDTR.MPR
1509 *!
1510 *!*****************************************************************
1502                                PROCEDURE _mhit
1503                                   PARAMETERS PROG
1504                                ┌IF RDLEVEL() > 1
1505                                │   tobedone = PROG
1506                                │   CLEAR READ
1507                                ├ELSE
1508                                │   DO (PROG) IN demo.prg
 =>                                │  && hll01
1509                                └ENDIF
1510
1521 *!*****************************************************************
1522 *!
1523 *!      Procedure: _QUIT
1524 *!
1525 *!      Called by: CAMD.MPR
1526 *!               : CAMDTR.MPR
1527 *!
1528 *!          Calls: POPMESSAGE()       (function  in U.PRG)
1529 *!
1530 *!*****************************************************************
1521                                PROCEDURE _quit
1522                                ┌IF EMPTY( WOUTPUT() )
 =>          && rwd07
1523                                │   dropdead = .T.
1524                                │   CLEAR READ ALL
1525                                ├ELSE
 =>          && rwd07
1526                                │   =popmessage("Close windows before quitt
 => ing")    && rwd07
1527                                └ENDIF
1528 <──────────────────────────────RETURN
1529
1530                                  *-------------------------------------
 => --------------
1531                                  *
```

```
1532                                  * edit patient info
1533                                  *
1534                                  *-------------------------------------
 => --------------
1546 *!*****************************************************************
1547 *!
1548 *!      Function: PATID()
1549 *!
1550 *!      Called by: M_REGPAT          (procedure in DEMO.PRG)
1551 *!
1552 *!          Calls: PUSHHELP          (procedure in U.PRG)
1553 *!               : W4.SPR
1554 *!               : POPHELP            (procedure in U.PRG)
1555 *!
1556 *!*****************************************************************
1546                                FUNCTION patid    *
1547                                   PARAMETERS mpatientid,msocsec,mlastname,mf
 => irstname,mdatebirth,msex,mraterank,isedit
1548
1549
1550                                  * open a dialog window
1551                                  PRIVATE mbuttons, mresult, mevent
1552                                  PRIVATE mebutton, mbbutton
1553                                  mbuttons = 0
1554                                  mebutton = 0
1555                                  mbbutton = 0
1556                                  PRIVATE quitting, saving
1557                                  quitting = .F.
1558                                  saving = .F.
1559                                  mevent = e_ok
1560                                  =pushhelp("PATIENT DETAILS")
 =>       && rwd02
1561                                  DO w4.spr
 =>                  && hll01
1562                                  =pophelp()
 =>       && rwd02
1563                                ┌IF mevent = e_ok  .AND.  saving
1564                                │   mevent = e_save
1565                                └ENDIF
1566 <──────────────────────────────RETURN mevent
1567
1568
1569                                  * Push button controls
1577 *!*****************************************************************
1578 *!
1579 *!      Procedure: _PVALID
1580 *!
1581 *!      Called by: W4.SPR
1582 *!
1583 *!*****************************************************************
1577                                FUNCTION _pvalid
1578                                ┌DO CASE
1579                                ├CASE mbuttons = 1
1580                                │   * OK button
1581                                │   quitting = .T.
1582                                │ ┌IF mpatientid != 0
1583                                │ │   mevent = e_ok
1584                                │ ├ELSE
1585                                │ │   * nothing selected
1586                                │ │   mevent = e_esc
1587                                │ └ENDIF
1588                                │   CLEAR READ
1589                                ├CASE mbuttons = 2
1590                                │   * new button
1591                                │   mevent = e_init
1592                                │   quitting = .T.
```

```
1593                    CLEAR READ
1594                 CASE mbuttons = 3
1595                    * cancel button
1596                    quitting = .T.
1597                    mevent = e_esc
1598                    CLEAR READ
1599                 ENDCASE
1600              SHOW GETS
1601              RETURN
1602
1603              *-------------------------------------
 => --------------
1604              *
1605              * undo data popup menu screen
1606              *
1607              *-------------------------------------
 => --------------
1617 *!*****************************************************************
1618 *!
1619 *!       Function: UNDOMENU()
1620 *!
1621 *!      Called by: MYUNDO              (procedure in DEMO.PRG)
1622 *!
1623 *!          Calls: W13.SPR
1624 *!
1625 *!*****************************************************************
1617              FUNCTION undomenu
1618
1619              * display menu prompts inside window
1620              PRIVATE mchoice, mevent, mbuttons, mpad
1621              DIMENSION mpad[4]
1622              mpad[1] = "Previous question"
1623              mpad[2] = "Continue current"
1624              mpad[3] = "Browse"
1625              mpad[4] = "Quit session"
1626              * get user's selection from menu
1627              mchoice = 0
1628              mbuttons = 0
1629              mevent = e_ok
1630              DO w13.spr
 =>        && hilo
1631              DO CASE
1632              CASE mchoice = 1
1633                 mevent = e_goback          && go back to
 => previous question
1634              CASE mchoice = 2
1635                 mevent = e_redo            && continue w
 => ith current dialog
1636              CASE mchoice = 3
1637                 mevent = e_change          && review and
 => change data
1638              CASE mchoice = 4
1639                 mevent = e_esc             && quit sessi
 => on
1640              OTHERWISE
1641                 mevent = e_redo            && default is
 => continue
1642              ENDCASE
1643              RETURN mevent
1644
1681 *!*****************************************************************
1682 *!
1683 *!       Procedure: PUSHHELP
1684 *!
1685 *!      Called by: WP2.PRG
1686 *!                 : USERINIT()        (function  in U.PRG)
1687 *!                 : WPAT              (procedure in DEMO.PRG)
1688 *!                 : EDITTEXT          (procedure in DEMO.PRG)
1689 *!                 : BACKUP            (procedure in DEMO.PRG)
1690 *!                 : RESTORE           (procedure in DEMO.PRG)
1691 *!                 : ENCNTTITLE()      (function  in U.PRG)
1692 *!                 : WCONFIRM()        (function  in U.PRG)
1693 *!                 : PATID()           (function  in U.PRG)
1694 *!                 : POPMESSAGE()      (function  in U.PRG)
1695 *!                 : GETDISLIST()      (function  in U.PRG)
1696 *!                 : GETAREALIST()     (function  in U.PRG)
1697 *!                 : GETPATLIST()      (function  in U.PRG)
1698 *!                 : GETENCLIST()      (function  in U.PRG)
1699 *!                 : EDNUM()           (function  in U.PRG)
1700 *!                 : EDBOOL()          (function  in U.PRG)
1701 *!                 : EDCHECK()         (function  in U.PRG)
1702 *!                 : WCHOICE           (procedure in U.PRG)
1703 *!                 : GETDISLOOKUPLIST  (procedure in U.PRG)
1704 *!                 : EDLIST            (procedure in U.PRG)
1705 *!                 : _EVALID           (procedure in U.PRG)
1706 *!                 : _EDITVDIAG        (procedure in U.PRG)
1707 *!                 : DPGETPATLIST      (procedure in U.PRG)
1708 *!                 : SEARCH            (procedure in U.PRG)
1709 *!                 : SFENCOUNT         (procedure in U.PRG)
1710 *!                 : SFENCLIST         (procedure in U.PRG)
1711 *!                 : SFGETPATLIST      (procedure in U.PRG)
1712 *!                 : _Q9POPFLOT        (procedure in CAND.MPR)
1713 *!
1714 *!          Calls: SETHELP             (procedure in U.PRG)
1715 *!
1716 *!*****************************************************************
1681              FUNCTION pushhelp
1682                 PARAMETERS helptopic
1683                 IF help_level  <  ALEN( help_stack )
1684                    help_level = help_level + 1
1685                 ENDIF
1686                 help_stack[help_level] = helptopic
1687                 =sethelp()
1688              RETURN help_level
1689
1725 *!*****************************************************************
1726 *!
1727 *!       Procedure: POPHELP
1728 *!
1729 *!      Called by: WP2.PRG
1730 *!                 : WPAT              (procedure in DEMO.PRG)
1731 *!                 : EDITTEXT          (procedure in DEMO.PRG)
1732 *!                 : BACKUP            (procedure in DEMO.PRG)
1733 *!                 : RESTORE           (procedure in DEMO.PRG)
1734 *!                 : ENCNTTITLE()      (function  in U.PRG)
1735 *!                 : WCONFIRM()        (function  in U.PRG)
1736 *!                 : PATID()           (function  in U.PRG)
1737 *!                 : POPMESSAGE()      (function  in U.PRG)
1738 *!                 : GETDISLIST()      (function  in U.PRG)
1739 *!                 : GETAREALIST()     (function  in U.PRG)
1740 *!                 : GETPATLIST()      (function  in U.PRG)
1741 *!                 : GETENCLIST()      (function  in U.PRG)
1742 *!                 : EDNUM()           (function  in U.PRG)
1743 *!                 : EDBOOL()          (function  in U.PRG)
1744 *!                 : EDCHECK()         (function  in U.PRG)
1745 *!                 : WCHOICE           (procedure in U.PRG)
1746 *!                 : GETDISLOOKUPLIST  (procedure in U.PRG)
1747 *!                 : EDLIST            (procedure in U.PRG)
1748 *!                 : _EVALID           (procedure in U.PRG)
1749 *!                 : _EDITVDIAG        (procedure in U.PRG)
1750 *!                 : DPGETPATLIST      (procedure in U.PRG)
1751 *!                 : SEARCH            (procedure in U.PRG)
```

```
1752  *|                : SFENCOUNT          (procedure in U.PRG)
1753  *|                : SFENCLIST          (procedure in U.PRG)
1754  *|                : SFGETPATLIST       (procedure in U.PRG)
1755  *|                : _Q9POPFLOY         (procedure in CAMD.MPR)
1756  *|
1757  *|          Calls: SETHELP             (procedure in U.PRG)
1758  *|
1759  *|*****************************************************************
1725                                   FUNCTION pophelp
1726                                     IF help_level > 1
1727                                        help_level = help_level - 1
1728                                        =sethelp()
1729                                     ENDIF
1730  <                                  RETURN help_level
1731
1740  *|*****************************************************************
1741  *|
1742  *|      Procedure: SETHELP
1743  *|
1744  *|      Called by: PUSHHELP            (procedure in U.PRG)
1745  *|              : POPHELP             (procedure in U.PRG)
1746  *|
1747  *|*****************************************************************
1740                                   FUNCTION sethelp
1741                                      help_id = help_stack[help_level]
1742                                      SET TOPIC TO help_id
1743  <                                  RETURN ""
1744
1745                                      *-------------------------------------
 => -------------
1746                                      *
1747                                      * on-line medical reference
1748                                      *
1749                                      *-------------------------------------
 => -------------
1759  *|*****************************************************************
1760  *|
1761  *|      Procedure: OLMR
1762  *|
1763  *|      Called by: INITHELP           (procedure in U.PRG)
1764  *|
1765  *|    Other Files: TEMP.WIN
1766  *|
1767  *|*****************************************************************
1759                                   FUNCTION olmr
 =>   && rwd06
1760                                      PRIVATE m.home, m.drv, m.dp
1761                                      SAVE WINDOW ALL TO temp.win
 =>     && rwd07
1762                                      CLEAR WINDOWS
 =>     && rwd07
1763                                      SAVE SCREEN
 =>     && rwd06
1764                                      m.dp = GETENV("DP")
 =>     && dp application directory
1765                                      m.drv = SYS(5)
 =>     && home drive
1766                                      m.home = SYS(2003)
 =>     && home directory
1767                                      SET DEFAULT TO (m.dp)
 =>     && switch to dp application
1768                                      ! /0 dp
 =>     && run dp application
1769                                      SET DEFAULT TO (m.drv)
 =>     && return to home drive
1770                                      SET DEFAULT TO (m.home)
```



```
 =>     && return to home directory
1771                                      RESTORE SCREEN
 =>     && rwd06
1772                                      RESTORE WINDOW ALL FROM temp.win
 =>     && rwd07
1773  <                                  RETURN ""
1774
1775
1787  *|*****************************************************************
1788  *|
1789  *|      Procedure: INITHELP
1790  *|
1791  *|      Called by: USERINIT()          (function  in U.PRG)
1792  *|
1793  *|          Calls: OLMR               (procedure in U.PRG)
1794  *|
1795  *|    Other Files: HELP
1796  *|
1797  *|*****************************************************************
1787                                   FUNCTION inithelp
1788                                      SET HELP TO HELP
1789                                      *SET HELPFILTER TO source = "A"
 =>     && rwd02
1790                                      ON KEY LABEL f1 HELP
1791                                      ON KEY LABEL f2 DO olmr
 =>     && rwd06  rwd07
1792  <                                  RETURN ""
1793
1794                                      * end user interface functions
1795
1796                                      * database interface functions
1797
1798                                      *-------------------------------------
 => -------------
1799                                      *
1800                                      * data base functions for diagnostic knowl
 => edge base domain
1801                                      *
1802                                      *-------------------------------------
 => -------------
1803
1804                                      *-------------------------------------
 => -------------
1805                                      *
1806                                      * count diagnostic area choices
1807                                      *
1808                                      *-------------------------------------
 => -------------
1818  *|*****************************************************************
1819  *|
1820  *|       Function: AREACOUNT()
1821  *|
1822  *|      Called by: GETDIAGAREA         (procedure in DEMO.PRG)
1823  *|
1824  *|          Uses: AREA.DBF
1825  *|
1826  *|*****************************************************************
1818                                   FUNCTION areacount
1819
1820                                      PRIVATE N
1821                                      PRIVATE msel
1822                                      msel = SELECT()              && save cur
 => rent database selection
1823                                      SELECT 0
1824                                      USE area AGAIN
1825                                      N = RECCOUNT()
```

```
1826                    USE                                          &&
=> close file
1827                    SELECT (msel)                                &&
=>  reselect original database
1828  <-----------------RETURN N                                 && list
=> of names
1829
1830                    *-------------------------------------
=> --------------
1831                    *
1832                    * get list of diagnostic area choices
1833                    *
1834                    *-------------------------------------
=> --------------
1846  *!*****************************************************************
1847  *!
1848  *!        Function: AREANAMES()
1849  *!
1850  *!       Called by: GETDIAGAREA        (procedure in DEMO.PRG)
1851  *!
1852  *!            Uses: AREA.DBF
1853  *!
1854  *!         Indexes: AREA.IDX
1855  *!*****************************************************************
1846                    FUNCTION areanames
1847                       PARAMETERS marea, mname
1848                       EXTERNAL ARRAY mname, marea
=>          && rwd07
1849
1850                       PRIVATE i
1851                       PRIVATE msel
1852                       msel = SELECT()                   && save cur
=> rent database selection
1853                       SELECT 0
1854                       USE area AGAIN INDEX area
=>     && list of diagnostic areas
1855                       GOTO TOP
1856                       i = 1
1857                       DO WHILE .NOT. EOF()
1858                          marea[i] = area
1859                          mname[i] = TRIM(name)
1860                          i = i + 1
1861                          SKIP
1862                       ENDDO
1863                       USE                                       &&
=> close file
1864                       SELECT (msel)                             &&
=>  reselect original database
1865  <-----------------RETURN i                                   &&
=> list of names
1866
1867                       *-------------------------------------
=> --------------
1868                       *
1869                       * get diagnostic area name
1870                       *
1871                       *-------------------------------------
=> --------------
1884  *!*****************************************************************
1885  *!
1886  *!        Function: GETEXAREA()
1887  *!
1888  *!       Called by: M_RESUME           (procedure in DEMO.PRG)
1889  *!                : _EDITCOMPL         (procedure in U.PRG)
1890  *!
1891  *!            Uses: AREA.DBF
1892  *!
1893  *!         Indexes: AREA.IDX
1895  *!*****************************************************************
1884                    FUNCTION getexarea
1885
1886                       PARAMETERS marea
1887
1888                       PRIVATE mresult
1889                       PRIVATE msel
1890
1891                       mresult = ""
1892                       IF !EMPTY(marea)
1893                          msel = SELECT()                        &&
=> save current database selection
1894                          SELECT 0
1895                          USE area AGAIN INDEX area
=>     && list of diagnostic areas
1896                          SEEK marea
1897                          mresult = IIF(FOUND(),  TRIM(name),  ""
=> )
1898                          USE                                    &&
=> close file
1899                          SELECT (msel)                          &&
=> reselect original database
1900                       ENDIF
1901  <-----------------RETURN mresult                             &&
=> extended name
1902
1903                       *-------------------------------------
=> --------------
1904                       *
1905                       * get rulebase for selected diagnostic are
=> a
1906                       *
1907                       *-------------------------------------
=> --------------
1919  *!*****************************************************************
1920  *!
1921  *!        Function: GETRULEBASE()
1922  *!
1923  *!       Called by: LODRULE            (procedure in DEMO.PRG)
1924  *!
1925  *!            Uses: AREA.DBF
1926  *!
1927  *!         Indexes: AREA.IDX
1928  *!
1929  *!*****************************************************************
19?2                    FUNCTION getrulebase
1920
1921                       PARAMETERS marea, cthreshold, cthreshpro,
=> cthresh(ik, nrules
1922
1923                       PRIVATE mrulebase
1924                       PRIVATE msel
1925                       msel = SELECT()                           &&
=> save current database selection
1926                       mrulebase = ""
1927                       nrules = 0
1928                       IF !EMPTY(marea)
1929                          SELECT 0
1930                          USE area AGAIN INDEX area
=>     && list of diagnostic areas
1931                          SEEK marea
1932                          IF FOUND()
```

```
1933                    mrulebase = TRIM(rulebase)      &&
 => extract rulebase name and
1934                    cthreshold = threshold          &&
 => possible diagnosis threshold
1935                    cthreshpro = probable           &&
 => probable diagnosis threshold
1936                    cthreshlik = likely             &&
 => most likely diagnosis threshold
1937                    nrules     = rules              &&
 => number of rules in rulebase
1938                  ENDIF
1939                  USE                               &&
 => close file
1940                  SELECT (msel)                     &&
 => reselect original database
1941                ENDIF
1942 <------------------------RETURN  mrulebase
1943
1944                *-------------------------------------
 => --------------
1945                *
1946                * access colors database
1947                *
1948                *-------------------------------------
 => --------------
1949
1950                *-------------------------------------
 => --------------
1951                *
1952                * load optional color set
1953                *
1954                *-------------------------------------
 => --------------
1962 *!*****************************************************************
1963 *!
1964 *!      Procedure: GETCOLORS
1965 *!
1966 *!      Called by: USERINIT()          (function  in U.PRG)
1967 *!
1968 *!*****************************************************************
1962                FUNCTION getcolors
1963
1964                PARAMETERS colorset
1965
1966                PRIVATE msel
1967                msel = SELECT()
1968                SELECT 0
1969                USE (colorset) AGAIN
1970                DO WHILE .NOT. EOF()
1971                   PRIVATE mname, mcolor
1972                   mname  = TRIM(name)
1973                   mcolor = TRIM(COLOR)
1974                   &mname = COLOR
1975                   SKIP
1976                ENDDO
1977                USE
1978                SELECT (msel)
1979 <------------------------RETURN .T.
1980
1981
1982                *-------------------------------------
 => --------------
1983                *
1984                * data base functions for diagnosis
1985                *
1986                *-------------------------------------
```

```
 => --------------
1987                *-------------------------------------
 => --------------
1988                *
1989                * update encounter data base with diagnose
 => s
1990                *
1991                *-------------------------------------
 => --------------
2001 *!*****************************************************************
2002 *!
2003 *!      Function: PUTDIAGNOS()
2004 *!
2005 *!     Called by: M_ENCSAV          (procedure in DEMO.PRG)
2006 *!
2007 *!         Calls: OPENDIAG          (procedure in U.PRG)
2008 *!
2009 *!*****************************************************************
2001                FUNCTION putdiagnos
2002
2003                PARAMETERS mencountid, mdiag, mconf
2004
2005                EXTERNAL ARRAY mdiag, mconf
 =>                && rwd07
2006                PRIVATE i
2007                PRIVATE nrecs                   && have exi
 => sting records
2008                PRIVATE trypool
2009                trypool = .T.          && try to get record
 =>  from free pool
2010                PRIVATE msel
2011                msel = SELECT()        && save current data
 => base selection
2012
2013                =opendiag()
2014
2015                * find existing transactions if any
2016                SEEK mencountid
2017                nrecs =  FOUND()
2018
2019                * save diagnostic result
2020                FOR  i = 1  TO  ALEN( mdiag )
2021                   IF  !EMPTY( mconf[i] )
2022                      IF !nrecs
2023                         * no more existing records, try t
 => he free pool
2024                         IF trypool
2025                            SEEK 0                  && use su
 => rplus records
2026                            trypool = FOUND()
2027                         ENDIF
2028                         IF !trypool
2029                            * free pool is exhausted
2030                            APPEND BLANK            &&  add n
 => ew record to file
2031                         ENDIF
2032                         REPLACE id  WITH mencountid
2033                      ENDIF
2034                      * edit new or existing item record
2035                      REPLACE disease    WITH  VAL( mdiag[
 => i] )
2036                      REPLACE probability WITH  mconf[i]
2037                      IF nrecs
2038                         * still updating existing records
2039                         * skip to next available record
2040                         SKIP
```

```
2041                                    nrecs  =  id = mencountid  .AND.
 => !EOF()
2042                                  ENDIF
2043                                ENDIF
2044                                NEXT
2045
2046                                * put any surplus records in pool with
 => index = 0
2047                                * (this is more efficient than deleting
 => /packing)
2048                                IF nrecs
2049                                   * still records remaining after abov
 => e update
2050                                   * put records in pool by changing th
 => eir IDs
2051                                   REPLACE id WITH 0 WHILE id = mencoun
 => tid
2052                                ENDIF
2053                                USE
2054                                SELECT (msel)                 && reselec
 => t original database
2055                                RETURN .T.
2056
2057                                *-----------------------------------
 => ----------------
2058                                *
2059                                * get diagnosis from encounter file
2060                                *
2061                                *-----------------------------------
 => ----------------
2072  *!*****************************************************************
2073  *!
2074  *!       Function: GETDIAGNOS()
2075  *!
2076  *!      Called by: LODENC              (procedure in DEMO.PRG)
2077  *!
2078  *!          Calls: OPENDIAG            (procedure in U.PRG)
2079  *!               : ADDDIAGNOS          (procedure in DEMO.PRG)
2080  *!
2081  *!*****************************************************************
2072                                FUNCTION getdiagnos
2073
2074                                PARAMETERS mencountid
2075
2076                                PRIVATE msel
2077                                msel = SELECT()               && save c
 => urrent database selection
2078                                =opendiag()
2079                                SCAN FOR id = mencountid
2080                                   =adddiagnos( STR( disease ), probabi
 => lty)
2081                                ENDSCAN
2082                                USE
2083                                SELECT (msel)                 && reselec
 => t original database
2084                                RETURN .T.
2085
2086                                *-----------------------------------
 => ----------------
2087                                *
2088                                * open diagnosis encounter file
2089                                *
2090                                *-----------------------------------
 => ----------------
2099  *!*****************************************************************
2100  *!
```

```
2101  *!      Procedure: OPENDIAG
2102  *!
2103  *!      Called by: PUTDIAGNOS()       (function  in U.PRG)
2104  *!               : GETDIAGNOS()       (function  in U.PRG)
2105  *!
2106  *!*****************************************************************
2099                                FUNCTION opendiag
2100                                   SELECT 0
2101                                   USE (f_diagnose) AGAIN INDEX (f_diagnos
 => e)     && open encounter/diagnosis
2102                                   RETURN SELECT()
2103
2104                                   *-----------------------------------
 => ----------------
2105                                   *
2106                                   * data base functions for data dictiona
 => ry
2107                                   *
2108                                   *-----------------------------------
 => ----------------
2109
2110                                   *-----------------------------------
 => ----------------
2111                                   *
2112                                   * open data dictionary files
2113                                   *
2114                                   *-----------------------------------
 => ----------------
2131  *!*****************************************************************
2132  *!
2133  *!      Procedure: OPENDICT
2134  *!
2135  *!      Called by: OPENFILES()        (function  in U.PRG)
2136  *!
2137  *!          Uses: DICT.DBF
2138  *!              : VAL.DBF                 Alias: DICTVAL
2139  *!              : ENUM.DBF
2140  *!
2141  *!       Indexes: DICTID.IDX
2142  *!              : DICTNAME.IDX
2143  *!              : VAL.IDX
2144  *!              : ENUM.IDX
2145  *!
2146  *!*****************************************************************
2131                                FUNCTION opendict
2132
2133                                   SELECT 0
2134                                   USF dict AGAIN INDEX dictid,dictname
2135                                   SELECT 0
2136                                   USE VAL AGAIN INDEX VAL ALIAS dictval
2137                                   SELECT 0
2138                                   USE enum AGAIN INDEX enum
2139                                   RETURN .T.
2140
2141                                   *-----------------------------------
 => ----------------
2142                                   *
2143                                   * close data dictionary files
2144                                   *
2145                                   *-----------------------------------
 => ----------------
2153  *!*****************************************************************
2154  *!
2155  *!      Procedure: CLOSEDICT
2156  *!
2157  *!      Called by: FLUSHFILES()       (function  in U.PRG)
```

```
2158  *|
2159  *|*****************************************************************
2153                                  FUNCTION closedict
2154
2155                                     PRIVATE msel
2156                                     msel = SELECT()              && save c
 => urrent database selection
2157                                     SELECT dict
2158                                     USE
2159                                     SELECT dictval
2160                                     USE
2161                                     SELECT enum
2162                                     USE
2163                                     SELECT (msel)
 =>   && reselect original database
2164  <-------------------------------RETURN .T.
2165
2166                                     *-----------------------------------
 => ----------------
2167                                     *
2168                                     * size data dictionary
2169                                     *
2170                                     *-----------------------------------
 => ----------------
2180  *|*****************************************************************
2181  *|
2182  *|       Function: DICTCOUNT()
2183  *|
2184  *|      Called by: INISYMPTOM()       (function  in DEMO.PRG)
2185  *|
2186  *|           Uses: DICT.DBF
2187  *|
2188  *|*****************************************************************
2180                                  FUNCTION dictcount
2181                                     PRIVATE nconditions
2182                                     PRIVATE msel
2183                                     msel = SELECT()
 =>  && save current database selection
2184                                     SELECT 0
2185                                     USE dict AGAIN
 =>  && list of elements
2186                                     nconditions= RECCOUNT()
 =>  && number of names in file
2187                                     USE
 =>  && close file
2188                                     SELECT (msel)
 =>  && reselect original database
2189  <-------------------------------RETURN nconditions
2190
2191
2192                                     *-----------------------------------
 => ----------------
2193                                     *
2194                                     * Lookup Dictionary item
2195                                     *
2196                                     *-----------------------------------
 => ----------------
2208  *|*****************************************************************
2209  *|
2210  *|       Function: DICTQ()
2211  *|
2212  *|      Called by: LOOKUPQ            (procedure in DEMO.PRG)
2213  *|               : LOOKUPNAM          (procedure in DEMO.PRG)
2214  *|               : EDITSYMP           (procedure in DEMO.PRG)
2215  *|               : PUTSYMPTOM()       (function  in U.PRG)
2216  *|               : GETSYMPTOM()       (function  in U.PRG)
```

```
2217  *|
2218  *|*****************************************************************
2208                                  FUNCTION dictq
2209                                     PARAMETERS mid, mask, mq, mname, mpromp
 => t
2210
2211                                     PRIVATE mtype
2212                                     PRIVATE msel
2213                                     msel = SELECT()              && save c
 => urrent database selection
2214                                     SELECT dict
2215                                     * look up identifier in dictionary
2216                                     SEEK VAL(mid)
2217                                     IF FOUND()
2218                                        * return associated attributes
2219                                        mtype    ^  = datatype
2220                                        IF PARAMETERS() > 1
2221                                           mask       = askable
2222                                        ENDIF
2223                                        IF PARAMETERS() > 2
2224                                           mq         = TRIM(question)
2225                                        ENDIF
2226                                        IF PARAMETERS() > 3
2227                                           mname      = name
2228                                        ENDIF
2229                                        IF PARAMETERS() > 4
2230                                           mprompt    = TRIM(PROMPT)
2231                                        ENDIF
2232                                     ELSE
2233                                        mtype = ""
2234                                     ENDIF
2235                                     SELECT (msel)                && restore
 =>  database selection
2236  <-------------------------------RETURN mtype
2237
2238                                     *-----------------------------------
 => ----------------
2239                                     *
2240                                     * Lookup Dictionary numeric value item
2241                                     *
2242                                     *-----------------------------------
 => ----------------
2250  *|*****************************************************************
2251  *|
2252  *|       Function: DICTQV()
2253  *|
2254  *|      Called by: EDITSYMP           (procedure in DEMO.PRG)
2255  *|
2256  *|*****************************************************************
2250                                  FUNCTION dictqv
2251                                     PARAMETERS mid, mhi, mlo, mwidth, mdec,
 =>  munits
2252
2253                                     PRIVATE mresult
2254                                     PRIVATE msel
2255                                     msel = SELECT()              && save c
 => urrent database selection
2256                                     SELECT dictval
2257                                     * look up identifier in dictionary
2258                                     SEEK VAL(mid)
2259                                     IF FOUND()
2260                                        mresult = .T.
2261                                        mhi     = hi
2262                                        mlo     = lo
2263                                        mwidth  = WIDTH
2264                                        mdec    = dec
```

```
2265                munits   = TRIM(units)
2266             ELSE
2267                mresult = .F.
2268             ENDIF
2269             SELECT (msel)                          &&
 => restore database selection
2270 <------------------------------RETURN mresult
2271
2272             *-----------------------------------
 => ----------------
2273             *
2274             * size dictionary enumerated item
2275             *
2276             *-----------------------------------
 => ----------------
2284 *!*****************************************************************
2285 *!
2286 *!       Function: DICTECOUNT()
2287 *!
2288 *!      Called by: EDITSYMP           (procedure in DEMO.PRG)
2289 *!
2290 *!*****************************************************************
2284          FUNCTION dictecount
2285             PARAMETERS mid
2286
2287             PRIVATE N
2288             PRIVATE mkey
2289             PRIVATE msel
2290             msel = SELECT()
 => && save current database selection
2291             SELECT enum
2292             * look up identifier in dictionary
2293             N = 0
2294             mkey = VAL(mid)
2295             SEEK mkey
2296             * count list of names
2297             COUNT WHILE mkey = id .AND. .NOT. EOF()
 => TO N
2298             SELECT (msel)
 =>  && reselect original database
2299 <------------------------------RETURN N
2300
2301             *-----------------------------------
 => ----------------
2302             *
2303             * Lookup Dictionary enumerated value it
 => em
2304             *
2305             *-----------------------------------
 => ----------------
2313 *!*****************************************************************
2314 *!
2315 *!       Function: DICTQE()
2316 *!
2317 *!      Called by: EDITSYMP           (procedure in DEMO.PRG)
2318 *!
2319 *!*****************************************************************
2313          FUNCTION dictqe
2314             PARAMETERS mid, names, mattrib
2315
2316             EXTERNAL ARRAY names, mattrib
 =>                && rwd07
2317
2318             PRIVATE i
2319             PRIVATE mkey
2320             PRIVATE msel
2321             msel = SELECT()                        && save c
 => urrent database selection
2322             SELECT enum
2323
2324             * look up identifier in dictionary
2325             i = 0
2326             mkey = VAL(mid)
2327             SEEK mkey
2328             IF FOUND()
2329                * create list of names
2330                i = 1
2331                DO WHILE mkey = id .AND. .NOT. EOF()
2332                   names[i] = TRIM( enumerate )
2333                   mattrib[i] = mutex
2334                   i = i + 1
2335                   SKIP
2336                ENDDO
2337             ENDIF
2338             SELECT (msel)                          &&
 => restore database selection
2339 <------------------------------RETURN i
2340
2341             *-----------------------------------
 => ----------------
2342             *
2343             * Lookup Dictionary enumerated item ord
 => inal
2344             *
2345             *-----------------------------------
 => ----------------
2353 *!*****************************************************************
2354 *!
2355 *!      Procedure: DICTORD
2356 *!
2357 *!      Called by: PUTSYMPTOM()       (function  in U.PRG)
2358 *!
2359 *!*****************************************************************
2353          FUNCTION dictord
2354             PARAMETERS mid, menum
2355
2356             PRIVATE mkey, mord
2357             PRIVATE msel
2358             msel = SELECT()                        && save c
 => urrent database selection
2359             SELECT enum
2360
2361             * look up identifier in dictionary
2362             mkey = VAL(mid)
2363             mord = 0
2364             SEEK mkey
2365             IF FOUND()
2366                * find enumeration
2367                DO WHILE  mkey = id  .AND. .NOT. EOF
 => ()
2368                   IF enumerate = menum
2369                      mord = ord                    && return
 =>  corresponding ordinal
2370                   EXIT
2371                   ENDIF
2372                   SKIP
2373                ENDDO
2374             ENDIF
2375             SELECT (msel)                          &&
 => restore database selection
2376 <------------------------------RETURN  ALLTRIM( STR( mord ) )
2377
```

```
2378                                    *-----------------------------------
 => -----------------
2379                                    *
2380                                    * Lookup Dictionary ordinal item enumer
 => ated val
2381                                    *
2382                                    *-----------------------------------
 => -----------------
2390  *!*****************************************************************
2391  *!
2392  *!      Procedure: DICTENUM
2393  *!
2394  *!      Called by: GETSYMPTOM()       (function  in U.PRG)
2395  *!
2396  *!*****************************************************************
2390                                  FUNCTION dictenum
2391                                     PARAMETERS mid, mord
2392
2393                                     PRIVATE mkey, menum, mvord
2394                                     PRIVATE msel
2395                                     msel = SELECT()              && save c
 => urrent database selection
2396                                     SELECT enum
2397
2398                                     * look up identifier in dictionary
2399                                     mkey = VAL(mid)
2400                                     mvord = VAL(mord)
2401                                     SEEK mkey
2402                                    IF FOUND()
2403                                       * find enumeration
2404                                      DO WHILE  mkey = id  .AND. .NOT. EOF
 => ()
2405                                        IF ord = mvord
2406                                           menum = TRIM(enumerate)   && r
 => eturn corresponding enumerate
2407                                      EXIT
2408                                        ENDIF
2409                                        SKIP
2410                                      ENDDO
2411                                    ENDIF
2412                                     SELECT (msel)                   &&
 => restore database selection
2413  <------------------------------RETURN  menum
2414
2415                                     *-----------------------------------
 => -----------------
2416                                     *
2417                                     * Lookup Dictionary enumerated item rep
 => ort
2418                                     *
2419                                     *-----------------------------------
 => -----------------
2428  *!*****************************************************************
2429  *!
2430  *!      Function: DICTREPORT()
2431  *!
2432  *!      Called by: STRNATURAL         (procedure in DEMO.PRG)
2433  *!               : SFSYMP             (procedure in DEMO.PRG)
2434  *!
2435  *!*****************************************************************
2428                                  FUNCTION dictreport
2429                                     PARAMETERS mid, menum
2430
2431                                     PRIVATE mkey, mreport
2432                                     PRIVATE msel
2433                                     msel = SELECT()              && save c
```

```
 => urrent database selection
2434                                     SELECT enum
2435
2436                                     * look up identifier in dictionary
2437                                     mkey = VAL(mid)
2438                                     mreport = ""
2439                                     SEEK mkey
2440                                    IF FOUND()
2441                                       * find enumeration
2442                                      DO WHILE  mkey = id  .AND. .NOT. EOF
 => ()
2443                                        IF enumerate = menum
2444                                           mreport = REPORT      && retur
 => n corresponding report
2445                                      EXIT
2446                                        ENDIF
2447                                        SKIP
2448                                      ENDDO
2449                                    ENDIF
2450                                     SELECT (msel)                   &&
 => restore database selection
2451  <------------------------------RETURN  ALLTRIM( mreport )
2452
2453                                     *-----------------------------------
 => -----------------
2454                                     *
2455                                     * Lookup question
2456                                     *
2457                                     *-----------------------------------
 => -----------------
2463  *!*****************************************************************
2464  *!
2465  *!      Procedure: DICTQID
2466  *!
2467  *!*****************************************************************
2463                                  FUNCTION dictqid
2464                                     PARAMETERS mname
2465
2466                                     PRIVATE mid
2467                                     PRIVATE msel
2468                                     msel = SELECT()              && save c
 => urrent database selection
2469                                     SELECT dict
2470                                     SET ORDER TO 3               && look u
 => p dictionary question
2471                                     SEEK mname
2472                                    IF FOUND()
2473                                       mid = ALLTRIM(STR(id))
2474                                    ELSE
2475                                       mid = ""
2476                                    ENDIF
2477                                     SET ORDER TO 1
2478                                     SELECT (msel)                && restore
 =>  database selection
2479  <------------------------------RETURN mid
2480
2481                                     *-----------------------------------
 => -----------------
2482                                     *
2483                                     * Lookup name
2484                                     *
2485                                     *-----------------------------------
 => -----------------
2493  *!*****************************************************************
2494  *!
2495  *!      Function: DICTQNAME()
```

```
2496  *|
2497  *|      Called by: ADDHISTORY          (procedure in DEMO.PRG)
2498  *|
2499  *|*****************************************************************
2493                                          FUNCTION dictqname
2494                                             PARAMETERS mname
2495
2496                                             PRIVATE mid
2497                                             PRIVATE msel
2498                                             msel = SELECT()                && save c
 => urrent database selection
2499                                             SELECT dict
2500                                             SET ORDER TO 2              && look up d
 => ictionary name
2501                                             SEEK mname
2502                                             IF FOUND()
2503                                                mid = ALLTRIM(STR(id))
2504                                             ELSE
2505                                                mid = ""
2506                                             ENDIF
2507                                             SET ORDER TO 1
2508                                             SELECT (msel)                  && restore
 =>  database selection
2509  <------------------------------------------RETURN mid
2510
2511                                             *-----------------------------------
 => ----------------
2512                                             *
2513                                             * data base functions for patient encou
 => nter
2514                                             *
2515                                             *-----------------------------------
 => ----------------
2516
2517                                             *-----------------------------------
 => ----------------
2518                                             *
2519                                             * open encounter files
2520                                             *
2521                                             *-----------------------------------
 => ----------------
2529  *|*****************************************************************
2530  *|
2531  *|      Procedure: OPENENCNT
2532  *|
2533  *|      Called by: OPENFILES()          (function  in U.PRG)
2534  *|
2535  *|*****************************************************************
2529                                          FUNCTION openencnt

2530
2531                                             PUBLIC fencountr
2532                                             IF !USED(f_encountr)
2533                                                SELECT 0
2534                                                USE (f_encountr) AGAIN INDEX (f_enco
 => untr), (f_encountd), (f_encountl)
2535                                                fencountr = SELECT()
2536                                             ELSE
2537                                                SELECT (f_encountr)
2538                                                fencountr = SELECT()
2539                                             ENDIF
2540  <------------------------------------------RETURN .T.
2541
2542                                             *-----------------------------------
 => ----------------
2543                                             *
2544                                             * flush encounter files
```

```
2545                                             *
2546                                             *-----------------------------------
 => ----------------
2554  *|*****************************************************************
2555  *|
2556  *|      Procedure: FLUSHENCNT
2557  *|
2558  *|      Called by: FLUSHFILES()         (function  in U.PRG)
2559  *|
2560  *|*****************************************************************
2554                                          FUNCTION flushencnt
2555                                             SELECT (fencountr)
2556                                             USE
2557                                             *=openencnt()
2558  <------------------------------------------RETURN .T.
2559
2560                                             *-----------------------------------
 => ----------------
2561                                             *
2562                                             * New patient encounter
2563                                             *
2564                                             *-----------------------------------
 => ----------------
2572  *|*****************************************************************
2573  *|
2574  *|      Function: NEWENCNT()
2575  *|
2576  *|      Called by: M_ENCSAV            (procedure in DEMO.PRG)
2577  *|
2578  *|*****************************************************************
2572                                          FUNCTION newencnt
2573
2574                                             PARAMETERS mencountid, mpatientid, mdat
 => eenc, mtimeenc,;
2575                                                mprovider, mstation, mdept, marea,
 => mdiagnosis,;
2576                                                mtreatment, mstate
2577
2578                                             PRIVATE msel
2579                                             msel = SELECT()                && save c
 => urrent database selection
2580
2581                                             SELECT (fencountr)
2582                                             APPEND BLANK                   && create a
 =>  record for the new encounter
2583                                             REPLACE patient    WITH mpatientid
2584                                             REPLACE    DATE    WITH mdateenc
2585                                             REPLACE    TIME    WITH mtimeenc
2586                                             REPLACE station    WITH mstation
2587                                             REPLACE department WITH mdept
2588                                             REPLACE provider   WITH mprovider
2589                                             REPLACE area       WITH marea
2590                                             REPLACE diagnose   WITH mdiagnosis
2591                                             REPLACE treatment  WITH mtreatment
2592                                             REPLACE state      WITH mstate
2593                                             REPLACE encountrid WITH mencountid
2594                                             SELECT (msel)                  && restore
 =>  database selection
2595  <------------------------------------------RETURN .T.
2596
2597                                             *-----------------------------------
 => ----------------
2598                                             *
2599                                             * Lookup encounter
2600                                             *
2601                                             *-----------------------------------
```

```
 => -----------------
2607  *!*****************************************************************
2608  *!
2609  *!      Procedure: ENCID
2610  *!
2611  *!*****************************************************************
2607                                  FUNCTION encid
2608                                     PARAMETERS mpatientid, mdate, mtime
2609
2610                                     PRIVATE mid
2611                                     mid = 0
2612                                     PRIVATE msel
2613                                     msel = SELECT()              && save c
 => urrent database selection
2614
2615                                     SELECT (fencountr)
2616                                     SEEK mpatientid
2617                                    IF FOUND()
2618                                      DO WHILE .NOT. EOF() .AND. patient =
 =>  mpatientid
2619                                        IF mdate = DATE
2620                                           * found encounter by date opti
 => onally match time
2621                                          IF IIF(TYPE("mtime")="C",mtime
 =>  = TIME(.T.)
2622                                             * found (first or exact) en
 => counter record
2623                                             mid = encountrid
2624                                          EXIT
2625                                        ENDIF
2626                                      ENDIF
2627                                      SKIP
2628                                    ENDDO
2629                                  ENDIF
2630                                  SELECT (msel)                 && restore
 =>  database selection
2631  <-------------------------------RETURN  mid
2632
2633                                  *-----------------------------------
 => -----------------
2634                                  *
2635                                  * Put patient encounter
2636                                  *
2637                                  *-----------------------------------
 => -----------------
2645  *!*****************************************************************
2646  *!
2647  *!      Function: PUTENCNT()
2648  *!
2649  *!      Called by: M_ENCSAV          (procedure in DEMO.PRG)
2650  *!
2651  *!*****************************************************************
2645                                  FUNCTION putencnt
2646
2647                                     PARAMETERS mencountid, mpatientid, mdat
 => eenc, mtimeenc,;
2648                                         mprovider, mstation, mdept, marea,
 => mdiagnosis,;
2649                                         mtreatment, mstate
2650
2651                                     PRIVATE mresult
2652                                     mresult = .F.
2653                                     PRIVATE msel
2654                                     msel = SELECT()              && save c
 => urrent database selection
2655
```

```
2656                                     SELECT (fencountr)
2657                                     SET ORDER TO 3               && key is e
 => ncounter id
2658
2659                                     * look up encounter record
2660                                     SEEK mencountid
2661                                    IF FOUND()
2662                                       mresult = .T.
2663                                       REPLACE    DATE    WITH mdateenc
2664                                       REPLACE    TIME    WITH mtimeenc
2665                                       REPLACE station    WITH mstation
2666                                       REPLACE department WITH mdept
2667                                       REPLACE provider   WITH mprovider
2668                                       REPLACE area       WITH marea
2669                                       REPLACE diagnose   WITH mdiagnosis
2670                                       REPLACE treatment  WITH mtreatment
2671                                       REPLACE state      WITH mstate
2672                                    ENDIF
2673                                     SET ORDER TO 1               && restore
 => index order
2674                                     SELECT (msel)                && restore
 =>  database selection
2675  <-------------------------------RETURN mresult
2676
2677                                     *-----------------------------------
 => -----------------
2678                                     *
2679                                     * Get encounter info
2680                                     *
2681                                     *-----------------------------------
 => -----------------
2689  *!*****************************************************************
2690  *!
2691  *!      Function: ENCGETINFO()
2692  *!
2693  *!      Called by: LODENC            (procedure in DEMO.PRG)
2694  *!
2695  *!*****************************************************************
2689                                  FUNCTION encgetinfo
2690
2691                                     PARAMETERS mencountid, mpatientid, mdat
 => eenc, mtimeenc,;
2692                                         mprovider, mstation, mdept, marea,
 => mdiagnosis,;
2693                                         mtreatment, mstate, mreport
2694
2695                                     PRIVATE mresult
2696                                     mresult = .F.
2697                                     PRIVATE msel
2698                                     msel = SELECT()              && save c
 => urrent database selection
2699
2700                                     SELECT (fencountr)
2701                                     SET ORDER TO 3               && key is e
 => ncounter id
2702
2703                                     * look up encounter record
2704                                     SEEK mencountid
2705                                    IF FOUND()
2706                                       mresult = .T.
2707                                       mdateenc   = DATE
2708                                       mtimeenc   = TIME
2709                                       mprovider  = provider
2710                                       mstation   = station
2711                                       mdept      = department
2712                                       marea      = area
```

```
2713                                    mdiagnosis = diagnose
2714                                    mtreatment = treatment
2715                                    mstate     = state
2716                                    mreport    = REPORT
2717                                 ENDIF
2718                                 SET ORDER TO 1                && restore
 => index order
2719                                 SELECT (msel)                 && restore
 => database selection
2720                                 RETURN mresult
2721
2722                                 *-----------------------------------
 => -----------------
2723                                 *
2724                                 * count number of past encounters
2725                                 *
2726                                 *-----------------------------------
 => -----------------
2735  *!*****************************************************************
2736  *!
2737  *!       Function: ENCCOUNT()
2738  *!
2739  *!      Called by: GETENCNT          (procedure in DEMO.PRG)
2740  *!              : SFGETENC           (procedure in U.PRG)
2741  *!
2742  *!*****************************************************************
2735                              FUNCTION enccount
2736                                 PRIVATE N
2737                                 PRIVATE msel
2738                                 msel = SELECT()
 => && save current database selection
2739                                 SELECT (fencountr)
2740                                 N= RECCOUNT()
 => && number of names in file
2741                                 SELECT (msel)
 => && reselect original database
2742                                 RETURN N
2743
2744                                 *-----------------------------------
 => -----------------
2745                                 *
2746                                 * Lookup encounter list
2747                                 *
2748                                 *-----------------------------------
 => -----------------
2762  *!*****************************************************************
2763  *!
2764  *!       Function: ENCNAMES()
2765  *!
2766  *!      Called by: GETENCNT          (procedure in DEMO.PRG)
2767  *!
2768  *!          Calls: PATSSN()          (function  in U.PRG)
2769  *!
2770  *!          Uses: AREA.DBF
2771  *!
2772  *!        Indexes: AREA.IDX
2773  *!
2774  *!*****************************************************************
2762                              FUNCTION encnames
2763                                 PARAMETERS names, ids, patients, mpatie
 => nt, mdate
2764
2765                                 EXTERNAL ARRAY names, ids, patients
 =>                              && rwd07
2766                                 * list encounter file by:
2767                                 *    patient
2768                                 *    patient + date
2769                                 *    date
2770                                 *    all
2771
2772                                 PRIVATE i
2773                                 PRIVATE mexact
2774                                 PRIVATE mid
2775                                 PRIVATE msocsec
2776                                 PRIVATE ispatient, isdate
2777                                 PRIVATE msel
2778                                 PRIVATE msel2
2779
2780                                 msel = SELECT()               && save c
 => urrent database selection
2781                                 SELECT 0
2782                                 USE area INDEX area
2783                                 msel2 = SELECT()
2784                                 SELECT (fencountr)
2785                                 SET RELATION TO area INTO area
2786                                 mid = 0
2787                                 mexact = SET("EXACT")
2788                                 SET EXACT OFF
2789                                 ispatient = .F.
2790                                 isdate    = .F.
2791                                 IF PARAMETERS() > 3  .AND. TYPE("mpatie
 => nt") = "N"
2792                                    ispatient = mpatient > 0
2793                                 ENDIF
2794                                 IF PARAMETERS() > 4
2795                                    isdate = TYPE("mdate") = "D"
2796                                 ENDIF
2797                                 SET ORDER TO 2             && index by d
 => ate
2798                                 IF ispatient
2799                                    IF isdate
2800                                       SET FILTER TO DATE = mdate  .AND.
 =>  patient = mpatient
2801                                    ELSE
2802                                       SET FILTER TO patient = mpatient
2803                                    ENDIF
2804                                 ELSE
2805                                    IF isdate
2806                                       SET FILTER TO DATE = mdate
2807                                    ENDIF
2808                                 ENDIF
2809                                 * create list of names
2810                                 i = 0
2811                                 GOTO TOP
2812                                 DO WHILE .NOT. EOF()
2813                                    IF patient != mid
2814                                       mid = patient
2815                                       msocsec = patssn(mid)
2816                                    ENDIF
2817                                    i = i + 1
2818                                    ids[i] = encountrid
2819                                    patients[i] = mid
2820                                    names[i] = IIF( ispatient, "", msoc
 => sec + " " ) ;
2821                                          + IIF( isdate, "", DTOC(DATE) +
 => " " ) ;
2822                                          + TIME + " " + area->name
2823                                    SKIP
2824                                 ENDDO
2825
2826                                 IF mexact = "ON"
2827                                    SET EXACT ON
```

```
2828                                    └ENDIF
2829                                      SET FILTER TO                          &
 => & remove filter condition
2830                                      SET ORDER TO 1                         &
 => & restore index order
2831                                      SELECT (msel2)
2832                                      USE
2833                                      SELECT (msel)
 => && restore database selection
2834                                    <------------------------RETURN i
 => && list of names
2835
2836                                      *-----------------------------------
 => -----------------
2837                                      *
2838                                      * get disease descriptions
2839                                      *
2840                                      *-----------------------------------
 => -----------------
2854  *!*****************************************************************
2855  *!
2856  *!        Procedure: DISEASEGET
2857  *!
2858  *!        Called by: GETDISLIST()        (function  in U.PRG)
2859  *!                 : GETDISLOOKUPLIST    (procedure in U.PRG)
2860  *!                 : _EDITVDIAG          (procedure in U.PRG)
2861  *!
2862  *!             Uses: DISEASE.DBF
2863  *!
2864  *!           Indexes: DISEASE.IDX
2865  *!
2866  *!*****************************************************************
2854                                    FUNCTION diseaseget
2855                                      PARAMETERS mdisease
2856
2857                                      PRIVATE mhelp
2858                                      mhelp = ""
2859                                      PRIVATE msel
2860                                      msel = SELECT()                 && save c
 => urrent database selection
2861                                      *SELECT 0
2862                                      *USE treat INDEX treat            && tre
 => atment descriptions
2863                                      SELECT 0
2864                                      USE disease INDEX disease
2865                                      *SET RELATION TO name INTO treat
2866                                      SEEK UPPER(mdisease)
2867                                      ┌IF FOUND()
2868                                      │  * get disease plus treatment descrip
 => tion
2869                                      │  mhelp = descript + CHR(10) + CHR(13)
 =>  + treatment
2870                                      └ENDIF
2871                                      USE                                 && clo
 => se files
2872                                      *SELECT treat
2873                                      *USE
2874                                      SELECT (msel)                        && re
 => select original database
2875                                    <------------------------RETURN  mhelp
2876
2877                                      *-----------------------------------
 => -----------------
2878                                      *
2879                                      * size disease list
2880                                      *
```



```
2881                                      *-----------------------------------
 => -----------------
2893  *!*****************************************************************
2894  *!
2895  *!        Function: DISEASECNT()
2896  *!
2897  *!        Called by: INIDIAGNOS()        (function  in DEMO.PRG)
2898  *!                 : ASKDIAG             (procedure in DEMO.PRG)
2899  *!                 : M_LOOKUP            (procedure in DEMO.PRG)
2900  *!
2901  *!             Uses: DISEASE.DBF
2902  *!
2903  *!*****************************************************************
2893                                    FUNCTION diseasecnt
2894                                      PRIVATE N
2895                                      PRIVATE msel
2896                                      msel = SELECT()
 => && save current database selection
2897                                      SELECT 0
2898                                      USE disease
 => && list of elements
2899                                      N = RECCOUNT()
 => && number of names in file
2900                                      USE
 => && close file
2901                                      SELECT (msel)
 => && reselect original database
2902                                    <------------------------RETURN N
2903
2904                                      *-----------------------------------
 => -----------------
2905                                      *
2906                                      * get index of disease topics
2907                                      *
2908                                      *-----------------------------------
 => -----------------
2922  *!*****************************************************************
2923  *!
2924  *!        Function: DISNAMES()
2925  *!
2926  *!        Called by: ASKDIAG             (procedure in DEMO.PRG)
2927  *!                 : M_LOOKUP            (procedure in DEMO.PRG)
2928  *!
2929  *!             Uses: DISEASE.DBF
2930  *!
2931  *!           Indexes: DISID.IDX
2932  *!                  : DISEASE.IDX
2933  *!
2934  *!*****************************************************************
2922                                    PROCEDURE disnames
2923                                      PARAMETERS mdiseases, marea
2924
2925                                      EXTERNAL ARRAY mdiseases
 =>                    && rwd07
2926                                      PRIVATE i
2927                                      PRIVATE msel
2928                                      msel = SELECT()                 && save c
 => urrent database selection
2929                                      SELECT 0
2930                                      USE disease
2931                                      ┌IF PARAMETERS() = 2
2932                                      │  SET INDEX TO disid             && disea
 => se name file
2933                                      │  SET FILTER TO area = marea        &&
 => area determines differential list
2934                                      │  GOTO TOP
```

```
2935                                  ELSE
2936                                     SET INDEX TO disease
2937                                  ENDIF
2938
2939                                  * create list of names
2940                                  i = 0
2941                                  DO WHILE .NOT. EOF()
2942                                     i = i + 1
2943                                     mdiseases[i] = TRIM( name )
2944                                     SKIP
2945                                  ENDDO
2946                                  DECLARE mdiseases[i]
2947                                  USE
2948                                  SELECT (msel)
 => && restore database selection
2949                                  RETURN i
 => && list of names
2950
2951                                  *-----------------------------------
 => ----------------
2952                                  *
2953                                  * update disease descriptions
2954                                  *
2955                                  *-----------------------------------
 => ----------------
2965  *!*****************************************************************
2966  *!
2967  *!      Procedure: DISEASEPUT
2968  *!
2969  *!           Uses: DISEASE.DBF
2970  *!
2971  *!        Indexes: DISEASE.IDX
2972  *!
2973  *!*****************************************************************
2965                                  FUNCTION diseaseput
2966                                     PARAMETERS mdisease, mhelp
2967
2968                                     PRIVATE msel
2969                                     msel = SELECT()                && sav
 => e current database selection
2970                                     SELECT 0
2971                                     USE disease INDEX disease
2972
2973                                     SEEK UPPER(mdisease)
2974                                     IF FOUND()
2975                                        REPLACE descript WITH mhelp
2976                                     ENDIF
2977                                     USE
2978                                     SELECT (msel)                  && re
 => select original database
2979                                     RETURN  ""
2980
2981                                     *-----------------------------------
 => ----------------
2982                                     *
2983                                     * get disease name
2984                                     *
2985                                     *-----------------------------------
 => ----------------
3001  *!*****************************************************************
3002  *!
3003  *!      Function: DISEASENAM()
3004  *!
3005  *!     Called by: BATCHTEST()        (function  in DEMO.PRG)
3006  *!             : M_RESUME            (procedure in DEMO.PRG)
3007  *!             : M_DIAG              (procedure in DEMO.PRG)
3008  *!             : STRDIAG             (procedure in DEMO.PRG)
3009  *!             : _EDITDIAG           (procedure in U.PRG)
3010  *!
3011  *!          Uses: DISEASE.DBF
3012  *!
3013  *!       Indexes: DISID.IDX
3014  *!
3015  *!*****************************************************************
3001                                     FUNCTION diseasenam
3002
3003                                        PARAMETERS mid, mname
3004
3005                                        EXTERNAL ARRAY mid, mname
 =>                          && rwd07
3006
3007                                        PRIVATE i, mresult
3008                                        mresult = ""
3009                                        PRIVATE msel
3010                                        msel = SELECT()          && save current
 => database selection
3011                                        SELECT 0
3012                                        USE disease INDEX disid
3013
3014                                        IF PARAMETERS() = 2
3015                                           * translate list
3016                                           FOR i = 1 TO ALEN(mid)
3017                                              SEEK mid[i]
3018                                              mname[i]  =  IIF( FOUND(),  TRIM(
 => name ),  "" )
3019                                           NEXT
3020                                           ELSE
3021                                              SEEK mid
3022                                              mresult  =  IIF( FOUND(), TRIM( n
 => ame ),  "")
**** Expected NEXT--Found ENDIF
3023                                           ENDIF
3024
3025                                        USE
3026                                        SELECT (msel)                  &&
 => reselect original database
3027                                        RETURN  mresult
3028
3029                                        *-----------------------------------
 => ----------------
3030                                        *
3031                                        * get disease id
3032                                        *
3033                                        *-----------------------------------
 => ----------------
3045  *!*****************************************************************
3046  *!
3047  *!      Function: DISEASEID()
3048  *!
3049  *!     Called by: M_DIAG             (procedure in DEMO.PRG)
3050  *!
3051  *!          Uses: DISEASE.DBF
3052  *!
3053  *!       Indexes: DISEASE.IDX
3054  *!
3055  *!*****************************************************************
3045                                        FUNCTION diseaseid
3046
3047                                        PARAMETERS dname, did
3048
3049                                        EXTERNAL ARRAY did, dname
 =>                          && rwd07
```

```
3050
3051                                    PRIVATE i, mresult
3052                                    mresult = ""
3053                                    PRIVATE msel
3054                                    msel = SELECT()          && save curr
 => ent database selection
3055                                    SELECT 0
3056                                    USE disease INDEX disease
3057
3058                                   ┌IF PARAMETERS() = 2
3059                                   │ ┌FOR i = 1 TO ALEN(dname)
3060                                   │ │   SEEK dname[i]
3061                                   │ │   did[i] = IIF(FOUND(), id, 0)
3062                                   │ └NEXT
3063                                   │ELSE
3064                                   │   SEEK dname
3065                                   │   mresult = IIF(FOUND(), id, 0)
**** Expected NEXT--Found ENDIF
3066                                   └ENDIF
3067
3068                                     USE
3069                                     SELECT (msel)
 => && reselect original database
3070  <------------------------------------RETURN  mresult
3071
3072                                     *-----------------------------
 => -----------------------
3073                                     *
3074                                     * open the major data bases neede
 => d throughout program
3075                                     *
3076                                     *-----------------------------
 => -----------------------
3095  *!*****************************************************************
3096  *!
3097  *!       Function: OPENFILES()
3098  *!
3099  *!      Called by: DEMO.PRG
3100  *!              : M_PAT              (procedure in DEMO.PRG)
3101  *!              : M_SF600            (procedure in DEMO.PRG)
3102  *!              : M_NEWENC           (procedure in DEMO.PRG)
3103  *!              : M_TRAINING         (procedure in DEMO.PRG)
3104  *!
3105  *!          Calls: POPUPSHOW()       (function  in U.PRG)
3106  *!              : OPENPAT            (procedure in U.PRG)
3107  *!              : OPENENCNT          (procedure in U.PRG)
3108  *!              : OPENDICT           (procedure in U.PRG)
3109  *!              : OPENRULE           (procedure in U.PRG)
3110  *!              : POPUPHIDE()        (function  in U.PRG)
3111  *!
3112  *!*****************************************************************
3095                                    FUNCTION openfiles
3096                                      PARAMETERS  testing
 =>              && rwd03
3097
3098                                     * set names of transaction (encou
 => nter files
3099                                    ┌IF EMPTY(testing)
3100                                    │   m.wpop=popupshow("Files loadin
 => g...")
3101                                    └ENDIF
3102                                     PUBLIC f_symptom
 =>              && rwd03
3103                                     PUBLIC f_diagnose
 =>              && rwd03
3104                                     PUBLIC f_patient
 =>              && rwd03
3105                                     PUBLIC f_patname
 =>              && rwd03
3106                                     PUBLIC f_patid
 =>              && rwd03
3107                                     PUBLIC f_encountr
 =>              && rwd03
3108                                     PUBLIC f_encountd
 =>              && rwd03
3109                                     PUBLIC f_encounti
 =>              && rwd03
3110                                     f_symptom  = "symptoms"
 =>              && rwd03
3111                                     f_diagnose = "diagnose"
 =>              && rwd03
3112                                     f_patient  = "patient"
 =>              && rwd03
3113                                     f_patname  = "patname"
 =>              && rwd03
3114                                     f_patid    = "patid"
 =>              && rwd03
3115                                     f_encountr = "encountr"
 =>              && rwd03
3116                                     f_encountd = "encountd"
 =>              && rwd03
3117                                     f_encounti = "encounti"
 =>              && rwd03
3118
3119                                    ┌IF !EMPTY(testing)
 =>              && rwd03
3120                                    │   * use test files
 =>              && rwd03
3121                                    │ ┌IF RIGHT(testing, 1) != "\"
 =>              && rwd03
3122                                    │ │   testing = testing + "\"
 =>              && rwd03
3123                                    │ └ENDIF
 =>              && rwd03
3124                                    │   f_symptom  = testing + f_sympt
 => om           && rwd03
3125                                    │   f_diagnose = testing + f_diagn
 => ose          && rwd03
3126                                    │   f_patient  = testing + f_patie
 => nt           && rwd03
3127                                    │   f_patname  = testing + f_patna
 => me           && rwd03
3128                                    │   f_patid    = testing + f_patid
 =>              && rwd03
3129                                    │   f_encountr = testing + f_encou
 => ntr          && rwd03
3130                                    │   f_encountd = testing + f_encou
 => ntd          && rwd03
3131                                    │   f_encounti = testing + f_encou
 => nti          && rwd03
3132                                    └ENDIF
 =>              && rwd03
3133
3134                                     =openpat()
3135                                     =openencnt()
3136                                     =opendict()
3137                                     =openrule()
3138                                    ┌IF EMPTY(testing)
3139                                    │   =popuphide(m.wpop)
3140                                    └ENDIF
3141  <------------------------------------RETURN .T.
3142
```

```
3143                                    *-----------------------------
 => ---------------------
3144                                    *
3145                                    * flush encounter files
3146                                    *
3147                                    *-----------------------------
 => ---------------------
3163  *|*****************************************************************
3164  *|
3165  *|       Function: FLUSHFILES()
3166  *|
3167  *|      Called by: M_PAT              (procedure in DEMO.PRG)
3168  *|               : M_SF600           (procedure in DEMO.PRG)
3169  *|               : M_NEWENC          (procedure in DEMO.PRG)
3170  *|               : M_TRAINING        (procedure in DEMO.PRG)
3171  *|
3172  *|          Calls: FLUSHPAT           (procedure in U.PRG)
3173  *|               : FLUSHENCNT        (procedure in U.PRG)
3174  *|               : CLOSEDICT         (procedure in U.PRG)
3175  *|               : CLOSERULE         (procedure in U.PRG)
3176  *|
3177  *|*****************************************************************
3163                                    FUNCTION flushfiles
3164                                       =flushpat()
3165                                       =flushencnt()
3166                                       =closedict()
3167                                       =closerule()
3168  <-------------------------------RETURN .T.
3169
3170                                    *-----------------------------
 => ---------------------
3171                                    *
3172                                    * close encounter files
3173                                    *
3174                                    *-----------------------------
 => ---------------------
3184  *|*****************************************************************
3185  *|
3186  *|      Procedure: CLOSEFILES
3187  *|
3188  *|      Called by: BACKUP.SPR
3189  *|               : RESTORE.SPR
3190  *|               : REINDEX          (procedure in U.PRG)
3191  *|
3192  *|*****************************************************************
3184                                    FUNCTION closefiles
3185                                       CLOSE DATABASES
3186  <-------------------------------RETURN .T.
3187
3188                                    *-----------------------------
 => ---------------------
3189                                    *
3190                                    * data base functions for patient
 => record
3191                                    *
3192                                    *-----------------------------
 => ---------------------
3193
3194                                    *-----------------------------
 => ---------------------
3195                                    *
3196                                    * open patient files and SAMS dbf
 => if it is exist
3197                                    * P: HLL
3198                                    *-----------------------------
 => ---------------------
```

```
3216  *|*****************************************************************
3217  *|
3218  *|      Procedure: OPENPAT
3219  *|
3220  *|      Called by: OPENFILES()        (function  in U.PRG)
3221  *|
3222  *|          Calls: FILEFIND          (procedure in U.PRG)
3223  *|               : ERRHAND           (procedure in U.PRG)
3224  *|               : OPENSAMS          (procedure in U.PRG)
3225  *|
3226  *|           Uses: &F_PATIENT
3227  *|
3228  *|        Indexes: &F_PATIENT
3229  *|               : &F_PATNAME
3230  *|               : &F_PATID
3231  *|
3232  *|*****************************************************************
3216                                    FUNCTION openpat
3217
3218                                       PUBLIC fpatient, msamsdate
3219                                       PRIVATE mffind
3220                                       mffind = filefind(f_patient + ".d
 => bf")
3221                                       IF !mffind
3222                                          errmsg = "Patient.dbf does not
 =>  found. This proccess will cancel"
3223                                          DO errhand WITH errmsg, .T.
3224                                       ENDIF
3225                                       IF EMPTY(fpatient) OR !USED(fpati
 => ent)
3226                                          SELECT 0
3227                                          USE &f_patient AGAIN INDEX &f_
 => patient, &f_patname, &f_patid
3228                                          fpatient = SELECT()
3229                                       ELSE
3230                                          SELECT (fpatient)
3231                                          fpatient = SELECT()
3232                                       ENDIF
3233                                       IF mtrain
3234                                          msamsdate = .F.
3235                                       ELSE
3236                                          msamsdate = opensams()
3237                                       ENDIF
3238  <-------------------------------RETURN
3239
3240                                       *-----------------------------
 => ---------------------
3241                                       *
3242                                       * flush patient file
3243                                       *
3244                                       *-----------------------------
 => ---------------------
3252  *|*****************************************************************
3253  *|
3254  *|      Procedure: FLUSHPAT
3255  *|
3256  *|      Called by: FLUSHFILES()       (function  in U.PRG)
3257  *|
3258  *|*****************************************************************
3252                                    FUNCTION flushpat
3253                                       SELECT (fpatient)
3254                                       USE
3255                                       fpatient = ""
3256  <-------------------------------RETURN .T.
3257
3258                                       *-----------------------------
```

```
 => ----------------------
3259                                                    *
3260                                                    * Lookup patient
3261                                                    *
3262                                                    *-----------------------------
 => ----------------------
3275  *|*****************************************************************
3276  *|
3277  *|       Function: PATSSN()
3278  *|
3279  *|      Called by: ENCOUNT            (procedure in DEMO.PRG)
3280  *|               : GETENCNT          (procedure in DEMO.PRG)
3281  *|               : ENCNAMES()        (function  in U.PRG)
3282  *|               : SFENCOUNT         (procedure in U.PRG)
3283  *|               : SFENCLIST         (procedure in U.PRG)
3284  *|               : SFENCNAMES        (procedure in U.PRG)
3285  *|
3286  *|*****************************************************************
3275                                                  FUNCTION patssn
3276                                                     PARAMETERS mid
3277
3278                                                     PRIVATE msocsec
3279                                                     PRIVATE msel
3280                                                     msel = SELECT()            && sa
 => ve current database selection
3281                                                     SELECT (fpatient)
3282                                                     * look up id in patient record
3283                                                     SET ORDER TO 3
3284                                                     SEEK mid
3285                                                     msocsec = IIF(FOUND(), socsec,  "
 => ")
3286                                                     SET ORDER TO 1             && re
 => store index order
3287                                                     SELECT (msel)              && r
 => estore database selection
3288  <------------------------------------------RETURN msocsec
3289
3290                                                     *-----------------------------
 => ----------------------
3291                                                     *
3292                                                     * Get patient info
3293                                                     *
3294                                                     *-----------------------------
 => ----------------------
3302  *|*****************************************************************
3303  *|
3304  *|       Function: PATGETINFO()
3305  *|
3306  *|      Called by: M_PATGET           (procedure in DEMO.PRG)
3307  *|
3308  *|*****************************************************************
3302                                                  FUNCTION patgetinfo
3303
3304                                                     PARAMETERS mpatientid,msocsec,mla
 => stname,mfirstname,mdatebirth,msex,mraterank
3305
3306                                                     PRIVATE mresult
3307                                                     mresult = .F.
3308                                                     PRIVATE msel
3309                                                     msel = SELECT()            &&
 => save current database selection
3310
3311                                                     SELECT (fpatient)
3312
3313                                                     * look up socsec in patient recor
 => d
```

```
3314
3315                                                     SEEK msocsec
3316                                                    IF FOUND()
3317                                                       mresult = .T.
3318                                                       mpatientid = id
3319                                                       mlastname  = lastname    && ge
 => t patient details
3320                                                       mfirstname = firstname
3321                                                       mdatebirth = datebirth
3322                                                       msex       = sex
3323                                                       mraterank  = raterank
3324                                                    ENDIF
3325                                                     SELECT (msel)              && r
 => estore database selection
3326  <------------------------------------------RETURN mresult
3327                                                        *
3328                                                     *-----------------------------
 => ----------------------
3329                                                     *
3330                                                     * Put patient info
3331                                                     *
3332                                                     *-----------------------------
 => ----------------------
3340  *|*****************************************************************
3341  *|
3342  *|       Function: PATPUTINFO()
3343  *|
3344  *|      Called by: M_PATPUT           (procedure in DEMO.PRG)
3345  *|
3346  *|*****************************************************************
3340                                                  FUNCTION patputinfo
3341
3342                                                     PARAMETERS mpatientid,msocsec,mla
 => stname,mfirstname,mdatebirth,msex,mraterank
3343
3344                                                     PRIVATE mresult
3345                                                     mresult = .F.
3346                                                     PRIVATE msel
3347                                                     msel = SELECT()            &&
 => save current database selection
3348                                                     SELECT (fpatient)
3349
3350                                                     * look up socsec in patient recor
 => d
3351                                                     SEEK msocsec
3352                                                    IF FOUND()
3353                                                       mresult = .T.
3354                                                       REPLACE lastname  WITH mlastna
 => me,;
3355                                                           firstname WITH mfirstname
3356                                                       REPLACE datebirth WITH mdatebi
 => rth,;
3357                                                           sex       WITH msex,;
3358                                                           raterank  WITH mraterank
3359                                                    ENDIF
3360                                                     SELECT (msel)              && r
 => estore database selection
3361  <------------------------------------------RETURN mresult
3362
3363                                                     *-----------------------------
 => ----------------------
3364                                                     *
3365                                                     * New patient record
3366                                                     *
3367                                                     *-----------------------------
 => ----------------------
```

```
3375  *|*****************************************************************
3376  *|
3377  *|        Function: NEWPATIENT()
3378  *|
3379  *|       Called by: M_PATADD            (procedure in DEMO.PRG)
3380  *|
3381  *|*****************************************************************
3375                                          FUNCTION newpatient
3376
3377                                            PARAMETERS mpatientid,msocsec,mla
 => stname,mfirstname,mdatebirth,msex,mraterank
3378
3379                                            PRIVATE msel
3380                                            msel = SELECT()                 &&
 => save current database selection
3381                                            SELECT (fpatient)
3382                                            APPEND BLANK                  && cr
 => eate a record for the new patient
3383                                            REPLACE id        WITH mpatientid
3384                                            REPLACE lastname  WITH mlastname,
 => ;
3385                                                firstname WITH mfirstname,;
3386                                                socsec    WITH msocsec
3387                                            REPLACE datebirth WITH mdatebirth
 => ,;
3388                                                sex       WITH msex,;
3389                                                raterank  WITH mraterank
3390                                            SELECT (msel)                 && r
 => estore database selection
3391  <-------------------------------------------RETURN .T.
3392
3393                                            *-----------------------------
 => ---------------------
3394                                            *
3395                                            * delete patient record
3396                                            *
3397                                            *-----------------------------
 => ---------------------
3403  *|*****************************************************************
3404  *|
3405  *|       Procedure: DELPATIENT
3406  *|
3407  *|*****************************************************************
3403                                          FUNCTION delpatient
3404
3405                                            PARAMETERS mpatientid
3406
3407                                            PRIVATE msel
3408                                            msel = SELECT()                 &&
 => save current database selection
3409
3410                                            SELECT (fpatient)
3411                                            SET ORDER TO 3                && ke
 => y is patient id
3412                                            SEEK mpatientid
3413                                            mresult = .NOT. FOUND()
3414                                            IF FOUND()
3415                                               DELETE
3416                                               PACK
3417                                            ENDIF
3418                                            SET ORDER TO 1                && re
 => store index order
3419                                            SELECT (msel)                 && r
 => estore database selection
3420  <-------------------------------------------RETURN mresult
3421
```

```
3422                                            *-----------------------------
 => ---------------------
3423                                            *
3424                                            * Check patient id is valid
3425                                            *
3426                                            *-----------------------------
 => ---------------------
3432  *|*****************************************************************
3433  *|
3434  *|       Procedure: ISVALIDPAT
3435  *|
3436  *|*****************************************************************
3432                                          FUNCTION isvalidpat
3433
3434                                            PARAMETERS mpatientid
3435
3436                                            PRIVATE mresult
3437                                            PRIVATE msel
3438                                            msel = SELECT()                 &&
 => save current database selection
3439
3440                                            SELECT (fpatient)
3441                                            SET ORDER TO 3                && ke
 => y is patient id
3442                                            SEEK mpatientid
3443                                            mresult = .NOT. FOUND()
3444                                            SET ORDER TO 1                && re
 => store index order
3445                                            SELECT (msel)                 && r
 => estore database selection
3446  <-------------------------------------------RETURN mresult
3447
3448                                            *-----------------------------
 => ---------------------
3449                                            *
3450                                            * Check patient SSN
3451                                            *
3452                                            *-----------------------------
 => ---------------------
3461  *|*****************************************************************
3462  *|
3463  *|       Function: ISVALIDSSN()
3464  *|
3465  *|       Called by: M_PATPUT            (procedure in DEMO.PRG)
3466  *|                : M_PATADD            (procedure in DEMO.PRG)
3467  *|
3468  *|*****************************************************************
3461                                          FUNCTION isvalidssn
3462
3463                                            PARAMETERS msocsec
3464
3465                                            PRIVATE mresult
3466                                            mresult = .F.
3467                                            PRIVATE msel
3468                                            msel = SELECT()                 &&
 => save current database selection
3469
3470                                            SELECT (fpatient)
3471
3472                                            * look up socsec in patient recor
 => d
3473
3474                                            SEEK msocsec
3475                                            IF FOUND()
3476                                               mresult = .T.
3477                                            ENDIF
```

```
3478                                    SELECT (msel)                   && r
 => estore database selection
3479                                    RETURN mresult
3480
3481                                    *-----------------------------
 => ----------------------
3482                                    *
3483                                    * size patient list
3484                                    *
3485                                    *-----------------------------
 => ----------------------
3493  *!*****************************************************************
3494  *!
3495  *!        Function: PATCOUNT()
3496  *!
3497  *!       Called by: GETPATIENT          (procedure in DEMO.PRG)
3498  *!
3499  *!*****************************************************************
3493                                    FUNCTION patcount
3494                                       PRIVATE N
3495                                       PRIVATE msel
3496                                       msel = SELECT()
 =>       && save current database selection
3497                                       SELECT (fpatient)
3498                                       N= RECCOUNT()
 =>       && number of names in file
3499                                       SELECT (msel)
 =>        && reselect original database
3500                                    RETURN N
3501
3502                                    *-----------------------------
 => ----------------------
3503                                    *
3504                                    * Lookup patient list
3505                                    *
3506                                    *-----------------------------
 => ----------------------
3516  *!*****************************************************************
3517  *!
3518  *!        Function: PATNAMES()
3519  *!
3520  *!       Called by: GETPATIENT          (procedure in DEMO.PRG)
3521  *!
3522  *!           Calls: STRNAME             (procedure in U.PRG)
3523  *!*****************************************************************
3516                                    FUNCTION patnames
3517                                       PARAMETERS names, DELIM, mkey
3518
3519                                       PRIVATE i, N
3520                                       PRIVATE mexact
3521                                       PRIVATE msel
3522
3523                                       msel = SELECT()                 &&
 => save current database selection
3524                                       SELECT (fpatient)
3525                                       mexact = SET("EXACT")
3526                                       SET EXACT OFF
3527
3528                                       * list patient file by: lastname
 =>   | socsec  |  all entries
3529                                       IF TYPE("mkey") = "C"
3530                                          mkey = ALLTRIM(mkey)
3531                                          IF TYPE(mkey) = "N"
3532                                             * key is social security nu
 => mber
```

```
3533                                             *"SET ORDER TO 1
 =>       && index by social security
3534                                             SET FILTER TO socsec = mkey
3535                                          ELSE
3536                                             * key is patient lastname
3537                                             SET ORDER TO 2
 =>        && index by lastname
3538                                             SET FILTER TO lastname = mk
 => ey
3539                                          ENDIF
3540                                       ELSE
3541                                          SET ORDER TO 2
 =>       && general lookup by lastname
3542                                       ENDIF
3543                                       GOTO TOP
 =>       && activate filter
3544                                       * create list of names
3545                                       i = 1
3546                                       N = ALEN(names)
3547                                       DO WHILE .NOT. EOF() .AND. i <= N
3548                                          names[i] = strname(socsec, las
 => tname, firstname, DELIM)
3549                                          i = i + 1
3550                                          SKIP
3551                                       ENDDO
3552                                       IF mexact = "ON"
 =>       && restore SET EXACT ON/OFF
3553                                          SET EXACT ON
3554                                       ENDIF
3555                                       SET FILTER TO
 =>       && remove filter condition
3556                                       SET ORDER TO 1
 =>       && restore index order
3557                                       SELECT (msel)
 =>        && restore database selection
3558                                    RETURN N
 =>       && number of names added
3559
3560                                    *-----------------------------
 => ----------------------
3561                                    *
3562                                    * make patient social security an
 => d name string
3563                                    *
3564                                    *-----------------------------
 => ----------------------
3572  *!*****************************************************************
3573  *!
3574  *!        Procedure: STRNAME
3575  *!
3576  *!        Called by: PATNAMES()         (function  in U.PRG)
3577  *!
3578  *!*****************************************************************
3572                                    FUNCTION strname
3573                                       PARAMETERS msocsec, mlastname, mf
 => irstname, DELIM
3574                                       PRIVATE s
3575                                       s = IIF(!EMPTY(msocsec),msocsec,"
 => *********)
3576                                       s = s  + DELIM + IIF(!EMPTY(mlast
 => name),TRIM(mlastname),"******")
3577                                       s = s  + IIF(!EMPTY(mfirstname),
 => ", " + TRIM(mfirstname),"")
3578                                    RETURN s
3579
3580                                    *-----------------------------
```

```
=> ----------------------
3581                                                    *
3582                                                    * open rule file
3583                                                    *
3584                                                    *-----------------------------
=> ----------------------
3601  *!*****************************************************************
3602  *!
3603  *!      Procedure: OPENRULE
3604  *!
3605  *!      Called by: OPENFILES()        (function  in U.PRG)
3606  *!
3607  *!           Uses: PREMISE.DBF
3608  *!               : ACTION.DBF
3609  *!               : RULE.DBF
3610  *!
3611  *!        Indexes: PREMISE.IDX
3612  *!               : ACTION.IDX
3613  *!               : SALIENCE.IDX
3614  *!               : RULE.IDX
3615  *!
3616  *!*****************************************************************
3601                                                FUNCTION openrule
3602
3603                                                    SELECT 0
3604                                                    USE premise INDEX premise
=>        && list of rule premises
3605                                                    SET RELATION TO id INTO dict
=>        && dictionary object associated
3606                                                    SELECT 0
3607                                                    USE action INDEX action
=>        && list of rule actions
3608                                                    SELECT 0
3609                                                    USE rule INDEX salience,rule
=>        && list of rules
3610                                                    SET FILTER TO area = marea
=>        && select rule segment
3611                                                    * SET RELATION TO premise INTO pr
=> emise, action INTO action    && associated clauses && rwd07
3612  <-------------------------------------------RETURN .T.
3613
3614                                                    *-----------------------------
=> ----------------------
3615                                                    *
3616                                                    * close rule file
3617                                                    *
3618                                                    *-----------------------------
=> ----------------------
3626  *!*****************************************************************
3627  *!
3628  *!      Procedure: CLOSERULE
3629  *!
3630  *!      Called by: FLUSHFILES()       (function  in U.PRG)
3631  *!
3632  *!*****************************************************************
3626                                                FUNCTION closerule
3627
3628                                                    PRIVATE msel
3629                                                    msel = SELECT()               &&
=> save current database selection
3630                                                    SELECT rule
3631                                                    USE
3632                                                    SELECT premise
3633                                                    USE
3634                                                    SELECT action
3635                                                    USE
3636                                                    SELECT (msel)
=>        && reselect original database
3637  <-------------------------------------------RETURN .T.
3638
3639                                                    *-----------------------------
=> ----------------------
3640                                                    *
3641                                                    * access rulebase
3642                                                    *
3643                                                    *-----------------------------
=> ----------------------
3649  *!*****************************************************************
3650  *!
3651  *!      Procedure: GETRULE
3652  *!*****************************************************************
3649                                                FUNCTION getrule
3650
3651                                                    PARAMETERS ruleid
3652
3653                                                    PRIVATE msel
3654                                                    msel = SELECT()               &&
=> save current database selection
3655                                                    PRIVATE mruletext
3656                                                    mruletext = ""
3657                                                    SEEK ruleid
3658                                                    IF FOUND()
3659                                                        mruletext = TEXT + NOTE
3660                                                    ENDIF
3661                                                    SELECT (msel)                 && r
=> eselect original database
3662  <-------------------------------------------RETURN mruletext
3663
3664                                                    *-----------------------------
=> ----------------------
3665                                                    *
3666                                                    * data base functions for patient
=>  symptoms
3667                                                    *
3668                                                    *-----------------------------
=> ----------------------
3669
3670                                                    *-----------------------------
=> ----------------------
3671                                                    *
3672                                                    * update data base with symptoms
=> from working memory
3673                                                    *
3674                                                    *-----------------------------
=> ----------------------
3687  *!*****************************************************************
3688  *!
3689  *!      Function: PUTSYMPTOM()
3690  *!
3691  *!      Called by: M_ENCSAV           (procedure in DEMO.PRG)
3692  *!
3693  *!          Calls: OPENSYMP           (procedure in U.PRG)
3694  *!               : SPLIT()            (function  in U.PRG)
3695  *!               : DICTQ()            (function  in U.PRG)
3696  *!               : DICTORD            (procedure in U.PRG)
3697  *!
3698  *!*****************************************************************
3687                                                FUNCTION putsymptom
3688
3689                                                    PARAMETERS mencountid, mcond, mva
=> (
```

```
3690
3691                                    EXTERNAL ARRAY mcond, mval
 => && rwd07
3692                                    PRIVATE i
3693                                    PRIVATE mitem
3694                                    PRIVATE mtype
3695                                    PRIVATE mid
3696                                    PRIVATE mq
3697                                    PRIVATE nrecs                    &&
 => have existing records
3698                                    PRIVATE trypool
3699                                    trypool = .T.          && try to g
 => et record from free pool
3700                                    PRIVATE msel
3701                                    msel = SELECT()                  &&
 => save current database selection
3702
3703                                    =opensymp()
3704                                    * find existing transactions if a
 => ny
3705                                    SEEK mencountid
3706                                    nrecs =  FOUND()
3707
3708                                    FOR i = 1 TO ALEN(mcond)
3709                                      IF !IF( !EMPTY( mcond[i] ),  !
 => EMPTY( mval[i] ), .F.)
3710                                        * condition has been define
 => d
3711                                        mq = mval[i]
3712                                        DO WHILE !EMPTY(mq)
3713                                          mitem = split(@mq)
 =>    && split off value of item
3714                                          IF !nrecs
3715                                            * no more existing re
 => cords, try the free pool
3716                                            IF trypool
3717                                              SEEK 0
 =>    && use surplus records
3718                                              trypool = FOUND()
3719                                            ENDIF
3720                                            IF !trypool
3721                                              * free pool 's exh
 => austed
3722                                              APPEND BLANK
 =>    &&  add new record to file
3723                                            ENDIF
3724                                            REPLACE id  WITH menc
 => ountid
3725                                          ENDIF
3726                                          * edit the new or existi
 => ng item record
3727                                          mid = mcond[i]
3728                                          REPLACE symptom WITH VAL
 => ( mid )
3729                                          mtype = dictq(mid)
3730                                          IF mtype = "E"  .OR.  mt
 => ype = "N"
3731                                            * translate string to
 =>  ordinal
3732                                            mitem = dictord(mid,
 => mitem)
3733                                          ENDIF
3734                                          REPLACE VALUE   WITH mit
 => em
3735                                          IF nrecs
3736                                            * still updating exis
```

```
 => ting records
3737                                            * skip to next availa
 => ble record
3738                                            SKIP
3739                                            nrecs  =  id = mencou
 => ntid  .AND. !EOF()
3740                                          ENDIF
3741                                        ENDDO
3742                                      ENDIF
3743                                      NEXT
3744
3745                                      * put any surplus records in p
 => ool with index = 0
3746                                      * (this is more efficient than
 =>  deleting/packing)
3747                                      IF nrecs
3748                                        * still records remaining a
 => fter above update
3749                                        * put records in pool by ch
 => anging their IDs
3750                                        REPLACE id WITH 0 WHILE id
 => = mencountid
3751                                      ENDIF
3752                                      USE
3753                                      SELECT (msel)                  &
 => & reselect original database
3754                                      RETURN .T.
3755
3756                                      *-------------------------
 => -------------------------
3757                                      *
3758                                      * split string into tokens
3759                                      *
3760                                      *-------------------------
 => -------------------------
3769  *!*****************************************************************
3770  *!
3771  *!        Function: SPLIT()
3772  *!
3773  *!       Called by: STRNATURAL          (procedure in DEMO.PRG)
3774  *!                : PUTSYMPTOM()        (function  in U.PRG)
3775  *!
3776  *!*****************************************************************
3769                                      FUNCTION split
3770                                        PARAMETERS mq
3771                                        PRIVATE   mitem
3772                                        PRIVATE   N
3773                                        N = AT(mconcat, mq)            &
 => & find end of item
3774                                      IF N > 0
3775                                        mitem   = LEFT(mq,N-1)     &
 => & get the item
3776                                        mq      = SUBSTR(mq,N+LEN(m
 => concat))
3777                                      ELSE
3778                                        mitem = mq
3779                                        mq    = ""
3780                                      ENDIF
3781                                      RETURN mitem
3782
3783                                      *-------------------------
 => -------------------------
3784                                      *
3785                                      * load symptoms from encounter
 =>  file into memory
3786                                      *
```

```
3787                                                    *---------------------------
 => ------------------------
3800  *|*****************************************************************
3801  *|
3802  *|       Function: GETSYMPTOM()
3803  *|
3804  *|      Called by: LODENC              (procedure in DEMO.PRG)
3805  *|
3806  *|          Calls: OPENSYMP            (procedure in U.PRG)
3807  *|              : DICTQ()             (function  in U.PRG)
3808  *|              : DICTENUM            (procedure in U.PRG)
3809  *|              : ADDSYMPTOM          (procedure in DEMO.PRG)
3810  *|
3811  *|*****************************************************************
3800                                          FUNCTION getsymptom
3801
3802                                            PARAMETERS mencountid
3803
3804                                            PRIVATE msel
3805                                            PRIVATE mid, mval, mtype
3806                                            msel = SELECT()
 => && save current database selection
3807                                            =opensymp()
3808                                            SEEK mencountid
3809
3810                                            SCAN FOR id = mencountid
3811                                                mval = ALLTRIM(VALUE)
3812                                                mid  = TRIM( STR( symptom )
 =>  )
3813                                                mtype = dictq(mid)
3814                                                IF mtype = "E"  .OR.  mtype
 =>  = "M"
3815                                                    * translate from ordinal
3816                                                    mval = dictenum(mid, mva
 => l)
3817                                                ENDIF
3818                                                =addsymptom( mid,  mval )
3819                                            ENDSCAN
3820
3821                                            USE
3822                                            SELECT (msel)                &
 => & reselect original database
3823  <-------------------------------------   RETURN .T.
3824
3825
3826                                            *---------------------------
 => ------------------------
3827                                            *
3828                                            * open encounter symptoms file
3829                                            *
3830                                            *---------------------------
 => ------------------------
3839  *|*****************************************************************
3840  *|
3841  *|       Procedure: OPENSYMP
3842  *|
3843  *|      Called by: PUTSYMPTOM()        (function  in U.PRG)
3844  *|              : GETSYMPTOM()         (function  in U.PRG)
3845  *|
3846  *|*****************************************************************
3839                                          FUNCTION opensymp
3840                                            SELECT 0
3841                                            USE (f_symptom) INDEX (f_sympt
 => om)
3842  <-------------------------------------   RETURN SELECT()
3843
```

```
3844                                                    *---------------------------
 => ------------------------
3845                                                    *
3846                                                    * system
3847                                                    *
3848                                                    *---------------------------
 => ------------------------
3858  *|*****************************************************************
3859  *|
3860  *|       Function: GETSTATION()
3861  *|
3862  *|      Called by: INIENCOUNT()        (function  in DEMO.PRG)
3863  *|
3864  *|          Uses: SYSTEM.DBF
3865  *|
3866  *|*****************************************************************
3858                                          FUNCTION getstation
3859                                            PARAMETERS mstation, mdept
3860
3861                                            * get station details
3862                                            PRIVATE msel
3863                                            msel = SELECT()
 =>          && save current database selection
3864                                            SELECT 0
3865                                            USE SYSTEM
3866                                            mstation = station
3867                                            mdept    = department
3868                                            USE
 =>          && close disease file
3869                                            SELECT (msel)
 =>          && reselect original database
3870  <-------------------------------------   RETURN .T.
3871
3872                                            *---------------------------
 => ------------------------
3873                                            *
3874                                            * load system options
3875                                            *
3876                                            *---------------------------
 => ------------------------
3886  *|*****************************************************************
3887  *|
3888  *|      Procedure: GETOPTS
3889  *|
3890  *|      Called by: SYSINIT()           (function  in DEMO.PRG)
3891  *|
3892  *|          Uses: SYSTEM.DBF
3893  *|
3894  *|*****************************************************************
3886                                          FUNCTION getopts
3887
3888                                            PARAMETERS is_debug, is_conf,
 => is_rule, is_novice, is_edit, mprivate, mdisclaim
3889                                            msel = SELECT()
3890                                            SELECT 0
3891                                            USE SYSTEM
3892                                            msyswarn  =  syswarning   && s
 => ystem warnign text
3893                                            mprivate  =  privacyact   && s
 => ystem privatecy act text
3894                                            mdisclaim =  sysdiscl     && s
 => ystem disclaiming text
3895                                            mpass     =  ALLTRIM(password)
 =>          && system password
3896                                            is_debug  =  DEBUG        && s
 => ystem debug mode
```

```
3897                                    is_conf   =  CONF        && d
 => isplay confidence factors
3898                                    is_rule   =  showrule     && d
 => isplay rules fired
3899                                    is_novice =  novice       && n
 => ovice user mode
3900                                    is_edit   =  edithelp     && e
 => nable help edit mode
3901                                    USE
3902                                    SELECT (msel)
3903  <------------------------------------RETURN .T.
3904
3914  *!*****************************************************************
3915  *!
3916  *!      Procedure: NEXTPAT
3917  *!
3918  *!      Called by: ADDPAT()          (function  in U.PRG)
3919  *!
3920  *!          Uses: SYSTEM.DBF
3921  *!
3922  *!*****************************************************************
3914                                    FUNCTION nextpat
3915                                       PARAMETERS inc
3916                                       PRIVATE N
3917                                       PRIVATE msel
3918                                       msel = SELECT()
 => && save current database selection
3919                                       SELECT 0
3920                                       USE SYSTEM
3921                                       N = patients + 1
3922                                    ┌─IF IIF(TYPE("inc") = "L", inc,
 =>  .F.)
3923                                    │   REPLACE patients WITH N
3924                                    └─ENDIF
3925                                       USE
3926                                       SELECT (msel)
3927  <------------------------------------RETURN N
3928
3939  *!*****************************************************************
3940  *!
3941  *!      Function: ADDPAT()
3942  *!
3943  *!      Called by: M_PATADD          (procedure in DEMO.PRG)
3944  *!
3945  *!          Calls: NEXTPAT           (procedure in U.PRG)
3946  *!
3947  *!*****************************************************************
3939                                    FUNCTION addpat
3940  <------------------------------------RETURN nextpat(.T.)
3941
3942
3953  *!*****************************************************************
3954  *!
3955  *!      Function: NEXTENCNT()
3956  *!
3957  *!      Called by: M_ENCNEW          (procedure in DEMO.PRG)
3958  *!               : ADDENCNT()        (function  in U.PRG)
3959  *!
3960  *!          Uses: SYSTEM.DBF
3961  *!
3962  *!*****************************************************************
3953                                    FUNCTION nextencnt
3954                                       PARAMETERS inc
3955                                       PRIVATE N
3956                                       PRIVATE msel
```



```
3957                                       msel = SELECT()
 => && save current database selection
3958                                       SELECT 0
3959                                       USE SYSTEM
3960                                       N = encounters + 1
3961                                    ┌─IF IIF(TYPE("inc") = "L", inc,
 =>  .F.)
3962                                    │   REPLACE encounters WITH N
3963                                    └─ENDIF
3964                                       USE
3965                                       SELECT (msel)
3966  <------------------------------------RETURN N
3967
3968
3978  *!*****************************************************************
3979  *!
3980  *!      Function: ADDENCNT()
3981  *!
3982  *!      Called by: M_ENCSAV          (procedure in DEMO.PRG)
3983  *!
3984  *!          Calls: NEXTENCNT()       (function  in U.PRG)
3985  *!
3986  *!*****************************************************************
3978                                    FUNCTION addencnt
3979  <------------------------------------RETURN nextencnt(.T.)
3980
3981
3989  *!*****************************************************************
3990  *!
3991  *!      Procedure: NENCNT
3992  *!
3993  *!          Uses: SYSTEM.DBF
3994  *!
3995  *!*****************************************************************
3989                                    FUNCTION nencnt
3990                                       PRIVATE N
3991                                       PRIVATE msel
3992                                       msel = SELECT()
 => && save current database selection
3993                                       SELECT 0
3994                                       USE SYSTEM
3995                                       N = encounters
3996                                       USE
3997                                       SELECT (msel)
3998  <------------------------------------RETURN N
3999
4009  *!*****************************************************************
4010  *!
4011  *!      Procedure: NEXTUSER
4012  *!
4013  *!      Called by: ADDUSER           (procedure in U.PRG)
4014  *!
4015  *!          Uses: SYSTEM.DBF
4016  *!
4017  *!*****************************************************************
4009                                    FUNCTION nextuser
4010                                       PARAMETERS inc
4011                                       PRIVATE N
4012                                       PRIVATE msel
4013                                       msel = SELECT()
 => && save current database selection
4014                                       SELECT 0
4015                                       USE SYSTEM
4016                                       N = providers + 1
4017                                    ┌─IF IIF(TYPE("inc") = "L", inc,
 =>  .F.)
```

```
4018                                        REPLACE providers WITH N
4019                                     ENDIF
4020                                     USE
4021                                     SELECT (msel)
4022                                     RETURN N
4023
4031  *!*****************************************************************
4032  *!
4033  *!      Procedure: ADDUSER
4034  *!
4035  *!          Calls: NEXTUSER          (procedure in U.PRG)
4036  *!
4037  *!*****************************************************************
4031                                     FUNCTION adduser
4032                                     RETURN nextuser(.T.)
4033
4034
4035                                     * end database interface funct
=> ions
4036                                     ******************************
=> ************************************
4037                                     * Browse patient
4038                                     ******************************
=> ************************************
4039
4050  *!*****************************************************************
4051  *!
4052  *!      Procedure: BROWSEPAT
4053  *!
4054  *!      Called by: _08D0YAR6U()      (function  in WP1.PRG)
4055  *!
4056  *!          Calls: ERRMSG.PRG
4057  *!               : WPGETPATLIST      (procedure in U.PRG)
4058  *!
4059  *!*****************************************************************
4050                                     FUNCTION browsepat
4051                                        PARAMETER msoc, mlast, mfirst,
=>  mmid
4052                                        PRIVATE msn
4053
4054                                        msel = SELECT()
4055                                        SELECT (fpatient)
4056                                        SET ORDER TO 1
4057                                        msize = RECCOUNT()
4058                                        IF msize = 0
4059                                           =errmsg("Patient file is em
=> pty",1)
4060                                           RETURN ""
4061                                        ENDIF
4062                                        mcount = 0
4063                                        DIMENSION patlist[msize,2]
4064                                        SCAN
4065                                           mcount = mcount + 1
4066                                           patlist[mcount,1] = "[" + s
=> ocsec + "] " + ALLTRIM(lastname) + ", " +   ALLTRIM(firstname) + " " +
=> ALLTRIM(midleinit)
4067                                           patlist[mcount,2] =
=> "patient" + STR(RECNO())
4068                                        ENDSCAN
4069                                        mfinal = wpgetpatlist(@patlist
=> )
4070                                        SELECT (msel)
4071
4072                                        RETURN mfinal
4073
4088  *!*****************************************************************
```

```
4089  *!
4090  *!      Procedure: WPGETPATLIST
4091  *!
4092  *!      Called by: BROWSEPAT         (procedure in U.PRG)
4093  *!
4094  *!          Calls: PUSHHELP          (procedure in U.PRG)
4095  *!               : WP3.SPR
4096  *!               : POPHELP           (procedure in U.PRG)
4097  *!               : DP.PRG
4098  *!               : GETPAT            (procedure in U.PRG)
4099  *!               : WP2.PRG
4100  *!
4101  *!*****************************************************************
4088                                     FUNCTION wpgetpatlist
4089                                        PARAMETERS mlist
4090                                        * mlist is list of social secu
=> rity numbers, names
4091
4092                                        EXTERNAL ARRAY mlist
4093
4094                                        PRIVATE mchoice
4095                                        IF ALEN(mlist) > 0
4096                                           PRIVATE mbuttons
4097                                           mchoice = 0
4098                                           mbuttons = 0
4099                                           =pushhelp("PATIENT BROWSE W
=> INDOW")                    && rwd02
4100                                           DO wp3.spr
=>                                  && hll01
4101                                           =pophelp()
4102                                           mfinal = ""
4103                                           DO CASE
4104                                           CASE mbuttons = 1
4105                                              i = mchoice
4106                                              IF i > 0
4107                                                 mrec = VAL(dp(mlist[i
=> ,2],u,2))
4108                                                 mfinal = getpat(mrec)
4109                                              ENDIF
4110                                           CASE mbuttons = 2
4111                                              i = mchoice
4112                                              IF i > 0
4113                                                 mrec = VAL(dp(mlist[i
=> ,2],u,2))
4114                                                 mtext = getpat(mrec)
4115                                                 mnew = .F.
4116                                                 mfinal = wp2(mnew,mte
=> xt)
4117                                              ENDIF
=>                                                   && rwd02
4118                                           CASE mbuttons = 3
4119                                              mfinal = ""
4120                                           ENDCASE
4121                                        ENDIF
4122                                        RETURN mfinal
4123
4124                                        ******************************
=> ************************************
4125                                        * Search patient
4126                                        * Programmer: HLL
4127                                        * Note: - smode: 0; The candid
=> ate list from patient; enabled key for select and search option
4128                                        *          - smode: 1; The
=> candidate list from SAMS database; disabled for select and search option.
4129                                        *       - smode: 2; Since it n
=> ot interface with SAMS, disabled for search option
```

```
4130                                                                  *
4131                                                                  ******************************
 => ****************************************
4132
4151  *!*****************************************************************
4152  *!
4153  *!      Procedure: SEARCH
4154  *!
4155  *!      Called by: _08D0YAR6U()        (function  in WP1.PRG)
4156  *!
4157  *!          Calls: POPMESSAGE()        (function  in U.PRG)
4158  *!               : GETCAMDPAT          (procedure in U.PRG)
4159  *!               : GETSAMSPAT          (procedure in U.PRG)
4160  *!               : DP.PRG
4161  *!               : GETSAMS             (procedure in U.PRG)
4162  *!               : POPUPHIDE()         (function  in U.PRG)
4163  *!               : PUSHHELP            (procedure in U.PRG)
4164  *!               : WPATS.PRG
4165  *!               : POPHELP             (procedure in U.PRG)
4166  *!               : WP2.PRG
4167  *!
4168  *!*****************************************************************
4151                                                                  FUNCTION search
4152                                                                     PARAMETER msoc, mlast, mfirst
4153                                                                     PRIVATE mssn
4154                                                                     mfinal = ""
4155                                                                     mmid = " "
4156                                                                     mitem = msoc + u + mlast + u +
 => mfirst + u + mmid
4157                                                                     SET EXACT OFF
4158                                                                     smode = 0
4159                                                                  IF EMPTY(msoc) AND EMPTY(mlast
 => ) AND EMPTY(mfirst)
4160                                                                        =popmessage("Could not sear
 => ch without criteria")
4161                                                                     RETURN ""
4162                                                                  ENDIF
4163                                                                     winpop = popupshow ("searching
 => ...")
4164                                                                     DO getcamdpat WITH msoc, mlast
 => , mfirst
4165                                                                     mcount = 0
4166                                                                     msize = 0
4167                                                                     smode = IIF(mtrain, 2, smode)
4168                                                                     smode = IIF(!msamsdata, 2, smo
 => de)
4169                                                                  IF RECCOUNT("wpat1") > 0
 =>          && Combine all candidate into one
4170                                                                        SELECT wpat1
4171                                                                        msize = RECCOUNT("wpat1")
4172                                                                        DIMENSION choice[msize,2],m
 => ssnlist[msize]
4173                                                                     SCAN
4174                                                                        mcount = mcount + 1
4175                                                                        mssnlist[mcount] = socse
 => c
4176                                                                        choice[mcount,1] = "[" +
 => socsec + "] " + ALLTRIM(lastname) + ", " +    ALLTRIM(firstname) + " " +
 => ALLTR'M(midleinit)
4177                                                                        choice[mcount,2] = "pati
 => ent" + STR(RECNO())
4178                                                                     ENDSCAN
4179                                                                  ELSE
4180                                                                     IF msamsdata
4181                                                                        DO getsamspat WITH msoc,
 => mlast, mfirst
```

```
4182                                                                  IF RECCOUNT("wpat2") > 1
 =>          && Combine all candidate into one
4183                                                                        SELECT wpat2
4184                                                                        msize = RECCOUNT("wpa
 => t2")
4185                                                                        DIMENSION choice[msiz
 => e,2]
4186                                                                     SCAN
4187                                                                        mcount = mcount +
 => 1
4188                                                                        mtext =   "[" + ss
 => n + "] " + dp(name," ",1) + ", " + dp(name," ",2) + " " + SUBSTR(dp(name,"
 => ",3),1,1)
4189                                                                        choice[mcount,1] =
 => mtext
4190                                                                        choice[mcount,2] =
 => "sams" + STR(RECNO())
4191                                                                     ENDSCAN
4192                                                                     smode = 1
4193                                                                  ELSE
4194                                                                     IF RECCOUNT("WPAT2")
 => = 1
4195                                                                        mitem = getsams(1)
4196                                                                     ENDIF
4197                                                                  ENDIF
4198                                                               ENDIF
4199                                                            ENDIF
4200                                                            =popuphide(winpop)
4201                                                            IF TYPE("choice") != "U"
4202                                                               DIMENSION choice[mcount,2]
4203                                                               IF mcount > 1
4204                                                                  =ASORT(choice,1)
4205                                                               ENDIF
4206                                                               =pushhelp("PATIENT CANDIDAT
 => E LIST")
4207                                                               mfinal = wpats(@choice,"Sel
 => ect following candidate:",mitem,smode)
4208                                                               =pophelp()
4209                                                            ELSE
4210                                                               mnew = .T.
4211                                                               mfinal = wp2(mnew,mitem)
4212                                                            ENDIF
4213                                                            RETURN mfinal
4214                                                            ******************************
 => ****************************************
4215                                                            * Search patient in sams datab
 => ase
4216                                                            ******************************
 => ****************************************
4217
4230  *!*****************************************************************
4231  *!
4232  *!      Procedure: MORESEARCH
4233  *!
4234  *!      Called by: _0710L5TJ5()        (function  in WPATS.PRG)
4235  *!
4236  *!          Calls: DP.PRG
4237  *!               : POPMESSAGE()        (function  in U.PRG)
4238  *!               : GETSAMSPAT          (procedure in U.PRG)
4239  *!               : POPUPHIDE()         (function  in U.PRG)
4240  *!
4241  *!*****************************************************************
4230                                                            PROCEDURE moresearch
4231                                                               PARAMETER mitem, choice
4232                                                               PRIVATE mssn
4233                                                               mfinal = ""
```

```
4234                                msoc = dp(mitem, u, 1)
4235                                mlast = dp(mitem, u, 2)
4236                                mfirst = dp(mitem, u, 3)
4237                                mmid = dp(mitem, u, 4)
4238                                mpresize = ALEN(choice,1)
4239                                DIMENSION mprelist[mpresize]
4240                                FOR i = 1 TO mpresize
4241                                   mtssn = dp(choice[i,1],"[",
 => 2)
4242                                   mtssn = dp(mtssn,"]",1)
4243                                   mprelist[i] = mtssn
4244                                ENDFOR
4245                                SET EXACT OFF
4246                                mnosams = .F.
4247                                IF EMPTY(msoc) AND EMPTY(mlast
 => ) AND EMPTY(mfirst)
4248                                   =popmessage("Could not sear
 => ch without critiria")
4249                                   RETURN ""
4250                                ENDIF
4251                                winpop = popupshow ("searching
 => ...")
4252                                DO getsamspat WITH msoc, mlast
 => , mfirst
4253                                msize = ALEN(mprelist)
4254                                mcount = ALEN(mprelist)
4255                                IF RECCOUNT("wpat2") > 0
4256                                   SELECT wpat2
4257                                   msize = msize + RECCOUNT("w
 => pat2")
4258                                   DIMENSION choice[msize,2]
4259                                   SCAN
4260                                      mtext = "[" + ssn + "] "
 =>  + dp(name," ",1) + ", " + dp(name," ",2) + " " + SUBSTR(dp(name," ",3),1,1
 => )
4261                                      IF TYPE("mprelist") = "U
 => "
4262                                         mnodup = .T.
4263                                      ELSE
4264                                         mnodup = IIF(ASCAN(mp
 => relist,ssn) = 0, .T., .F.)
4265                                      ENDIF
4266                                      IF mnodup
4267                                         mcount = mcount + 1
4268                                         choice[mcount,1] = mt
 => ext
4269                                         choice[mcount,2] = "s
 => ams" + STR(RECNO())
4270                                      ENDIF
4271                                   ENDSCAN
4272                                   RELEASE mprelist
4273                                ENDIF
4274                                =popuphide(winpop)
4275                                IF TYPE("choice") != "U"
4276                                   DIMENSION choice[mcount,2]
4277                                   IF mcount > 1
4278                                      =ASORT(choice,1)
4279                                   ENDIF
4280                                ENDIF
4281                                RETURN
4282
4283                                ******************************
 => ****************************************
4284                                *  Get the matching list in CA
 => MD
4285                                ******************************
```

```
 => ****************************************
4294  *!*****************************************************************
4295  *!
4296  *!      Procedure: GETCAMDPAT
4297  *!
4298  *!      Called by: SEARCH              (procedure in U.PRG)
4299  *!               : SFGETPAT            (procedure in U.PRG)
4300  *!
4301  *!*****************************************************************
4294                                PROCEDURE getcamdpat
4295                                   PARAMETER mssn, mlast, mfirst
4296                                   PRIVATE mcom, olddir, mdatadir
4297                                   mcom = SET("EXACT")
4298                                   SET EXACT OFF
4299                                   tssn = mssn
4300                                   tlast = mlast
4301                                   tfirst = mfirst
4302                                   IF OCCURS(" ",tssn) = 0
4303                                      tlast = ""
4304                                      tfirst = ""
4305                                   ELSE
4306                                      tssn = STRTRAN(tssn," ","_"
 => )
4307                                   ENDIF
4308                                   tssn = STRTRAN(tssn,"?"," ")
4309                                   tlast = STRTRAN(ALLTRIM(tlast)
 => ,"?"," ") + "%"
4310                                   tfirst = STRTRAN(ALLTRIM(tfirs
 => t),"?"," ") + "%"
4311                                   tlast = ALLTRIM(tlast)
4312                                   tfirst = ALLTRIM(tfirst)
4313
4314                                   SELECT socsec,lastname,firstna
 => me,midleinit,datebirth,sex,raterank,id,RECNO();
4315                                      FROM (f_patient);
4316                                      INTO CURSOR wpati;
4317                                      WHERE patient.socsec LIKE
 => (tssn);
4318                                      AND UPPER(patient.lastname
 => ) LIKE (UPPER(tlast));
4319                                      AND UPPER(patient.firstnam
 => e) LIKE (UPPER(tfirst));
4320
4321                                   SET EXACT &mcom
4322                                   RETURN
4323
4324                                   ******************************
 => ****************************************
4325                                   ** Get the matching list in SA
 => MS
4326                                   ******************************
 => ****************************************
4335  *!*****************************************************************
4336  *!
4337  *!      Procedure: GETSAMSPAT
4338  *!
4339  *!      Called by: SEARCH              (procedure in U.PRG)
4340  *!               : MORESEARCH          (procedure in U.PRG)
4341  *!
4342  *!*****************************************************************
4335                                PROCEDURE getsamspat
4336                                   PARAMETER mssn, mlast, mfirst
4337                                   PRIVATE olddir, mdatadir
4338
4339                                   tssn = mssn
4340                                   tlast = mlast
```

```
4341                                                    tfirst = mfirst
4342                                                   IF msamsdata
4343                                                      IF OCCURS(" ",mssn) = 0
4344                                                         tlast = ""
4345                                                         tfirst = ""
4346                                                      ELSE
4347                                                         tssn = STRTRAN(tssn," ",
 => "_")
4348                                                      ENDIF
4349                                                      tssn = STRTRAN(tssn,"?","_"
 => )
4350                                                      tlast = STRTRAN(ALLTRIM(tla
 => st),"?","_") + "%"
4351                                                      tfirst = STRTRAN(ALLTRIM(tf
 => irst),"?","_") + "%"
4352                                                      tlast = ALLTRIM(tlast)
4353                                                      tfirst = ALLTRIM(tfirst)
4354                                                      SELECT ssn,name,crwdob,crws
 => ex,RANK "SAMS";
4355                                                         FROM (f_mkey);
4356                                                         INTO CURSOR wpat2;
4357                                                         WHERE ssn LIKE tssn;
4358                                                         AND name LIKE (UPPER(tl
 => ast));
4359                                                         AND dp(name," ",2) LIKE
 =>  (UPPER(tfirst));
4360
4361                                                   ENDIF
4362                                                   RETURN
4363
4364                                                   ******************************
 => ****************************************
4365                                                   *  Get the data from Query.
4366                                                   ******************************
 => ****************************************
4379  *!*****************************************************************
4380  *!
4381  *!      Procedure: GETPATDATA
4382  *!
4383  *!      Called by: RESULT            (procedure in WPATS.PRG)
4384  *!               : PATEDIT           (procedure in WPATS.PRG)
4385  *!
4386  *!          Calls: DP.PRG
4387  *!               : GETPATTEMP        (procedure in U.PRG)
4388  *!               : GETSAMS           (procedure in U.PRG)
4389  *!*****************************************************************
4379                                                FUNCTION getpatdata
4380                                                   PARAMETER mitem
4381                                                   mfile = dp(mitem,u,2)
4382                                                   mrecno = VAL(dp(mitem,u,3))
4383                                                   IF UPPER(mfile)$"PATIENT"
4384                                                      mtext = getpattemp(mrecno)
4385                                                   ELSE
4386                                                      mtext = getsams(mrecno)
4387                                                   ENDIF
4388                                                   RETURN mtext
4389
4390                                                   ******************************
 => ****************************************
4391                                                   *  getdata from sams database
4392                                                   ******************************
 => ****************************************
4403  *!*****************************************************************
4404  *!
4405  *!      Procedure: GETSAMS
```

```
4406  *!
4407  *!      Called by: SEARCH            (procedure in U.PRG)
4408  *!               : GETPATDATA        (procedure in U.PRG)
4409  *!
4410  *!          Calls: SAMSDATA          (procedure in U.PRG)
4411  *!
4412  *!*****************************************************************
4403                                                FUNCTION getsams
4404                                                   PARAMETER mrec
4405                                                   PRIVATE msel
4406                                                   msel = SELECT()
4407                                                   SELECT wpat2
4408                                                   GO RECORD mrec
4409                                                   mt = samsdata()
4410                                                   SELECT (msel)
4411                                                   RETURN mt
4412
4413
4424  *!*****************************************************************
4425  *!
4426  *!      Procedure: SAMSDATA
4427  *!
4428  *!      Called by: GETSAMS           (procedure in U.PRG)
4429  *!
4430  *!          Calls: DP.PRG
4431  *!               : COMBINED          (procedure in U.PRG)
4432  *!
4433  *!*****************************************************************
4424                                                FUNCTION samsdata
4425                                                   mssn = ssn
4426                                                   mlast = dp(name," ",1)
4427                                                   mfirst = dp(name," ",2)
4428                                                   mmid = SUBSTR(dp(name," ",3),1
 => ,1)
4429                                                   mdob = crwdob
4430                                                   msx = crwsex
4431                                                   mrank = RANK
4432                                                   mt = combined(mssn,mlast,mfirs
 => t,mmid,DTOC(mdob),msx,mrank)
4433                                                   RETURN mt
4434
4435                                                   ******************************
 => ****************************************
4436                                                   *  getdata from patient databa
 => se
4437                                                   ******************************
 => ****************************************
4447  *!*****************************************************************
4448  *!
4449  *!      Procedure: GETPAT
4450  *!
4451  *!      Called by: WPGETPATLIST      (procedure in U.PRG)
4452  *!
4453  *!          Calls: PATDATA           (procedure in U.PRG)
4454  *!
4455  *!*****************************************************************
4447                                                FUNCTION getpat
4448                                                   PARAMETER mrec
4449                                                   PRIVATE msel
4450                                                   msel = SELECT()
4451                                                   SELECT (fpatient)
4452                                                   SET ORDER TO 1
4453                                                   GO RECORD mrec
4454                                                   mt = patdata()
4455                                                   mt = mt + u + STR(mrec)
4456                                                   SELECT (msel)
```

```
4457  <------------------------------------RETURN mt
4458
4459                                         ******************************
 => ****************************************
4460                                         *  getdata from patient databa
 => se (temporary file)
4461                                         ******************************
 => ****************************************
4472  *!*******************************************************************
4473  *!
4474  *!      Procedure: GETPATTEMP
4475  *!
4476  *!      Called by: GETPATDATA          (procedure in U.PRG)
4477  *!               : SFGETPATLIST        (procedure in U.PRG)
4478  *!
4479  *!          Calls: PATDATA             (procedure in U.PRG)
4480  *!
4481  *!*******************************************************************
4472                                         FUNCTION getpattemp
4473                                            PARAMETER mrec
4474                                            PRIVATE msel
4475                                            msel = SELECT()
4476                                            SELECT wpat1
4477                                            GO RECORD mrec
4478                                            mt = patdata()
4479                                            mrec = exp_9
4480                                            mt = mt + u + STR(mrec)
4481                                            SELECT (msel)
4482  <------------------------------------RETURN mt
4483
4484                                         ******************************
 => ****************************************
4485                                         *  getdata from database
4486                                         ******************************
 => ****************************************
4497  *!*******************************************************************
4498  *!
4499  *!      Procedure: PATDATA
4500  *!
4501  *!      Called by: GETPAT              (procedure in U.PRG)
4502  *!               : GETPATTEMP          (procedure in U.PRG)
4503  *!
4504  *!          Calls: COMBINED            (procedure in U.PRG)
4505  *!
4506  *!*******************************************************************
4497                                         FUNCTION patdata
4498                                            mssn = socsec
4499                                            mlast = lastname
4500                                            mfirst = firstname
4501                                            mmid = midleinit
4502                                            mdob = datebirth
4503                                            msx = sex
4504                                            mrank = raterank
4505                                            mpatid = id
4506                                            mt = combined(mssn,mlast,mfirs
 => t,mmid,DTOC(mdob),msx,mrank,STR(mpatid))
4507  <------------------------------------RETURN mt
4508
4509                                         ******************************
 => ****************************************
4510                                         *  Combined all datas into one
 =>  string
4511                                         ******************************
 => ****************************************
4522  *!*******************************************************************
4523  *!
```

```
4524  *!      Procedure: COMBINED
4525  *!
4526  *!      Called by: SAMSDATA            (procedure in U.PRG)
4527  *!               : PATDATA             (procedure in U.PRG)
4528  *!               : SAVREC              (procedure in U.PRG)
4529  *!               : _Q8D0TWDV1()        (function  in WP2.PRG)
4530  *!
4531  *!*******************************************************************
4522                                         FUNCTION combined
4523                                            PARAMETER mt1, mt2, mt3, mt4,
 => mt5, mt6, mt7, mt8, mt9
4524                                            mtext = ""
4525                                            IF PARAMETER() > 0
4526                                               FOR i = 1 TO PARAMETER()
4527                                                  mvar = 'mt' + ALLTRIM(ST
 => R(i))
4528                                                  mtext = IIF(i = 1, EVAL(
 => mvar), (mtext + u + EVAL(mvar)))
4529                                               ENDFOR
4530                                            ENDIF
4531  <------------------------------------RETURN mtext
4532
4533                                         ******************************
 => ****************************************
4534                                         *  Save the record into patien
 => t database
4535                                         ******************************
 => ****************************************
4547  *!*******************************************************************
4548  *!
4549  *!      Procedure: SAVREC
4550  *!
4551  *!      Called by: _Q8D0TWDV1()        (function  in WP2.PRG)
4552  *!               : RESULT              (procedure in WPATS.PRG)
4553  *!
4554  *!          Calls: DP.PRG
4555  *!               : COMBINED            (procedure in U.PRG)
4556  *!
4557  *!*******************************************************************
4547                                         FUNCTION savrec
4548                                            PARAMETER mflag, mitem
4549                                            IF PARAMETER() = 0
4550                                               mflag = .T.
4551                                            ENDIF
4552                                            mnew = mflag
4553                                            PRIVATE msel,mssn,mlast,mfirst
 => ,mmid,mdob,msx,mrank,mpatid
4554
4555                                            mssn = dp(mitem,u,1)
4556                                            mlast = UPPER(dp(mitem,u,2))
4557                                            mfirst = UPPER(dp(mitem,u,3))
4558                                            mmid = UPPER(dp(mitem,u,4))
4559                                            mdob = CTOD(dp(mitem,u,5))
4560                                            msx = UPPER(dp(mitem,u,6))
4561                                            mrank = UPPER(dp(mitem,u,7))
4562                                            mpatid = VAL(dp(mitem,u,8))
4563                                            mrec = VAL(dp(mitem,u,9))
4564                                            msel = SELECT()
4565
4566                                            SELECT (fpatient)
4567                                            IF mnew
4568                                               SET ORDER TO 3
4569                                               GO BOTTOM
4570                                               mpatid = id + 1
4571                                               APPEND BLANK
4572                                            ELSE
```

```
4573                                                    IF EMPTY(mrec)
4574                                                       SELECT (msel)
4575  <-------------------------------------------------RETURN
4576                                                    ENDIF
4577                                                      GO RECORD mrec
4578                                                  ENDIF
4579                                                  REPLACE id WITH mpatid
4580                                                  REPLACE socsec WITH mssn
4581                                                  REPLACE lastname WITH mlast
4582                                                  REPLACE firstname WITH mfirst
4583                                                  REPLACE midleinit WITH mmid
4584                                                  REPLACE datebirth WITH mdob
4585                                                  REPLACE sex WITH msx
4586                                                  REPLACE raterank WITH mrank
4587                                                  mdata = combined(mssn,mlast,mf
 => irst,mmid,DTOC(mdob),msx,mrank,STR(mpatid),STR(mrec))
4588                                                  SELECT (msel)
4589  <-------------------------------------------------RETURN mdata
4590                                                  ******************************
 => ****************************************
4591                                                  * Open SAMS database.
4592                                                  ******************************
 => ****************************************
4605  *!*****************************************************************
4606  *!
4607  *!      Procedure: OPENSAMS
4608  *!
4609  *!      Called by: OPENPAT              (procedure in U.PRG)
4610  *!
4611  *!          Calls: SETPATH              (procedure in U.PRG)
4612  *!               : FILEFIND             (procedure in U.PRG)
4613  *!
4614  *!           Uses: &F_MKEY
4615  *!
4616  *!*****************************************************************
4605                                                FUNCTION opensams
4606                                                  PUBLIC f_mkey, f_mkeyssn, fsam
 => s
4607                                                  f_mkey = "mkey"
4608                                                  f_mkeyssn = "mkeyssn"
4609                                                  IF !EMPTY(GETENV("SAMS"))
4610                                                     DO setpath WITH GETENV("SAM
 => S")
4611                                                  ENDIF
4612                                                  mffind = filefind(f_mkey + ".d
 => bf")
4613                                                  IF !mffind
4614  <-------------------------------------------------RETURN .F.
4615                                                  ENDIF
4616                                                  IF EMPTY(fsams) OR !USED(f_mke
 => y)
4617                                                     SELECT 0
4618                                                     USE &f_mkey AGAIN INDEX (f_
 => mkeyssn)
4619                                                     fsams = SELECT()
4620                                                  ELSE
4621                                                     SELECT (fsams)
4622                                                     fsams = SELECT()
4623                                                  ENDIF
4624  <-------------------------------------------------RETURN .T.
4625
4626                                                  ******************************
 => ****************************************
4627                                                  * Convert first letter of sent
 => ense to lower case or upper case
4628                                                  ******************************
```

```
 => ****************************************
4636  *!*****************************************************************
4637  *!
4638  *!      Procedure: CASECONVERT
4639  *!
4640  *!      Called by: SFSYMP               (procedure in DEMO.PRG)
4641  *!
4642  *!*****************************************************************
4636                                                FUNCTION caseconvert
4637                                                   PARAMETER mstr, mcase
4638                                                   DO CASE
4639                                                   CASE mcase = "U"
4640                                                      mtext = UPPER(LEFT(mstr,1))
 =>  + SUBSTR(mstr,2)
4641                                                   CASE mcase = "L"
4642                                                      mtext = LOWER(LEFT(mstr,1))
 =>  + SUBSTR(mstr,2)
4643                                                   OTHERWISE
4644                                                      mtext = ""
4645                                                   ENDCASE
4646  <-------------------------------------------------RETURN mtext
4647
4648                                                   *---------------------------
 => ---------------------
4649                                                   * Get the encounter for sf600
 => report
4650                                                   *---------------------------
 => ---------------------
4651
4669  *!*****************************************************************
4670  *!
4671  *!      Procedure: SFENCOUNT
4672  *!
4673  *!      Called by: M_SF600              (procedure in DEMO.PRG)
4674  *!
4675  *!          Calls: M_PATGET             (procedure in U.PRG)
4676  *!               : PUSHHELP             (procedure in U.PRG)
4677  *!               : WP4.SPR
4678  *!               : POPHELP              (procedure in U.PRG)
4679  *!               : SFGETPAT             (procedure in U.PRG)
4680  *!               : SFGETENC             (procedure in U.PRG)
4681  *!               : POPMESSAGE()         (function  in U.PRG)
4682  *!               : PATSSN()             (function  in U.PRG)
4683  *!               : LODENC               (procedure in DEMO.PRG)
4684  *!
4685  *!*****************************************************************
4669                                                FUNCTION sfencount
4670                                                   PRIVATE mprevpat, mprevdate
4671                                                   mprevpat = mpatientid
 =>   && previous patient selected
4672                                                   mprevdate = mdateenc
4673                                                   mdateenc = CTOD("  /  /  ")
4674                                                   mbuttons = 1
4675                                                   IF mpatientid > 0
4676                                                      =m_patget(mpatientid,msocse
 => c,@mlastname,@mfirstname,@mdatebirth,@msex,@mraterank)
4677                                                   ENDIF
4678                                                   mssn = IIF(EMPTY(msocsec),"
 =>      ",msocsec)
4679                                                   mlast = mlastname
4680                                                   mfirst = mfirstname
4681                                                   mdob = mdatebirth
4682                                                   msx = msex
4683                                                   mrate = mraterank
4684                                                   =pushhelp("SF500 REPORT SELECT
 => ION")
```

```
4685                                    DO wp4.spr
4686                                    =pophelp()
4687                                    DO CASE
4688                                    CASE mbuttons = 1
4689                                       mpatientid = sfgetpat(mssn,
 => mlast, mfirst)
4690                                       DO CASE
4691                                       CASE mpatientid > 0
4692                                          IF !EMPTY(mdateenc)
4693                                             mencountid = sfgetenc
 => (mpatientid, mdateenc)
4694                                          ELSE
4695                                             mencountid = sfgetenc
 => (mpatientid)
4696                                          ENDIF
4697                                       CASE mpatientid = 0
4698                                          IF !EMPTY(mdateenc)
4699                                             mencountid = sfgetenc
 => (mdateenc)
4700                                          ELSE
4701                                             mencountid = 0
4702                                          ENDIF
4703                                       CASE mpatientid < 0
4704                                          =popmessage("This Patien
 => t doesn't exist")
4705                                          mencountid = 0
4706                                       ENDCASE
4707                                    CASE mbuttons = 2
4708                                       mencountid = sfgetenc()
4709                                    OTHERWISE
4710                                       mencountid = 0
4711                                    ENDCASE
4712                                    IF mencountid > 0
4713                                       * pending encounter
4714                                       msocsec = patssn(mpatientid
 => )
4715                                       * get patient record
4716                                       =m_patget(mpatientid,msocse
 => c,@mlastname,@mfirstname,@mdateBirth,@msex,@mraterank)
4717                                       =lodenc(mencountid)
4718                                    ENDIF
4719                                    mpatientid = mprevpat
4720                                    mdateenc = mprevdate
4721
4722                                    RETURN mencountid
4723
4724                                    *---------------------------
 => -----------------------
4725                                    *
4726                                    * get previous patient encount
 => er for processing/review
4727                                    *
4728                                    *---------------------------
 => -----------------------
4741  *!*****************************************************************
4742  *!
4743  *!      Procedure: SFGETENC
4744  *!
4745  *!      Called by: SFENCOUNT          (procedure in U.PRG)
4746  *!
4747  *!          Calls: ENCCOUNT()         (function  in U.PRG)
4748  *!               : SFENCNAMES         (procedure in U.PRG)
4749  *!               : SFENCLIST          (procedure in U.PRG)
4750  *!               : POPMESSAGE()       (function  in U.PRG)
4751  *!
4752  *!*****************************************************************
```

```
4741                                    FUNCTION sfgetenc
4742                                       PARAMETERS par1, par2
4743
4744                                       PRIVATE N
4745                                       PRIVATE names
4746                                       PRIVATE mencntid
4747                                       DO CASE
4748                                       CASE PARAMETER() = 0
4749                                          mpatientid = 0
4750                                          mdate = CTOD("")
4751                                       CASE PARAMETER() = 1
4752                                          IF TYPE("par1") = "D"
4753                                             mdate = par1
4754                                             mpatientid = 0
4755                                          ELSE
4756                                             IF TYPE("par1") = "N"
4757                                                mpatientid = par1
4758                                                mdate = CTOD("")
4759                                             ENDIF
4760                                          ENDIF
4761                                       CASE PARAMETER() = 2
4762                                          mpatientid = par1
4763                                          mdate = par2
4764                                       ENDCASE
4765                                       mencntid = 0
4766                                       PRIVATE names, ids, patients,
 => N, mchoice
4767                                       N = enccount()
4768                                       IF N > 0
 => && number of encounters
4769                                          * user selects an encounter
 => from list
4770                                          DIMENSION names[n], ids[n],
 => patients[n]
4771
4772                                          N = sfencnames(names, ids,
 => patients, mpatientid, mdate)      && create list of names
4773                                          IF N > 0
4774                                             DIMENSION names[n], ids[
 => n], patients[n]
4775                                             mchoice = sfenclist(name
 => s, mpatientid, mdate)
4776                                             IF mchoice > 0
4777                                                mencntid = ids[mchoic
 => e]
4778                                                mpatientid = patients
 => [mchoice]
4779                                             ELSE
4780                                                mencntid = 0
4781                                             ENDIF
4782                                          ELSE
4783                                             =popmessage("No encounte
 => r....")
4784                                          ENDIF
4785                                       ELSE
4786                                          =popmessage("No encounter..
 => ..")
4787                                       ENDIF
4788                                       RETURN mencntid
4789
4790                                       *---------------------------
 => -----------------------
4791                                       *
4792                                       * select a previous encounter
 => from list
4793                                       *
```

```
4794                                                    *---------------------------
 => -------------------------
4807  *|*****************************************************************
4808  *|
4809  *|      Procedure: SFENCLIST
4810  *|
4811  *|      Called by: SFGETENC          (procedure in U.PRG)
4812  *|
4813  *|          Calls: PATSSN()          (function  in U.PRG)
4814  *|                : PUSHHELP         (procedure in U.PRG)
4815  *|                : WP5.SPR
4816  *|                : POPHELP          (procedure in U.PRG)
4817  *|
4818  *|*****************************************************************
4807                                        FUNCTION sfenclist
4808                                          PARAMETERS mlist, mpat, mdaten
 => c
4809
4810                                          EXTERNAL ARRAY mlist
4811                                          PRIVATE mchoice
4812                                          mchoice = 0
4813                                        IF ALEN(mlist) > 0
4814                                          * display menu of names in
 => window
4815                                          PRIVATE mbuttons
4816                                          mkey =IIF(EMPTY(mpat), "",
 => patssn(mpat))
4817                                          mdate = IIF(EMPTY(mdatenc),
 => "", mdatenc)
4818                                          mchoice = 0
4819                                          mbuttons = 0
4820                                          =pushhelp("ENCOUNTER VERIFI
 => CATION")
4821                                          DO wp5.spr
 =>                                  && hilo
4822                                          =pophelp()
4823                                        DO CASE
4824                                        CASE mbuttons = 1
4825                                          * RESUME button
4826                                          IF mchoice = 0
4827                                            * nothing to resume,
 => so cancel
4828                                            mchoice = -1
4829                                          ENDIF
4830                                        CASE mbuttons = 2
4831                                          * CANCEL button
4832                                          mchoice = -1
4833                                        OTHERWISE
4834                                          mchoice = -1
4835                                        ENDCASE
4836                                      ENDIF
4837  <-------------------------------------RETURN mchoice
4838
4839                                        *---------------------------
 => ------------------------------
4840                                        * Validation patient in file
4841                                        *---------------------------
 => ------------------------------
4853  *|*****************************************************************
4854  *|
4855  *|      Procedure: SFGETPAT
4856  *|
4857  *|      Called by: SFENCOUNT         (procedure in U.PRG)
4858  *|
4859  *|          Calls: GETCAMDPAT        (procedure in U.PRG)
4860  *|                : SFGETPATLIST     (procedure in U.PRG)
```

```
4861  *|                : DP.PRG
4862  *|
4863  *|*****************************************************************
4853                                        FUNCTION sfgetpat
4854                                          PARAMETER msoc, mlastt, mfirst
 => t
4855                                          PRIVATE mssn, mlast, mfirst
4856                                          mssn = IIF(EMPTY(msoc),"
 =>     " msoc)
4857                                          mlast = ALLTRIM(mlastt)
4858                                          mfirst = ALLTRIM(mfirstt)
4859                                        IF EMPTY(mssn) AND EMPTY(mlast
 => ) AND EMPTY(mfirst)
4860                                          mpatid = 0
4861                                        ELSE
4862                                          DO getcamdpat WITH mssn, ml
 => ast, mfirst
4863                                          IF RECCOUNT("wpat1") > 0
4864                                            u = "`"
4865                                            SELECT wpat1
4866                                            mcount = 0
4867                                            msize = RECCOUNT("wpat1"
 => )
4868                                            DIMENSION choice[msize,2
 => ],mssnlist[msize]
4869                                            SCAN
4870                                              mcount = mcount + 1
4871                                              choice[mcount,1] = "[
 => " + socsec + "] " + ALLTRIM(lastname) + ", " +  ALLTRIM(firstname) + " " +
 => ALLTRIM(middleinit)
4872                                              choice[mcount,2] = "p
 => atient" + STR(RECNO())
4873                                            ENDSCAN
4874                                            mfinal = =fgetpatlist(@c
 => hoice)
4875                                            mpatid = VAL(dp(mfinal,u
 => ,8))
4876                                            SELECT wpat1
4877                                            USE
4878                                          ELSE
4879                                            mpatid = -1
4880                                          ENDIF
4881                                        ENDIF
4882  <-------------------------------------RETURN mpatid
4883                                        *---------------------------
 => ------------------------
4884                                        *
4885                                        * Lookup encounter list
4886                                        *
4887                                        *---------------------------
 => ------------------------
4901  *|*****************************************************************
4902  *|
4903  *|      Procedure: SFENCNAMES
4904  *|
4905  *|      Called by: SFGETENC          (procedure in U.PRG)
4906  *|
4907  *|          Calls: PATSSN()          (function  in U.PRG)
4908  *|
4909  *|           Uses: AREA.DBF
4910  *|
4911  *|        Indexes: AREA.IDX
4912  *|
4913  *|*****************************************************************
4901                                        FUNCTION sfencnames
4902                                          PARAMETERS names, ids, patient
```

```
 => s, mpatient, mdate
4903
4904                                                    EXTERNAL ARRAY names, ids, pat
 => ients                                        && rwd07
4905                                                    * list encounter file by:
4906                                                    *    patient
4907                                                    *    patient + date
4908                                                    *    date
4909                                                    *    all
4910
4911                                                    PRIVATE i
4912                                                    PRIVATE mexact
4913                                                    PRIVATE mid
4914                                                    PRIVATE msocsec
4915                                                    PRIVATE ispatient, isdate
4916                                                    PRIVATE msel
4917                                                    PRIVATE msel2
4918
4919                                                    msel = SELECT()
 => && save current database selection
4920                                                    SELECT 0
4921                                                    USE area AGAIN INDEX area
4922                                                    msel2 = SELECT()
4923                                                    SELECT (fencountr)
4924                                                    SET RELATION TO area INTO area
4925                                                    mid = 0
4926                                                    mexact = SET("EXACT")
4927                                                    SET EXACT OFF
4928                                                    ispatient = .F.
4929                                                    isdate    = .F.
4930                                                    IF PARAMETERS() > 3  .AND. TYP
 => E("mpatient") = "N"
4931                                                        ispatient = IIF(IEMPTY(mpat
 => ient),.T.,.F.)
4932                                                    ENDIF
4933                                                    IF PARAMETERS() > 4
4934                                                        isdate = IIF(IEMPTY(mdate),
 => .T., .F.)
4935                                                    ENDIF
4936                                                    SET ORDER TO 2              && i
 => ndex by date
4937                                                    IF ispatient
4938                                                      IF isdate
4939                                                          SET FILTER TO DATE = mda
 => te  .AND. patient = mpatient
4940                                                      ELSE
4941                                                          SET FILTER TO patient =
 => mpatient
4942                                                      ENDIF
4943                                                    ELSE
4944                                                      IF isdate
4945                                                          SET FILTER TO DATE = mda
 => te
4946                                                      ENDIF
4947                                                    ENDIF
4948                                                    * create list of names
4949                                                    i = 0
4950                                                    GOTO TOP
4951                                                    DO WHILE .NOT. EOF()
4952                                                      IF patient != mid
4953                                                          mid = patient
4954                                                          msocsec = patssn(mid)
4955                                                      ENDIF
4956                                                      i = i + 1
4957                                                      ids[i] = encountrid
4958                                                      patients[i] = mid
```

```
4959                                                      names[i] = IIF( ispatient,
 => "",  msocsec + " " ) ;
4960                                                          + IIF( isdate, "", DTOC
 => (DATE) + " " ) ;
4961                                                          + TIME + " " + area->na
 => me
4962                                                      SKIP
4963                                                    ENDDO
4964
4965                                                    IF mexact = "ON"
4966                                                      SET EXACT ON
4967                                                    ENDIF
4968                                                    SET FILTER TO
 =>          && remove filter condition
4969                                                    SET ORDER TO 1
 =>          && restore index order
4970                                                    SELECT (msel2)
4971                                                    USE
4972                                                    SELECT (msel)
 =>          && restore database selection
4973                                                    RETURN i
 =>          && list of names
4974                                                    *-----------------------------
 => ----------------------------------------
4975
4989  *!*****************************************************************
4990  *!
4991  *!      Procedure: SFGETPATLIST
4992  *!
4993  *!      Called by: SFGETPAT            (procedure in U.PRG)
4994  *!
4995  *!          Calls: PUSHHELP            (procedure in U.PRG)
4996  *!               : WP8.SPR
4997  *!               : POPHELP             (procedure in U.PRG)
4998  *!               : DP.PRG
4999  *!               : GETPATTEMP          (procedure in U.PRG)
5000  *!
5001  *!*****************************************************************
4989                                                    FUNCTION sfgetpatlist
4990                                                      PARAMETERS mlist
4991                                                      * mlist is list of social secu
 => rity numbers, names
4992
4993                                                      EXTERNAL ARRAY mlist
4994
4995                                                      PRIVATE mchoice
4996                                                      IF ALEN(mlist) > 0
4997                                                        PRIVATE mbuttons
4998                                                        mchoice = 0
4999                                                        mbuttons = 0
5000                                                        =pushhelp("PATIENT VERIFICA
 => TION")                                  && rwd02
5001                                                        DO wp8.spr
 =>                                                  && h110
5002                                                        =pophelp()
5003                                                        mfinal = ""
5004                                                        DO CASE
5005                                                        CASE mbuttons = 1
5006                                                            i = mchoice
5007                                                            IF i > 0
5008                                                                mrec = VAL(dp(mlist[i
 => ,2],0,2))
5009                                                                mfinal = getpattemp(m
 => rec)
5010                                                            ENDIF
5011                                                        CASE mbuttons = 2
```

```
5012                                                    mfinal = ""
5013                                                 ENDCASE
5014                                              ENDIF
5015 <-------------------------------------------RETURN mfinal
5016
5017                                              *---------------------------
 => ------------------------
5018                                              *
5019                                              * Output sf600 Form
5020                                              *
5021                                              *---------------------------
 => ------------------------
5035 *!*****************************************************************
5036 *!
5037 *!      Procedure: SFPRINT
5038 *!
5039 *!      Called by: _Q710L17F7()        (function  in WP7.SPR)
5040 *!
5041 *!           Uses: SF600TEX.DBF
5042 *!
5043 *!        Indexes: SF600TEX.IDX
5044 *!
5045 *!   Report Forms: SF600.FRX
5046 *!
5047 *!*****************************************************************
5035                                           PROCEDURE sfprint
5036                                              PARAMETER mprintfile, mafstr
5037                                              PRIVATE msel, mtext, mwidth, m
 => oldssn
5038                                              moldssn = msocsec
5039                                              msocsec = SUBSTR(msocsec,1,3)
 => + "-" + SUBSTR(msocsec,4,2) + "-" + SUBSTR(msocsec,6,4)
5040                                              mtorg = mafstr
5041                                              msel = SELECT()
5042                                           IF !USED("SF600TEX")
5043                                              SELECT 0
5044                                              USE sf600tex INDEX sf600tex
5045                                           ELSE
5046                                              SELECT sf600tex
5047                                           ENDIF
5048                                              mwidth = SET("memowidth")
5049                                              SET MEMOWIDTH TO 65
5050                                              mlength = MEMLINES(mtorg)
5051                                              newline = CHR(13) + CHR(10)
5052                                              SET CONSOLE OFF
5053                                              SET PRINTER ON
5054                                           IF mprintfile != "PRINT"
5055                                              mafty = SET("SAFETY")
5056                                              SET PRINTER TO (mprintfile)
5057                                              ? "Encounter: "
5058                                              ?? mdateenc
5059                                              ?? mtimeenc
5060                                              ? "Patient:                "
5061                                              ?? "SSN: "
5062                                              ?? msocsec
5063                                              ?? "   "
5064                                              ?? TRIM(mlastname)
5065                                              ?? ","
5066                                              ?? TRIM(mfirstname)
5067                                              ?? "   "
5068                                              ?? "sex: "
5069                                              ?? IIF(msex = "F", "Female"
 => , "Male")
5070                                              ?? DTOC(mdatebirth)
5071                                              ?
5072                                              ? mtorg
```

```
5073                                              SET SAFETY &mafty
5074                                           ELSE
5075                                              pageno = 0
5076                                              mnext = 1
5077                                              mstop = IIF(mlength > 39, 3
 => 9, mlength)
5078                                              DO WHILE mnext < mlength
5079                                                 pageno = pageno + 1
5080                                                 mtext =IIF(pageno > 1, "
 => (Continue - SF600 )",")
5081                                                 mstart = mnext
5082                                                 mstop = IIF(mstart + 39
 => > mlength, mlength, mstart + 39)
5083                                                 FOR i = mstart TO mstop
5084                                                    mtext = IIF(EMPTY(mte
 => xt),"",mtext + newline) + MLINE(mtorg,i)
5085                                                 ENDFOR
5086                                                 REPORT FORM sf600.frx FO
 => R RECNO() = 1
5087                                                 EJECT
5088                                                 mnext = mstop + 1
5089                                              ENDDO
5090                                           ENDIF
5091                                           SET PRINTER OFF
5092                                           SET PRINTER TO
5093                                           SET CONSOLE ON
5094                                           USE
5095                                           SELECT (msel)
5096                                           SET MEMOWIDTH TO (mwidth)
5097                                           msocsec = moldssn
5098 <-------------------------------------------RETURN
5099                                           *---------------------------
 => ------------------------
5100                                           * Creat foxpro path
5101                                           *---------------------------
 => ------------------------
5111 *!*****************************************************************
5112 *!
5113 *!      Procedure: SETPATH
5114 *!
5115 *!      Called by: SYSINIT()          (function  in DEMO.PRG)
5116 *!                 : OPENSARS         (procedure in U.PRG)
5117 *!                 : PKUNZIP          (procedure in RESTORE.SPR)
5118 *!
5119 *!*****************************************************************
5111                                           PROCEDURE setpath
5112                                              PARAMETER newpath
5113                                              PRIVATE mnewpath
5114                                              IF !EMPTY(newpath)
5115                                                 mnewpath = FULLPATH(newpath
 => )
5116                                                 mcurpath = IIF(EMPTY(SET("P
 => ATH")),FULLPATH(CURDIR()),SET("PATH"))
5117                                                 IF NOT mnewpath $ mcurpath
5118                                                    mcurpath = mcurpath + ";
 => " + mnewpath      && add same subdirectory to path
5119                                                    SET PATH TO (mcurpath)
5120                                                 ENDIF
5121                                              ENDIF
5122 <-------------------------------------------RETURN
5123                                           *---------------------------
 => ------------------------
5124                                           * Error handling
5125                                           *---------------------------
 => ------------------------
5135 *!*****************************************************************
```

```
5136  *|
5137  *|      Procedure: ERRHAND
5138  *|
5139  *|      Called by: OPENPAT            (procedure in U.PRG)
5140  *|
5141  *|          Calls: ERRMSG.PRG
5142  *|
5143  *|*****************************************************************
5135                                        PROCEDURE errhand
5136
5137                                          PARAMETER errmsg, quitflag
5138                                        ┌IF PARAMETER() = 0
5139  <------------------------------------------RETURN
5140                                        └ENDIF
5141                                        ┌IF PARAMETER() < 2
5142                                        │   quitflag = .F.
5143                                        └ENDIF
5144                                        ┌IF !EMPTY(errmsg)
5145                                        │   =errmsg(errmsg,3)
5146                                        └ENDIF
5147                                        ┌IF quitflag
5148  <------------------------------------------CANCEL
5149                                        └ENDIF
5150  <------------------------------------------RETURN
5151
5152                                          *--------------------------
=> ------------------------
5153                                          *
5154                                          * Find the location of the fil
=> e and file path will update into new path
5155                                          * if the file path has not in
=> the foxpro path yet.
5156                                          *
5157                                          *--------------------------
=> ------------------------
5169  *|*****************************************************************
5170  *|
5171  *|      Procedure: FILEFIND
5172  *|
5173  *|      Called by: KEYLOG.PRG
5174  *|              : OPENPAT            (procedure in U.PRG)
5175  *|              : OPENSAMS           (procedure in U.PRG)
5176  *|
5177  *|          Calls: DP.PRG
5178  *|
5179  *|*****************************************************************
5169                                        FUNCTION filefind
5170                                          PARAMETER mfilename
5171                                          PRIVATE mflag
5172                                          mflag = .T.
5173                                        ┌IF PARAMETER() = 0
5174  <------------------------------------------RETURN .F.
5175                                        └ENDIF
5176                                          ON ERROR RETURN .F.
5177                                        ┌IF !FILE(mfilename)
5178                                        │   mfile = dp(mfilename,".",1)
5179                                        │   mext = dp(mfilename,".",2)
5180                                        │   =LOCFILE(mfile,mext,"where
=> is the " + mfilename + "?")
5181                                        └ENDIF
5182                                          ON ERROR
5183  <------------------------------------------RETURN mflag
5184
5185                                          *--------------------------
5186                                          * Reindex all the file
5187                                          *--------------------------
```

```
5238  *|*****************************************************************
5239  *|
5240  *|      Procedure: REINDEX
5241  *|
5242  *|      Called by: CAMD.MPR
5243  *|              : CAMDTR.MPR
5244  *|
5245  *|          Calls: POPUPSHOW()        (function  in U.PRG)
5246  *|              : CLOSEFILES         (procedure in U.PRG)
5247  *|              : POPUPHIDE()        (function  in U.PRG)
5248  *|
5249  *|           Uses: PATIENT.DBF
5250  *|              : ENCOUNTR.DBF
5251  *|              : SYMPTOMS.DBF
5252  *|              : DICT.DBF
5253  *|              : VAL.DBF                Alias: DICTVAL
5254  *|              : ENUM.DBF
5255  *|              : PREMISE.DBF
5256  *|              : ACTION.DBF
5257  *|              : RULE.DBF
5258  *|              : DISEASE.DBF
5259  *|              : DIAGNOSE.DBF
5260  *|              : REPORT.DBF
5261  *|              : SECTION.DBF
5262  *|              : SF600TEX.DBF
5263  *|
5264  *|        Indexes: PATIENT.IDX
5265  *|              : PATNAME.IDX
5266  *|              : PATID.IDX
5267  *|              : ENCOUNTR.IDX
5268  *|              : ENCOUNTD.IDX
5269  *|              : ENCOUNTI.IDX
5270  *|              : SYMPTOMS.IDX
5271  *|              : DICTID.IDX
5272  *|              : DICTNAME.IDX
5273  *|              : VAL.IDX
5274  *|              : ENUM.IDX
5275  *|              : PREMISE.IDX
5276  *|              : ACTION.IDX
5277  *|              : SALIENCE.IDX
5278  *|              : RULE.IDX
5279  *|              : DISEASE.IDX
5280  *|              : DISID.IDX
5281  *|              : DISAREA.IDX
5282  *|              : DIAGNOSE.IDX
5283  *|              : REPORT.IDX
5284  *|              : SECTION.IDX
5285  *|              : SF600TEX.IDX
5286  *|
5287  *|*****************************************************************
5238                                        FUNCTION REINDEX
5239                                          PRIVATE ALL
5240                                        ┌IF EMPTY(WOUTPUT())
5241                                        │   winpop = popupshow("Reindex
=> ......")
5242                                        │   =closefiles()
5243                                        │   USE patient INDEX patient,
=> patname, patid
5244                                        │   REINDEX COMPACT
5245                                        │   USE encountr INDEX encountr
=> , encountd, encount
5246                                        │   REINDEX COMPACT
5247                                        │   USE symptoms INDEX symptoms
5248                                        │   REINDEX COMPACT
5249                                        │   USE dict INDEX dictid, dict
=> name
```

```
5250                                    REINDEX COMPACT
5251                                    USE VAL INDEX VAL
5252                                    REINDEX COMPACT
5253                                    USE enum INDEX enum
5254                                    REINDEX COMPACT
5255                                    USE premise INDEX premise
5256                                    REINDEX COMPACT
5257                                    USE action INDEX action
5258                                    REINDEX COMPACT
5259                                    USE rule INDEX salience, ru
 => le
5260                                    REINDEX COMPACT
5261                                    USE disease INDEX disease,
 => disid, disaree
5262                                    REINDEX COMPACT
5263                                    USE diagnose INDEX diagnose
5264                                    REINDEX COMPACT
5265                                    USE REPORT INDEX REPORT
5266                                    REINDEX COMPACT
5267                                    USE SECTION INDEX SECTION
5268                                    REINDEX COMPACT
5269                                    USE sf600tex INDEX sf600tex
5270                                    REINDEX COMPACT
5271                                    =popuphide(winpop)
5272                                  ENDIF
5273                                  RETURN
5275                                  *: EOF: U.ACT
```

```
1     *
2     *
3     *        07/20/92                W2.SPR                    10:01:15
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                    Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_prn")
39       DEFINE WINDOW w_prn ;
40            FROM INT((SROW()-13)/2),INT((SCOL()-43)/2) ;
41            TO INT((SROW()-13)/2)+12,INT((SCOL()-43)/2)+42 ;
42            TITLE "Print Encounter" ;
43            FLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *                 W2 Setup Code - SECTION 2
54    *
55    *
56    *
57
58    #REGION 1
59    PRIVATE mchoice, mprintfile, mfilename, i
60    mchoice = 1
61    mfilename = "report.txt"
62    mprintfile = mfilename
63
64    *
65    *
66    *                    W2 Screen Layout
67    *
68    *
69    *
70
71    #REGION 1
72    IF WVISIBLE("w_prn")
73       ACTIVATE WINDOW w_prn SAME
74    ELSE
75       ACTIVATE WINDOW w_prn NOSHOW
76    ENDIF
77    @ 0,0 TO 3,40
78    @ 5,0 TO 8,40
79    @ 6,1 SAY "Filename:"
80    @ 6,11 GET mprintfile ;
81         SIZE 1,29 ;
82         DEFAULT " " ;
83         VALID _q7i0lh8ot()
84    @ 10,9 GET mbuttons ;
85         PICTURE "@*HT OK;\?Cancel" ;
86         SIZE 1,8,5 ;
87         DEFAULT 1 ;
88         VALID _q7i0lh8wd()
89    @ 1,24 GET mchoice ;
90         PICTURE "@*RVN File;Printer" ;
91         SIZE 1,11,0 ;
92         DEFAULT 1 ;
93         VALID _q7i0lh94q()
94    @ 1,1 SAY "Print Encounter to:"
95
96    IF NOT WVISIBLE("w_prn")
97       ACTIVATE WINDOW w_prn
98    ENDIF
99
100   READ CYCLE MODAL
101
102   RELEASE WINDOW w_prn
103
104   #REGION 0
105   IF m.talkstat = "ON"
106      SET TALK ON
107   ENDIF
108   IF m.compstat = "ON"
109      SET COMPATIBLE ON
110   ENDIF
111
112
113   *
114   *
115   *        _Q7I0LH8OT            mprintfile VALID
116   *
117   *        Function Origin:
118   *
119   *        From Screen:          W2,     Record Number:    5
120   *        Variable:             mprintfile
121   *        Called By:            VALID Clause
122   *        Object Type:          Field
123   *        Snippet Number:       1
124   *
125   *
126   *
127   FUNCTION _q7i0lh8ot      && mprintfile VALID
128   #REGION 1
129   mfilename = mprintfile
130   RETURN .T.
131
132   *
```

```
133     *
134     *          _q7IOLH8WD            mbuttons VALID
135     *
136     *          Function Origin:
137     *
138     *          From Screen:          W2,     Record Number:    6
139     *          Variable:             mbuttons
140     *          Called By:            VALID Clause
141     *          Object Type:          Push Button
142     *          Snippet Number:       2
143     *
144     *
145     *
146     FUNCTION _q7iOlh8wd      && mbuttons VALID
147     #REGION 1
148     IF mbuttons = 1
149        = printenc(ALLTRIM(mprintfile))
150     ENDIF
151     RETURN .T.
152
153
154     *
155     *
156     *          _q7IOLH94Q            mchoice VALID
157     *
158     *          Function Origin:
159     *
160     *          From Screen:          W2,     Record Number:    7
161     *          Variable:             mchoice
162     *          Called By:            VALID Clause
163     *          Object Type:          Radio Button
164     *          Snippet Number:       3
165     *
166     *
167     *
168     FUNCTION _q7iOlh94q      && mchoice VALID
169     #REGION 1
170     IF mchoice = 1
171        mprintfile = mfilename
172         CUROBJ = OBJNUM(mprintfile)
173     ELSE
174        mprintfile = "PRINT"
175     ENDIF
176     SHOW GETS
177     RETURN .T.
179     *: EOF: W2.AC1
```

```
1    *
2    *
3    *        07/20/92              W3.SPR                10:01:25
4    *
5    *
6    *
7    *        Author's Name
8    *
9    *        Copyright (c) 1992 Company Name
10   *        Address
11   *        City,     Zip
12   *
13   *        Description:
14   *        This program was automatically generated by GENSCRN.
15   *
16   *
17
18
19   #REGION 0
20   REGIONAL m.currarea, m.talkstat, m.compstat
21
22   IF SET("TALK") = "ON"
23      SET TALK OFF
24      m.talkstat = "ON"
25   ELSE
26      m.talkstat = "OFF"
27   ENDIF
28   m.compstat = SET("COMPATIBLE")
29   SET COMPATIBLE FOXPLUS
30
31   *
32   *
33   *              Window definitions
34   *
35   *
36   *
37
38   IF NOT WEXIST("w_patlist")
39      DEFINE WINDOW w_patlist ;
40          FROM INT((SROW()-15)/2),INT((SCOL()-37)/2) ;
41          TO INT((SROW()-15)/2)+14,INT((SCOL()-37)/2)+36 ;
42          TITLE "Patients on File" ;
43          FLOAT ;
44          NOCLOSE ;
45          SHADOW ;
46          DOUBLE ;
47          COLOR SCHEME 5
48   ENDIF
49
50
51   *
52   *
53   *              W3 Setup Code - SECTION 2
54   *
55   *
56   *
57
58   #REGION 1
59   mchoice=1
60
61   *
62   *
63   *              W3 Screen Layout
64   *
65   *
66   *
67
68   #REGION 1
69   IF WVISIBLE("w_patlist")
70      ACTIVATE WINDOW w_patlist SAME
71   ELSE
72      ACTIVATE WINDOW w_patlist NOSHOW
73   ENDIF
74   @ 0,0 GET mchoice ;
75       PICTURE "@&T" ;
76       FROM mlist ;
77       SIZE 11,34 ;
78       DEFAULT 1 ;
79       COLOR SCHEME 6
80   @ 12,6 GET mbuttons ;
81       PICTURE "@*HT Select;\?Cancel" ;
82       SIZE 1,8,5 ;
83       DEFAULT 1
84
85   IF NOT WVISIBLE("w_patlist")
86      ACTIVATE WINDOW w_patlist
87   ENDIF
88
89   READ CYCLE MODAL
90
91   RELEASE WINDOW w_patlist
92
93   #REGION 0
94   IF m.talkstat = "ON"
95      SET TALK ON
96   ENDIF
97   IF m.compstat = "ON"
98      SET COMPATIBLE ON
99   ENDIF
101  *: EOF: W3.AC1
```

```
1     *
2     *
3     *        07/28/92              W4.SPR                  16:29:59
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *        Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_patedit")
39       DEFINE WINDOW w_patedit ;
40            FROM INT((SROW()-23)/2),INT((SCOL()-80)/2) ;
41            TO INT((SROW()-23)/2)+22,INT((SCOL()-80)/2)+79 ;
42            TITLE "Patient Details" ;
43            FLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *        W4 Screen Layout
54    *
55    *
56    *
57
58    #REGION 1
59    IF WVISIBLE("w_patedit")
60       ACTIVATE WINDOW w_patedit SAME
61    ELSE
62       ACTIVATE WINDOW w_patedit NOSHOW
63    ENDIF
64    @ 1,1 SAY "Social security"
65    @ 3,1 SAY "Last name"
66    @ 5,1 SAY "First name"
67    @ 7,1 SAY "Date of birth"
68    @ 9,1 SAY "Sex"
69    @ 11,1 SAY "Rate / rank"
70    @ 1,18 GET msocsec ;
71       SIZE 1,12 ;
72       DEFAULT " " ;
73       PICTURE "@R 999-99-9999" ;
74       DISABLE
75    @ 3,18 GET mlastname ;
76       SIZE 1,27 ;
77       DEFAULT " " ;
78       DISABLE
79    @ 5,18 GET mfirstname ;
80       SIZE 1,27 ;
81       DEFAULT " " ;
82       DISABLE
83    @ 7,18 GET mdatebirth ;
84       SIZE 1,8 ;
85       DEFAULT {  /  /  } ;
86       DISABLE
87    @ 9,18 GET msex ;
88       SIZE 1,1 ;
89       DEFAULT " " ;
90       PICTURE "@A!" ;
91       DISABLE
92    @ 11,18 GET mraterank ;
93       SIZE 1,28 ;
94       DEFAULT " " ;
95       DISABLE
96    @ 7,66 GET mbuttons ;
97       PICTURE "@*VT OK;New;\?Cancel" ;
98       SIZE 1,9,2 ;
99       DEFAULT 1 ;
100      VALID _pvalid()
101   @ 1,66 GET mebutton ;
102      PICTURE "@*HN Edit" ;
103      SIZE 1,6,1 ;
104      DEFAULT 1 ;
105      VALID _q7q0zd5ur()
106   @ 4,66 GET mbbutton ;
107      PICTURE "@*HN Browse" ;
108      SIZE 1,8,1 ;
109      DEFAULT 1 ;
110      VALID _q7q0zd65j()
111
112   IF NOT WVISIBLE("w_patedit")
113      ACTIVATE WINDOW w_patedit
114   ENDIF
115
116   READ CYCLE ;
117      DEACTIVATE _stopread('w_patedit')
118
119   RELEASE WINDOW w_patedit
120
121   #REGION 0
122   IF m.talkstat = "ON"
123      SET TALK ON
124   ENDIF
125   IF m.compstat = "ON"
126      SET COMPATIBLE ON
127   ENDIF
128
129
130   *
131   *
132   *        W4 Cleanup Code
```

```
133     *
134     *
135     *
136
137     #REGION 1
138     IF quitting
139        RELEASE WINDOW w_patedit
140     ENDIF
141
142     *
143     *
144     *        _q7Q0ZD5UR          mebutton VALID
145     *
146     *        Function Origin:
147     *
148     *        From Screen:        W4,     Record Number:   15
149     *        Variable:           mebutton
150     *        Called By:          VALID Clause
151     *        Object Type:        Push Button
152     *        Snippet Number:     1
153     *
154     *
155     *
156     FUNCTION _q7q0zd5ur      && mebutton VALID
157     #REGION 1
158     IF saving
159        saving = .F.
160        SHOW GETS disabled
161     ELSE
162        saving = .T.
163        SHOW GETS enabled
164     ENDIF
165     RETURN .T.
166
167     *
168     *
169     *        _q7Q0ZD65J          mbbuttor. VALID
170     *
171     *        Fu .tion Origin:
172     *
173     *        From Screen:        W4,     Record Number:   16
174     *        Variable:           mbbutton
175     *        Called By:          VALID Clause
176     *        Object Type:        Push Button
177     *        Snippet Number:     2
178     *
179     *
180     *
181     FUNCTION _q7q0zd65j      && mbbutton VALID
182     #REGION 1
183     DO m_patlist IN (startdir + "\" + "demo.prg")
184     SHOW GETS
185     RETURN .T.
187     *: EOF: W4.AC1
```

```
1     *
2     *
3     *        07/20/92              W5.SPR                  10:01:21
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_area")
39       DEFINE WINDOW w_area ;
40           FROM INT((SROW()-11)/2),INT((SCOL()-46)/2) ;
41           TO INT((SROW()-11)/2)+10,INT((SCOL()-46)/2)+45 ;
42           TITLE "Complaints" ;
43           FLOAT ;
44           NOCLOSE ;
45           SHADOW ;
46           DOUBLE ;
47           COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *                 W5 Setup Code - SECTION 2
54    *
55    *
56    *
57
58    #REGION 1
59    mchoice = 1
60
61    *
62    *
63    *                 W5 Screen Layout
64    *
65    *
66    *
67
68    #REGION 1
69    IF WVISIBLE("w_area")
70       ACTIVATE WINDOW w_area SAME
71    ELSE
72       ACTIVATE WINDOW w_area NOSHOW
73    ENDIF
74    @ 2,32 GET mbuttons ;
75        PICTURE "@*VT Select;\?Cancel" ;
76        SIZE 1,8,1 ;
77        DEFAULT 1
78    @ 0,1 GET mchoice ;
79        PICTURE "@&T" ;
80        FROM mlist ;
81        SIZE 9,29 ;
82        DEFAULT 1 ;
83        COLOR SCHEME 6
84
85    IF NOT WVISIBLE("w_area")
86       ACTIVATE WINDOW w_area
87    ENDIF
88
89    READ CYCLE MODAL
90
91    RELEASE WINDOW w_area
92
93    #REGION 0
94    IF m.talkstat = "ON"
95       SET TALK ON
96    ENDIF
97    IF m.compstat = "ON"
98       SET COMPATIBLE ON
99    ENDIF
101   *: EOF: W5.AC1
```

```
1   *
2   *
3   *        07/20/92                W6.SPR                    10:00:49
4   *
5   *
6   *
7   *        Author's Name
8   *
9   *        Copyright (c) 1992 Company Name
10  *        Address
11  *        City,     Zip
12  *
13  *        Description:
14  *        This program was automatically generated by GENSCRN.
15  *
16  *
17
18
19  #REGION 0
20  REGIONAL m.currarea, m.talkstat, m.compstat
21
22  IF SET("TALK") = "ON"
23      SET TALK OFF
24      m.talkstat = "ON"
25  ELSE
26      m.talkstat = "OFF"
27  ENDIF
28  m.compstat = SET("COMPATIBLE")
29  SET COMPATIBLE FOXPLUS
30
31  *
32  *
33  *                    Window definitions
34  *
35  *
36  *
37
38  IF NOT WEXIST("w_num")
39      DEFINE WINDOW w_num ;
40          FROM INT((SROW()-15)/2),INT((SCOL()-55)/2) ;
41          TO INT((SROW()-15)/2)+14,INT((SCOL()-55)/2)+54 ;
42          TITLE "Numeric Data Entry" ;
43          FLOAT ;
44          NOCLOSE ;
45          SHADOW ;
46          DOUBLE ;
47          COLOR SCHEME 5
48  ENDIF
49
50
51  *
52  *
53  *                    W6 Screen Layout
54  *
55  *
56  *
57
58  #REGION 1
59  IF WVISIBLE("w_num")
60      ACTIVATE WINDOW w_num SAME
61  ELSE
62      ACTIVATE WINDOW w_num NOSHOW
63  ENDIF
64  @ 8,6 GET mval ;
65      SIZE 1,10 ;
66      RANGE mlo, mhi ;
67      DEFAULT 0 ;
68      PICTURE "@KZ"
69  @ 1,1 SAY mprompt ;
70      SIZE 3,51 ;
71      PICTURE "@T"
72  @ 6,2 TO 10,48
73  @ 5,23 SAY mhi ;
74      SIZE 1,10 ;
75      PICTURE "@BZ"
76  @ 8,30 SAY munits ;
77      SIZE 1,14 ;
78      PICTURE "@T"
79  @ 5,7 SAY mlo ;
80      SIZE 1,10 ;
81      PICTURE "@BZ"
82  @ 5,19 SAY "to"
83  @ 12,14 GET mbuttons ;
84      PICTURE "@*HT OK;\?Cancel" ;
85      SIZE 1,8,5 ;
86      DEFAULT 1
87  @ 5,1 SAY "Range"
88
89  IF NOT WVISIBLE("w_num")
90      ACTIVATE WINDOW w_num
91  ENDIF
92
93  READ CYCLE MODAL
94
95  RELEASE WINDOW w_num
96
97  #REGION 0
98  IF m.talkstat = "ON"
99      SET TALK ON
100 ENDIF
101 IF m.compstat = "ON"
102     SET COMPATIBLE ON
103 ENDIF
105 *: EOF: W6.AC1
```

```
1   *
2   *
3   *        07/20/92                W68.SPR                    10:00:58
4   *
5   *
6   *
7   *        Author's Name
8   *
9   *        Copyright (c) 1992 Company Name
10  *        Address
11  *        City,     Zip
12  *
13  *        Description:
14  *        This program was automatically generated by GENSCRN.
15  *
16  *
17
18
19  #REGION 0
20  REGIONAL m.currarea, m.talkstat, m.compstat
21
22  IF SET("TALK") = "ON"
23     SET TALK OFF
24     m.talkstat = "ON"
25  ELSE
26     m.talkstat = "OFF"
27  ENDIF
28  m.compstat = SET("COMPATIBLE")
29  SET COMPATIBLE FOXPLUS
30
31  *
32  *
33  *                    Window definitions
34  *
35  *
36  *
37
38  IF NOT WEXIST("w_check")
39     DEFINE WINDOW w_check ;
40          FROM INT((SROW()-17)/2),INT((SCOL()-63)/2) ;
41          TO INT((SROW()-17)/2)+16,INT((SCOL()-63)/2)+62 ;
42          TITLE "Single Choice Data Entry" ;
43          FLOAT ;
44          NOCLOSE ;
45          SHADOW ;
46          DOUBLE ;
47          COLOR SCHEME 5
48  ENDIF
49
50
51  *
52  *
53  *                    W68 Screen Layout
54  *
55  *
56  *
57
58  #REGION 1
59  IF WVISIBLE("w_check")
60     ACTIVATE WINDOW w_check SAME
61  ELSE
62     ACTIVATE WINDOW w_check NOSHOW
63  ENDIF
64  @ 14,20 GET mbuttons ;
65      PICTURE "@*HT OK;\?Cancel" ;
66      SIZE 1,8,5 ;
67      DEFAULT 1
68  @ 1,1 SAY mprompt ;
69      SIZE 3,59 ;
70      PICTURE "@f"
71  @ 5,1 GET mchoice ;
72      PICTURE "@&T" ;
73      FROM choices ;
74      SIZE 8,59 ;
75      DEFAULT 1 ;
76      COLOR SCHEME 6
77
78  IF NOT WVISIBLE("w_check")
79     ACTIVATE WINDOW w_check
80  ENDIF
81
82  READ CYCLE MODAL
83
84  RELEASE WINDOW w_check
85
86  #REGION 0
87  IF m.talkstat = "ON"
88     SET TALK ON
89  ENDIF
90  IF m.compstat = "ON"
91     SET COMPATIBLE ON
92  ENDIF
94  *: EOF: W68.AC1
```

```
1     *
2     *
3     *        07/20/92              W6C.SPR                10:00:53
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_check")
39       DEFINE WINDOW w_check ;
40           FROM INT((SROW()-14)/2),INT((SCOL()-54)/2) ;
41           TO INT((SROW()-14)/2)+13,INT((SCOL()-54)/2)+53 ;
42           TITLE "Logical Data Entry" ;
43           FLOAT ;
44           NOCLOSE ;
45           SHADOW ;
46           DOUBLE ;
47           COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *                 W6C Screen Layout
54    *
55    *
56    *
57
58    #REGION 1
59    IF WVISIBLE("w_check")
60       ACTIVATE WINDOW w_check SAME
61    ELSE
62       ACTIVATE WINDOW w_check NOSHOW
63    ENDIF
64    @ 11,14 GET mbuttons ;
65        PICTURE "@*HT OK;\?Cancel" ;
66        SIZE 1,8,5 ;
67        DEFAULT 1
68    @ 1,1 SAY mprompt ;
69        SIZE 3,50 ;
70        PICTURE "@T"
71    @ 6,22 GET mchoice ;
72        PICTURE "@*RVN Yes;No" ;
73        SIZE 1,7,0 ;
74        DEFAULT 1
75    @ 4,13 TO 9,38
76
77    IF NOT WVISIBLE("w_check")
78       ACTIVATE WINDOW w_check
79    ENDIF
80
81    READ CYCLE MODAL
82
83    RELEASE WINDOW w_check
84
85    #REGION 0
86    IF m.talkstat = "ON"
87       SET TALK ON
88    ENDIF
89    IF m.compstat = "ON"
90       SET COMPATIBLE ON
91    ENDIF
93    *: EOF: W6C.AC1
```

```
1   *
2   *
3   *        07/20/92              W7.SPR                  10:00:40
4   *
5   *
6   *
7   *        Author's Name
8   *
9   *        Copyright (c) 1992 Company Name
10  *        Address
11  *        City,     Zip
12  *
13  *        Description:
14  *        This program was automatically generated by GENSCRN.
15  *
16  *
17
18
19  #REGION 0
20  REGIONAL m.currarea, m.talkstat, m.compstat
21
22  IF SET("TALK") = "ON"
23     SET TALK OFF
24     m.talkstat = "ON"
25  ELSE
26     m.talkstat = "OFF"
27  ENDIF
28  m.compstat = SET("COMPATIBLE")
29  SET COMPATIBLE FOXPLUS
30
31  *
32  *
33  *                 Window definitions
34  *
35  *
36  *
37
38  IF NOT WEXIST("w_disease")
39     DEFINE WINDOW w_disease ;
40          FROM INT((SROW()-16)/2),INT((SCOL()-39)/2) ;
41          TO INT((SROW()-16)/2)+15,INT((SCOL()-39)/2)+38 ;
42          TITLE "Diseases on File" ;
43          FLOAT ;
44          NOCLOSE ;
45          SHADOW ;
46          DOUBLE ;
47          COLOR SCHEME 5
48  ENDIF
49
50
51  *
52  *
53  *                 W7 Setup Code - SECTION 2
54  *
55  *
56  *
57
58  #REGION 1
59  mchoice = 1
60
61  *
62  *
63  *                 W7 Screen Layout
64  *
65  *
66  *
67
68  #REGION 1
69  IF WVISIBLE("w_disease")
70     ACTIVATE WINDOW w_disease SAME
71  ELSE
72     ACTIVATE WINDOW w_disease NOSHOW
73  ENDIF
74  @ 0,0 GET mchoice ;
75       PICTURE "@&T" ;
76       FROM mlist ;
77       SIZE 11,36 ;
78       DEFAULT 1 ;
79       COLOR SCHEME 6
80  @ 12,1 GET mbuttons ;
81       PICTURE "@*HT Select;View;\?Cancel" ;
82       SIZE 1,8,4 ;
83       DEFAULT 1
84
85  IF NOT WVISIBLE("w_disease")
86     ACTIVATE WINDOW w_disease
87  ENDIF
88
89  READ CYCLE MODAL
90
91  RELEASE WINDOW w_disease
92
93  #REGION 0
94  IF m.talkstat = "ON"
95     SET TALK ON
96  ENDIF
97  IF m.compstat = "ON"
98     SET COMPATIBLE ON
99  ENDIF
101 *: EOF: W7.AC1
```

```
1     *
2     *
3     *        08/03/92              W7A.SPR                16:06:10
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_dislook")
39       DEFINE WINDOW w_dislook ;
40            FROM INT((SROW()-19)/2),INT((SCOL()-40)/2) ;
41            TO INT((SROW()-19)/2)+18,INT((SCOL()-40)/2)+39 ;
42            TITLE "Disease/Treatment Reference" ;
43            NOFLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *                 W7A Setup Code - SECTION 2
54    *
55    *
56    *
57
58    #REGION 1
59    mchoice = 1
60
61    *
62    *
63    *                 W7A Screen Layout
64    *
65    *
66    *
67
68    #REGION 1
69    IF WVISIBLE("w_dislook")
70       ACTIVATE WINDOW w_dislook SAME
71    ELSE
72       ACTIVATE WINDOW w_dislook NOSHOW
73    ENDIF
74    @ 0,1 GET mchoice ;
75        PICTURE "@&T" ;
76        FROM mlist ;
77        SIZE 15,35 ;
78        DEFAULT 1 ;
79        VALID _q7w0yiii2() ;
80        COLOR SCHEME 6
81    @ 15,10 GET mbuttons ;
82        PICTURE "@*HT \!\<View;\<Quit" ;
83        SIZE 1,8,4 ;
84        DEFAULT 1
85
86    IF NOT WVISIBLE("w_dislook")
87       ACTIVATE WINDOW w_dislook
88    ENDIF
89
90    READ CYCLE
91
92    RELEASE WINDOW w_dislook
93
94    #REGION 0
95    IF m.talkstat = "ON"
96       SET TALK ON
97    ENDIF
98    IF m.compstat = "ON"
99       SET COMPATIBLE ON
100   ENDIF
101
102
103   *
104   *
105   *        _q7W0YIII2            mchoice VALID
106   *
107   *        Function Origin:
108   *
109   *        From Screen:          W7A,     Record Number:    2
110   *        Variable:             mchoice
111   *        Called By:            VALID Clause
112   *        Object Type:          List
113   *        Snippet Number:       1
114   *
115   *
116   *
117   FUNCTION _q7w0yiii2      && mchoice VALID
118   #REGION 1
119   mbuttons = 1
121   *: EOF: W7A.AC1
```

```
1   *
2   *
3   *        10/13/92              WB.PRG                  09:54:30
4   *
5   *
6   *
7   *        Roy dobbin & Hoa Ly
8   *
9   *        Copyright (c) 1992
10  *        Naval Health Research Center, code 22
11  *        P.O.Box 85122
12  *        San Diego, CA  92186-5122
13  *
14  *        Description:
15  *        This program was automatically generated by GENSCRN.
16  *
17  *
18
19
20  #REGION 0
21  REGIONAL m.currarea, m.talkstat, m.compstat
22
23  IF SET("TALK") = "ON"
24     SET TALK OFF
25     m.talkstat = "ON"
26  ELSE
27     m.talkstat = "OFF"
28  ENDIF
29  m.compstat = SET("COMPATIBLE")
30  SET COMPATIBLE FOXPLUS
31
32  m.currarea = SELECT()
33
34
35  *
36  *
37  *        Window definitions
38  *
39  *
40  *
41
42  IF NOT WEXIST("w_disview")
43     DEFINE WINDOW w_disview ;
44          FROM INT((SROW()-19)/2),INT((SCOL()-80)/2) ;
45          TO INT((SROW()-19)/2)+18,INT((SCOL()-80)/2)+79 ;
46          TITLE "Disease Description" ;
47          FLOAT ;
48          NOCLOSE ;
49          SHADOW ;
50          DOUBLE ;
51          COLOR SCHEME 5
52  ENDIF
53
54
55  *
56  *
57  *        WB Screen Layout
58  *
59  *
60  *
61
62  #REGION 1
63  IF WVISIBLE("w_disview")
64     ACTIVATE WINDOW w_disview SAME
65  ELSE
66     ACTIVATE WINDOW w_disview NOSHOW
67  ENDIF
68   @ 1,0 EDIT mhelp ;
69        SIZE 15,78,0 ;
70        DEFAULT " " ;
71        SCROLL;
72        NOMODIFY
73   @ 16,34 GET mbuttons ;
74        PICTURE "@*HT OK" ;
75        SIZE 1,8,4 ;
76        DEFAULT 1
77
78  IF NOT WVISIBLE("w_disview")
79     ACTIVATE WINDOW w_disview
80  ENDIF
81
82   READ CYCLE MODAL
83
84   RELEASE WINDOW w_disview
85   SELECT (m.currarea)
86
87
88   #REGION 0
89  IF m.talkstat = "ON"
90     SET TALK ON
91  ENDIF
92  IF m.compstat = "ON"
93     SET COMPATIBLE ON
94  ENDIF
96   *: EOF: WB.ACT
```

```
1     *
2     *
3     *        07/02/92              W9.SPR                  11:15:35
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_popmsg")
39       DEFINE WINDOW w_popmsg ;
40           FROM INT((SROW()-8)/2),INT((SCOL()-36)/2) ;
41           TO INT((SROW()-8)/2)+7,INT((SCOL()-36)/2)+35 ;
42           TITLE "Warning" ;
43           FLOAT ;
44           NOCLOSE ;
45           SHADOW ;
46           DOUBLE ;
47           COLOR SCHEME 7
48    ENDIF
49
50
51    *
52    *
53    *                 W9 Screen Layout
54    *
55    *
56    *
57
58    #REGION 1
59    IF WVISIBLE("w_popmsg")
60       ACTIVATE WINDOW w_popmsg SAME
61    ELSE
62       ACTIVATE WINDOW w_popmsg NOSHOW
63    ENDIF
64    @ 1,1 SAY errstr ;
65        SIZE 3,31
66    @ 5,14 GET mbuttons ;
67        PICTURE "@*HT OK" ;
68        SIZE 1,4,1 ;
69        DEFAULT 1
70
71    IF NOT WVISIBLE("w_popmsg")
72       ACTIVATE WINDOW w_popmsg
73    ENDIF
74
75    READ CYCLE MODAL
76
77    RELEASE WINDOW w_popmsg
78
79    #REGION 0
80    IF m.talkstat = "ON"
81       SET TALK ON
82    ENDIF
83    IF m.compstat = "ON"
84       SET COMPATIBLE ON
85    ENDIF
87    *: EOF: W9.AC1
```

```
1   *
2   *
3   *        07/20/92              W10.SPR                10:00:36
4   *
5   *
6   *
7   *        Author's Name
8   *
9   *        Copyright (c) 1992 Company Name
10  *        Address
11  *        City,     Zip
12  *
13  *        Description:
14  *        This program was automatically generated by GENSCRN.
15  *
16  *
17
18
19  #REGION 0
20  REGIONAL m.currarea, m.talkstat, m.compstat
21
22  IF SET("\ALK") = "ON"
23     SET TALK OFF
24     m.talkstat = "ON"
25  ELSE
26     m.talkstat = "OFF"
27  ENDIF
28  m.compstat = SET("COMPATIBLE")
29  SET COMPATIBLE FOXPLUS
30
31  *
32  *
33  *                Window definitions
34  *
35  *
36  *
37
38  IF NOT WEXIST(" q7i0lgdbl")
39     DEFINE WINDOW _q7i0lgdbl ;
40          FROM INT((SROW()-9)/2),INT((SCOL()-35)/2) ;
41          TO INT((SROW()-9)/2)+8,INT((SCOL()-35)/2)+34 ;
42          TITLE "Confirm" ;
43          FLOAT ;
44          CLOSE ;
45          SHADOW ;
46          COLOR SCHEME 1
47  ENDIF
48
49
50  *
51  *
52  *                W10 Screen Layout
53  *
54  *
55  *
56
57  #REGION 1
58  IF WVISIBLE(" q7i0lgdbl")
59     ACTIVATE WINDOW _q7i0lgdbl SAME
60  ELSE
61     ACTIVATE WINDOW _q7i0lgdbl NOSHOW
62  ENDIF
63  @ 1,1 SAY m.prompt ;
64      SIZE 4,31
65  @ 6,7 GET mbuttons ;
66      PICTURE "@*HT OK;Cancel" ;
67      SIZE 1,8,1 ;
68      DEFAULT 1
69
70  IF NOT WVISIBLE(" q7i0lgdbl")
71     ACTIVATE WINDOW _q7i0lgdbl
72  ENDIF
73
74  READ CYCLE MODAL
75
76  RELEASE WINDOW _q7i0lgdbl
77
78  #REGION 0
79  IF m.talkstat = "ON"
80     SET TALK ON
81  ENDIF
82  IF m.compstat = "ON"
83     SET COMPATIBLE ON
84  ENDIF
86  *: EOF: W10.AC1
```

```
1     *
2     *
3     *        07/20/92              W11.SPR                10:01:02
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_pend")
39       DEFINE WINDOW w_pend ;
40            FROM INT((SROW()-19)/2),INT((SCOL()-59)/2) ;
41            TO INT((SROW()-19)/2)+18,INT((SCOL()-59)/2)+58 ;
42            TITLE "Pending Encounters" ;
43            FLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *              W11 Setup Code - SECTION 2
54    *
55    *
56    *
57
58    #REGION 1
59    mchoice = 1
60
61    *
62    *
63    *                W11 Screen Layout
64    *
65    *
66    *
67
68    #REGION 1
69    IF WVISIBLE("w_pend")
70       ACTIVATE WINDOW w_pend SAME
71    ELSE
72       ACTIVATE WINDOW w_pend NOSHOW
73    ENDIF
74    @ 6,1 GET mchoice ;
75         PICTURE "@&N" ;
76         FROM mlist ;
77         SIZE 9,55 ;
78         DEFAULT 1 ;
79         COLOR SCHEME 6
80    @ 16,11 GET mbuttons ;
81         PICTURE "@*HT Resume;New;\?Cancel" ;
82         SIZE 1,8,5 ;
83         DEFAULT 1
84    @ 3,1 SAY "Date"
85    @ 1,1 SAY "Patient"
86    @ 1,10 SAY mkey ;
87         SIZE 1,10
88    @ 3,10 SAY mdate ;
89         SIZE 1,10
90
91    IF NOT WVISIBLE("w_pend")
92       ACTIVATE WINDOW w_pend
93    ENDIF
94
95    READ CYCLE MODAL
96
97    RELEASE WINDOW w_pend
98
99    #REGION 0
100   IF m.talkstat = "ON"
101      SET TALK ON
102   ENDIF
103   IF m.compstat = "ON"
104      SET COMPATIBLE ON
105   ENDIF
107   *: EOF: W11.AC1
```

```
1     *
2     *
3     *        10/13/92              W12.SPR                14:17:09
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_encount")
39       DEFINE WINDOW w_encount ;
40            FROM 1, 0 ;
41            TO 22,79 ;
42            TITLE "Patient Encounter" ;
43            NOFLOAT ;
44            NOCLOSE ;
45            DOUBLE ;
46            COLOR SCHEME 5
47    ENDIF
48
49    IF NOT WEXIST("w_warning")
50       DEFINE WINDOW w_warning ;
51            FROM 23, 0 ;
52            TO 24,79 ;
53            NOFLOAT ;
54            NOCLOSE ;
55            NONE ;
56            COLOR SCHEME 5
57    ENDIF
58
59
60    *
61    *
62    *                 W12 Screen Layout
63    *
64    *
65    *
66
67    #REGION 1
68    IF WVISIBLE("w_encount")
69       ACTIVATE WINDOW w_encount SAME
70    ELSE
71       ACTIVATE WINDOW w_encount NOSHOW
72    ENDIF
73    @ 0,5 GET me_dateenc ;
74          SIZE 1,8 ;
75          DEFAULT (  /  /  ) ;
76          DISABLE
77    @ 0,20 GET me_timeenc ;
78          SIZE 1,5 ;
79          DEFAULT " " ;
80          DISABLE
81    @ 0,34 GET me_state ;
82          SIZE 1,4 ;
83          DEFAULT " " ;
84          DISABLE
85    @ 0,52 GET me_providr ;
86          SIZE 1,24 ;
87          DEFAULT " " ;
88          DISABLE
89    @ 1,13 GET me_lastnam ;
90          SIZE 1,28 ;
91          DEFAULT " " ;
92          DISABLE
93    @ 1,49 GET me_firstna ;
94          SIZE 1,22 ;
95          DEFAULT " " ;
96          PICTURE "@!" ;
97          DISABLE
98    @ 2,4 GET me_ssn ;
99          SIZE 1,11 ;
100         DEFAULT " " ;
101         PICTURE "@R 999-99-9999" ;
102         DISABLE
103   @ 2,31 GET me_sex ;
104         SIZE 1,6 ;
105         DEFAULT " " ;
106         PICTURE "@!" ;
107         DISABLE
108   @ 2,47 GET me_age ;
109         SIZE 1,3 ;
110         DEFAULT " " ;
111         DISABLE
112   @ 4,0 GET mckcompl ;
113         PICTURE "@*HN \<Complaint" ;
114         SIZE 1,12,1 ;
115         DEFAULT 1 ;
116         VALID _editcompl()
117   @ 4,13 GET me_area ;
118         SIZE 1,24 ;
119         DEFAULT " " ;
120         DISABLE
121   @ 5,0 TO 13,38
122   @ 5,15 GET mckhandp ;
123         PICTURE "@*HN \<Symptom" ;
124         SIZE 1,9,1 ;
125         DEFAULT 1 ;
126         VALID _edithandp()
127   @ 6,1 GET mcondl ;
128         PICTURE "@&N" ;
129         FROM me_cond ;
130         SIZE 7,37 ;
131         DEFAULT 1 ;
132         VALID _editcond(mcondl) ;
```

```
133          COLOR SCHEME 6
134      @ 4,40 GET mckdiag ;
135          PICTURE "@*HN \<Diagnosis" ;
136          SIZE 1,11,1 ;
137          DEFAULT 1 ;
138          VALID _editdiag()
139      @ 4,52 GET me_diagnos ;
140          SIZE 1,24 ;
141          DEFAULT " " ;
142          DISABLE
143      @ 5,39 TO 13,77
144      @ 5,55 GET mckassist ;
145          PICTURE "@*HN \<Assist" ;
146          SIZE 1,8,1 ;
147          DEFAULT 1 ;
148          VALID _editassis()
149      @ 6,40 GET mdiagl ;
150          PICTURE "@&N" ;
151          FROM me_diag ;
152          SIZE 7,37 ;
153          DEFAULT 1 ;
154          VALID _editvdiag(mdiagl) ;
155          COLOR SCHEME 6
156      @ 14,0 TO 14,76
157      @ 14,33 GET mcktreat ;
158          PICTURE "@*HN \<Treatment" ;
159          SIZE 1,12,1 ;
160          DEFAULT 1 ;
161          VALID _edittreat()
162      @ 14,76 GET mtmax ;
163          PICTURE "@*IVN " ;
164          SIZE 1,1,1 ;
165          DEFAULT 0 ;
166          VALID _q9v0umcrg()
167      @ 14,76 SAY CHR(30) ;
168          SIZE 1,1
169      @ 15,0 EDIT me_treat ;
170          SIZE 3,77,0 ;
171          DEFAULT " " ;
172          SCROLL
173      @ 19,25 GET mbuttons ;
174          PICTURE "@*HT \<Print;\f\?Sa\<ve;Ca\<ncel" ;
175          SIZE 1,8,4 ;
176          DEFAULT 1 ;
177          VALID _evalid()
178      @ 14,75 GET mtnorm ;
179          PICTURE "@*IVN " ;
180          SIZE 1,1,1 ;
181          DEFAULT 0 ;
182          VALID _q9v0umd2m()
183      @ 14,75 SAY CHR(31) ;
184          SIZE 1,1
185      @ 0,43 SAY "Provider"
186      @ 1,8 SAY "Name"
187      @ 0,15 SAY "Time"
188      @ 0,0 SAY "Date"
189      @ 1,43 SAY "First"
190      @ 0,27 SAY "Status"
191      @ 1,0 SAY "Patient"
192      @ 5,35 GET msnorm ;
193          PICTURE "@*IVN " ;
194          SIZE 1,1,1 ;
195          DEFAULT 0 ;
196          VALID _q9v0umddi()
197      @ 5,36 GET msmax ;
198          PICTURE "@*IVN " ;
199          SIZE 1,1,1 ;
200          DEFAULT 0 ;
201          VALID _q9v0umdjj()
202      @ 5,75 GET manorm ;
203          PICTURE "@*IVN " ;
204          SIZE 1,1,1 ;
205          DEFAULT 0 ;
206          VALID _q9v0umdqf()
207      @ 5,76 GET mamax ;
208          PICTURE "@*IVN " ;
209          SIZE 1,1,1 ;
210          DEFAULT 0 ;
211          VALID _q9v0umdwg()
212      @ 2,0 SAY "SSN"
213      @ 2,27 SAY "Sex"
214      @ 2,43 SAY "Age"
215      @ 5,35 SAY CHR(31) ;
216          SIZE 1,1
217      @ 5,36 SAY CHR(30) ;
218          SIZE 1,1
219      @ 5,75 SAY CHR(31) ;
220          SIZE 1,1
221      @ 5,76 SAY CHR(30) ;
222          SIZE 1,1
223
224
225
226
227      *
228      *
229      *
230      *
231      *
232      *
233
234      #REGION 2
235      IF WVISIBLE("w_warning")
236          ACTIVATE WINDOW w_warning SAME
237      ELSE
238          ACTIVATE WINDOW w_warning NOSHOW
239      ENDIF
240      @ 0,0 SAY mprivateact ;
241          SIZE 2,80
242
243      IF NOT WVISIBLE("w_warning")
244          ACTIVATE WINDOW w_warning
245      ENDIF
246      IF NOT WVISIBLE("w_encount")
247          ACTIVATE WINDOW w_encount
248      ENDIF
249
250
251      READ CYCLE MODAL ;
252          VALID _q9v0ume80() ;
253          DEACTIVATE _stopread('w_encount')
254
255      RELEASE WINDOW w_encount
256      RELEASE WINDOW w_warning
257
258      #REGION 0
259      IF m.talkstat = "ON"
260          SET TALK ON
261      ENDIF
262      IF m.compstat = "ON"
263          SET COMPATIBLE ON
264      ENDIF
```

W12E Screen Layout

```
265
266
267     *
268     *                 W12 Cleanup Code
269     *
270     *
271     *
272     *
273
274     #REGION 1
275     IF quitting
276        RELEASE WINDOW w_encount
277        IF WEXIST('w_warning')
278           RELEASE WINDOW w_warning
279        ENDIF
280     ENDIF
281
282
283     *
284     *
285     *        _Q9V0UMCRG          mtmax VALID
286     *
287     *        Function Origin:
288     *
289     *        From Screen:        W12,     Record Number:   23
290     *        Variable:           mtmax
291     *        Called By:          VALID Clause
292     *        Snippet Number:     1
293     *
294     *
295     *
296     FUNCTION _q9v0umcrg      && mtmax VALID
297     #REGION 1
298     IF NOT WONTOP("w_treat")
299        DO w12c.spr
300        SHOW GETS
301     ENDIF
302
303
304     *
305     *
306     *        _Q9V0UMD2M          mtnorm VALID
307     *
308     *        Function Origin:
309     *
310     *        From Screen:        W12,     Record Number:   27
311     *        Variable:           mtnorm
312     *        Called By:          VALID Clause
313     *        Snippet Number:     2
314     *
315     *
316     *
317     FUNCTION _q9v0umd2m      && mtnorm VALID
318     #REGION 1
319     =RELEASE("w_treat")
320
321     *
322     *
323     *        _Q9V0UMDDI          msnorm VALID
324     *
325     *        Function Origin:
326     *
327     *        From Screen:        W12,     Record Number:   36
328     *        Variable:           msnorm
329     *        Called By:          VALID Clause
330     *        Snippet Number:     3
331     *
332     *
333     *
334     FUNCTION _q9v0umddi      && msnorm VALID
335     #REGION 1
336     =RELEASE("w_handp")
337
338     *
339     *
340     *        _Q9V0UMDJJ          msmax VALID
341     *
342     *        Function Origin:
343     *
344     *        From Screen:        W12,     Record Number:   37
345     *        Variable:           msmax
346     *        Called By:          VALID Clause
347     *        Snippet Number:     4
348     *
349     *
350     *
351     FUNCTION _q9v0umdjj      && msmax VALID
352     #REGION 1
353     IF NOT WONTOP("w_handp")
354        DO w12a.spr
355        SHOW GETS
356     ENDIF
357
358     *
359     *
360     *        _Q9V0UMDQF          mnorm VALID
361     *
362     *        Function Origin:
363     *
364     *        From Screen:        W12,     Record Number:   38
365     *        Variable:           mnorm
366     *        Called By:          VALID Clause
367     *        Snippet Number:     5
368     *
369     *
370     *
371     FUNCTION _q9v0umdqf      && mnorm VALID
372     #REGION 1
373     =RELEASE("w_assist")
374
375     *
376     *
377     *        _Q9V0UMDWG          mmax VALID
378     *
379     *        Function Origin:
380     *
381     *        From Screen:        W12,     Record Number:   39
382     *        Variable:           mmax
383     *        Called By:          VALID Clause
384     *        Snippet Number:     6
385     *
386     *
387     *
388     FUNCTION _q9v0umdwg      && mmax VALID
389     #REGION 1
390     IF NOT WONTOP("w_assist")
391        DO w12b.spr
392        SHOW GETS
393     ENDIF
394
395     *
396     *
```

```
397     *          _09V0UME80            Read Level Valid
398     *
399     *          Function Origin:
400     *
401     *          From Screen:          Multiple Screens
402     *          Called By:            READ Statement
403     *          Snippet Number:       7
404     *
405     *
406     *
407     FUNCTION _q9v0ume80      && Read Level Valid
408     *
409     * Valid Code from screen: W12
410     *
411     #REGION
412     IF WONTOP("w12e")
413        ACTIVATE WINDOW w_encount
414     ENDIF
416     *: EOF: W12.AC1
```

```
1     *
2     *
3     *        07/20/92              W13.SPR                10:01:36
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_undo")
39       DEFINE WINDOW w_undo ;
40            FROM INT((SROW()-12)/2),INT((SCOL()-32)/2) ;
41            TO INT((SROW()-12)/2)+11,INT((SCOL()-32)/2)+31 ;
42            TITLE "Undo Options" ;
43            FLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 1
48    ENDIF
49
50
51    *
52    *
53    *                W13 Screen Layout
54    *
55    *
56    *
57
58    #REGION 1
59    IF WVISIBLE("w_undo")
60       ACTIVATE WINDOW w_undo SAME
61    ELSE
62       ACTIVATE WINDOW w_undo NOSHOW
63    ENDIF
64    @ 1,2 GET mchoice ;
65        PICTURE "@&T" ;
66        FROM mped ;
67        SIZE 7,26 ;
68        DEFAULT 1 ;
69        COLOR SCHEME 2
70    @ 9,5 GET mbuttons ;
71        PICTURE "@*HT OK;\?Cancel" ;
72        SIZE 1,8,1 ;
73        DEFAULT 1
74
75    IF NOT WVISIBLE("w_undo")
76       ACTIVATE WINDOW w_undo
77    ENDIF
78
79    READ CYCLE MODAL
80
81    RELEASE WINDOW w_undo
82
83    #REGION 0
84    IF m.talkstat = "ON"
85       SET TALK ON
86    ENDIF
87    IF m.compstat = "ON"
88       SET COMPATIBLE ON
89    ENDIF
91    *: EOF: W13.AC1
```

```
1   *
2   *
3   *        07/20/92              W14.SPR                10:00:31
4   *
5   *
6   *
7   *        Author's Name
8   *
9   *        Copyright (c) 1992 Company Name
10  *        Address
11  *        City,     Zip
12  *
13  *        Description:
14  *        This program was automatically generated by GENSCRN.
15  *
16  *
17
18
19  #REGION 0
20  REGIONAL m.currarea, m.talkstat, m.compstat
21
22  IF SET("TALK") = "ON"
23     SET TALK OFF
24     m.talkstat = "ON"
25  ELSE
26     m.talkstat = "OFF"
27  ENDIF
28  m.compstat = SET("COMPATIBLE",
29  SET COMPATIBLE FOXPLUS
30
31  *
32  *
33  *                Window definitions
34  *
35  *
36  *
37
38  IF NOT WEXIST("w_list")
39     DEFINE WINDOW w_list ;
40          FROM INT((SROW()-14)/2),INT((SCOL()-52)/2) ;
41          TO INT((SROW()-14)/2)+13,INT((SCOL()-52)/2)+51 ;
42          TITLE "Change Data" ;
43          FLOAT ;
44          NOCLOSE ;
45          SHADOW ;
46          DOUBLE ;
47          COLOR SCHEME 1
48  ENDIF
49
50
51  *
52  *
53  *                W14 Screen Layout
54  *
55  *
56  *
57
58  #REGION 1
59  IF WVISIBLE("w_list")
60     ACTIVATE WINDOW w_list SAME
61  ELSE
62     ACTIVATE WINDOW w_list NOSHOW
63  ENDIF
64  @ 1,0 GET mchoice ;
65       PICTURE "@&T" ;
66       FROM mpad ;
67       SIZE 9,50 ;
68       DEFAULT 1 ;
69       COLOR SCHEME 2
70  @ 11,16 GET mbuttons ;
71       PICTURE "@*HT OK;\?Cancel" ;
72       SIZE 1,8,1 ;
73       DEFAULT 1
74
75  IF NOT WVISIBLE("w_list")
76     ACTIVATE WINDOW w_list
77  ENDIF
78
79  READ CYCLE MODAL
80
81  RELEASE WINDOW w_list
82
83  #REGION 0
84  IF m.talkstat = "ON"
85     SET TALK ON
86  ENDIF
87  IF m.compstat = "ON"
88     SET COMPATIBLE ON
89  ENDIF
91  *: EOF: W14.AC1
```

```
1     *
2     *
3     *        08/03/92              WP3.SPR                16:06:01
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                    Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("wp3")
39       DEFINE WINDOW wp3 ;
40            FROM INT((SROW()-16)/2),INT((SCOL()-41)/2) ;
41            TO INT((SROW()-16)/2)+15,INT((SCOL()-41)/2)+40 ;
42            TITLE "Patient in File" ;
43            NOFLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *                 WP3 Setup Code - SECTION 2
54    *
55    *
56    *
57
58    #REGION 1
59    mchoice=1
60
61    *
62    *
63    *                    WP3 Screen Layout
64    *
65    *
66    *
67
68    #REGION 1
69    IF WVISIBLE("wp3")
70       ACTIVATE WINDOW wp3 SAME
71    ELSE
72       ACTIVATE WINDOW wp3 NOSHOW
73    ENDIF
74    @ 0,2 GET mchoice ;
75        PICTURE "@&T" ;
76        FROM mlist ;
77        SIZE 12,34 ;
78        DEFAULT 1 ;
79        VALID _q7w0yica9() ;
80        COLOR SCHEME 6
81    @ 12,2 GET mbuttons ;
82        PICTURE "@*HT \<Select;\<Edit;\?\<Cancel" ;
83        SIZE 1,8,5 ;
84        DEFAULT 1
85
86    IF NOT WVISIBLE("wp3")
87       ACTIVATE WINDOW wp3
88    ENDIF
89
90    READ CYCLE MODAL
91
92    RELEASE WINDOW wp3
93
94    #REGION 0
95    IF m.talkstat = "ON"
96       SET TALK ON
97    ENDIF
98    IF m.compstat = "ON"
99       SET COMPATIBLE ON
100   ENDIF
101
102
103   *
104   *
105   *        _q7W0YICA9          mchoice VALID
106   *
107   *        Function Origin:
108   *
109   *        From Screen:        WP3,     Record Number:    2
110   *        Variable:           mchoice
111   *        Called By:          VALID Clause
112   *        Object Type:        List
113   *        Snippet Number:     1
114   *
115   *
116   *
117   FUNCTION _q7w0yica9      &&  mchoice VALID
118   #REGION 1
119   mbuttons = 1
121   *: EOF: WP3.AC1
```

```
1     *
2     *
3     *        07/20/92              WP4.SPR                 10:01:46
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *              Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("wp4")
39       DEFINE WINDOW wp4 ;
40            FROM INT((SROW()-13)/2),INT((SCOL()-56)/2) ;
41            TO INT((SROW()-13)/2)+12,INT((SCOL()-56)/2)+55 ;
42            TITLE "Encounter to Report" ;
43            NOFLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *              WP4 Screen Layout
54    *
55    *
56    *
57
58    #REGION 1
59    IF WVISIBLE("wp4")
60       ACTIVATE WINDOW wp4 SAME
61    ELSE
62       ACTIVATE WINDOW wp4 NOSHOW
63    ENDIF
64    @ 1,20 GET mssn ;
65        SIZE 1,12 ;
66        DEFAULT " " ;
67        PICTURE "@R 999-99-9999"
68    @ 1,3 SAY "Social Security:"
69    @ 3,3 SAY "Last Name:"
70    @ 5,3 SAY "First Name: "
71    @ 3,20 GET mlast ;
72        SIZE 1,31 ;
73        DEFAULT " "
74    @ 5,20 GET mfirst ;
75        SIZE 1,31 ;
76        DEFAULT " "
77    @ 7,3 SAY "Encounter date:"
78    @ 7,20 GET mdateenc ;
79        SIZE 1,10 ;
80        DEFAULT { / / }
81    @ 9,11 GET mbuttons ;
82        PICTURE "@*HT \!\<OK;\<Browse;\<Cancel" ;
83        SIZE 1,8,3 ;
84        DEFAULT 1
85
86    IF NOT WVISIBLE("wp4")
87       ACTIVATE WINDOW wp4
88    ENDIF
89
90    READ CYCLE MODAL
91
92    RELEASE WINDOW wp4
93
94    #REGION 0
95    IF m.talkstat = "ON"
96       SET TALK ON
97    ENDIF
98    IF m.compstat = "ON"
99       SET COMPATIBLE ON
100   ENDIF
101
102
103   *
104   *
105   *              WP4 Cleanup Code
106   *
107   *
108   *
109
110   #REGION 1
111   RETURN mbuttons
113   *: EOF: WP4.AC1
```

```
1     *
2     *
3     *        07/20/92                WP5.SPR                10:01:51
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("wp4")
39       DEFINE WINDOW wp4 ;
40            FROM INT((SROW()-17)/2),INT((SCOL()-59)/2) ;
41            TO INT((SROW()-17)/2)+16,INT((SCOL()-59)/2)+58 ;
42            NOFLOAT ;
43            NOCLOSE ;
44            SHADOW ;
45            DOUBLE ;
46            COLOR SCHEME 5
47    ENDIF
48
49
50    *
51    *
52    *                 WP5 Setup Code - SECTION 2
53    *
54    *
55    *
56
57    #REGION 1
58    mchoice = 1
59
60    *
61    *
62    *                 WP5 Screen Layout
63    *
64    *
65    *
66
67    #REGION 1
68    IF WVISIBLE("wp4")
69       ACTIVATE WINDOW wp4 SAME
70    ELSE
71       ACTIVATE WINDOW wp4 NOSHOW
72    ENDIF
73    @ 4,0 GET mchoice ;
74          PICTURE "@&N" ;
75          FROM mlist ;
76          SIZE 9,55 ;
77          DEFAULT 1 ;
78          COLOR SCHEME 6
79    @ 14,18 GET mbuttons ;
80          PICTURE "@*HT \<OK;\?Cancel" ;
81          SIZE 1,8,5 ;
82          DEFAULT 1
83    @ 2,0 SAY "Date"
84    @ 0,0 SAY "Patient"
85    @ 0,9 SAY mkey ;
86          SIZE 1,45
87    @ 2,9 SAY mdate ;
88          SIZE 1,10
89
90    IF NOT WVISIBLE("wp4")
91       ACTIVATE WINDOW wp4
92    ENDIF
93
94    READ CYCLE MODAL
95
96    RELEASE WINDOW wp4
97
98    #REGION 0
99    IF m.talkstat = "ON"
100      SET TALK ON
101   ENDIF
102   IF m.compstat = "ON"
103      SET COMPATIBLE ON
104   ENDIF
105
106
107   *
108   *
109   *                 WP5 Cleanup Code
110   *
111   *
112   *
113
114   #REGION 1
115   RETURN mchoice
117   *: EOF: WP5.AC1
```

```
1   *
2   *
3   *        07/20/92              WP6.SPR                10:01:55
4   *
5   *
6   *
7   *        Author's Name
8   *
9   *        Copyright (c) 1992 Company Name
10  *        Address
11  *        City,     Zip
12  *
13  *        Description:
14  *        This program was automatically generated by GENSCRN.
15  *
16  *
17
18
19  #REGION 0
20  REGIONAL m.currarea, m.talkstat, m.compstat
21
22  IF SET("TALK") = "ON"
23     SET TALK OFF
24     m.talkstat = "ON"
25  ELSE
26     m.talkstat = "OFF"
27  ENDIF
28  m.compstat = SET("COMPATIBLE")
29  SET COMPATIBLE FOXPLUS
30
31  *
32  *
33  *                Window definitions
34  *
35  *
36  *
37
38  IF NOT WEXIST("wp6")
39     DEFINE WINDOW wp6 ;
40          FROM INT((SROW()-20)/2),INT((SCOL()-77)/2) ;
41          TO INT((SROW()-20)/2)+19,INT((SCOL()-77)/2)+76 ;
42          TITLE "Report Editing" ;
43          NOFLOAT ;
44          NOCLOSE ;
45          SHADOW ;
46          DOUBLE ;
47          COLOR SCHEME 5
48  ENDIF
49
50
51  *
52  *
53  *                WP6 Screen Layout
54  *
55  *
56  *
57
58  #REGION 1
59  IF WVISIBLE("wp6")
60     ACTIVATE WINDOW wp6 SAME
61  ELSE
62     ACTIVATE WINDOW wp6 NOSHOW
63  ENDIF
64  @ 1,1 EDIT matr ;
65       SIZE 15,73,0 ;
66       DEFAULT " " ;
67       TAB ;
68       SCROLL
69  @ 17,23 GET wbuttons ;
70       PICTURE "@*HT \<Print;\<Save;\<Cancel" ;
71       SIZE 1,8,4 ;
72       DEFAULT 1
73
74  IF NOT WVISIBLE("wp6")
75     ACTIVATE WINDOW wp6
76  ENDIF
77
78  READ CYCLE MODAL
79
80  RELEASE WINDOW wp6
81
82  #REGION 0
83  IF m.talkstat = "ON"
84     SET TALK ON
85  ENDIF
86  IF m.compstat = "ON"
87     SET COMPATIBLE ON
88  ENDIF
90  *: EOF: WP6.AC1
```

```
1     *
2     *
3     *        07/20/92                WP7.SPR                10:02:00
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                    Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("wp7")
39       DEFINE WINDOW wp7 ;
40            FROM INT((SROW()-13)/2),INT((SCOL()-43)/2) ;
41            TO INT((SROW()-13)/2)+12,INT((SCOL()-43)/2)+42 ;
42            TITLE "Print Option" ;
43            FLOAT ;
44            NOCLOSE ;
45            SHADOW ;
46            DOUBLE ;
47            COLOR SCHEME 5
48    ENDIF
49
50
51    *
52    *
53    *                 WP7 Setup Code - SECTION 2
54    *
55    *
56    *
57
58    #REGION 1
59    PRIVATE mchoice, mprintfile, mfilename, i
60    mchoice = 1
61    mfilename = "sf600.txt"
62    mprintfile = mfilename
63
64    *
65    *
66    *                    WP7 Screen Layout
```



```
67    *
68    *
69    *
70
71    #REGION 1
72    IF WVISIBLE("wp7")
73       ACTIVATE WINDOW wp7 SAME
74    ELSE
75       ACTIVATE WINDOW wp7 NOSHOW
76    ENDIF
77    @ 0,0 TO 3,40
78    @ 5,0 TO 8,40
79    @ 6,1 SAY "Filename:"
80    @ 6,11 GET mprintfile ;
81       SIZE 1,29 ;
82       DEFAULT " " ;
83       VALID _q7i0li77m()
84    @ 10,9 GET mbuttons ;
85       PICTURE "@*HT OK;\?Cancel" ;
86       SIZE 1,8,5 ;
87       DEFAULT 1 ;
88       VALID _q7i0li7f7()
89    @ 1,24 GET mchoice ;
90       PICTURE "@*RVN File;Printer" ;
91       SIZE 1,11,0 ;
92       DEFAULT 1 ;
93       VALID _q7i0li7oj()
94    @ 1,1 SAY "Print Encounter to:"
95
96    IF NOT WVISIBLE("wp7")
97       ACTIVATE WINDOW wp7
98    ENDIF
99
100   READ CYCLE MODAL
101
102   RELEASE WINDOW wp7
103
104   #REGION 0
105   IF m.talkstat = "ON"
106      SET TALK ON
107   ENDIF
108   IF m.compstat = "ON"
109      SET COMPATIBLE ON
110   ENDIF
111
112
113   *
114   *
115   *        _Q7I0LI77M          mprintfile VALID
116   *
117   *        Function Origin:
118   *
119   *        From Screen:        WP7,     Record Number:    5
120   *        Variable:           mprintfile
121   *        Called By:          VALID Clause
122   *        Object Type:        Field
123   *        Snippet Number:     1
124   *
125   *
126   *
127   FUNCTION _q7i0li77m      &&  mprintfile VALID
128   #REGION 1
129   mfilename = mprintfile
130   RETURN .T.
131
132   *
```

```
133     *
134     *          _Q7IOLI7F7          mbuttons VALID
135     *
136     *          Function Origin:
137     *
138     *          From Screen:        WP7,     Record Number:    6
139     *          Variable:           mbuttons
140     *          Called By:          VALID Clause
141     *          Object Type:        Push Button
142     *          Snippet Number:     2
143     *
144     *
145     *
146     FUNCTION _q7i0li7f7      && mbuttons VALID
147     #REGION 1
148     IF mbuttons = 1
149        IF !EMPTY(mprintfile)
150           DO sfprint WITH mprintfile, mtext
151        ENDIF
152     ENDIF
153     RETURN
154
155     *
156     *
157     *          _Q7IOLI7DJ          mchoice VALID
158     *
159     *          Function Origin:
160     *
161     *          From Screen:        WP7,     Record Number:    7
162     *          Variable:           mchoice
163     *          Called By:          VALID Clause
164     *          Object Type:        Radio Button
165     *          Snippet Number:     3
166     *
167     *
168     *
169     FUNCTION _q7i0li7oj      && mchoice VALID
170     #REGION 1
171     IF mchoice = 1
172        mprintfile = mfilename
173        CUROBJ = OBJNUM(mprintfile)
174     ELSE
175        mprintfile = "PRINT"
176     ENDIF
177     SHOW GETS
178     RETURN .T.
180     *: EOF: WP7.AC1
```

```
1   *
2   *
3   *        07/20/92                WP8.SPR                10:02:06
4   *
5   *
6   *
7   *        Author's Name
8   *
9   *        Copyright (c) 1992 Company Name
10  *        Address
11  *        City,     Zip
12  *
13  *        Description:
14  *        This program was automatically generated by GENSCRN.
15  *
16  *
17
18
19  #REGION 0
20  REGIONAL m.currarea, m.talkstat, m.compstat
21
22  IF SET("TALK") = "ON"
23     SET TALK OFF
24     m.talkstat = "ON"
25  ELSE
26     m.talkstat = "OFF"
27  ENDIF
28  m.compstat = SET("COMPATIBLE")
29  SET COMPATIBLE FOXPLUS
30
31  *
32  *
33  *                Window definitions
34  *
35  *
36  *
37
38  IF NOT WEXIST("wp3")
39     DEFINE WINDOW wp3 ;
40          FROM INT((SROW()-16)/2),INT((SCOL()-41)/2) ;
41          TO INT((SROW()-16)/2)+15,INT((SCOL()-41)/2)+40 ;
42          TITLE "Patient in File" ;
43          NOFLOAT ;
44          NOCLOSE ;
45          SHADOW ;
46          DOUBLE ;
47          COLOR SCHEME 5
48  ENDIF
49
50
51  *
52  *
53  *                WP8 Setup Code - SECTION 2
54  *
55  *
56  *
57
58  #REGION 1
59  mchoice=1
60
61  *
62  *
63  *                WP8 Screen Layout
64  *
65  *
66  *
67
68  #REGION 1
69  IF WVISIBLE("wp3")
70     ACTIVATE WINDOW wp3 SAME
71  ELSE
72     ACTIVATE WINDOW wp3 NOSHOW
73  ENDIF
74  @ 0,2 GET mchoice ;
75       PICTURE "@&T" ;
76       FROM mlist ;
77       SIZE 12,34 ;
78       DEFAULT 1 ;
79       COLOR SCHEME 6
80  @ 12,8 GET mbuttons ;
81       PICTURE "@*HT \<Select;\?\<Cancel" ;
82       SIZE 1,8,5 ;
83       DEFAULT 1
84
85  IF NOT WVISIBLE("wp3")
86     ACTIVATE WINDOW wm3
87  ENDIF
88
89  READ CYCLE MODAL
90
91  RELEASE WINDOW wp3
92
93  #REGION 0
94  IF m.talkstat = "ON"
95     SET TALK ON
96  ENDIF
97  IF m.compstat = "ON"
98     SET COMPATIBLE ON
99  ENDIF
101 *: EOF: WP8.AC1
```

```
1     *
2     *
3     *        07/06/92              W12A.SPR                10:46:49
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_handp")
39       DEFINE WINDOW w_handp ;
40            FROM INT((SROW()-21)/2),INT((SCOL()-80)/2) ;
41            TO INT((SROW()-21)/2)+20,INT((SCOL()-80)/2)+79 ;
42            NOFLOAT ;
43            NOCLOSE ;
44            NONE ;
45            COLOR SCHEME 5
46    ENDIF
47
48
49    *
50    *
51    *                 W12A Screen Layout
52    *
53    *
54    *
55
56    #REGION 1
57    IF WVISIBLE("w_handp")
58       ACTIVATE WINDOW w_handp SAME
59    ELSE
60       ACTIVATE WINDOW w_handp NOSHOW
61    ENDIF
62    @ 0,0 TO 20,79
63    @ 1,2 GET mcondl ;
64        PICTURE "@&N" ;
65        FROM me_cond ;
66        SIZE 19,76 ;
67        DEFAULT 1 ;
68        VALID _editcond(mcondl) ;
69        COLOR SCHEME 6
70    @ 0,36 GET mckhandp ;
71        PICTURE "@*HN \<Symptom" ;
72        SIZE 1,9,1 ;
73        DEFAULT 1 ;
74        VALID _edithandp()
75    @ 0,76 GET msnorm ;
76        PICTURE "@*IVN " ;
77        SIZE 1,1,1 ;
78        DEFAULT 0 ;
79        VALID _q740n3u9y()
80    @ 0,76 SAY CHR(31) ;
81        SIZE 1,1
82    @ 0,77 GET msmax ;
83        PICTURE "@*IVN " ;
84        SIZE 1,1,1 ;
85        DEFAULT 0 ;
86        VALID _q740n3uhf()
87    @ 0,77 SAY CHR(30) ;
88        SIZE 1,1
89
90    IF NOT WVISIBLE("w_handp")
91       ACTIVATE WINDOW w_handp
92    ENDIF
93
94    READ CYCLE MODAL
95
96    RELEASE WINDOW w_handp
97
98    #REGION 0
99    IF m.talkstat = "ON"
100      SET TALK ON
101   ENDIF
102   IF m.compstat = "ON"
103      SET COMPATIBLE ON
104   ENDIF
105
106
107   *
108   *
109   *        _Q740N3U9Y            msnorm VALID
110   *
111   *        Function Origin:
112   *
113   *        From Screen:          W12A,     Record Number:    5
114   *        Variable:             msnorm
115   *        Called By:            VALID Clause
116   *        Snippet Number:       1
117   *
118   *
119   *
120   FUNCTION _q740n3u9y      && msnorm VALID
121   #REGION 1
122   =RELEASE("w_handp")
123
124   *
125   *
126   *        _Q740N3UHF            msmax VALID
127   *
128   *        Function Origin:
129   *
130   *        From Screen:          W12A,     Record Number:    7
131   *        Variable:             msmax
132   *        Called By:            VALID Clause
```

```
*        Snippet Number:      2
*
*
*
FUNCTION _q740n3uhf      && msmax VALID
#REGION 1
IF NOT WONTOP("W12A")
   DO w12a.spr
ENDIF
*: EOF: W12A.AC1
```

```
1     *
2     *
3     *        07/06/92              W12B.SPR                  10:46:54
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                 Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_assist")
39       DEFINE WINDOW w_assist ;
40            FROM INT((SROW()-21)/2),INT((SCOL()-80)/2) ;
41            TO INT((SROW()-21)/2)+20,INT((SCOL()-80)/2)+79 ;
42            NOFLOAT ;
43            NOCLOSE ;
44            NONE ;
45            COLOR SCHEME 5
46    ENDIF
47
48
49    *
50    *
51    *                 W12B Screen Layout
52    *
53    *
54    *
55
56    #REGION 1
57    IF WVISIBLE("w_assist")
58       ACTIVATE WINDOW w_assist SAME
59    ELSE
60       ACTIVATE WINDOW w_assist NOSHOW
61    ENDIF
62    @ 0,0 TO 20,79
63    @ 0,77 GET menorm ;
64        PICTURE "@*IVN " ;
65        SIZE 1,1,1 ;
66        DEFAULT 0 ;
67        VALID _q740n3xwa()
68    @ 1,1 GET mdiagl ;
69        PICTURE "@&N" ;
70        FROM me_diag ;
71        SIZE 19,77 ;
72        DEFAULT 1 ;
73        VALID _editvdiag(mdiagl) ;
74        COLOR SCHEME 6
75    @ 0,38 GET mckassist ;
76        PICTURE "@*HN \<Assist" ;
77        SIZE 1,8,1 ;
78        DEFAULT 1 ;
79        VALID _editassis()
80    @ 0,78 GET memax ;
81        PICTURE "@*IVN " ;
82        SIZE 1,1,1 ;
83        DEFAULT 0 ;
84        VALID _q740n3y5o()
85    @ 0,78 SAY CHR(30) ;
86        SIZE 1,1
87    @ 0,77 SAY CHR(31) ;
88        SIZE 1,1
89
90    IF NOT WVISIBLE("w_assist")
91       ACTIVATE WINDOW w_assist
92    ENDIF
93
94    READ CYCLE MODAL
95
96    RELEASE WINDOW w_assist
97
98    #REGION 0
99    IF m.talkstat = "ON"
100      SET TALK ON
101   ENDIF
102   IF m.compstat = "ON"
103      SET COMPATIBLE ON
104   ENDIF
105
106
107   *
108   *
109   *        _Q740N3XWA            menorm VALID
110   *
111   *        Function Origin:
112   *
113   *        From Screen:          W12B,     Record Number:    3
114   *        Variable:             menorm
115   *        Called By:            VALID Clause
116   *        Snippet Number:       1
117   *
118   *
119   *
120   FUNCTION _q740n3xwa      && menorm VALID
121   #REGION 1
122   =RELEASE("w_assist")
123
124
125   *
126   *
127   *        _Q740N3Y5O            memax VALID
128   *
129   *        Function Origin:
130   *
131   *        From Screen:          W12B,     Record Number:    6
132   *        Variable:             memax
```

```
133     *          Called By:          VALID Clause
134     *          Snippet Number:     2
135     *
136     *
137     *
138     FUNCTION _q740n3y5o      && mamax VALID
139     #REGION 1
140     IF NOT WONTOP("w_assist")
141        DO w12b.spr
142     ENDIF
144     *: EOF: W12B.AC1
```

```
1     *
2     *
3     *        07/06/92              W12C.SPR                10:46:59
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    #REGION 0
20    REGIONAL m.currarea, m.talkstat, m.compstat
21
22    IF SET("TALK") = "ON"
23       SET TALK OFF
24       m.talkstat = "ON"
25    ELSE
26       m.talkstat = "OFF"
27    ENDIF
28    m.compstat = SET("COMPATIBLE")
29    SET COMPATIBLE FOXPLUS
30
31    *
32    *
33    *                    Window definitions
34    *
35    *
36    *
37
38    IF NOT WEXIST("w_treat")
39       DEFINE WINDOW w_treat ;
40           FROM INT((SROW()-21)/2),INT((SCOL()-80)/2) ;
41           TO INT((SROW()-21)/2)+20,INT((SCOL()-80)/2)+79 ;
42           TITLE "Treatment" ;
43           NOFLOAT ;
44           NOCLOSE ;
45           NONE ;
46           COLOR SCHEME 5
47    ENDIF
48
49
50    *
51    *
52    *                    W12C Screen Layout
53    *
54    *
55    *
56
57    #REGION 1
58    IF WVISIBLE("w_treat")
59       ACTIVATE WINDOW w_treat SAME
60    ELSE
61       ACTIVATE WINDOW w_treat NOSHOW
62    ENDIF
63    @ 0,0 TO 20,79
64    @ 2,2 EDIT me_treat ;
65        SIZE 17,76,0 ;
66        DEFAULT " " ;
```

```
67        SCROLL
68    @ 0,76 GET mtmax ;
69        PICTURE "@*IVN " ;
70        SIZE 1,1,1 ;
71        DEFAULT 0 ;
72        VALID _q740n41w1()
73    @ 0,75 GET mtnorm ;
74        PICTURE "@*IVN " ;
75        SIZE 1,1,1 ;
76        DEFAULT 0 ;
77        VALID _q740n423h()
78    @ 0,76 SAY CHR(30) ;
79        SIZE 1,1
80    @ 0,75 SAY CHR(31) ;
81        SIZE 1,1
82    @ 0,36 GET mcktreat ;
83        PICTURE "@*HN \<Treatment" ;
84        SIZE 1,12,1 ;
85        DEFAULT 1 ;
86        VALID _edittreat
87
88    IF NOT WVISIBLE("w_treat")
89       ACTIVATE WINDOW w_treat
90    ENDIF
91
92    READ CYCLE MODAL
93
94    RELEASE WINDOW w_treat
95
96    #REGION 0
97    IF m.talkstat = "ON"
98       SET TALK ON
99    ENDIF
100   IF m.compstat = "ON"
101      SET COMPATIBLE ON
102   ENDIF
103
104
105   *
106   *
107   *        _Q740N41W1            mtmax VALID
108   *
109   *        Function Origin:
110   *
111   *        From Screen:          W12C,     Record Number:    4
112   *        Variable:             mtmax
113   *        Called By:            VALID Clause
114   *        Snippet Number:       1
115   *
116   *
117   *
118   FUNCTION _q740n41w1      &&  mtmax VALID
119   #REGION 1
120   IF NOT WONTOP("w_treat")
121      DO w12c.spr
122   ENDIF
123
124
125   *
126   *
127   *        _Q740N423H            mtnorm VALID
128   *
129   *        Function Origin:
130   *
131   *        From Screen:          W12C,     Record Number:    5
132   *        Variable:             mtnorm
```

```
133     *          Called By:          VALID Clause
134     *          Snippet Number:     2
135     *
136     *
137     *
138     FUNCTION _q740n423h      && mtnorm VALID
139     #REGION 1
140     =RELEASE("w_treat")
142     *: EOF: W12C.AC1
```

```
 1   *
 2   *
 3   *        08/20/92              WP1.PRG                 16:00:07
 4   *
 5   *
 6   *
 7   *        Author's Name
 8   *
 9   *        Copyright (c) 1992 Company Name
10   *        Address
11   *        City,     Zip
12   *
13   *        Description:
14   *        This program was automatically generated by GENSCRN.
15   *
16   *
17
18
19   #REGION 0
20   REGIONAL m.currarea, m.talkstat, m.compstat
21
22   IF SET("TALK") = "ON"
23      SET TALK OFF
24      m.talkstat = "ON"
25   ELSE
26      m.talkstat = "OFF"
27   ENDIF
28   m.compstat = SET("COMPATIBLE")
29   SET COMPATIBLE FOXPLUS
30
31   m.currarea = SELECT()
32
33
34   *
35   *
36   *        Window definitions
37   *
38   *
39   *
40
41   IF NOT WEXIST("wp1")
42      DEFINE WINDOW wp1 ;
43           FROM INT((SROW()-11)/2),INT((SCOL()-57)/2) ;
44           TO INT((SROW()-11)/2)+10,INT((SCOL()-57)/2)+56 ;
45           TITLE "Patient Selection" ;
46           NOFLOAT ;
47           NOCLOSE ;
48           SHADOW ;
49           DOUBLE ;
50           COLOR SCHEME 5
51   ENDIF
52
53
54   *
55   *
56   *        WP1 Setup Code - SECTION 2
57   *
58   *
59   *
60
61   #REGION 1
62   msocsec = "          "
63   mlastname = ""
64   mfirstname = ""
65   mresult = ""
66
```

```
67   *
68   *
69   *        WP1 Screen Layout
70   *
71   *
72   *
73
74   #REGION 1
75   IF WVISIBLE("wp1")
76      ACTIVATE WINDOW wp1 SAME
77   ELSE
78      ACTIVATE WINDOW wp1 NOSHOW
79   ENDIF
80   @ 1,3 SAY "Social Security: "
81   @ 3,3 SAY "Lastname:"
82   @ 5,3 SAY "Firstname:"
83   @ 1,20 GET msocsec ;
84        SIZE 1,11 ;
85        DEFAULT " " ;
86        PICTURE "@R 999-99-9999"
87   @ 3,20 GET mlastname ;
88        SIZE 1,31 ;
89        DEFAULT " " ;
90        PICTURE "@!T"
91   @ 5,20 GET mfirstname ;
92        SIZE 1,31 ;
93        DEFAULT " " ;
94        PICTURE "@!T"
95   @ 8,9 GET mchoice ;
96        PICTURE "@*HN \!\<OK;\<BROWSE;\<CANCEL" ;
97        SIZE 1,10,3 ;
98        DEFAULT 1 ;
99        VALID _qb0yar6u()
100
101  IF NOT WVISIBLE("wp1")
102     ACTIVATE WINDOW wp1
103  ENDIF
104
105  READ CYCLE MODAL
106
107  RELEASE WINDOW wp1
108  SELECT (m.currarea)
109
110
111  #REGION 0
112  IF m.talkstat = "ON"
113     SET TALK ON
114  ENDIF
115  IF m.compstat = "ON"
116     SET COMPATIBLE ON
117  ENDIF
118
119
120  *
121  *
122  *        WP1 Cleanup Code
123  *
124  *
125  *
126
127  #REGION 1
128  DEACTIVATE WINDOW wp1
129  RELEASE WINDOW wp1
130  RETURN mresult
131
132
```

```
133
134     *
135     *
136     *        _Q8D0YAR6U          mchoice VALID
137     *
138     *        Function Origin:
139     *
140     *        From Screen:        WP1,     Record Number:    8
141     *        Variable:           mchoice
142     *        Called By:          VALID Clause
143     *        Object Type:        Push Button
144     *        Snippet Number:     1
145     *
146     *
147     *
148     FUNCTION _q8d0yar6u      &&  mchoice VALID
149     #REGION 1
150     DO CASE
151     CASE mchoice = 1
152        mresult = search(msocsec, mlastname, mfirstname)
153        IF !EMPTY(mresult)
154           CLEAR READ
155        ENDIF
156     CASE mchoice = 2
157        mresult = browsepat()
158        IF !EMPTY(mresult)
159           CLEAR READ
160        ENDIF
161     CASE mchoice = 3
162        mresult = ""
163        CLEAR READ
164     ENDCASE
165     RETURN mresult
166
168     *: EOF: WP1.ACT
```

```
1     *
2     *
3     *        08/20/92              WP2.PRG                13:56:56
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    *
20    *
21    *                WP2 Setup Code - SECTION 1
22    *
23    *
24    *
25
26    #REGION 1
27    PARAMETERS mflag, mitem
28    IF PARAMETERS() = 0
29       mflag = .T.
30    ENDIF
31    mnew = mflag
32    mnosams = .F.
33    mpatopen = .T.
34    u = "^"
35    mfinal = ""
36    IF PARAMETERS() < 2
37       mlast  = ""
38       mfirst = ""
39       mmidle = ""
40       mssn    = ""
41       mdob = CTOD("  /  /  ")
42       msx        = "M"
43       mrank  = ""
44    ELSE
45       mlast = dp(mitem,u,2)
46       mfirst = dp(mitem,u,3)
47       mmidle = dp(mitem,u,4)
48       mssn    = dp(mitem,u,1)
49       mdob = CTOD(dp(mitem,u,5))
50       msx  = dp(mitem,u,6)
51       msx = IIF(EMPTY(msx),"M",msx)
52       mrank  = dp(mitem,u,7)
53       mpatid = VAL(dp(mitem,u,8))
54       mrec = VAL(dp(mitem,u,9))
55    ENDIF
56
57    mlast = IIF(EMPTY(mlast),SPACE(25),mlast)
58    mfirst = IIF(EMPTY(mfirst),SPACE(25),mfirst)
59
60    moldssn = mssn
61    =pushhelp("PATIENT DETAILS")
62
63    #REGION 0
64    REGIONAL m.currarea, m.talkstat, m.compstat
65
66    IF SET("TALK") = "ON"
67       SET TALK OFF
68       m.talkstat = "ON"
69    ELSE
70       m.talkstat = "OFF"
71    ENDIF
72    m.compstat = SET("COMPATIBLE")
73    SET COMPATIBLE FOXPLUS
74
75    m.currarea = SELECT()
76
77
78    *
79    *
80    *                Window definitions
81    *
82    *
83    *
84
85    IF NOT WEXIST("wp2")
86       DEFINE WINDOW wp2 ;
87            FROM INT((SROW()-17)/2),INT((SCOL()-50)/2) ;
88            TO INT((SROW()-17)/2)+16,INT((SCOL()-50)/2)+49 ;
89            TITLE "Patient Enter/Edit" ;
90            NOFLOAT ;
91            NOCLOSE ;
92            SHADOW ;
93            DOUBLE ;
94            COLOR SCHEME 5
95    ENDIF
96
97
98    *
99    *
100   *                WP2 Screen Layout
101   *
102   *
103   *
104
105   #REGION 1
106   IF WVISIBLE("wp2")
107      ACTIVATE WINDOW wp2 SAME
108   ELSE
109      ACTIVATE WINDOW wp2 NOSHOW
110   ENDIF
111   @ 0,2 SAY "Social security"
112   @ 2,2 SAY "Last name"
113   @ 4,2 SAY "First name"
114   @ 8,2 SAY "Date of birth"
115   @ 10,2 SAY "Sex"
116   @ 12,2 SAY "Rate / rank"
117   @ 6,2 SAY "Middle Initial"
118   @ 0,19 GET mssn ;
119        SIZE 1,11 ;
120        DEFAULT " " ;
121        PICTURE "@R 999-99-9999" ;
122        VALID _q8d0twcz4()
123   @ 2,19 GET mlast ;
124        SIZE 1,27 ;
125        DEFAULT " " ;
126        PICTURE "@!T"
127   @ 4,19 GET mfirst ;
128        SIZE 1,27 ;
129        DEFAULT " " ;
130        PICTURE "@!T"
131   @ 6,19 GET mmidle ;
132        SIZE 1,1 ;
```

```
133         DEFAULT " " ;
134         PICTURE "@A!"
135     @ 8,19 GET mdob ;
136         SIZE 1,8 ;
137         DEFAULT (  /  /  ) ;
138         VALID _q8d0twdkt()
139     @ 10,19 GET msx ;
140         SIZE 1,1 ;
141         DEFAULT " " ;
142         PICTURE "@A!"
143     @ 12,19 GET mrank ;
144         SIZE 1,28 ;
145         DEFAULT " " ;
146         PICTURE "@!T"
147     @ 14,13 GET maction ;
148         PICTURE "@*HN \<OK;\<CANCEL" ;
149         SIZE 1,8,3 ;
150         DEFAULT 1 ;
151         VALID _q8d0twdv1()
152
153     IF NOT WVISIBLE("wp2")
154         ACTIVATE WINDOW wp2
155     ENDIF
156
157     READ CYCLE MODAL
158
159     RELEASE WINDOW wp2
160     SELECT (m.currarea)
161
162
163     #REGION 0
164     IF m.talkstat = "ON"
165         SET TALK ON
166     ENDIF
167     IF m.compstat = "ON"
168         SET COMPATIBLE ON
169     ENDIF
170
171
172     *
173     *                    WP2 Cleanup Code
174     *
175     *
176     *
177     *
178
179     #REGION 1
180     =pophelp()
181     DEACTIVATE WINDOW wp2
182     RELEASE WINDOW wp2
183     RETURN mfinal
184
185
186     *
187     *
188     *   _Q8D0TWCZ4          mssn VALID
189     *
190     *   Function Origin:
191     *
192     *   From Screen:        WP2,     Record Number:    9
193     *   Variable:           mssn
194     *   Called By:          VALID Clause
195     *   Object Type:        Field
196     *   Snippet Number:     1
197     *
198     *
199     *
200
201     FUNCTION _q8d0twcz4      && mssn VALID
202     #REGION 1
203     IF moldssn = mssn
204         mvalid = .T.
205     ELSE
206         msel = SELECT()
207         SELECT patient
208         SET ORDER TO 1
209         IF SEEK(mssn)
210             mvalid = .F.
211             mssn = moldssn
212             =popmessage("Duplicate Social Security Number")
213             CUROBJ = OBJNUM(mssn)
214             SHOW GETS
215         ELSE
216             mvalid = .T.
217             moldssn = mssn
218         ENDIF
219         SELECT (msel)
220     ENDIF
221     RETURN
222
223     *
224     *
225     *   _Q8D0TWDKT          mdob VALID
226     *
227     *   Function Origin:
228     *
229     *   From Screen:        WP2,     Record Number:   13
230     *   Variable:           mdob
231     *   Called By:          VALID Clause
232     *   Object Type:        Field
233     *   Snippet Number:     2
234     *
235     *
236     *
237     FUNCTION _q8d0twdkt      && mdob VALID
238     #REGION 1
239     IF mdob > DATE()
240         RETURN .F.
241     ENDIF
242
243     *
244     *
245     *   _Q8D0TWDV1          maction VALID
246     *
247     *   Function Origin:
248     *
249     *   From Screen:        WP2,     Record Number:   16
250     *   Variable:           maction
251     *   Called By:          VALID Clause
252     *   Object Type:        Push Button
253     *   Snippet Number:     3
254     *
255     *
256     *
257     FUNCTION _q8d0twdv1      && maction VALID
258     #REGION 1
259     DO CASE
260     CASE maction = 1
261         mfinal = ""
262         merr = .F.
263         mtssn = STRTRAN(mssn," ")
264         DO CASE
```

```
265         CASE LEN(ALLTRIM(mtssn)) != 9
266             = popmessage("Invalid Social Security !!")
267             _CUROBJ = OBJNUM(mssn)
268             merr = .T.
269             *       CASE empty(mdob)
=>                  && hll06 no require field
270             *               = popmessage("Please enter patient date of birth
=> ")
271             *               _CUROBJ = OBJNUM(mdob)
272             *               merr = .t.
273
274             *       CASE empty(msx)
275             *               = popmessage("Please enter patient sex")
276             *               _CUROBJ = OBJNUM(msex)
277             *               merr = .t.
278         ENDCASE
279         IF merr
280             SHOW GETS
281         ELSE
282             mfinal = combined(mssn,mlast,mfirst,mmidle,DTOC(mdob),msx,mrank
=> ,STR(mpatid),STR(mrec))
283             mfinal = savrec(mnew, mfinal)
284             CLEAR READ
285         ENDIF
286
287     CASE maction = 2
288         mfinal = ""
289         CLEAR READ
290     ENDCASE
291     RETURN mfinal
293     *: EOF: WP2.ACT
```

```
1     *
2     *
3     *        07/20/92              WP9.SPR                10:02:11
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    *
20    *
21    *                 WP9 Setup Code - SECTION 1
22    *
23    *
24    *
25
26    #REGION 1
27    PARAMETER msyswar
28    IF PARAMETER() = 0
29        RETURN
30    ENDIF
31    mtext = msyswar
32
33    #REGION 0
34    REGIONAL m.currarea, m.talkstat, m.compstat
35
36    IF SET("TALK") = "ON"
37       SET TALK OFF
38       m.talkstat = "ON"
39    ELSE
40       m.talkstat = "OFF"
41    ENDIF
42    m.compstat = SET("COMPATIBLE")
43    SET COMPATIBLE FOXPLUS
44
45    *
46    *
47    *                 Window definitions
48    *
49    *
50    *
51
52    IF NOT WEXIST("wp9")
53       DEFINE WINDOW wp9 ;
54           FROM 1, 0 ;
55           TO 6,76 ;
56           NOFLOAT ;
57           NOCLOSE ;
58           COLOR SCHEME 7
59    ENDIF
60
61
62    *
63    *
64    *                 WP9 Screen Layout
65    *
66    *
67    *
68
69    #REGION 1
70    IF WVISIBLE("wp9")
71       ACTIVATE WINDOW wp9 SAME
72    ELSE
73       ACTIVATE WINDOW wp9 NOSHOW
74    ENDIF
75    @ 0,1 SAY mtext ;
76        SIZE 4,73
77
78    IF NOT WVISIBLE("wp9")
79       ACTIVATE WINDOW wp9
80    ENDIF
81
82    READ CYCLE MODAL
83
84    RELEASE WINDOW wp9
85
86    #REGION 0
87    IF m.talkstat = "ON"
88       SET TALK ON
89    ENDIF
90    IF m.compstat = "ON"
91       SET COMPATIBLE ON
92    ENDIF
94    *: EOF: WP9.AC1
```

```
1     *
2     *
3     *        07/20/92            WPATS1.PRG                09:52:21
4     *
5     *
6     *
7     *        Author's Name
8     *
9     *        Copyright (c) 1992 Company Name
10    *        Address
11    *        City,     Zip
12    *
13    *        Description:
14    *        This program was automatically generated by GENSCRN.
15    *
16    *
17
18
19    *
20    *
21    *                WPATS Setup Code - SECTION 1
22    *
23    *
24    *
25
26      #REGION 1
27      PARAMETERS mlist, MESSAGE, mcriteria,msearch
28      IF PARAMETERS() = 0
29          RETURN ""
30      ENDIF
31      IF PARAMETERS() < 2
32          MESSAGE = "Select following candidate:"
33      ELSE
34          MESSAGE = IIF(LEN(MESSAGE)>60, SUBSTR(MESSAGE,1,60), MESSAGE)
35      ENDIF
36      mcriter = IIF(PARAMETER() < 3, "", mcriteria)
37      msearch = IIF(PARAMETER() < 4, 2, msearch)
38
39
40      #REGION 0
41      REGIONAL m.currarea, m.talkstat, m.compstat
42
43      IF SET("TALK") = "ON"
44          SET TALK OFF
45          m.talkstat = "ON"
46      ELSE
47          m.talkstat = "OFF"
48      ENDIF
49      m.compstat = SET("COMPATIBLE")
50      SET COMPATIBLE FOXPLUS
51
52      m.currarea = SELECT()
53
54
55    *
56    *
57    *                Window definitions
58    *
59    *
60    *
61
62      IF NOT WEXIST("wpats")
63          DEFINE WINDOW wpats ;
64              FROM 10, 11 ;
65              TO 23,70 ;
66              FLOAT ;
67              NOCLOSE ;
68              SHADOW ;
69              DOUBLE ;
70              COLOR SCHEME 5
71      ENDIF
72
73
74    *
75    *
76    *                WPATS Setup Code - SECTION 2
77    *
78    *
79    *
80
81      #REGION 1
82      * initial the push button to the SELECT button
83      maction = 1
84      EXTERNAL ARRAY mlist
85      mchoose = mlist[1]
86      u = "^"
87      mdata = ""
88
89
90    *
91    *
92    *                WPATS Screen Layout
93    *
94    *
95    *
96
97      #REGION 1
98      IF WVISIBLE("wpats")
99          ACTIVATE WINDOW wpats SAME
100     ELSE
101         ACTIVATE WINDOW wpats NOSHOW
102     ENDIF
103     @ 0,1 SAY MESSAGE ;
104         SIZE 1,56
105     @ 1,3 GET mchoose ;
106         PICTURE "@&N" ;
107         FROM mlist ;
108         SIZE 10,52 ;
109         DEFAULT 1 ;
110         VALID  _q7i0l5tb2() ;
111         COLOR SCHEME 6
112     @ 11,2 GET maction ;
113         PICTURE "@*HN \!\<Select;\<Edit;\<New;\<Search;\<Cancel" ;
114         SIZE 1,11,0 ;
115         DEFAULT 1 ;
116         VALID _q7i0l5tj5()
117
118     IF NOT WVISIBLE("wpats")
119         ACTIVATE WINDOW wpats
120     ENDIF
121
122     READ CYCLE MODAL ;
123         WHEN _q7i0l5txd()
124
125     RELEASE WINDOW wpats
126     SELECT (m.currarea)
127
128
129     #REGION 0
130     IF m.talkstat = "ON"
131         SET TALK ON
132     ENDIF
```

```
133     IF m.compatat = "ON"
134        SET COMPATIBLE ON
135     ENDIF
136
137
138     *
139     *                 WPATS Cleanup Code
140     *
141     *
142     *
143     *
144
145     #REGION 1
146     DEACTIVATE WINDOW wpats
147     RELEASE WINDOW wpat
148     RETURN mdata
149
150     ************************************************************************
=> *****
151     * This function return the result deliminated by "^"
152     ************************************************************************
=> *****
153     FUNCTION result
154     PARAMETER mitem
155     PRIVATE mpos
156
157     mnew = .F.
158     mpos = ASCAN(mlist, mitem)
159     IF mpos > 0
160        mfile = dp(mlist[mpos+1],u,1)
161        IF UPPER(mfile) $ "SAMS"
162           mnew = .T.
163        ENDIF
164        mtemp = mlist[mpos] + u + mlist[mpos+1]
165        mdata = getpatdata(mtemp)
166        IF mnew
167           mdata = savrec(mnew, mdata)
168        ENDIF
169     ELSE
170        mdata =  ""
171     ENDIF
172     RETURN mdata
173
174     ************************************************************************
=> *****
175     * This function call edit window
176     ************************************************************************
=> *****
177     PROCEDURE patedit
178     PARAMETER mitem
179     PRIVATE mpos
180     mpos = ASCAN(mlist,mitem)
181     IF mpos > 0
182        mfile = dp(mlist[mpos+1],u,1)
183        IF UPPER(mfile) $ "SAMS"
184           mnew = .T.
185        ELSE
186           mnew = .F.
187        ENDIF
188        mtemp = mlist[mpos] + u + mlist[mpos+1]
189        mtext = getpatdata(mtemp)
190        DEACTIVATE WINDOW wpats
191        RELEASE WINDOW wpats
192        mdata = wp2(mnew,mtext)
193     ELSE
194        mdata = ""
195     ENDIF
196     RETURN mdata
197
198
199     ************************************************************************
=> *****
200     * This function call new window
201     ************************************************************************
=> *****
202     PROCEDURE patnew
203     PRIVATE mpos,mnew
204     mnew = .T.
205     DEACTIVATE WINDOW wpats
206     RELEASE WINDOW wpats
207     mdata = wp2(mnew,mcriter)
208     RETURN mdata
209
210     *
211     *
212     *        _q7i0l5tb2          mchoose VALID
213     *
214     *        Function Origin:
215     *
216     *        From Screen:        WPATS,     Record Number:    3
217     *        Variable:           mchoose
218     *        Called By:          VALID Clause
219     *        Object Type:        List
220     *        Snippet Number:     1
221     *
222     *
223     *
224     *Start the READ on the list box
225     FUNCTION _q7i0l5tb2      &&  mchoose VALID
226     #REGION 1
227     _CUROBJ = OBJNUM(mchoose)
228
229
230     *
231     *
232     *        _q7i0l5tj5          maction VALID
233     *
234     *        Function Origin:
235     *
236     *        From Screen:        WPATS,     Record Number:    4
237     *        Variable:           maction
238     *        Called By:          VALID Clause
239     *        Object Type:        Push Button
240     *        Snippet Number:     2
241     *
242     *
243     *
244     * SELECT: return the select item
245     * EDIT: Let the user edit the text and return
246     * New: Enter the new patient
247     * CANCEL: QUIT
248
249     FUNCTION _q7i0l5tj5      &&  maction VALID
250     #REGION 1
251     DO CASE
252     CASE maction = 1
253        mpat = result(mchoose)
254        CLEAR READ
255     CASE maction = 2
256        mpat = patedit(mchoose)
257        CLEAR READ
258     CASE maction = 3
```

```
259        mpat = patnew()
260        CLEAR READ
261     CASE maction = 4
262        DO moresearch WITH mcriter, mlist
263        _CUROBJ = OBJNUM(mchoose)
264        mpat = ""
265        SHOW GETS
266        SHOW GET maction, 4 disabled
267     CASE maction = 5
268        mpat = ""
269        CLEAR READ
270     ENDCASE
271     RETURN mpat
272
273     *
274     *
275     *        _Q710L5TXD            Read Level When
276     *
277     *        Function Origin:
278     *
279     *        From Screen:          WPATS
280     *        Called By:            READ Statement
281     *        Snippet Number:       3
282     *
283     *
284     *
285     FUNCTION _q710l5txd      && Read Level When
286     *
287     * When Code from screen: WPATS
288     *
289     #REGION 1
290     DO CASE
291     CASE msearch = 0
292        SHOW GET maction enabled
293     CASE msearch = 1
294        SHOW GET maction, 4 disabled
295        SHOW GET maction, 1 disabled
296     CASE msearch = 2
297        SHOW GET maction, 4 disabled
298     ENDCASE
300     *: EOF: WPATS.ACT
```

```
 1  *:*****************************************************************
 2  *:
 3  *: Procedure file: C:\CAMD\WORK\YESNO.PRG
 4  *:
 5  *:          System: COMPUTER ASSISTED MEDICAL DIAGNOSIS
 6  *:          Author: ROY DOBBINS and HOA LY
 7  *:       Copyright (c) 1992, NAVAL HEALTH RESEARCH CENTER
 8  *:   Last modified: 11/26/91      2:50
 9  *:
10  *:          Set by: SFREPORT          (procedure in DEMO.PRG)
11  *:                : _Q9RORVE7X()      (function  in BACKUP.SPR)
12  *:                : _Q9RORVEV4()      (function  in BACKUP.SPR)
13  *:                : _Q9ROUD7U5()      (function  in RESTORE.SPR)
14  *:
15  *:       Documented 10/14/92 at 10:52              FoxDoc  version 2.10
16  *:*****************************************************************
17      *       ************************************************************
18      *       *                                                          *
19      *       * 11/26/91              YESNO.PRG                02:50:49 *
20      *       *                                                          *
21      *       ************************************************************
22      *       *                                                          *
23      *       * Adam Green                                               *
24      *       *                                                          *
25      *       * Copyright (c) 1991 Adam Green Seminars                   *
26      *       * One Faneuil Hall                                         *
27      *       * Boston, MA  02174                                        *
28      *       *                                                          *
29      *       * Description:                                             *
30      *       * This program was automatically generated by GENSCRN.     *
31      *       *                                                          *
32      *       ************************************************************
33
34
35      *       ************************************************************
36      *       *                                                          *
37      *       *              YESNO Setup Code - SECTION 1                *
38      *       *                                                          *
39      *       ************************************************************
40      *
41
42      #REGION 1
43      PARAMETERS MESSAGE, ok, CANCEL
44      PRIVATE ALL
45
46      *HLL, If message was empty print a plain window just only need respon
=> se
47      IF PARAMETERS() = 0
48         * Default to plain prompts
49         m.message = ""
50      ENDIF
51
52      * Ok and Cancel are used for prompts in the push buttons
53      IF PARAMETERS() = 1
54         * Default to normal prompts
55         m.ok = "OK"
56         m.cancel = "Cancel"
57      ENDIF
58
59      * Truncate any message longer than 50 characters
60      IF LEN( m.message ) > 50
61         * The message is centered in an @SAY
62         m.message = SUBSTR( m.message, 1, 50 )
63      ENDIF
64
65      PUSH KEY CLEAR
66
67      #REGION 0
68      REGIONAL m.currarea, m.talkstat, m.compstat
69
70      IF SET("TALK") = "ON"
71         SET TALK OFF
72         m.talkstat = "ON"
73      ELSE
74         m.talkstat = "OFF"
75      ENDIF
76      m.compstat = SET("COMPATIBLE")
77      SET COMPATIBLE FOXPLUS
78
79      *       ************************************************************
80      *       *                                                          *
81      *       *                     Window definitions                   *
82      *       *                                                          *
83      *       ************************************************************
84      *
85
86      IF NOT WEXIST("yesno")
87         DEFINE WINDOW yesno ;
88              FROM INT((SROW()-7)/2),INT((SCOL()-48)/2) ;
89              TO INT((SROW()-7)/2)+6,INT((SCOL()-48)/2)+47 ;
90              FLOAT ;
91              NOCLOSE ;
92              SHADOW ;
93              DOUBLE ;
94              COLOR SCHEME 5
95      ENDIF
96
97
98      *       ************************************************************
99      *       *                                                          *
100     *       *                   YESNO Screen Layout                    *
101     *       *                                                          *
102     *       ************************************************************
103     *
104
105     #REGION 1
106     IF WVISIBLE("yesno")
107        ACTIVATE WINDOW yesno SAME
108     ELSE
109        ACTIVATE WINDOW yesno NOSHOW
110     ENDIF
111     @ 1,2 SAY PADC( m.message, 40 ) ;
112          SIZE 1,41
113     @ 3,8 GET m.answer ;
114          PICTURE "@*HT \!\<&OK;\<&Cancel" ;
115          SIZE 1,12,5 ;
116          DEFAULT 1
117
118     IF NOT WVISIBLE("yesno")
119        ACTIVATE WINDOW yesno
120     ENDIF
121
122     READ CYCLE MODAL
123
124     RELEASE WINDOW yesno
125
126     #REGION 0
127     IF m.talkstat = "ON"
128        SET TALK ON
129     ENDIF
130     IF m.compstat = "ON"
131        SET COMPATIBLE ON
```

```
132   └ENDIF
133
134
135     *       ************************************************************
136     *       *                                                          *
137     *       *                    YESNO Cleanup Code                    *
138     *       *                                                          *
139     *       ************************************************************
140     *
141
142     #REGION 1
143     POP KEY
144
145     * Convert the numeric value of m.Answer from 1|2 to .T.|.F.
146     * If the user selected OK and didn't exit with Escape
147   ┌IF m.answer = 1 AND LASTKEY() <> 27
148 <────RETURN .T.
149   ├ELSE
150   │   * Cancel or Escape returns false
151 <────RETURN .F.
152   └ENDIF
154     *: EOF: YESNO.ACT
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATE COVERED
	October 1992	Final Sept 91- Oct 92

4. TITLE AND SUBTITLE
Computer Assisted Medical Diagnosis System Technical Manual

5. FUNDING NUMBERS
Program Element: 63706N
Work Unit Number:
M0095.005-6103

6. AUTHOR(S)
Ly, Hoa L.; Dobbins, Roy; Pearsall, Dianna M.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Health Research Center
P. O. Box 85122
San Diego, CA 92186-5122

8. PERFORMING ORGANIZATION Report No.
92-25

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Naval Medical Research and Development Command
National Naval Medical Center
Building 1, Tower 2
Bethesda, MD 20889-5044

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This technical manual contains the information on the program code, data elements, file structures needed to maintain the Computer Assisted Medical Diagnosis System. This documentation was created using the FOXDOC Version 2.0 program.

14. SUBJECT TERMS
Computer Assisted Medical Diagnosis Expert system
Technical
Medical Automated System

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited