DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

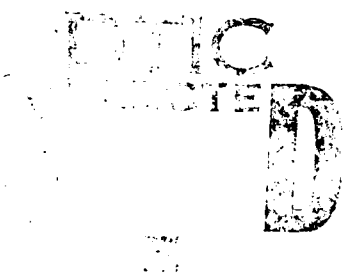
2

AFIT/GLM/LSM/88S-2

DEVELOPMENT OF A dBASE III PLUS DATABASE FOR OFFICE AUTOMATION WITHIN THE DEPARTMENT OF LOGISTICS MANAGEMENT, SCHOOL OF SYSTEMS AND LOGISTICS

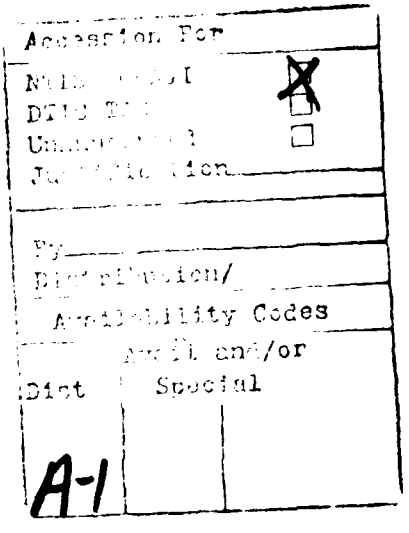
THESIS

John H. Barnes B.S.C.E
GS-0896-12, USAF
AFIT/GLM/LSM/88S-2



Approved for public release; distribution unlimited

The contents of the document are technically accurate, and no sensitive items, detrimental ideas, or deleterious information is contained therein. Furthermore, the views expressed in the document are those of the author and do not necessarily reflect the views of the School of Systems and Logistics, the Air University, the United States Air Force, or the Department of Defense.



AFIT/GLM/LSM/88S-2

DEVELOPMENT OF A dBASE III PLUS DATABASE FOR OFFICE AUTOMATION WITHIN THE DEPARTMENT OF LOGISTICS MANAGEMENT, SCHOOL OF SYSTEMS AND LOGISTICS

THESIS

Presented to the Faculty of the School of Systems and Logistics
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Logistics Management

John H. Barnes B.S.C.E
GS-0896-12, USAF

September 1988

Table of Contents

Abstract

The purpose of this thesis was to program a database system written in dBase III PLUS for use in AFIT/LSM. The system was to be hosted on a personal computer so that it could be transportable between the different computers in the office and conveniently used by as many personnel as possible. The program was intended to replace as much manual paperwork as possible and to provide a means to quickly access, sort, and to efficiently answer queries regarding the department faculty. The final result was intended to automate the routine office functions. This automation would relieve the department personnel from mundane duties so that they could concentrate their time on more productive pursuits.

The product of these efforts was the Integrated Faculty Information System (IFIS). The IFIS is a menu driven system that utilizes 19 databases that are interrelated to form one large database system. The database system has the capability to contain all the vital professional and demographic information on each member of the staff. The IFIS provides a means to input all the appropriate information and to modify preexisting information within the database system.

The IFIS has many checks to ensure that the users provide all the necessary information on specially designed and fully descriptive input screens. The input screens were also used to modify the data by typing over or making minor changes to existing data. Several allowances had been made to inform the user of possible mistakes while using the system. The users can access the information by developing reports to suit the need.

The IFIS does not represent a complete office automation system. There are many functions that could be added to the system to make it more usable. The expansion of IFIS capabilities would make a good subject for follow-on thesis research.

DEVELOPMENT OF A dBASE III PLUS DATA BASE FOR OFFICE AUTOMATION WITHIN THE DEPARTMENT OF LOGISTICS MANAGEMENT, SCHOOL OF SYSTEMS AND LOGISTICS

I. Introduction

General Issue

The Air Force Institute of Technology, School of Systems and Logistics, Department of Logistics Management has a requirement for the development of an office data base system. This system, titled the Integrated Faculty Information System (IFIS) has automated many of the manual record keeping responsibilities involving the professional information required for each of the faculty. The information contained in the IFIS is linked throughout the data bases using the faculties first and last names which are then keyed on the social security numbers to insure unique queries and replicable results. The nineteen data bases include areas such as personal and professional demographics, academic history, civic involvement, work experience, and course assignments, among others. A full list of the data bases is shown in appendix A. The data base program will be primarily menu-driven for ease of use and time savings. The user need only be prepared with the information on each faculty member and the system will prompt for the information when applicable and then the system will file the information consistently for easy recall.

Reports using the faculty information will be made by the user of the IFIS. The software package "R&R Relational Report Writer" (5), which is designed for use with the dBase III programming language, is one of the methods available to make reports. R&R provides more reporting capability as compared to the reporting capability provided with dBase III PLUS and allows for the development of new reports to the IFIS as the system matures.

The IFIS has been designed using structured programming, also known as modular programming (6:31). Top down programming is a technique that utilizes program modules in a systematic fashion. It is analogous to an underground building with many floors. The first programmed module in the system is comparable to the floor where the user enters the building. Sufficient information is provided to the user so that he may enter the appropriate compartment of the next lower level. Each lower level needs to get progressively more detailed in function and description so that the user does not get lost in the system and knows what options are available at that level. When the user has gone as deeply into the program as he desires, quick exits to higher levels should be provided. The many options that are available down into the program may be many and varied but should be completely self explanatory to the user. This top down programming technique allows for the addition of new program modules to the system by

the insertion of the new module in the appropriate section of the hierarchal system of menus.

Specific Problem

AFIT/LSM is using manual procedures using paper forms and bulky inconsistent filing to accomplish many tasks that could be easily automated. This causes the head of the department and other members of the his staff to spend excessive amounts of time in the preparation, filing, and maintenance of office information. More importantly, the laborious tasks required to build the many administrative reports through manual searching through piles of diverse forms consumes inordinate amounts of time. By automating some of these processes, the department personnel can manipulate information in seconds using the computer instead of spending hours in the gathering, collating, and maintenance of paper born information.

Investigative Questions

The development of a data base requires a structured approach to ensure there is a proper phasing of programming activities. The investigative questions defined below are require to accomplish this.

1. What are the routine administrative products of the office to be automated? This question poses the first step in the development of a data base. A variety of

forms have been identified by the for inclusion in the IFIS: AFIT Form 44, "Faculty Personnel"; AFIT Form 56, "Faculty Course Assignments"; AFIT Form 0-126, "Faculty Publications"; AFIT Form 127, "Faculty Presentations". These forms are incorporated into the database design. Other forms and data are being considered as the IFIS matures.

2. What are the administrative demands for the data base application? This includes usability factors such as the number and types of input and output products, the ability to quickly provide personally tailored answers to what-if questions, ease of use, and the ability to withstand inadvertent user abuse. Some of the more common concerns such as what staff members are due for an OER, or who is available/qualified to teach a specific course, etc. are available through the use of software such as R&R Relational Report Writer (5).

3. What, if any, are the new functions that the faculty and staff require of the data base? This is a "wish list" of ideas from the users. The newly requested functions of the data base are prioritized jointly by the programmer and the users to determine which are to be coded.

4. How is the system designed to accommodate the changing needs of the user? As described in the introduction, the system is designed in a modular

hierarchal format. If, for example, a new report format is to be installed into the system, then the report menu would be slightly modified to include the report in the display and then a new case would be added to tell where the report file is located. The new report file may then be programmed as a stand alone file with a simple feature added to return the processing back to the main program when the report is complete. The changes to the IFIS will require only a few minutes for an experienced dBase III programmer.

Glossary

This paper uses many common phrases that have a very specific meaning when discussing database system design. In an effort to limit confusion or incongruities this may cause, this glossary has been provided to define the meaning and usage of some of the important terminology. The words used in the definitions that are underlined are defined elsewhere in the glossary.

Database. A set of information with a common link among individual members of the set (3:339). A group of several small databases is sometimes referred to as a masterbase or a database system.

Data Dictionary. A preliminary list of fields within a database that form the structure of that database prior to actual programming.

Field. A single component in the structure of a database. In a database containing clients names and addresses, the first names of the clients would constitute a field. Each field can have a wide variety of attributes to meet the requirements of virtually any information.

Format Screen. A selected group of fields with descriptive documentation designed to be displayed using the full computer monitor screen for the purpose of requesting a single record of information from the user.

Index File. Provides a logical arrangement of the records in a database sorted according to the contents of a specific key field.

Key Field. The field whose contents are used in the arrangement of an index file. It is also known as an index key.

Keyword. A dBase III PLUS subcommand that typically implies a specific condition limits when used in conjunction with a more powerful dBase III PLUS command. For example, when using the @...GET command to have the user input a value, the RANGE keyword is used to limit the acceptable user inputs to only those falling in the specified range.

Menu. A list of linked tasks, normally using one full computer monitor screen, that provides the user with some control over the database system (3:339).

RAM. Random Access Memory. A computer's microprocessor can read information from RAM and store information to RAM. The information stored in RAM is lost when the computer is turned off. The available RAM represents a constraint to a computer's information processing capability and speed.

Record. One line of information in a database that contains one piece of information in each field. In a database of client's names and addresses, a record would be the name and address of one client.

Structure. The design of a database that is described by fields. The structure is initially determined by the database dictionary and implemented with a working database.

II. Methodology

Initial Development Concerns

This database is intended to replace a manual office system, therefore, a detailed study of the current paper system is the logical first step. The requirements of the manual system have been objectively examined. Capt Phillip Beard, et al. (2) completed a thorough review of the database requirements of AFIT/LSM and developed a comprehensive database dictionary. This data dictionary was used as the starting point in the development of the structure of the multiple databases used in the programming of the IFIS. Capt Beard's database was revised and edited slightly to improve programmability and the final product can be seen in Appendix A. This database layout does not presuppose that further modifications will not be necessary. During the maturation of the IFIS, it is likely that new and changing requirements will cause additional changes to the existing database design. Continuing efforts should be maintained by the using office to analyze system effectiveness and to monitor user suggestions to try and improve the useability of the database operations. Also, the users will identify new procedures to be included in the database that are not part of the original manual procedures. The growth and maturation of the database is expected to continue through the life of the IFIS.

The manual system employed by the AFIT/LSM office prior to the implementation of the IFIS is examined in three basic parts: input products, processing functions, and output products (4:3-4). The IFIS itself is similarly divided except that the processing functions are wholely internal to the system's programming.

Input Products

The user can input data into the database through several means. The users have the capability to update and correct the database through the use of menus and format screens. The input menus are designed to be self-explanatory and request the information in an obvious and systematic way. If properly designed, the IFIS should not present any surprises nor distractions to the user. Faculty and staff will have access and be able to make minor changes to the database at the discretion of the department head. The secretary and/or clerical assistants will input a majority of the information via a series of menus that maintain as many of the advantages of the manual system as possible within the confines of the dBase III PLUS programming environment.

A study of some of the paper products that are included in the database was undertaken early in the system design. Several standard AFIT forms were identified by the users for inclusion in this LSM database. These forms are

available on each member of the department and are filed by last name and first name. The IFIS also prompts the user for input on last name and first name but uses that person's social security number for record maintenance within the systems processing functions. Therefore, it is extremely important that the social security number for each person is correct and unambiguous. Also, the social security number must be consistently applied to each record of information in every database pertaining to that individual. Several checks have been installed into the system to help ensure that the social security numbers are adequately maintained. The old paper products that now form a part of the IFIS are listed below with a brief description of each.

AFIT Form 44 - Faculty Personnel. This form lists the societies and civic organizations in which each faculty member is involved. It also contains lists of academic institutions attended, professional positions held, professional development programs completed, and awards received.

AFIT Form 56 - Faculty Course Assignments. This form identifies courses each faculty member has taught and, if updated, is currently teaching. This form also lists the student theses in which each professor has advised or has acted as the technical reader.

AFIT Form 0-126 - Faculty Publications. This is a list

of each professor's publications.

AFIT Form 127 - Faculty Presentations. This form lists all presentations given by a professor and includes the subject of the presentation, the meeting title, and the time and place of the meeting.

The information from these forms will be stored in the database. This does not mean that the information will be stored in dBase III Plus files that are exact duplicates of the forms. This would waste disk space by having excessive duplication of information between database files. The database dictionary shown in appendix A will be developed to define the information in each file. The database dictionary will be explained in the next section.

Processing Functions

The manual system requires study to determine the flow of information through the office. This should identify areas such as the present and projected workflow as well as the processing schedule. A thorough understanding of how the information is entered, updated, manipulated and eventually printed is vital to the development of a comprehensive database (4:5). To gather the knowledge of the system, conversations have been held with the faculty to address some questions and concerns. Some important questions have been derived and solutions found in order to determine the optimum configuration of the IFIS.

Database Dictionary. The content of the databases was accomplished primarily through a study conducted by Capt Beard, et al. (2). The resulting database dictionary is a vital first step in the development and programming of a database system. The research must include a study of all the candidate information requested by the user of the database system. The information must be sifted with a knowledge of the database programming language's capabilities. An example of this involves the separation of the information that is permanently a part of the database record, such as an employee name, and that which can change regularly, such as the latest publication. It would be wasteful in both processing time and memory utilization to lug around large portions of permanent information when the changes to variable information must be made.

Another principle area of concern occurs when multiple records of information must be attached to a given employee. For example, it is expected that many of the faculty would have more than one publication. To account for this, all the information on each publication is placed in the GFPUBS.DBF (see Appendix A) which is then tied to the appropriate author through his or her social security number.

The culmination of these two programming concerns in the structuring of databases cause the programmer to use many

small databases. This is the case for the IFIS. The nineteen databases can each be traced to either variable data or multiple records to show why each is used as a separate database file. GFDEMO.DBF contains the demographics information on each employee that is considered permanent. GFEDHIST.DBF holds the employee's educational background that is permanent but may also contain multiple records in some instances, such as several undergraduate institutions attended or multiple majors at a given educational level. GFCOMITE.DBF shows the various faculty committees on which a member of the faculty sits, each requiring a separate record. As a final example, GFHIST.DBF encloses the academic posture of each of the faculty which needs occasional revision.

Faculty Size. The number of faculty members that are expected to be entered into the database is an viable concern. If the number of personnel were to be large, possibly in excess of one hundred, then the number of fields in some of the more complex databases should be reduced. The means to reduce the number of fields in a database would be to section it into smaller, more manageable pieces. All the fields would be maintained, just the number of databases would be increased.

The process of sectioning the structure of the databases allows a database program to be written so that the computer utilizes less computer memory at any given point

in the program by putting the minimum needed information into memory (3:34-6). This can be illustrated by examining the impact of USEing (a dBase III Plus command that is self explanatory) GFDEMO.DBF. This database uses approximately one kilobyte (1K) of Random Access Memory (RAM) for each filled record. Often there are ten or more files open (stored in RAM) at any given time during the processing of a database system such as IFIS (9:20). These include two for the disk operating system (DOS), IO.SYS and MSDOS.SYS (almost 41K combined) (7:2-2,4-88), two for dBase III PLUS (3:36), DBASE.EXE (138K) and DBASE.OVL (272K). These four files require the combined use of about 452K of RAM just to operate in dBase III PLUS. If this exceeds the available RAM on a system or if the size of the open databases causes the demand to exceed the available RAM, then the computer must read the excess capacity from the user's disk storage which radically slows processing speed. Programming to minimize the need to read from disks by maximizing the utilization of system RAM was considered a primary objective.

Information Gathering. Once the appropriate data needs have been established, then the programmer is faced with determining a means to obtain the information in a cohesive, unambiguous way from the user. The user should be separated from the rigors of programming and not be irritated with any unnecessary surprises. As Maj Mary K.

Allen pointed out (1), it is vital for the programmer to provide "graceful degradation" from the program operations. In simple terms, this means that catastrophic errors, errors that halt the program execution and leave the user with an unintelligible error message from dBase III PLUS, must be avoided. This requires the programmer to foresee all user responses and to warn the user of the error and to provide a means to correct the error. This often proves to be an elusive task.

In the development of the IFIS, the programming procedures to find and correct errors entailed the larger portion of the time it took to program the database maintenance function. Typically, the error checking is performed during or immediately after the requested data is input so that it forms a part of the active procedural files. dBase III PLUS offers a several good command options to do this. The one used most of in the IFIS is the @...GET command utilizing the PICTURE and RANGE keywords (8:13-16).

The @...GET command, when used with the READ command (8:390-393), is a tool the programmer has available to request information from the user. The PICTURE keyword (8:356-364) presents the user with a specific format in which to enter the information. An example of the usefulness of the PICTURE keyword is for the input of a social security number. The computer is highly sensitive

to the difference between a social security number with the two dashes in place as opposed to one with nine consecutive numbers or, thirdly, a numeric expression displaying a three digit number subtracting a two digit number minus a four digit number. To avoid confusion, the PICTURE keyword written

PICTURE "999-99-9999"

forces the two dashes in the correct location in a character expression. The 9's are in place so that only numbers can be input (alpha characters are ignored) and the user need not type the dashes. The RANGE keyword (8:388-390) is used with the @...GET command to identify upper and lower limits for a variable. This keyword does not hold for any character type variable. If the range were from 3 to 5 then the request would simply stay unchanged and apparently lifeless until the user input either 3, 4, or 5. It is the programmers responsibility to inform the user about this requirement. Again, this command is just one example of the programmer's first line of defense against errors that will either stop the program execution or cause the program to act upon unintended data that produces the wrong results.

The next line of defense against errors is the programmer's responsibility. The IFIS has four program files dedicated to informing the user of errors and providing the appropriate solutions. These programs are

presented with a brief explanation of their functions.

NOFIND.PRG. This program informs the user that the record requested is not in the main database, GFDEMO.DBF. The options available are either to enter the record into GFDEMO.DBF or to retype the request.

ENDFILE.PRG. This file states that the requested record is found in GFDEMO.DBF but it has no corresponding record in the database currently being queried. The user is asked if he would like to add the record to the database. If not, the program returns to a file called CASER.PRG.

ISBLANK.PRG. This file checks to see if the user left the social security number field blank. If so, he is warned of the severity of this oversight and is forced to input something into this field before proceeding.

NOADD.PRG. This program simply presents a screen that informs the user that his attempt to add a record directly to a database without a corresponding record existing in GFDEMO.DBF cannot be done. If the user is unsure whether a corresponding record exists in GFLEMO.DBF, he can attempt to add a record into another database and IFIS will check GFDEMO for a match and notify him with NOADD.PRG if no match is found. The user is then notified to update GFDEMO.PRG. This prevents the input of records that are not retrievable by IFIS.

These defenses against user errors do not exclude all

possibilities. There are ways to verify that all information placed in the records is according to the desires of the system manager and the programmer, but these extend beyond the needs of this application. To avoid inadvertent deletions of important information, it was decided that the removal of records from the databases cannot be done from within the IFIS. Deleting records must be done from the dBase III PLUS dot prompt. A function that should be incorporated into the IFIS is a check to ensure that all social security numbers contain nine uniquely combined numbers.

Output Products

Output that will be used to display and print the output products, known as reports, will be generated by the user. These reports may be written using R&R Relational Report Writer (5), a software package written to create dBase reports by Concentric Data Systems,Inc. These reports generated with R&R will require minimal effort by the user because the commands are in a succinct menu-driven format. The flexibility is driven by multiple menus and user queries that are as simple and direct as possible.

III. Technical Analysis

The technical analysis is a close look at the Integrated Faculty Information System's programs, format files, and indices. The emphasis is on the dBase III PLUS programming language code as it was used and the results of the code. The commentary will include the merits of some sections, the shortcomings on others and areas for improvement. To facilitate the maturation process of the IFIS, special attention will be paid to explain how the system can be modified.

The code has been commented and indented as is the standard programming practice. Also, the full name of the commands is used in many locations to promote readability. These techniques, though, are meaningless to the computer and even cause a significant amount of extra processing time. It is suggested that when modifications become few and the system becomes a daily tool, the master documented copy of the IFIS should be put in a safe storage location and the working copy should be stripped of all documentation and indentation. All commands longer than four letters should also be stripped to the four letter root commands. This stripping process will allow the IFIS to run at its peak efficiency (3:14-17) and make its utilization a more pleasurable and less time consuming experience.

All the files discussed in this chapter are listed in Appendices B, C, and D. The reader should refer to these appendices in order to find the code being discussed.

Database Modifications

It is anticipated that there will numerous changes to the databases' contents and structure during the system's infancy. The user's will find that some fields are too short while others may be too long. These problems will be simple to remedy. Other modifications such as subtracting fields or adding new fields may prove to be more involved, but will still be relatively easy to accomplish by a novice dBase III programmer. This section contains the necessary procedures to make the anticipated modifications to the IFIS databases. However, one point must be emphasized, under NO conditions should anyone attempt to change the names of any of the databases nor the names of any of the format files accompanying the databases! Changing database names or attempting to add or subtract entire databases should be done only by an experienced dBase III Plus programmer in order to preclude the inadvertent erasure of valuable office information.

Modifying Database Fields. Most of the database fields within the IFIS can be modified with no disruption of the programming functions. However, there may be some adverse consequences experienced on specific reports generated that

attempt to call a field that has been changed. All reports built into IFIS should be carefully examined and the system manager should be notified prior to making the changes. There are three specific fields that should not be changed. In the GFDEMO.DBF database, the FIRST_NAME field and the LAST_NAME field should not be changed. Also, the SSAN field should not be modified in any of the databases.

With the above conditions in mind, the character type and width parameters of the remaining fields may be modified using the following procedure. The dBase III PLUS program should be entered and the ASSIST Section should be bypassed by pressing the ESC key. At the dot prompt, the user must tell the computer which database he wishes to modify with the

USE dbname

command (8:602-607). The word dbname should be substituted with the name of the appropriate database. At the next dot prompt, the user should type the

MODIFY STRUCTURE

command (8:330-334). The computer will show a new screen with all the existing fields in the active database on it. Each field has four parameters associated with it that describe its attributes: Field Name, Type, Width, and Dec (abbreviation for Decimal). Use the arrow keys on the keyboard to place the curser in the attribute box that is to be changed. Then type in the change and hit the ENTER

key. Continue this procedure until all the changes have been made. It must be stressed that the user should not change the names of any of the fields. When done, hold down the CTRL key and hit the END key (Ctrl-End). The computer will then display "Should data be COPIED from backup for all fields? (Y/N)". The user should type Y to indicate that all of the data from the original database file should be retained (8:331). When the dot prompt is returned then the user has successfully completed the changes for that database.

Removing Fields from a Database. This procedure has more implications on the IFIS than the previous modification procedure. It should not be done without the consent of the System Manager. To repeat, DO NOT remove the FIRST_NAME or LAST_NAME fields from the GFDEMO.DBF database nor the SSAN field from any database.

Open the database that is to be changed with the USE command as described in the last section. Similarly, type the MODIFY STRUCTURE command (8:30-334) to make changes to the database. Using the arrow keys, highlight the field that is to be removed. Keep in mind that all the information contained in this field will be lost when the field is deleted. When the correct field is highlighted, hold down the CTRL key and type U (Ctrl-U). Continue this procedure until the unnecessary fields have been removed. Then type Ctrl-End to save the database and exit the

session. Press ENTER to confirm the save (8:330-334). Caution cannot be overly emphasized when removing a field from a database. When in doubt, ensure that a copy has been made of the database prior to the change and set safely aside. If the change proves to adversely affect the IFIS, simply reinstall the original database.

Now that the database has less fields, the format file that accompanies this database must be changed to remove the data call for input into this field. If the computer finds the command asking to input information into a field that no longer exists, it will "crash" or stop operation. This may be avoided by the procedures shown in the section "Modifying the Format Files".

Adding Fields to the Database. Adding a field should have no adverse affect on the IFIS. It will, however, require a bit of modification to the format file so that data can be input into the new field. Updating the format file will be discussed in the next section.

Open the selected database and prepare to modify it using the USE and MODIFY STRUCTURE commands as described earlier. Use the down arrow key to move the curser down to the last field in the database. When at the last field, hit the down arrow key once more and a blank line will appear. Type in a name for the field. Only appropriate characters will be accepted, all others are ignored. Press enter when the field name is complete. There are five

choices in the column title "Type": Character, Numeric, Date, Logical, and Memo (8:330; 4:72-73).

Character. The character type is probably the most commonly used and represents an alpha-numeric (letters, numbers, and symbols) field that cannot be used in mathematical calculations and can have any width up to 254 characters (4:72-73).

Numeric. A numeric field contains only numbers and is the only type that can be used in mathematical calculations. Numeric fields can contain simple integer numbers (containing no decimal point, such as 142) or real numbers (with a decimal point, such as 13.5). The numeric field intended to hold real numbers should be made wide enough to hold a sign, decimal point and all the numbers after the decimal point. So, a numeric field with a specified width of 7 and decimal length of three can hold a number no larger than 999.999 and no smaller than -99.999 (4:72-73).

Date. The date field is always 8 characters wide and the standard date format is mm/dd/yy (4:73).

Logical. The logical field is always on character long and will only accept T, F, Y, or N for true, false, yes, or no respectively (4:73; 3:167).

Memo. The memo field should only be used by an experienced programmer.

Choose the field type by pressing the space bar until the appropriate type displays on the screen, then press ENTER to select it. The width is governed by the type of field and the users discretion. The Dec column is only used for numeric fields and should be zero for integers. Continue this process until all the fields have been added. Then type Ctrl-End to save your changes and exit the MODIFY STRUCTURE session (8:334).

Adding fields, as was the case with removing fields, requires modification of the format file associated with the database. The format file should have an data input procedure built so that the user can input information into the new fields through the IFIS.

Modification of the Format Files. The modification of the format files requires a basic knowledge of dBase III PLUS programming. A firm grasp in the use of @...GET (8:13-16) and @...SAY (8:17-21) commands is a prerequisite.

It is the responsibility of the user to determine the best placement of the new field's data request in the existing format file. The placement of the new fields as well as any existing fields that are to be moved must be determined exactly. The appropriate spacing must be carefully counted so that the placement is not in error before the modification process should begin.

The MODIFY SCREEN command (8:329) must not be used on existing format files. These files have been manipulated

and special graphics characters have been added. If the user attempts to use the MODIFY SCREEN command, the format file must be rebuilt for scratch. Use the

MODIFY COMMAND xxxxxxxx.FMT

command (8:325-329)to update the format files and be sure to include the .FMT suffix after the format file name. Place the @...GET and @...SAY commands in the appropriate places and type Ctrl-W (8:327) to save the file and exit the MODIFY COMMAND session. If the new field data requests are not in the appropriate locations, then this process must be repeated iteratively until the user is satisfied.

Program Files

Program files are the control files of a database system. They control all the operations and processes that will take place when running the system. Program files have a file name no longer than eight alpha-numeric characters long followed by a .PRG suffix. The program files used in the IFIS will be discussed in detail in this section.

The Introduction File - INTRO.PRG. The introduction file is the first file in the IFIS. Its first task is to set the selected dBase III PLUS environmental settings. These settings could easily be controlled through the CONFIG.DB file so that programming them in the IFIS would not be necessary. The AFIT/LSM Office will probably have

more than one dBase III application. Because of the diversity in this office's using requirements, it was decided that this system should not presume a standard environment that may confuse other applications.

The first few lines are simply comment lines, those starting with an asterisk (*), and are not read by the computer. Similar comment lines are interspersed throughout the programs to assist in reading the program code and to debug the program. The reader may see the use of a double ampersand (&&) after some commands in the IFIS programs. Double ampersands are, again, not read by the computer and are used for in-line comments.

The PUBLIC command is used to identify variables that will be used in more that one program. Any variable listed with this command can be used by any program that follows. The CHOICE variable is a numeric variable that identifies which database that the user would like to modify in the maintenance section of the IFIS. It is probably the most widely used variable. CHECK is the variable used to identify which section the user plans to operate and is found primarily in the file MAIN.PRG. ULINE is a variable used to hold eighty underlines to draw a line across the screen in several programs. FIRST and LAST are for holding the name of the person who's records are being modified.

The SET STATUS OFF command removes the status bar from line twenty three and the message line from line twenty

four. This creates a cleaner and larger screen for the menu and information files within the IFIS. When the status is toggled off, dBase III uses line zero, known as the scoreboard, to convey information for when CAP LOCK is on and/or when the insert mode is on. To provide a completely uncluttered screen the scoreboard has also been toggled off with the SET SCOREBOARD OFF command. To minimize user annoyance and unnecessary computer generated messages, the bell and the talk functions have been turned off.

The next few line are to initialize the variables. The computer will halt operation (crash) if there is nothing in a variable when it is called into use. Following the initializing lines are the program code to display the introduction screen. First, there is a loop that moves the IFIS text image from the bottom of the screen to the top and climbs two lines each loop. Then comes the code that displays the offset solid block IFIS image. To complete the introduction screen, several lines of text are displayed in various color patterns for information and programming credits.

The next to last line sets the scoreboard back on. This line is written in its four letter roots to improve efficiency. Lastly, the DO command sends the computer processing to the next program in system. INTRO.PRG will

stay open and active in computer memory for the duration of the IFIS session.

The Main Menu File - MAIN.PRG. This program consists of one large loop that can be exited through the DO CASE command. The first command is to empty the screen so that the first menu can be presented. The menu gives the user a choice of sections within IFIS in which to work. The user's response one letter is placed into the CHECK variable. If the user puts in an invalid response, one not presented on the menu screen, then the OTHERWISE case, within the DO CASE command, is initiated. The CHECK variable is loaded with AA and the program loops. The IF command now registers as true and the variable named VALID that was identified in INTRO.PRG is displayed on the last line of the screen. Another request for the user's preference of sections is presented. When an appropriate response is given the corresponding case is initiated in the DO CASE command. MAIN.PRG, like INTRO.PRG, will remain open and active during the entire IFIS session.

The First Maintenance Menu File - MAINT-0.PRG. If the user opts to maintain the databases in the IFIS, the MAINT-0.PRG is the third file to be processed. This program also is one large loop that can be exited through the DO CASE command. The program displays a menu presenting the numbers associated with starting the maintenance procedure for the first six of nineteen total IFIS databases. The

practiced user can avoid the next three menu screens by entering the number of any of the other databases at this prompt. For instance, if the user wanted to maintain the GFADDUTY.DBF and he knew the number was 33, as shown in the menu from MAINT-3.PRG, he could simply type in 33 at the request and begin maintaining that database. This capability will help speed the experienced user's through the IFIS while maintaining an informational back-up for the occasional user.

The CHOICE variable is used for the first time in this program. CHOICE is used to route between the maintenance section programs. The changes in its value can be confusing when an observer tries to track it through the programs, but the theory is quite simple. The user can input only two digit numbers. The only valid numbers that can be input are shown on the menus in files MAINT-0.PRG, MAINT-1.PRG, MAINT-2.PRG, and MAINT-3.PRG. All numeric values, valid and invalid, are accepted by MAINT-0.PRG. All values except the menu movement values, 7 and 8 for this program, are transferred to the CHOICES.PRG file through the OTHERWISE case in the DO CASE command. CHOICES.PRG processes the valid values and if an invalid value for the CHOICE variable is used, then it forces a value of 150 into the CHOICE variable and routes it back to MAINT-0.PRG file. Back in MAINT-0.PRG, the program loops, displays the menu again and checks to see if CHOICE now

equals 150. Since it does, the variable VALID, from INTRO.PRG, is displayed on the last line and the user is asked to respond again. This is a moderately inefficient procedure but highly effective.

IFIS forces the value 150 into the CHOICE variable to denote invalid responses from any of the maintenance menu screens. IFIS also forces two other numbers into the CHOICE variable in appropriate situations, 100 and 101. These two values allow quick movement between menus. If the user is at three of the four maintenance menu screens (MAINT-1.PRG, MAINT-2.PRG, or MAINT-3.PRG) he has the option of returning to the main menu, MAIN.PRG, directly. Because every file remains open in dBase III PLUS that has been called with the DO command until the corresponding RETURN has been issued, all files in line that call the next file in IFIS are open. Only 16 files are allowed to be open by the IFIS at any given time, for reasons beyond the scope of this paper. This condition requires that to go from file MAINT-2.PRG to MAIN.PRG, for example, then processing must be transferred from MAINT-2.PRG by the RETURN command to MAINT-1.PRG, which in then is returned to MAINT-0.PRG, which can finally return to MAIN.prg. This type processing is analogous to a person climbing a flight of stairs using a DO command to the next floor, which correlates to a file. When the programmer climbs up five or six floors (files) he can build a new floor two place a

duplicate of the second floor or he can go down several flights of steps with return commands. By examining the code, it can be seen that the value of 101 in the CHOICE variable allows the IFIS to quickly close the files down through the MAIN.PRG file. The CHOICE variable forced with the value 100 is a special case and will be discussed in the section titled "The Fourth Maintenance Menu File - MAINT-3.PRG".

The Second Maintenance Menu File - MAINT-1.PRG. This file serves the same function as MAINT-0.PRG for the seventh through the ninth databases. In this file, as with the other maintenance menu files, any of the 19 databases can be accessed by entering the number shown to the left of it in the appropriate file maintenance menu.

The Third Maintenance Menu File - MAINT-2.PRG. This file is the same as the previous two except that is supports the twelfth through the sixteenth databases.

The Fourth Maintenance Menu File - MAINT-3.PRG. This maintenance menu file supports the seventeenth through nineteenth databases and contains two openings for any databases that are added at some future date. This file forces the value 100 into the CHOICE variable in the case where the user inputs the value where CHOICE equals 37, "GO to Next Menu". When the value of CHOICE is 101, the IFIS steps back through the programs to MAIN.PRG. Similarly, when the value of CHOICE is 100, the IFIS steps back to

MAINT-0.PRG. The theory and the execution behind these two processes is identical.

File to Choose Database Route - CHOICES.PRG. This program works invisibly from the user's perspective. That is, it processes information received from the maintenance menu files through the CHOICE variable and routes the processing to the appropriate files. There are no direct screen displays from this program.

CHOICES.PRG starts by identifying two new public variables that will be used in subsequent programs. The MOD_NEW variable is used initially in CHECKER.PRG to record whether the user wants to modify an existing record or to add a new record to a database. The SSN variable is set aside to hold the social security number of the person whose records are being maintained. SSN is used to identify the connection between the databases and keys on the SSAN field contained in each IFIS database. All the databases have an index that arranges the records by the order of the social security number to facilitate searches.

The next two IF commands check to ensure that the user has input a valid database number from the four file maintenance programs. The first IF command uses simple arithmetic checks. The second IF command is a bit more esoteric checking sequence. The intent is to determine if the second number in the numeric variable CHOICE is either a nine or a zero. This difficulty in this simple process

becomes more apparent when the programmer learns that this type of check can only be done with character variables. So, CHOICE is converted to a character variable with the STR() function, leading zeros and blanks are removed with the LTRIM() function, and the SUBSTR() function as listed removes the second character from the variable which is then stored in K. If CHOICES is a one digit number, it makes no sense to pull out the second digit which explains the IF command check to ensure CHOICES is greater or equal to 10.

The CHECKER.PRG file is called with the DO command. The file returns the name of the person whose records the user is interested in examining. The GFDEMO.DBF database is always active in the maintenance of any database because it is the central database and the only one that contains the names of the people whose records are in the IFIS. GFDEMO.DBF is stored in location 2. The database being modified is stored in location 1. When processing returns from CHECKER.PRG to CHOICES.PRG, GFDEMO.DBF is set with the appropriate record active using the LOCATE command. When a record is active, that means that it is the record that dBase III PLUS will pull information from when a given field name is identified.

The DO CASE command is then initiated. This command identifies which database is to be modified by the user's choice stored in the CHOICES variable. Each database is

treated essentially the same with the exception of the GFDEMO.DBF. GFDEMO.DBF is selected by itself, no other databases need be open or active, and is then modified by the user using the DEMO.FMT file within the CASER.PRG file. The other eighteen databases are processed similarly except that each of the others is modified in conjunction with the demographics database. The SET RELATION command tells dBASE III PLUS how to connect the two databases so that they can temporarily be treated as one large database.

File to Check Whether to Modify/Append - CHECKER.PRG. This program, as mentioned in the last section, is used to determine whether the user plans to modify or add to the database that was previously selected. This at first glance may seem like a fine distinction, but in dBase III PLUS there is a large difference between modifying and adding. In order to modify a database, dBase III PLUS must know which record is to be changed. To append, though, a new record must be prepared and added on to the end of the database. Therefore, when the user chooses to modify, CHECKER is programmed to request the person's name whose record is going to be used. If this person has no record in the GFDEMO.DBF database or if his name is no in the database as the user typed it, then the user is notified of this and has the opportunity to enter the name again.

When an appropriate name is found, then the ISBLANK.PRG file is called. This file is discussed in another section.

When processing returns from ISBLANK.PRG then information is pulled from GFDEMO.DBF and placed in the SSN, LAST, and FIRST variables. Processing is then returned to CHOICES.PRG.

File to Find Record to Edit - CASER.PRG. This program actually implements the information that the user supplied into the MOD_NEW variable during the execution of the CHECKER.PRG file. If the user chose to modify a record the first case in the DO CASE command is initiated. The first step is to find the right record in the database to be modified. If it is not found, ENDFILE.PRG is called. If it is found, then it is the format file for the chosen database is activated so that the database can be modified using the EDIT command. When editing is complete, ISBLANK.PRG is called. Assuming that the user chose to add a record to the database then a similar process is conducted using the APPEND command. This command opens a blank record at the end of the database and allows the user to input information into that record. Lastly, if the user chose to exit the processing on the active database, then processing is transferred back to CHOICES.PRG which then transfers processing back to the file maintenance menu program that was last used. This allows the user to work on a different database or to leave the maintenance area and go to the main menu.

File to Say No Record in System - NOFIND.PRG. This program is a contingency file that is activated to tell the user that the demographics database has no information on the person whose records are to be modified. The user is then given the choice of whether to add a record for this person to the demographics database or to choose another person's records to modify. If the user decides to add a record, then the MOD_NEW variable is changed to the add mode and the CASER.PRG file treats this process as it would a simple record addition as described in the CASER.PRG section. If the user chooses to try a different record, then processing is transferred back to CHECKER.PRG.

File to Say No Record in Database - ENDFILE.PRG. This file is very similar to the NOFILE.PRG file, but it is used for the eighteen databases besides the demographics database. The NOFILE.PRG file is for the demographics database alone. If the demographics database has information on the person that the user has selected and the active database does not, then this program is activated. Once this program is activated, the user is informed of this dilemma and is presented with a choice to add a record to the active database or not. If the user wants to add a record, the APPEND command is activated. If not, then processing is sent back to CHECKER.PRG.

File to Test for Blank SSN - ISBLANK.PRG. This file is a simple check to determine if the user has placed some

value into the social security number field. If there is something in this field the rest of ISBLANK.PRG is skipped. If the field has been left blank, then the user is told that this will make the record unreadable by the IFIS since it locates all records based on social security numbers. The user must then place something in this field in order to get out of the ISBLANK.PRG program loop.

A problem with this system is that it does not check to determine whether the user has put in the correct social security number. If the wrong number has been input, then the IFIS still will not be able to find a record for future maintenance nor for reporting purposes.

File to Explain Adding Records - NOADD.PRG. This is a very simple program built solely for informational purposes. The CHECKER.PRG will not allow the user to add a record to a database, besides the demographics database, directly. The IFIS cannot find records for a person in any of the databases if that person does not have a record in the demographics database. This program serves as a reminder to fill the demographics database first, then input the data into the other databases in any order.

The Basic Screen Building File - SCR.PRG. This program sets up the basic screen format used in several of the other programs. This program was built in the name of programming efficiency. This program allowed the IFIS

programmer to type in information that would be repetitive for many programs just one time.

The Last File - CLOSE.PRG. This program is simply a screen showing the IFIS logo to the user prior to leaving the dBase III PLUS environment at the conclusion of an IFIS session.

Format Files

Format files are essentially cosmetic files used to present a screen that will layout database input requests and/or display database field values in a logical fashion. They are primarily used with the EDIT and APPEND commands, but can also be called with a READ command. Format files align closely with the databases they support. Database files and format files use the same names except that the latter ends in a .FMT suffix. The format files used in the IFIS are shown in Appendix C. All the format files have essentially the same intent and differ only slightly in character, so they require no individual discussion.

Format files may require some modification to account for slight changes in the IFIS database structure as the IFIS matures. This was discussed at length in the database modification section located near the beginning of this chapter. Care should be taken to ensure all changes have been concluded successfully before the original copy of the format file is erased. Also note that the format files were built using the CREATE(MODIFY) SCREEN command but have

all been modified using the MODIFY COMMAND command afterward. These modifications include the addition of graphics characters that are not supported in the CREATE(MODIFY) SCREEN environment. If the one of the screen commands is attempted on one of the format files it will be lost and significant reprogramming will be required.

Index Files

Index files are denoted by the suffix .NDX and are listed in appendix D. All of the databases, with the exclusion of the demographics database, have one index file that is arranged according to the social security number field, SSAN.

The demographics database has two index files. The first index for the demographics database is arranged on the two name fields. It is first arranged alphabetically according to the LAST_NAME field and then by the FIRST_NAME field. This means that if there are two Smiths in the database, they will be arranged alphabetically by first name. This index file is named DEMO.NDX. The second index file for this database is named DEMOSSN.NDX and is arranged according to the social security number field.

IV. Conclusions and Recommendations

Some analysis of the IFIS has already been presented in previous chapters. Inefficiencies and uncompleted programming involving the system are summarized here and further conclusions regarding the finished product will be examined. An attempt will also be made to discern the lessons learned during the conduct of this thesis so that they may promote more effective programming efforts in office automation projects. Lastly, and arguably most importantly, some recommendations are fashioned for furthering the IFIS programming effort to make it into a complete office automation package.

Summary of Shortfalls

The IFIS is inefficient. The inefficiency is due primarily to two major weaknesses in the programming concept. First, this program was written while the programmer was learning dBase III PLUS. Many techniques used in the initial programming were used without the knowledge of some helpful dBase III PLUS commands. These commands could have sped the processing of the system while gaining the same overall results. Many of the inefficient techniques are still used in the IFIS because they were not sufficiently detrimental to the operation of the system to warrant correction. Even with this knowledge in mind,

several attempts were made near the end of the programming of the IFIS to improve the efficiency of several program modules. The repercussions of these seemingly innocent changes sent tendrils of havoc rummaging though the entirety of the IFIS programs. It soon became apparent that the old saying, "If it isn't broken, don't fix it!" is wise indeed. The only remedy to the learning process in computer programming is experience. Early efforts, therefore, will be inherently inefficient.

The most glaring example of programming inefficiency is shown in the program CHOICES.PRG shown in appendix B. The reader may notice that each CASE in this program, with the exception of the first, shows a marked similarity with all the others. This could be rectified through the use of a well programmed subroutine. It is estimated that a subroutine could save in excess of 80 lines of code, which is almost half of the executable lines in this program.

The second major cause of inefficiency in the IFIS is due to standard programming practices. These practices require that a program be fully documented and formatted so that the computer code is readable. The use of comment lines and indenting the lines of code are of no practical use to the computer. In fact, these practices use valuable processing time. In order to improve efficiency, two different copies of the system should be available; one for the computer and one for people. The computer copy should

be identical to the copy for people with all comment lines and indentures removed.

Lessons Learned

Careful research prior to the start of actual programming is essential to the production of a successful database system. The programmer needs to spend a significant amount of time in close association with the people that will eventually use the system to gain the insight needed to meet the user's requirements. This is a difficult task that requires much practice to perform. Often the user does not know the depth of the capabilities that they require at the onset of the project. The programmer must be prepared to offer suggestions and deny some requests based on his knowledge the programming language's capabilities. This complex task can be confounding to the first time programmer. Therefore, it is advisable for a programmer to complete several small projects to gain proficiency in the necessary process prior to undertaking a large database system.

It is best for the programmer to begin a database project from the initial phases of needs determination. The development of the database dictionary is an important step for the programmer so that he can begin the mental preparation of the system prior to undertaking a written plan. This written plan should be centered around a rough

draft of the schematic or flow chart of the proposed system. This schematic then needs to be discussed in detail with the users. This discussion should be the cornerstone of the entire programming effort. The user's must realize that at the completion of this discussion and all their input into the design of the system has been heard and understood, the basic design of the system is then cast in stone. Once the actual programming efforts have been started, radical changes to the system design can be devastating. If the programmer is unsure of the requirements of the system because of insufficient or ambiguous guidance, then the outcome of the project is similarly likely to suffer. A firm understanding between the users and the programmer prior to the actual coding of the system is vital to the successful outcome of the project. The burden of ensuring this understanding exists must be borne by the programmer.

Any good program is never fully complete, time available just runs out. This may be the most difficult realization for any conscientious programmer. The true test of any program is whether it works within acceptable presupposed limits. Once this level has been achieved, stop. Additional work is fruitless and may even be detrimental to the project. If additional improvements are envisioned by the programmer, he should make a mental note and use this information on subsequent system designs. The energy a

programmer would use to improve one workable system would be better spent on the next project. When programming, good enough is good enough and better is a waste of valuable productivity.

Recommendations for Future Study

The IFIS represents a good start at an overall system, yet it is incomplete. The IFIS is a system programmed for database development and maintenance only. There are several additions, however, that need to be incorporated into the system before it is a truly viable tool for office automation.

The IFIS requires a good reporting section before it will be advantageous to the using organization. There are numerous documents required by AFIT/LSM that can be generated by programmed reports. The R&R Relational Report Writer (5) package is available and could be readily incorporated into the IFIS. The use of R&R would be relatively simple when compared to programming with the dBase III PLUS report generator. Also, selected queries of the information provided in the database system would be useful to the users. The ability to perform these queries quickly within the IFIS would alleviate the need for senior faculty members to search through stack of paper to answer questions concerning faculty teaching capabilities, tenure requirements, and faculty replacements, among others.

The IFIS could be adapted to encompass the needs of the entire School of Systems and Logistics. This would probably entail an expansion of the capabilities of some of the databases and possibly the addition of several new databases. Also the reporting procedures would similarly have a broader base of requirements. The growth of the IFIS could include connectivity with other dBase III PLUS systems currently in use within the School of Systems and Logistics. If this growth of capability were successfully implemented, the Dean of the school should be given access to the system so that he could gain a better understanding of the daily functions of his organization. The potential growth of the system is large and varied.

The advancement of this system would be a rigorous and rewarding challenge for future thesis work. The Head of the Department of Logistics Management should be contacted for more information.

Appendix A: Database Dictionary

DATABASE NAME: Demographics

IFIS File: GFDEMO.DBF

Field Name	Type	Width	Dec
LAST_NAME	Character	20	
FIRST_NAME	Character	20	
SSAN	Character	11	
ADDRESS	Character	30	
CITY	Character	12	
STATE	Character	2	
ZIP	Character	5	
HOME_PHONE	Character	8	
GENDER	Character	1	
CLEARANCE	Character	10	
RANK_GRADE	Character	6	
DOR	Date	8	
DOS	Date	8	
DOB	Date	8	
SPOUSE_NAM	Character	10	
BIRTH_PLAC	Character	25	
NO_CHILD	Numeric	2	0
KIDS_NAMES	Character	40	
DEPARTMENT	Character	6	
POS_NUMBER	Numeric	10	0
AFSC	Character	7	
AFSC_AUTH	Character	7	
AFSC_ASSGN	Character	7	
ARRIVAL	Date	8	
DEPARTURE	Date	8	
REPLACEMNT	Logical	1	
REMARKS	Character	75	

DATABASE NAME: AFIT Academic History

IFIS File: GFHIST.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
ACADEM_RNK	Character	10	
DOR	Date	8	
PROF_MEMB	Character	40	
PROF_REG	Character	40	

DATABASE NAME: Professional Experience

IFIS File: GFEXPER.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
ORG	Character	40	
JOB_TITLE	Character	20	
START	Date	8	
END	Date	8	
REMARKS	Character	80	

DATABASE NAME: Course Assignments

IFIS File: GFCOURSE.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
QUARTER	Character	2	
COURSE_NO	Numeric	2	
TITLE	Character	254	
PCE_CREDIT	Character	3	
PCE_DT_HRS	Numeric	6	1
PCE_CLS_SZ	Numeric	3	
UG_CREDIT	Character	3	
UG_LAB_HRS	Character	3	
UG_LEC_HRS	Character	3	
UG_NO_SECS	Numeric	3	
UG_CLS_SZ	Numeric	3	

DATABASE NAME: Faculty Replacements

IFIS File: GFREPLAC.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
STUDT_SSAN	Character	11	
NAME_LAST	Character	20	
NAME_FIRST	Character	20	
SCHOOL	Character	40	
MAJOR	Character	20	
ENROLLED	Date	8	
GRADUATE	Date	8	
REPORT_DATE	Date	8	
REP_INFO	Character	40	

DATABASE NAME: Faculty TDY

IFIS File: GFTDY.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
FUND_OFICE	Character	5	
TDY_NO	Character	3	
LOCATION	Character	25	
START_DATE	Date	8	
RTURN_DATE	Date	8	
TRAVEL_CST	Numeric	7	2
PER_DIEM	Numeric	6	2
REG_FEE	Numeric	7	2
NO_DAYS	Numeric	3	0
PURPOSE	Character	130	

DATABASE NAME: Educational History

IFIS File: GFEDHIST.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
SCHOOL_UG	Character	30	
DEGREE_UG	Character	10	
START_UG	Date	8	
END_UG	Date	8	
MAJOR_UG	Character	20	
MINOR_UG	Character	20	
SCHOOL_MS	Character	30	
DEGREE_MS	Character	10	
START_MS	Date	8	
END_MS	Date	8	
THESIS_TTL	Character	80	
MAJOR_MS	Character	20	
SCHOOL_PHD	Character	30	
DEGREE_PHD	Character	10	
START_PHD	Date	8	
END_PHD	Date	8	
DISS_TITLE	Character	80	
MAJOR_PHD	Character	20	

DATABASE NAME: Professional Development Courses

IFIS File: GFPRODEV.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
COURSE_TTL	Character	30	
COURSE_LOC	Character	30	
COURSE_DES	Character	80	
START	Date	8	
END	Date	8	

DATABASE NAME: Military Development Courses

IFIS File: GFCOURSE.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
COUR6E_TTL	Character	30	
COURSE_LOC	Character	30	
COURSE_DES	Character	80	
START	Date	8	
END	Date	8	

DATABASE NAME: Continuing Education

IFIS File: GFCONTED.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
SCHOOL	Character	25	
LOCATION	Character	25	
COURSE_NAM	Character	65	
COURSE_NO	Character	10	

DATABASE NAME: Honors and Awards

IFIS File: GFHONAWD.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
AWD_NAME	Character	30	
DATE_RECD	Date	8	

DATABASE NAME: Civic Activities

IFIS File: GFCIVIC.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
CIV_ORG1	Character	40	

DATABASE NAME: Faculty Publications

IFIS File: GFPUBS.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
PUB_TITLE	Character	26	
PUB_VOL	Character	4	
PUB_NO	Character	4	
PUB_DATE	Character	8	
PUB_PG_NOS	Character	7	
ART_NAME	Character	130	
ART_PGS	Numeric	3	0
COAUTHORS	Character	40	

DATABASE NAME: Faculty Presentations

IFIS File: GFPRESEN.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
MEETING	Character	40	
LOCATION	Character	25	
DATE	Date	8	
PRE_TITLE	Character	130	
ADD_INFO	Character	65	

DATABASE NAME: Consulting Activities

IFIS File: GFCNSULT.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
POC	Character	30	
POC_OFFICE	Character	20	
POC_PH_COM	Character	13	
POC_PH_AV	Character	8	
ORG	Character	15	
BASE_ADDR	Character	15	
STATE	Character	2	
ZIP	Character	5	
CON_HRS	Numeric	3	0
NATION	Character	10	
CITY	Character	12	
INFO	Character	75	

DATABASE NAME: Research Hours

IFIS File: GFRESRCH.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
TOPIC	Character	65	
RESRCH_HRS	Numeric	3	0
PAPER_HRS	Numeric	3	0
PRESENT_HRS	Numeric	3	0
TOPIC_HRS	Numeric	3	0
TOT_RES_HRS	Numeric	3	0
RES_INFO	Character	50	

DATABASE NAME: Committee Responsibilities

IFIS File: GFCOMITE.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
COMMITTEE	Character	30	
COM_DATE	Date	8	
COM_HRS	Numeric	3	0
COM_INFO	Character	50	
COM_DT_END	Date	8	

DATABASE NAME: Applicable OER Dates

IFIS File: GFOER.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
RATER	Character	20	
OER_DUE	Date	8	
LAST_OER	Date	8	
UIF	Logical	1	
TAFSD	Date	8	

DATABASE NAME: Additional Duties

IFIS File: GFADDUTY.DBF

Field Name	Type	Width	Dec
SSAN	Character	11	
DUTY	Character	40	
DUTY_INFO	Character	75	

Appendix B: Program Library

```
* THE FOLLOWING IS AN INTRO SCREEN WITH SOME  -  INTRO.PRG
* INITIALIZED VARIABLES
*
*
*------------------------ Initialization
*
PUBLIC CHOICE, CHECK, ULINE, FIRST, LAST
CLOSE ALL
SET SCOREBOARD OFF
SET BELL OFF
SET STATUS OFF
SET TALK OFF
FIRST = SPACE(10)
LAST = SPACE(20)
CHOICE = 0
CHECK = " "
DARK7 = REPLICATE(CHR(219),7)
DARK5 = REPLICATE(CHR(219),5)
ULINE = REPLICATE("_",80)
VALID = "Please use a valid choice."
*
*------------------------ Display Introduction Screen
*
CLEAR
*
*
X=17
DO WHILE X>1
X=X-2
@X,1 SAY " "
TEXT
                    IIIIIII   FFFFFFF   IIIIIII    SSSSS
                       I      F           I      S     S
                       I      F           I      S
                       I      FFFFF       I       SSSSS
                       I      F           I            S
                       I      F           I      S     S
                    IIIIIII   F        IIIIIII    SSSSS


ENDTEXT
ENDDO
*
*
SET COLOR TO W+
@3,21 SAY DARK7
@3,31 SAY DARK7 + "    " + DARK7
@3,51 SAY DARK7
```

```
@4,24 SAY CHR(219)
@4,31 SAY CHR(219)
@4,44 SAY CHR(219)
@4,51 SAY CHR(219)
@4,57 SAY CHR(223)
@5,24 SAY CHR(219)
@5,31 SAY CHR(219)
@5,44 SAY CHR(219)
@5,51 SAY CHR(219)
@6,24 SAY CHR(219)
@6,31 SAY DARK5
@6,44 SAY CHR(219)
@6,51 SAY DARK7
@7,24 SAY CHR(219)
@7,31 SAY CHR(219)
@7,44 SAY CHR(219)
@7,57 SAY CHR(219)
@8,24 SAY CHR(219)
@8,31 SAY CHR(219)
@8,44 SAY CHR(219)
@8,51 SAY CHR(220)
@8,57 SAY CHR(219)
@9,21 SAY DARK7 + "    " + CHR(219) + "          " + DARK7 +;

"    " + DARK7
*
*
@13,19 SAY CHR(16)+" Integrated Faculty Information ;
System "+CHR(17)
SET COLOR TO
@18,15 SAY "Written for the Air Force Institute of
Technology"
@17,13 TO 19,65
@19,25 TO 21,54
@19,25 SAY CHR(191)+"                                    "+CHR(218)

@20,27 SAY "By: John H. Barnes GLM-88S"
@0,0 TO 24,79 DOUBLE
SET COLOR TO W*,X
@5,6 SAY "AIR"
@6,5 SAY "FORCE"
@5,70 SAY "AIR"
@6,69 SAY "FORCE"
DUMMY=" "
@14,37 SAY " " GET DUMMY
READ
SET COLOR TO
SET SCOR ON
DO MAIN
```

```
*******   MAIN MENU PROGRAM - MAIN.PRG
*
*
*---------------Display menu and get user's choice.
*
DO WHILE .T.
    CLEAR
    @2,1 SAY "Main Menu"
    @2,60 SAY DTOC(DATE()) + "    " + TIME()
    @3,0 SAY ULINE
    @7,20 SAY "    Code       Description"
    @9,20 SAY "      M      MAINTAIN the Databases"
    @16,21 SAY "     X      EXIT IFIS"
    @22,1 SAY "Enter Your Choice (M or X) " GET CHECK;
        PICTURE "A"
    IF CHECK = "AA"
        @24,30 SAY VALID
    ENDIF
    READ
*
*--------------------BRANCH TO REQUESTED PROGRAM
*
    DO CASE
        CASE UPPER(CHECK)="M"
            DO MAINT-0
        CASE UPPER(CHECK)="X"
            DO CLOSE
*--------------------- Mistake check
        OTHERWISE
            CHECK = "AA"
            LOOP
    ENDCASE
ENDDO
```

```
*******  THE FILE MAINTENANCE PROGRAMS - MENU # ZERO
*        MAINT-0.PRG
*
*---------------Display menu and get user's choice.
*
DO WHILE .T.
    CLEAR
    @2,1 SAY "File Maintenance Menu #1"
    @2,60 SAY DTOC(DATE()) + "    " + TIME()
    @3,0 SAY ULINE
    @5,15 SAY "Add to or modify the area of your choice."
    @7,20 SAY "      1     Demographics"
    @9,20 SAY "      2     AFIT Academic History"
    @11,21 SAY "     3     Professional Experience"
    @13,21 SAY "     4     Course Assignments"
    @15,21 SAY "     5     Faculty Replacements"
    @17,21 SAY "     6     Faculty TDY"
    @19,21 SAY "     7     Go to Next Menu"
    @21,21 SAY "     8     Return to Main Menu"
    @23,1 SAY "Enter Your Choice (1-8) " GET CHOICE;
        PICTURE "99"
    IF CHOICE = 150
        @24,30 SAY VALID
    ENDIF
    READ
*--------------------PARTIAL CASE
*
    DO CASE
        CASE CHOICE = 7
            DO MAINT-1
*
*------------ Check for "Return to Main Menu" and Undo Do's
*
            IF CHOICE = 101
                RETURN
            ENDIF
            LOOP
        CASE CHOICE = 8
            RETURN
        OTHERWISE
******---------------FINISH CASE IN CHOICES PROGRAM
*
            DO CHOICES
*
*--------------------Check for Completed File Maintenance
*
            IF CHOICE = 101
                RETURN
            ENDIF
    ENDCASE
LOOP
ENDDO
```

```
*******  THE FILE MAINTENANCE PROGRAMS - MENU ONE
*        MAINT1.PRG
*---------------Display menu and get user's choice.
*
DO WHILE .T.
    CLEAR
    @2,1 SAY "File Maintenance Menu #2"
    @2,60 SAY DTOC(DATE()) + "    " + TIME()
    @3,0 SAY ULINE
    @5,15 SAY "Add to or modify the area of your choice."
    @7,20 SAY "      11     Educational History"
    @9,20 SAY "      12     Professional Development
Courses"
    @11,21 SAY "     13     Military Development Courses"
    @13,21 SAY "     14     Continuing Education"
    @15,21 SAY "     15     Honors and/or Awards"
    @17,21 SAY "     16     Go to Previous Menu"
    @19,21 SAY "     17     Go to Next Menu"
    @21,21 SAY "     18     Return to Main Menu"
    @23,1 SAY "Enter Your Choice (11-18) " GET CHOICE;
        PICTURE "99"
    IF CHOICE = 150
        @24,30 SAY VALID
    ENDIF
    READ
*--------------------PARTIAL CASE
*
    DO CASE
        CASE CHOICE = 16
            RETURN
        CASE CHOICE = 17
            DO MAINT-2
*
*------------Check for "Return to Main Menu" and Undo Do's
*
            IF CHOICE >= 100
                RETURN
            ENDIF
        CASE CHOICE = 18
            CHOICE = 101
            RETURN
        OTHERWISE
*
******---------------FINISH CASE IN CHOICES PROGRAM
            DO CHOICES
*--------------------Check fo Completed File Maintenance
            IF CHOICE = 101
                RETURN
            ENDIF
    ENDCASE
LOOP
ENDDO
```

```
*******  THE FILE MAINTENANCE PROGRAMS - MENU # TWO
*        MAINT2.PRG
*---------------Display menu and get user's choice.
*
DO WHILE .T.
    CLEAR
    @2,1 SAY "File Maintenance Menu #3"
    @2,60 SAY DTOC(DATE()) + "    " + TIME()
    @3,0 SAY ULINE
    @5,15 SAY "Add to or modify the area of your choice."
    @7,20 SAY "      21     Civic Activities"
    @9,20 SAY "      22     Publications"
    @11,21 SAY "     23     Presentations"
    @13,21 SAY "     24     Consulting Activities"
    @15,21 SAY "     25     Research Hours"
    @17,21 SAY "     26     Got to Previous Menu"
    @19,21 SAY "     27     Go to Next Menu"
    @21,21 SAY "     28     Return to Main Menu"
    @23,1 SAY "Enter Your Choice (21-28) " GET CHOICE;
        PICTURE "99"
    IF CHOICE = 150
        @24,30 SAY VALID
    ENDIF
    READ
*--------------------PARTIAL CASE
*
    DO CASE
       CASE CHOICE = 26
            RETURN
       CASE CHOICE = 27
            DO MAINT-3
*-------------Check for 'Return to Main Menu' and Undo Do's
*
            IF CHOICE >= 100
                RETURN
            ENDIF
       CASE CHOICE = 28
            CHOICE = 101
            RETURN
        OTHERWISE
*
******---------------FINISH CASE IN CHOICES PROGRAM
            DO CHOICES
*--------------------Check for Completed File Maintenance
*
            IF CHOICE = 101
                RETURN
            ENDIF
    ENDCASE
LOOP
ENDDO
```

```
*******  THE FILE MAINTENANCE PROGRAMS - MENU THREE
*        MAINT3.PRG
*
*---------------Display menu and get user's choice.
*
DO WHILE .T.
    CLEAR
    @2,1 SAY "File Maintenance Menu #4"
    @2,60 SAY DTOC(DATE()) + "    " + TIME()
    @3,0 SAY ULINE
    @5,15 SAY "Add to or modify the area of you choice."
    @7,20 SAY "      31     Committee Responsibilities"
    @9,20 SAY "      32     Applicable OER Dates"
    @11,21 SAY "     33     Additional Duties"
    @13,21 SAY "     34     ** Currently not in use."
    @15,21 SAY "     35     ** Currently not in use."
    @17,21 SAY "     36     Go to Previous Menu"
    @19,21 SAY "     37     Go to Next Menu"
    @21,21 SAY "     38     Return to Main Menu"
    @23,1 SAY "Enter Your Choice (31-38) " GET CHOICE;
        PICTURE "99"
    IF CHOICE = 150
        @24,30 SAY VALID
    ENDIF
    READ
*
*--------------------PARTIAL CASE
*
    DO CASE
        CASE CHOICE = 36
            RETURN
        CASE CHOICE = 37
            CHOICE = 100
            RETURN
        CASE CHOICE = 38
            CHOICE = 101
            RETURN
        OTHERWISE
*
******---------------FINISH CASE IN CHOICES PROGRAM
*
            DO CHOICES
*
*--------------------Check for Completed File Maintenance
*
            IF CHOICE = 101
                RETURN
            ENDIF
    ENDCASE
LOOP
ENDDO
```

```
**********  IFIS Choices Case Program - CHOICES.PRG
*
*
PUBLIC MOD_NEW, SSN, LOOPER
SSN = "           "
STORE "  " TO MOD_NEW
LOOPER=0
DO WHILE LOOPER=0
LOOPER=1
*
DO CHECKER
* ----- Selecting Demographics database for SET RELATION
Commands SELECT 2
USE GFDEMO INDEX DEMOSSN
LOCATE FOR LAST_NAME=LAST .AND. FIRST_NAME=FIRST
*
DO CASE
*
SET ECHO ON
*------------------------- CHOICES FROM THE MENU-0.PROG
MENU
CASE CHOICE = 1
    SELECT 2
    REINDEX
    SET FORMAT TO DEMO
    DO CASER
CASE CHOICE = 2
    SELECT 1
    USE GFHIST INDEX GFHIST
    SELECT 2
    SET RELATION TO SSAN INTO GFHIST
    SELECT 1
    SET FORM TO GFHIST
    DO CASER
CASE CHOICE = 3
    SELECT 1
    USE GFEXPER INDEX GFEXPER
    SELECT 2
    SET RELATION TO SSAN INTO GFEXPER
    SELECT 1
    SET FORM TO GFEXPER
    DO CASER
CASE CHOICE = 4
    SELECT 1
    USE GFCOURSE INDEX GFCOURSE
    SELECT 2
    SET RELATION TO SSAN INTO GFCOURSE
    SELECT 1
    SET FORM TO GFCOURSE
    DO CASER
CASE CHOICE = 5
    SELECT 1
```

```
     USE GFREPLAC INDEX GFREPLAC
     SELECT 2
     SET RELATION TO SSAN INTO GFREPLAC
     SELECT 1
     SET FORM TO GFREPLAC
     DO CASER
CASE CHOICE = 6
     SELECT 1
     USE GFTDY INDEX GFTDY
     SELECT 2
     SET RELATION TO SSAN INTO GFTDY
     SELECT 1
     SET FORM TO GFTDY
     DO CASER
*
*--------------- CHOICE FROM THE MAINT-1.PRG MENU
*
CASE CHOICE = 11
     SELECT 1
     USE GFEDHIST INDEX GFEDHIST
     SELECT 2
     SET RELATION TO SSAN INTO GFEDHIST
     SELECT 1
     SET FORM TO GFEDHIST
     DO CASER
CASE CHOICE = 12
     SELECT 1
     USE GFPRODEV INDEX GFPRODEV
     SELECT 2
     SET RELATION TO SSAN INTO GFPRODEV
     SELECT 1
     SET FORM TO GFPRODEV
     DO CASER
CASE CHOICE = 13
     SELECT 1
     USE GFMILDEV INDEX GFMILDEV
     SELECT 2
     SET RELATION TO SSAN INTO GFMILDEV
     SELECT 1
     SET FORM TO GFMILDEV
     DO CASER
CASE CHOICE = 14
     SELECT 1
     USE GFCONTED INDEX GFCONTED
     SELECT 2
     SET RELATION TO SSAN INTO GFCONTED
     SELECT 1
     SET FORM TO GFCONTED
     DO CASER
CASE CHOICE = 15
     SELECT 1
     USE GFHONAWD INDEX GFHONAWD
```

```
     SELECT 2
     SET RELATION TO SSAN INTO GFHONAWD
     SELECT 1
     SET FORM TO GFHONAWD
     DO CASER
*
*--------------- CHOICES FROM THE MAINT-2.PRG MENU
*
CASE CHOICE = 21
     SELECT 1
     USE GFCIVIC INDEX GFCIVIC
     SELECT 2
     SET RELATION TO SSAN INTO GFCIVIC
     SELECT 1
     SET FORM TO GFCIVIC
     DO CASER
CASE CHOICE = 22
     SELECT 1
     USE GFPUBS INDEX GFPUBS
     SELECT 2
     SET RELATION TO SSAN INTO GFPUBS
     SELECT 1
     SET FORM TO GFPUBS
     DO CASER
CASE CHOICE = 23
     SELECT 1
     USE GFPRESEN INDEX GFPRESEN
     SELECT 2
     SET RELATION TO SSAN INTO GFPRESEN
     SELECT 1
     SET FORM TO GFPRESEN
     DO CASER
CASE CHOICE = 24
     SELECT 1
     USE GFCNSULT INDEX GFCNSULT
     SELECT 2
     SET RELATION TO SSAN INTO GFCNSULT
     SELECT 1
     SET FORM TO GFCNSULT
     DO CASER
CASE CHOICE = 25
     SELECT 1
     USE GFRESRCH INDEX GFRESRCH
     SELECT 2
     SET RELATION TO SSAN INTO GFRESRCH
     SELECT 1
     SET FORM TO GFRESRCH
     DO CASER
*
*--------------- CHOICES FROM THE MAINT-3.PRG MENU
*
CASE CHOICE = 31
```

```
     SELECT 1
     USE GFCOMITE INDEX GFCOMITE
     SELECT 2
     SET RELATION TO SSAN INTO GFCOMITE
     SELECT 1
     SET FORM TO COMITE
     DO CASER
CASE CHOICE = 32
     SELECT 1
     USE GFOER INDEX GFOER
     SELECT 2
     SET RELATION TO SSAN INTO GFOER
     SELECT 1
     SET FORM TO GFOER
     DO CASER
CASE CHOICE = 33
     SELECT 1
     USE GFADDUTY INDEX GFADDUTY
     SELECT 2
     SET RELATION TO SSAN INTO GFADDUTY
     SELECT 1
     SET FORM TO GFADDUTY
     DO CASER
*
*--------------- UNRECOGNIZED AND OTHER CHOICES
*
CASE CHOICE=101
CASE CHOICE=149
OTHERWISE
     CHOICE = 150
     RETURN
ENDCASE
ENDDO
RETURN
```

```
** IFIS Modify or New Record Checking Program - CHECKER.PRG
*
PUBLIC Z
Z=0
*
DO WHILE Z=0     &&   NOFIND.PRG Do Loop to start over!!
STORE 1 TO Z,ZZ
*
DO SCR
@2,2 SAY "MODIFY OR NEW"
X=0
*
DO WHILE X=0
X=1
    @8,5 SAY "Would you like to:"
    @10,20 SAY "M)     Modify a record"
    @11,20 SAY "A)     Add a new record"
    @12,20 SAY "       (Demographics only!)"
    @14,20 SAY "X)     Leave this section."
    @17,5 SAY "Enter choice: (M, A, or X) " GET MOD_NEW
PICTURE "!"
    READ
    DO CASE
    CASE MOD_NEW = "M"
       @17,5 SAY "Please Provide the Last Name " GET LAST ;
PICTURE "!!!!!!!!!!!!!!!!!!!!"
       READ
     @19,5 SAY "Please Provide the First Name " GET FIRST ;
PICTURE "!!!!!!!!!!"
       READ
    CASE MOD_NEW = "X"
       RETURN
    CASE MOD_NEW = "A"
       IF CHOICE # 1
          STORE 0 TO Z,ZZ
          DO NOADD
       ELSE
          RETURN
       ENDIF
    OTHERWISE
       @19,25 SAY "TRY AGAIN."
       X=0
    ENDCASE
ENDDO
IF ZZ = 1
USE GFDEMO INDEX DEMO
REINDEX
GO TOP
LOCATE FOR LEFT(LAST_NAME,4) = LEFT("&LAST",4) .AND. ;
    LEFT(FIRST_NAME,3) = LEFT("&FIRST",3)
IF EOF()
    IF CHOICE = 1
```

```
        DO NOFIND
    ELSE
        @20,5 SAY "This person has no record in the
Demographics database.  Please enter"
        @21,5 say "the appropriate information into
Demographics for this person first."
        Z=0
    ENDIF
ELSE
    DO ISBLANK
    SSN = SSAN
    LAST = LAST_NAME
    FIRST = FIRST_NAME
ENDIF
CLOS DATA
*
ENDIF
ENDDO     &&     NOFIND.PRG Do Loop to start over
*
RETURN
```

```
****    IFIS Program to test MOD/ADD/ESC Choices - CASER.PRG
*
*
REQUEST= SPACE(1)
CLEAR TYPE
W=0
*
**--Determine MOD/ADD/ESC Case From User Input Var. MOD_NEW
*
DO CASE
    CASE MOD_NEW="M"
        GO TOP
        SEEK "&SSN"
        IF EOF()
            DO ENDFILE
        ELSE
            SET STATUS ON
            IF CHOICE=1 .OR. CHOICE=5 .OR. CHOICE=32
                EDIT NEXT 1
            ELSE
                EDIT
            ENDIF
            SET STATUS OFF
            DO ISBLANK
        ENDIF
    CASE MOD_NEW="A"
        SET STAT ON
        APPEND
        IF CHOICE # 1
            REPLACE SSAN WITH SSN
        ELSE
            DO ISBLANK
        ENDIF
        SET STAT OFF
    CASE MOD_NEW="X"
        CLOS DATA
        CLOS FORM
        RETURN
ENDCASE
*
**------------------REQUEST TO CONTINUE
*
CLOS FORM
DO SCR
@2,2 SAY "CONTINUE"
DO WHILE W=0
W=1
@10,5 SAY "Would you like to continue with this database? ;
(Y or N)" GET REQUEST
READ
*
DO CASE
```

```
    CASE REQUEST="Y" .OR. REQUEST="y"
        LOOPER=0 && To continue with do loop in choices.prg
        CLOS DATA
    CASE REQUEST="N" .OR. REQUEST="n"
        CLOS DATA
    OTHERWISE
        @12,30 SAY "Please Try Again."
        W=0
ENDCASE
*
ENDDO
RETURN
```

```
*****  IFIS Program to verify record exists - NOFIND.PRG
*
*
KEY=" "
X = 0
*
DO SCR
@2,2 SAY "Record Not Found"
*
@7,5 SAY "The record for "+TRIM(FIRST)+" "+TRIM(LAST)+"
cannot;
 be found."
@9,5 SAY "Would you like to:"
@11,15 SAY "A)     Add this record to your data base, or"

@12,15 SAY "T)     Try a different record."
*
DO WHILE X=0
*
@14,15 SAY "Please enter your choice: " GET KEY ;
PICTURE "!"
READ
*
DO CASE
    CASE KEY="A"
        MOD_NEW = "A"
        X=1
    CASE KEY="T"
        LAST="                    "
        FIRST="          "
        Z=0    && For use with do loop in checker.prg
        RETURN
    OTHERWISE
        @17,20 SAY CHR(7)+"Please enter A or T."
ENDCASE
*
ENDDO
RETURN
```

```
**    IFIS Program to append records to database without
**    a relational record to gfdemo.dbf  -        ENDFILE.PRG
*
*
DO SCR
@2,2 SAY "NO CORRESPONDING RECORD"
@7,5 SAY TRIM(FIRST)+" "+TRIM(LAST)+" exists in the system
but ;
has no corresponding"
@8,5 SAY "record in this database."
DO WHILE .T.
@10,2 SAY " "
ACCEPT CHR(186)+"    Would you like to add this person to
the ;
database? (Y or N) " TO ADDS
DO CASE
    CASE ADDS="Y" .OR. ADDS="y"
        SET STAT ON
        APPEND
        REPLACE SSAN WITH GFDEMO->SSAN
        SET STAT OFF
    CASE ADDS="N" .OR. ADDS="n"
        RETURN
    OTHERWISE
        LOOP
ENDCASE
```

```
* IFIS Program to find if SSAN field is empty.- ISBLANK.PRG
*
*
DO WHILE SSAN = "           " .OR. SSAN = "   -  -    "
    LOCATE FOR SSAN = SSN
    IF TRIM(SSAN) # "" .OR. SSAN # "   -  -    "
        RETURN
    ENDIF
    CLEAR
    @1,0 TO 17,78 DOUBLE
    @3,1 TO 3,77
    @2,20 SAY "NO  SOCIAL  SECURITY  NUMBER  INPUT"
    @5,5 SAY CHR(7)+"There is NO Social Security Number for
the ;
current record."
    @8,9 SAY "This number is used to allow communication
between;
 the various"
    @10,5 SAY "segments of the IFIS.  If a UNIQUE number is
not ;
placed in the"
    @12,5 SAY "SSAN field, then the program will produce
UNPRE;
DICTABLE and"
    @14,5 SAY "ERRONEOUS results.  Please correct this
record ;
now."
    @18,5
    WAIT "     Type any key to continue..."
    IF CHOICE = 1 .OR. CHOICE=5 .OR. CHOICE=32
        EDIT NEXT 1
    ELSE
        EDIT
    ENDIF
ENDDO
RETURN
```

```
*****  Program to explain why user can add new
*      records only to GFDEMO.DBF                - NOADD.PRG
*
DO SCR
@2,2 SAY "Cannot Add New Records To This Database"
@6,7 SAY "To ensure proper file maintenance, the user can
only ;
add new"
@8,5 SAY "records to the DEMOGRAPHICS DATABASE.  This
procedure ;
will help"
@10,5 SAY "keep the filing system clean and fast by
avoiding ;
records that"
@12,5 SAY "are stuck on the end of the working files that ;
IFIS can neither"
@14,5 SAY "locate nor use."
@16,7 SAY "IF you would like to ADD a new person, please
ADD ;
this person"
@18,5 SAY "into the DEMOGRAPHICS database FIRST and then
MODIFY ;
the other"
@20,5 SAY "databases.  Thanks..."
@23,0 SAY " "
WAIT
RETURN
```

```
*****    BASIC SCREEN DESIGN FOR IFIS -- SCR.PRG
*
*
CLEAR
@1,0 TO 22,79 DOUBLE
@2,58 SAY DTOC(DATE())+"     "+TIME()
@3,1 TO 3,78
RETURN
```

```
* THE FOLLOWING IS THE GOOD-BYE SCREEN - CLOSE.PRG
*
*----------------------Bold IFIS Logo
*
SET SCOREBOARD OFF
DARK7 = REPLICATE(CHR(219),7)
DARK5 = REPLICATE(CHR(219),5)
CLEAR
SET COLOR TO W+
@3,21 SAY DARK7
@3,31 SAY DARK7 + "    " + DARK7
@3,51 SAY DARK7
@4,24 SAY CHR(219)
@4,31 SAY CHR(219)
@4,44 SAY CHR(219)
@4,51 SAY CHR(219)
@4,57 SAY CHR(223)
@5,24 SAY CHR(219)
@5,31 SAY CHR(219)
@5,44 SAY CHR(219)
@5,51 SAY CHR(219)
@6,24 SAY CHR(219)
@6,31 SAY DARK5
@6,44 SAY CHR(219)
@6,51 SAY DARK7
@7,24 SAY CHR(219)
@7,31 SAY CHR(219)
@7,44 SAY CHR(219)
@7,57 SAY CHR(219)
@8,24 SAY CHR(219)
@8,31 SAY CHR(219)
@8,44 SAY CHR(219)
@8,51 SAY CHR(220)
@8,57 SAY CHR(219)
@9,21 SAY DARK7 + "    " + CHR(219) + "          " + DARK7 +;
"    " + DARK7
*
* ---------------- Words below and beside IFIS Logo
*
@13,19 SAY CHR(16)+" Integrated Faculty Information ;
System "+CHR(17)
SET COLOR TO
@20,28 SAY "THANKS FOR STOPPING BY"+CHR(19)+CHR(19)
@19,26 TO 21,53
@0,0 TO 23,79 DOUBLE
SET COLOR TO W*,X
@5,6 SAY "AIR"
@6,5 SAY "FORCE"
@5,70 SAY "AIR"
@6,69 SAY "FORCE"
DUMMY = " "
*
```

```
*  -----------  Set Curser on Screen and Wait for User to
*  -----------  Hit ENTER Key
*
@15,37 SAY " " GET DUMMY
READ
*
*  -----------  Set Color to Normal, Close Databases, and
*  -----------  Clear all Memory.
*
SET COLOR TO
CLOSE ALL
CLEAR ALL
@24,0 SAY " "
QUIT
```

Appendix C: Formatted Screen Library

DEMOGRAPHICS

```
@  1, 12  SAY "****  INTEGRATED  FACULTY  INFORMATION
SYSTEM  ****"
@  3, 25  SAY "- Demographics Database -"
@  5, 15  SAY "First  MI  Last"
@  6,  2  SAY "Name:"
@  6, 15  GET  GFDEMO->FIRST_NAME  PICTURE "!!!!!!!!!!"
@  6, 26  GET  GFDEMO->LAST_NAME  PICTURE
"!!!!!!!!!!!!!!!!!!!!"
@  6, 51  SAY "Rank/Grade:"
@  6, 63  GET  GFDEMO->RANK_GRADE
@  7,  2  SAY "Street:"
@  7, 15  GET  GFDEMO->ADDRESS
@  7, 51  SAY "SSAN:"
@  7, 63  GET  GFDEMO->SSAN  PICTURE "999-99-9999"
@  8,  2  SAY "City, State:"
@  8, 15  GET  GFDEMO->CITY
@  8, 27  SAY ","
@  8, 29  GET  GFDEMO->STATE
@  8, 33  SAY "ZIP:"
@  8, 38  GET  GFDEMO->ZIP  PICTURE "99999"
@  8, 51  SAY "Sex:"
@  8, 63  GET  GFDEMO->GENDER
@  9,  2  SAY "Home Phone:"
@  9, 15  GET  GFDEMO->HOME_PHONE  PICTURE "999-9999"
@  9, 51  SAY "# of Kids:"
@  9. 63  GET  GFDEMO->NO_CHILD
@ 11,  2  SAY "Birth Date:"
@ 11, 17  GET  GFDEMO->DOB
@ 11, 31  SAY "Birth Place:"
@ 11, 44  GET  GFDEMO->BIRTH_PLAC
@ 12,  2  SAY "Spouse's Name:"
@ 12, 17  GET  GFDEMO->SPOUSE_NAM
@ 12, 31  SAY "Kids' Names:"
@ 12, 44  GET  GFDEMO->KIDS_NAMES  FUNCTION "S35"
@ 14,  2  SAY "AFIT Dept:"
@ 14, 14  GET  GFDEMO->DEPARTMENT
@ 14, 27  SAY "Position #:"
@ 14, 40  GET  GFDEMO->POS_NUMBER
@ 14, 54  SAY "Clearance:"
@ 14, 66  GET  GFDEMO->CLEARANCE
@ 15,  2  SAY "Arriv Date:"
@ 15, 14  GET  GFDEMO->ARRIVAL
@ 15, 27  SAY "Depart Date:"
@ 15, 40  GET  GFDEMO->DEPARTURE
@ 15, 54  SAY "Replacemt Known?:"
@ 15, 72  GET  GFDEMO->REPLACEMNT
```

```
@ 17,  2  SAY "AFSC:"
@ 17, 14  GET  GFDEMO->AFSC
@ 17, 27  SAY "AFSC Auth:"
@ 17, 40  GET  GFDEMO->AFSC_AUTH
@ 17, 54  SAY "AFSC Assgn:"
@ 17, 66  GET  GFDEMO->AFSC_ASSGN
@ 18,  2  SAY "Date Rank:"
@ 18, 14  GET  GFDEMO->DOR
@ 18, 27  SAY "Date Serv:"
@ 18, 40  GET  GFDEMO->DOS
@ 19,  2  SAY "Remarks:"
@ 19, 14  GET  GFDEMO->REMARKS  FUNCTION "S66"  PICTURE
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX-
XXXXXXXXXXXXXXXXX"
@  2, 11  TO  2, 65
@  0,  0  TO 20, 79    DOUBLE
```

AFIT ACADEMIC HISTORY

```
@  2, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  4, 27  SAY "-  AFIT Academic History  -"
@  7,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" ";
+TRIM(GFDEMO->LAST_NAME)
@  7, 35  SAY "Rank: "+GFDEMO->RANK_GRADE
@  7, 57  SAY "SSAN: "+GFDEMO->SSAN
@  9,  3  SAY "Academic Rank:"
@  9, 18  GET  GFHIST->ACADEM_RNK
@  9, 39  SAY "Social Security No.: "+GFHIST->SSAN
@ 10,  3  SAY "Date of Rank:"
@ 10, 18  GET  GFHIST->DOR
@ 12,  3  SAY "Professional Memberships:"
@ 12, 31  GET  GFHIST->PROF_MEMB
@ 13,  3  SAY "Professional Registration:"
@ 13, 31  GET  GFHIST->PROF_REG
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:       ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@ 20,  6  SAY "Insert Mode:  Ins
  Abandon:          Esc"
@  1, 20  TO  3, 60    DOUBLE
@ 15,  2  TO 21, 77    DOUBLE
@ 16, 27  TO 20, 27
@ 16, 49  TO 20, 49
```

PROFESSIONAL EXPERIENCE

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 26  SAY "-  Professional Experience  -"
@  5,  3  SAY "Name: "+;
TRIM(GFDEMO->FIRST_NAME)+" "+TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  6, 57  SAY "SSAN:"
@  6, 64  GET  GFEXPER->SSAN
@  7,  3  SAY "Organization:"
@  7, 17  GET  GFEXPER->ORG
@  8,  3  SAY "Job Title:"
@  8, 17  GET  GFEXPER->JOB_TITLE
@  8, 39  SAY "Start Date:"
@  8, 51  GET  GFEXPER->START
@  8, 61  SAY "End Date:"
@  8, 70  GET  GFEXPER->END
@  9,  3  SAY "Remarks:"
@  9, 17  GET  GFEXPER->REMARKS  FUNCTION "S65"
@ 11,  3  SAY "1. Enter only ONE job per screen.  Retype
the SSAN for each job."
@ 12,  3  SAY "2. DO NOT type over a displayed job unless
you wish to erase it."
@ 13,  3  SAY "3. For more jobs, type ENTER for a new
screen.  Type CTRL-END when done."
@ 15,  4  SAY "CURSER                    DELETE
  RECORD"
@ 16,  6  SAY "Character: "+CHR(27)+" "+CHR(26)

@16,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 17,  6  SAY "Word:  Home or End       Field:       ^Y
  Next Record:      PgDn"
@ 18,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@18,31 SAY "Record:     ^U        Done/Save:        ^End"
@ 19,  6  SAY "Insert Mode:  Ins
  Abandon:          Esc"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 14,  2  TO 20, 77    DOUBLE
@ 15, 27  TO 19, 27
@ 15, 49  TO 19, 49
```

FACULTY COURSE ASSIGNMENTS

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 24  SAY "-  Faculty Course Assignments  -"
@  4, 57  SAY "SSAN:"+GFDEMO->SSAN
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+;
" "+TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank:"+ GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN:"
@  5, 64  GET  GFCOURSE->SSAN
@  6,  3  SAY "Course Title:"
@  6, 17  GET  GFCOURSE->TITLE  FUNCTION "S62"
@  7,  3  SAY "Quarter Taught:"
@  7, 20  GET  GFCOURSE->QUARTER
@  7, 25  SAY "Course Number:"
@  7, 40  GET  GFCOURSE->COURSE_NO
@  8,  3  SAY "PCE Classes:      Credit:"
@  8, 30  GET  GFCOURSE->PCE_CREDIT
@  8, 37  SAY "DT Hours:"
@  8, 48  GET  GFCOURSE->PCE_DT_HRS
@  8, 58  SAY "Class Size:"
@  8, 71  GET  GFCOURSE->PCE_CLS_SZ
@ 10,  3  SAY "U/G Classes:      Credit:"
@ 10, 30  GET  GFCOURSE->UG_CREDIT
@ 10, 37  SAY "Lect Hours:"
@ 10, 49  GET  GFCOURSE->UG_LEC_HRS
@ 10, 58  SAY "Lab Hours:"
@ 10, 71  GET  GFCOURSE->UG_LAB_HRS
@ 11, 21  SAY "No. of Sections:"
@ 11, 39  GET  GFCOURSE->UG_NO_SECS
@ 11, 49  SAY "Class Size:"
@ 11, 62  GET  GFCOURSE->UG_CLS_SZ
@ 12,  3  SAY "1. Enter only ONE class per screen.  Retype
the SSAN for each class."
@ 13,  3  SAY "2. DO NOT type over a displayed class unless
you wish to erase it."
@ 14,  3  SAY "3. For more classes, type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Word:  Home or End        Character:  Del
  Previous Record:  PgUp"
@ 18,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@18,31 SAY "Field:      ^Y        Next Record:      PgDn"
@ 19,  6  SAY "Insert Mode:  Ins
  Done/Save:          ^End"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

FACULTY REPLACEMENTS

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 27  SAY "-  Faculty Replacements  -"
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+ GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+ GFDEMO->SSAN
@  6,  3  SAY "Replacement's Information:"
@  7, 10  SAY "First  MI            Last"
@  8,  3  SAY "Name:"
@  8, 10  GET  GFREPLAC->NAME_FIRST
@  8, 31  GET  GFREPLAC->NAME_LAST
@  8, 57  SAY "SSAN:"
@  8, 64  GET  GFREPLAC->STUDT_SSAN  PICTURE "999-99-9999"
@ 10,  3  SAY "School:"
@ 10, 13  GET  GFREPLAC->SCHOOL
@ 11,  3  SAY "Major:"
@ 11, 13  GET  GFREPLAC->MAJOR
@ 13,  3  SAY "Date Enrolled:"
@ 13, 18  GET  GFREPLAC->ENROLLED
@ 13, 30  SAY "Grad Date:"
@ 13, 41  GET  GFREPLAC->GRADUATE
@ 13, 53  SAY "Report Date:"
@ 13, 66  GET  GFREPLAC->REPORT_DAT
@ 14,  3  SAY "Reporting Information:"
@ 14, 27  GET  GFREPLAC->REP_INFO
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Word:  Home or End        Character:  Del
  Previous Record:  PgUp"
@ 18,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@18,31 SAY "Field:      ^Y        Next Record:      PgDn"
@ 19,  6  SAY "Insert Mode:  Ins
  Done/Save:          ^End"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

FACULTY TDY

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 32  SAY "-  Faculty TDY   -"
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank:"
@  5, 46  SAY  GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN:"
@  5, 64  SAY  GFDEMO->SSAN
@  7,  3  SAY "Funding Office:"
@  7, 21  GET  GFTDY->FUND_OFICE
@  7, 34  SAY "TDY Number:"
@  7, 47  GET  GFTDY->TDY_NO
@  8,  3  SAY "TDY Destination:"
@  8, 21  GET  GFTDY->LOCATION
@  9,  3  SAY "Purpose:"
@  9, 13  GET  GFTDY->PURPOSE  FUNCTION "S65"
@ 10,  6  SAY "{Note: The Purpose block holds more than
shown, data will scroll.}"
@ 12,  3  SAY "Start Date:"
@ 12, 16  GET  GFTDY->START_DATE
@ 12, 35  SAY "Return Date:"
@ 12, 49  GET  GFTDY->RTURN_DATE
@ 13,  3  SAY "Number of Days:"
@ 13, 20  GET  GFTDY->NO_DAYS
@ 13, 35  SAY "Per Diem (One Day Cost):"
@ 13, 61  GET  GFTDY->PER_DIEM
@ 14,  3  SAY "Transportation Cost:"
@ 14, 25  GET  GFTDY->TRAVEL_CST
@ 14, 35  SAY "Registration Fee:"
@ 14, 54  GET  GFTDY->REG_FEE
@ 16,  4  SAY "CURSER                        DELETE
  RECORD"
@ 17,  6  SAY "Word:  Home or End            Character:  Del
  Previous Record:  PgUp"
@ 18,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@18,31 SAY "Field:      ^Y          Next Record:      PgDn"
@ 19,  6  SAY "Insert Mode:  Ins
  Done/Save:          ^End"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

EDUCATIONAL HISTORY

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 28  SAY "-  Educational History  -"
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  7,  3  SAY "UNDERGRADUATE DEGREE:"
@  8,  6  SAY "School:"
@  8, 19  GET  GFEDHIST->SCHOOL_UG
@  8, 53  SAY "Degree:"
@  8, 62  GET  GFEDHIST->DEGREE_UG
@  9,  6  SAY "Start Date:"
@  9, 19  GET  GFEDHIST->START_UG
@  9, 30  SAY "End Date:"
@  9, 41  GET  GFEDHIST->END_UG
@ 10,  6  SAY "Major:"
@ 10, 19  GET  GFEDHIST->MAJOR_UG
@ 10, 44  SAY "Minor:"
@ 10, 52  GET  GFEDHIST->MINOR_UG
@ 11,  3  SAY "MASTER'S DEGREE:"
@ 12,  6  SAY "School:"
@ 12, 19  GET  GFEDHIST->SCHOOL_MS
@ 12, 53  SAY "Degree:"
@ 12, 62  GET  GFEDHIST->DEGREE_MS
@ 13,  6  SAY "Start Date:"
@ 13, 19  GET  GFEDHIST->START_MS
@ 13, 30  SAY "End Date:"
@ 13, 41  GET  GFEDHIST->END_MS
@ 14,  6  SAY "Major:"
@ 14, 19  GET  GFEDHIST->MAJOR_MS
@ 15,  6  SAY "Thesis Title:"
@ 15, 21  GET  GFEDHIST->THESIS_TTL  FUNCTION "S62"
@ 16,  3  SAY "DOCTORATE DEGREE:"
@ 17,  6  SAY "School:"
@ 17, 19  GET  GFEDHIST->SCHOOL_PHD
@ 17, 53  SAY "Degree:"
@ 17, 62  GET  GFEDHIST->DEGREE_PHD
@ 18,  6  SAY "Start Date:"
@ 18, 19  GET  GFEDHIST->START_PHD
@ 18, 30  SAY "End Date:"
@ 18, 41  GET  GFEDHIST->END_PHD
@ 19,  6  SAY "Major:"
@ 19, 19  GET  GFEDHIST->MAJOR_PHD
@ 20,  6  SAY "Dissertation Title:"
@ 20, 27  GET  GFEDHIST->DISS_TITLE  FUNCTION "S62"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
```

PROFESSIONAL DEVELOPMENT COURSES

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 21  SAY "-  Professional Development Courses  -"
@5,3 SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  6, 57  SAY "SSAN:"
@  6, 64  GET  GFPRODEV->SSAN
@  7,  3  SAY "Course Title:"
@  7, 24  GET  GFPRODEV->COURSE_TTL
@  8,  3  SAY "Course Location:"
@  8, 24  GET  GFPRODEV->COURSE_LOC
@  9,  3  SAY "Course Description:"
@  9, 24  GET  GFPRODEV->COURSE_DES  FUNCTION "S60"
@ 10,  3  SAY "Course Start Date:"
@ 10, 24  GET  GFPRODEV->START
@ 10, 36  SAY "Course End Date:"
@ 10, 55  GET  GFPRODEV->END
@ 12,  3  SAY "1. Enter only ONE course per screen.  Retype
the SSAN for each course."
@ 13,  3  SAY "2. DO NOT type over a displayed course
unless you wish to erase it."
@ 14,  3  SAY "3. For more courses, type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

MILITARY DEVELOPMENT COURSES

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 23  SAY "-  Military Development Courses  -"
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  6, 57  SAY "SSAN:"
@  6, 64  GET  GFMILDEV->SSAN
@  7,  3  SAY "Course Title:"
@  7, 24  GET  GFMILDEV->COURSE_TTL
@  8,  3  SAY "Course Location:"
@  8, 24  GET  GFMILDEV->COURSE_LOC
@  9,  3  SAY "Course Description:"
@  9, 24  GET  GFMILDEV->COURSE_DES  FUNCTION "S65"
@ 10,  3  SAY "Course Start Date:"
@ 10, 24  GET  GFMILDEV->START
@ 10, 36  SAY "Course End Date:"
@ 10, 55  GET  GFMILDEV->END
@ 12,  3  SAY "1. Enter only ONE course per screen.  Retype
the SSAN for each course."
@ 13,  3  SAY "2. DO NOT type over a displayed course
unless you wish to erase it."
@ 14,  3  SAY "3. For more courses, type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(26)+" "+CHR(27)

@ 17, 31  SAY "Character:  Del       Previous Record:
PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)

@ 19, 31  SAY "Record:     ^U        Done/Save:
^End"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

CONTINUING EDUCATION COURSES

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 23  SAY "-  Continuing Education Courses  -"
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  6, 57  SAY "SSAN:"
@  6, 64  GET  GFCONTED->SSAN
@  7,  3  SAY "School:"
@  7, 18  GET  GFCONTED->SCHOOL
@  8,  3  SAY "Location:"
@  8, 18  GET  GFCONTED->LOCATION
@  9,  3  SAY "Course Name:"
@  9, 18  GET  GFCONTED->COURSE_NAM
@ 10,  3  SAY "Course Number:"
@ 10, 18  GET  GFCONTED->COURSE_NO
@ 12,  3  SAY "1. Enter only ONE course per screen.  Retype
the SSAN for each course."
@ 13,  3  SAY "2. DO NOT type over a displayed course
unless you wish to erase it."
@ 14,  3  SAY "3. For more courses, type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

HONORS AND AWARDS

```
@  2, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  4, 29  SAY "-  Honors and Awards  -"
@  6,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  6, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  6, 57  SAY "SSAN: "+GFDEMO->SSAN
@  7, 57  SAY "SSAN:"
@  7, 64  GET  GFHONAWD->SSAN
@  8,  3  SAY "Honor/Award Name:"
@  8, 23  GET  GFHONAWD->AWD_NAME
@  9,  3  SAY "Date Received:"
@  9, 23  GET  GFHONAWD->DATE_RECD
@ 11,  3  SAY "1. Enter only ONE award per screen.  Retype
the SSAN for each award."
@ 12,  3  SAY "2. DO NOT type over a displayed award unless
you wish to erase it."
@ 13,  3  SAY "3. For more awards, type ENTER for a new
screen.  Type CTRL-END when done."
@ 15,  4  SAY "CURSER                    DELETE
  RECORD"
@ 16,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@16,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 17,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 18,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@18,31 SAY "Record:     ^U        Done/Save:        ^End"
@ 19,  6  SAY "Insert Mode:  Ins
  Abandon:          Esc"
@  0,  0  TO 21, 79
@  1, 20  TO  3, 60    DOUBLE
@ 14,  2  TO 20, 77    DOUBLE
@ 15, 27  TO 19, 27
@ 15, 49  TO 19, 49
```

CIVIC ACTIVITIES

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 29  SAY "-  Civic Activities  -"
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  6, 57  SAY "SSAN:"
@  6, 64  GET  GFCIVIC->SSAN
@  8,  3  SAY "Civic Activity or Organization:"
@  8, 35  GET  GFCIVIC->CIV_ORG1
@ 10,  3  SAY "1. Retype the SSAN for each activity."
@ 11,  3  SAY "2. Please enter only ONE activity per
screen."
@ 12,  3  SAY "3. DO NOT type over a displayed activity
unless you wish to erase it."
@ 13,  3  SAY "4. For more activities type ENTER for new
screen.  Type CTRL-END when done."
@ 15,  4  SAY "CURSER                    DELETE
  RECORD"
@ 16,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@16,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 17,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 18,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@18,31 SAY "Record:     ^U        Done/Save:        ^End"
@ 19,  6  SAY "Insert Mode:  Ins
  Abandon:          Esc"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 14,  2  TO 20, 77    DOUBLE
@ 15, 27  TO 19, 27
@ 15, 49  TO 19, 49
```

PROFESSIONAL PUBLICATIONS

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 25  SAY "-  Professional Publications  -"
@  4, 57  SAY "SSAN: "+GFDEMO->SSAN
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN:"
@  5, 64  GET  GFPUBS->SSAN
@  7,  3  SAY "Publication: Title:"
@  7, 23  GET  GFPUBS->PUB_TITLE
@  7, 51  SAY "Volume:"
@  7, 59  GET  GFPUBS->PUB_VOL
@  7, 65  SAY "Number:"
@  7, 73  GET  GFPUBS->PUB_NO
@  8, 16  SAY "Date:"
@  8, 23  GET  GFPUBS->PUB_DATE
@  8, 35  SAY "Page Numbers:"
@  8, 49  GET  GFPUBS->PUB_PG_NOS
@  9,  3  SAY "Article Name:"
@  9, 17  GET  GFPUBS->ART_NAME  FUNCTION "S63"
@ 10,  3  SAY "Art. No. of Pages:"
@ 10, 22  GET  GFPUBS->ART_PGS
@ 10, 27  SAY "Coathors:"
@ 10, 37  GET  GFPUBS->COAUTHORS
@ 12,  3  SAY "1. Enter only ONE pub. per screen.  Retype
the SSAN for each pub."
@ 13,  3  SAY "2. DO NOT type over a displayed pub. unless
you wish to erase it."
@ 14,  3  SAY "3. For more pubs., type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

PROFESSIONAL PUBLICATIONS

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  2, 25  SAY "-  Professional Presentations  -"
@  4,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  4, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  4, 57  SAY "SSAN: "+GFDEMO->SSAN
@  5, 57  SAY "SSAN:"
@  5, 64  GET  GFPRESEN->SSAN
@  6,  3  SAY "Meeting:"
@  6, 14  GET  GFPRESEN->MEETING
@  7,  3  SAY "Location:"
@  7, 14  GET  GFPRESEN->LOCATION
@  7, 51  SAY "Date:"
@  7, 58  GET  GFPRESEN->DATE
@  8,  3  SAY "Presentation Title:"
@  9,  3  GET  GFPRESEN->PRE_TITLE  FUNCTION "S75"
@ 10,  3  SAY "Remarks:"
@ 10, 13  GET  GFPRESEN->ADD_INFO
@ 12,  4  SAY "1. Enter only ONE presentation per screen.
Retype the SSAN for each!"
@ 13,  4  SAY "2. DO NOT type over a displayed screen
unless you wish to erase it."
@ 14,  4  SAY "3. Complete screen, type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@  0,  0  TO 21, 79
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

PROFESSIONAL CONSULTATIONS

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  2, 24  SAY "-  Professional Consultations  -"
@  4,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  4, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  4, 57  SAY "SSAN: "+GFDEMO->SSAN
@  5, 57  SAY "SSAN:"
@  5, 64  GET  GFCNSULT->SSAN
@  6,  3  SAY "Point of Contact (POC):"
@  6, 27  GET  GFCNSULT->POC
@  7,  3  SAY "POC Office:"
@  7, 16  GET  GFCNSULT->POC_OFFICE
@  8,  3  SAY "POC Phone No.-  Commercial:"
@  8, 31  GET  GFCNSULT->POC_PH_COM  PICTURE
"(999)999-9999"
@  8, 48  SAY "AutoVon:"
@  8, 58  GET  GFCNSULT->POC_PH_AV  PICTURE "999-9999"
@ 10,  3  SAY "Organization:"
@ 10, 18  GET  GFCNSULT->ORG
@ 11,  3  SAY "Address:"
@ 11, 18  GET  GFCNSULT->BASE_ADDR
@ 12,  3  SAY "City, State:"
@ 12, 18  GET  GFCNSULT->CITY
@ 12, 30  SAY ","
@ 12, 32  GET  GFCNSULT->STATE
@ 12, 36  SAY "Zip:"
@ 12, 42  GET  GFCNSULT->ZIP
@ 13,  3  SAY "Nation:"
@ 13, 18  GET  GFCNSULT->NATION
@ 14,  3  SAY "Consultation Hours:"
@ 14, 24  GET  GFCNSULT->CON_HRS
@ 15,  3  SAY "Remarks:"
@ 16,  3  GET  GFCNSULT->INFO
@ 18,  3  SAY "1. Enter only ONE job per screen.  Retype
the SSAN for each job."
@ 19,  3  SAY "2. DO NOT type over a displayed job unless
you wish to erase it."
@ 20,  3  SAY "3. Finish screen then type ENTER for a new
screen.  Type CTRL-END when done."
@  0,  0  TO 21, 79
```

FACULTY RESEARCH HOURS

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  2, 26  SAY "-  Faculty Research Hours  -"
@  4,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  4, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  4, 57  SAY "SSAN: "+GFDEMO->SSAN
@  5, 57  SAY "SSAN:"
@  5, 64  GET  GFRESRCH->SSAN
@  7,  3  SAY "Topic:"
@  7, 10  GET  GFRESRCH->TOPIC
@  9,  3  SAY "Research Hrs:"
@  9, 17  GET  GFRESRCH->RESRCH_HRS
@  9, 22  SAY "Paper Hrs:"
@  9, 33  GET  GFRESRCH->PAPER_HRS
@  9, 38  SAY "Presentation Hrs:"
@  9, 56  GET  GFRESRCH->PRESENT_HR
@  9, 61  SAY "Topic Hrs:"
@  9, 72  GET  GFRESRCH->TOPIC_HRS
@ 10,  3  SAY "Added Information:"
@ 10, 23  GET  GFRESRCH->RES_INFO
@ 12,  3  SAY "1. Enter only ONE topic per screen.  Retype
the SSAN for each topic."
@ 13,  3  SAY "2. DO NOT type over a displayed topic unless
you wish to erase it."
@ 14,  3  SAY "3. For more topics, type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@  0,  0  TO 21, 79
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

COMMITTEE RESPONSIBILITIES

```
@  1, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  3, 24  SAY "-  Committee Responsibilities  -"
@  5,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  5, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  6, 57  SAY "SSAN:"
@  6, 64  GET  GFCOMITE->SSAN
@  7,  3  SAY "Committee:"
@  7, 15  GET  GFCOMITE->COMMITTEE
@  8,  3  SAY "Committee Start Date:"
@  8, 25  GET  GFCOMITE->COM_DATE
@  8, 36  SAY "End Date:"
@  8, 46  GET  GFCOMITE->COM_DT_END
@  8, 57  SAY "Total Hrs:"
@  8, 69  GET  GFCOMITE->COM_HRS
@  9,  3  SAY "Added Information:"
@  9, 22  GET  GFCOMITE->COM_INFO
@ 11,  3  SAY "1. Enter only ONE job per screen.  Retype
the SSAN for each job."
@ 12,  3  SAY "2. DO NOT type over a displayed job unless
you wish to erase it."
@ 13,  3  SAY "3. For more jobs, type ENTER for a new
screen.  Type CTRL-END when done."
@ 15,  4  SAY "CURSER                    DELETE
  RECORD"
@ 16,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@16,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 17,  6  SAY "Word:  Home or End       Field:       ^Y
  Next Record:      PgDn"
@ 18,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@18,31 SAY "Record:     ^U        Done/Save:        ^End"
@ 19,  6  SAY "Insert Mode:  Ins
  Abandon:          Esc"
@  0,  0  TO 21, 79
@  2, 20  TO  2, 60    DOUBLE
@ 14,  2  TO 20, 77    DOUBLE
@ 15, 27  TO 19, 27
@ 15, 49  TO 19, 49
```

APPLICABLE OER DATES

```
@  2, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  4, 28  SAY "-  Applicable OER Dates  -"
@  6,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  6, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  6, 57  SAY "SSAN: "+GFDEMO->SSAN
@  7, 57  SAY "SSAN:"
@  7, 64  GET  GFOER->SSAN  PICTURE "999-99-9999"
@  9,  3  SAY "Rater:"
@  9, 14  GET  GFOER->RATER
@ 10,  3  SAY "Last OER:"
@ 10, 14  GET  GFOER->LAST_OER
@ 10, 27  SAY "Next OER Due:"
@ 10, 42  GET  GFOER->OER_DUE
@ 11,  3  SAY "TAFSD:"
@ 11, 14  GET  GFOER->TAFSD
@ 11, 36  SAY "UIF:"
@ 11, 42  GET  GFOER->UIF
@ 13,  3  SAY "NOTE:  The two SSAN's should match.  Correct
the highlighted one now, if"
@ 14,  8  SAY "necessary.  Correct the top SSAN through the
Demographics section."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:       ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@  0,  0  TO 21, 79
@  1, 20  TO  3, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

ADDITIONAL DUTIES

```
@  2, 22  SAY "INTEGRATED FACULTY INFORMATION SYSTEM"
@  4, 29  SAY "-  Additional Duties  -"
@  5, 57  SAY "SSAN: "+GFDEMO->SSAN
@  6,  3  SAY "Name: "+TRIM(GFDEMO->FIRST_NAME)+" "+;
TRIM(GFDEMO->LAST_NAME)
@  6, 39  SAY "Rank: "+GFDEMO->RANK_GRADE
@  6, 57  SAY "SSAN: "+GFADDUTY->SSAN
@  8,  3  SAY "Added Duty:"
@  8, 22  GET  GFADDUTY->DUTY
@  9,  3  SAY "Duty Information:"
@ 10,  3  GET  GFADDUTY->DUTY_INFO
@ 12,  3  SAY "1. Enter only ONE duty per screen.  Retype
the SSAN for each duty."
@ 13,  3  SAY "2. DO NOT type over a displayed duty unless
you wish to erase it."
@ 14,  3  SAY "3. For more duties, type ENTER for a new
screen.  Type CTRL-END when done."
@ 16,  4  SAY "CURSER                    DELETE
  RECORD"
@ 17,  6  SAY "Character: "+CHR(27)+" "+CHR(26)
@17,31 SAY "Character:  Del       Previous Record:  PgUp"
@ 18,  6  SAY "Word:  Home or End       Field:      ^Y
  Next Record:      PgDn"
@ 19,  6  SAY "Field: "+CHR(24)+" "+CHR(25)
@19,31 SAY "Record:     ^U        Done/Save:        ^End"
@  0,  0  TO 21, 79
@  1, 20  TO  3, 60    DOUBLE
@ 15,  2  TO 20, 77    DOUBLE
@ 16, 27  TO 19, 27
@ 16, 49  TO 19, 49
```

Bibliography

1. Allen, Maj Mary Kay Class lecture in LOGM 615, Logistics Decision Support Systems. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, May 1988.

2. Beard, Capt Phillip H. Class assignment in LOGM 490, Computer Programming Concepts for Managers. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1987.

3. Carrabis, Joseph-David. dBASE III PLUS: Advanced Programming. Indianapolis IN: Que Corporation, 1986.

4. Chou, George Tsu-der. dBASE III PLUS Handbook. Carmel IN: Que Corporation, 1986.

5. Concentric Data Systems, Inc. R&R Relational Report Writer: User's Manual. Concentric Data Systems, Inc., Westborough MA, 1988.

6. Liskin, Miriam. Advanced dBASE III PLUS Programming and Techniques. Berkeley CA: Osborne McGraw-Hill, 1985.

7. NCR Corporation. NCR-DOS Manual. NCR Corporation, Dayton OH, 1985.

8. Rettig, Tom and Debby Moody. Expert Advisor: dBASE III PLUS. Reading MA: Addison-Wesley Publishing Co., Inc., 1988.

9. Simpson, Alan. Advanced Techniques in dBASE III Plus. Berkeley CA: SYBEX Inc., 1986.

VITA

John H. Barnes was born on [redacted] graduated from the local high school in the spring of 1977. He entered college in the fall of that year at the University of Cincinnati and majored in Business Administration. In 1978, after sampling several majors, Mr. Barnes applied and was accepted into the College of Engineering five year cooperative education program. He received a degree of Bachelor of Science in Civil Engineering in 1983 which required a total of seven quarters of cooperative education which he satisfied in Base Civil Engineering at Wright-Patterson AFB. Upon graduation, he was hired into the Facilities and Equipment Engineering Branch in DCS Maintenance at HQ AFLC, Wright-Patterson AFB, through the Maintenance Recruitment and Development Program. Mr. Barnes spent two years at Warner Robins ALC in various offices as a condition of this program. He returned to HQ AFLC in 1985 where he worked until he entered the School of Systems and Logistics, Air Force Institute of Technology, in May 1987.

[redacted]

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

AD-A202 625

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GLM/LSM/88S-2	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION School of Systems and Logistics 6b. OFFICE SYMBOL (If applicable) AFIT/LSM	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Air Force Institute of Technology (AU) Wright-Patterson AFB OH 45433-6583	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
DEVELOPMENT OF A DBASE III PLUS DATABASE FOR OFFICE AUTOMATION WITHIN THE DEPARTMENT OF LOGISTICS MANAGEMENT, SCHOOL OF SYSTEMS AND LOGISTICS

12. PERSONAL AUTHOR(S)
John H. Barnes, B.S.C.E., GS-12

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
MS Thesis	FROM _____ TO _____	1988 September	118

16. SUPPLEMENTARY NOTATION
Fr back

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Office Automation, Databases, Computer Programming, Personal Computers, Database Management Systems,
12	05		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Thesis Chairman: John M. Halliday, Lt Col, USAF
Head, Dept of Log Mgt

Approved for public release IAW AFR 190-1.

WILLIAM A. MAUER 17 Oct 88
Associate Dean
School of Systems and Logistics
Air Force Institute of Technology (AU)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL John M. Halliday, Lt Col, USAF	22b. TELEPHONE (Include Area Code) (513) 255-5023
	22c. OFFICE SYMBOL AFIT/LSM

DD Form 1473, JUN 86 Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

ABSTRACT

The purpose of this thesis was to program a database system written in dBase III Plus for use in AFIT/LSM. The system was to be hosted on a personal computer so that it could be transportable between the different computers in the office and conveniently used by as many personnel as possible. The program was intended to replace as much manual paperwork as possible and to provide a means to quickly access, sort, and to efficiently answer queries regarding the department faculty. The final result was intended to automate the routine office functions. This automation would relieve the department personnel from mundane duties so that they could concentrate their time on more productive pursuits.

The product of these efforts was the Integrated Faculty Information System (IFIS). The IFIS is a menu driven system that utilizes 19 databases that are interrelated to form one large database system. The database system has the capability to contain all the vital professional and demographic information on each member of the staff. The IFIS provides a means to input all the appropriate information and to modify preexisting information within the database system. Keywords: → FTD 18

The IFIS has many checks to ensure that the users provide all the necessary information on specially designed and fully descriptive input screens. The input screens were also used to modify the data by typing over or making minor changes to existing data. Several allowances had been made to inform the user of possible mistakes while using the system. The users can access the information by developing reports to suit the need.

The IFIS does not represent a complete office automation system. There are many functions that could be added to the system to make it more usable. The expansion of IFIS capabilities would make a good subject for follow-on thesis research.