AFWL BENCHMARK COMPUTER PROGRAM

B. J. Henderson
Boeing Aerospace Company
P.O. Box 3999
Seattle, Washington 98124

1 May 1982

Final Report for Period 3 March 1980—1 May 1982

CONTRACT No. DNA 001-80-C-0056

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

THIS WORK WAS SPONSORED BY THE DEFENSE NUCLEAR AGENCY UNDER RDT&E RMSS CODE B310080464 P99DAXDB08853 H2590D.

Prepared for
Director
DEFENSE NUCLEAR AGENCY
Washington, DC 20305

Publication of this material does not imply Defense Nuclear Agency (DNA) endorsement of factual accuracy or opinions, nor does DNA accept any responsibility for maintenance or support of this software.
Destroy this report when it is no longer needed. Do not return to sender.

PLEASE NOTIFY THE DEFENSE NUCLEAR AGENCY, ATTN: STTI, WASHINGTON, D.C. 20305, IF YOUR ADDRESS IS INCORRECT, IF YOU WISH TO BE DELETED FROM THE DISTRIBUTION LIST, OR IF THE ADDRESSEE IS NO LONGER EMPLOYED BY YOUR ORGANIZATION.
This report includes the description of a computer program that partially translates IBM, CDC, DEC and VAX FORTRAN into the subset language of ANSI FORTRAN 77. This contract supports a DNA task to provide benchmark computer programs to vendors who bid on a DNA computer procurement project.
This is the final report of an effort conducted under Defense Nuclear Agency Contract DNA-80-C-0056. The contract supported a DNA task of procuring an advanced scientific computer system by collecting representative software for evaluating the performance of candidate computer systems (benchmarking, Ref. 3). One sample benchmark program, a large hydrodynamics code named CSQ, has been processed into ANSI Standard Fortran using SCOFF, a translating computer program developed on this contract. The resulting version of CSQ has been tested and is now available in the DNA Cyber 176 computer located at Kirtland AFB, New Mexico.

This study was sponsored by the Office of Technical Information with LCDR Harold R. Gladwin serving as technical monitor. Major John M. Anderson reviewed the final report. Their comments and support are gratefully acknowledged. Special appreciation is extended to Michael Warshaw of RDA whose patience, cooperation, and advice provided great support for this report. The technical work of BAC staff members Loren Milliman and Dave Cruikshank made the successful completion of this contract possible.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREFACE</td>
<td>1</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>3</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>5</td>
</tr>
<tr>
<td>2 THE SCOFF COMPUTERIZED CONVERSION METHOD.</td>
<td>6</td>
</tr>
<tr>
<td>2-1 RESTRUCTURING (R)</td>
<td>6</td>
</tr>
<tr>
<td>2-2 TRANSLATION (T)</td>
<td>13</td>
</tr>
<tr>
<td>2-3 EXAMPLES</td>
<td>14</td>
</tr>
<tr>
<td>3 DEMONSTRATED USES OF SCOFF</td>
<td>15</td>
</tr>
<tr>
<td>4 SUMMARY</td>
<td>16</td>
</tr>
<tr>
<td>5 REFERENCES</td>
<td>17</td>
</tr>
<tr>
<td>APPENDIX A - SAMPLE COMPUTER RUN OF SCOFF.</td>
<td>19</td>
</tr>
<tr>
<td>APPENDIX B - LISTING OF SCOFF.</td>
<td>23</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Statement handling by SCOFF</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>Actions taken by SCOFF</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>Conventions of ANSI X3.9-1978, Fortran 77</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Job control language required to execute SCOFF</td>
<td>12</td>
</tr>
</tbody>
</table>
SECTION 1

INTRODUCTION

Although the Fortran language was designed to be a universal and machine-independent medium for development of computer programs, computer system manufacturers have corrupted this by not fully implementing the ANSI Standard Language (Refs. 1,2) and by providing various enhancements to ANSI Fortran. Although a particular vendor's Fortran language may in itself be excellent, the irregularity makes it difficult to assemble a collection of Fortran programs which is free from non-standard coding. A computer program, SCOFF (Semi-automated Conversion of Fortran Four) was developed to translate programs written in the various dialects of the Fortran language to a standard, ANSI-compatible language. SCOFF converts most of the non-standard features of major vendor's Fortran to a standard Fortran language. For example, Control Data Corporation (CDC) Fortran allows variable names to be up to seven characters in length, while Digital Equipment Corporation (DEC) Fortran allows names up to 15 characters long which may contain dollar marks ($) and underscores(_). The ANSI Standard specified that variable names be a maximum of six characters in length and that all characters be from the set (A-Z, 0-9). SCOFF replaces each instance of an improper variable name with a truncated, proper variable name.

The SCOFF output meets the requirements of ANSI X3.9-1978 Fortran 77 Subset with two additions: that of COMPLEX statements and BLOCK Data structures. These additions were made because the preponderance of computer programs used in DNA activities require one or both of these features and because there is no convenient alternative.

SCOFF presently runs on IBM 360/370 series computers and requires approximately 200K bytes memory. SCOFF uses an efficient input/output method, so that the program execution time is comparable to the IBM-supplied copy utility IEBGENER. Several input formats (corresponding to source language maintenance program conventions) are accepted by SCOFF. Output may be in ASCII, EBCDIC, or BCD character codes. The Job Control Language required to execute the SCOFF program is shown in Table 4. A listing of SCOFF appears in Appendix B.

In Section 2 of this report, we describe the development of SCOFF. Section 3 briefly describes a demonstrated use of SCOFF.
SECTION 2

THE SCOFF COMPUTERIZED CONVERSION METHOD

SCOFF reads statements (lines or cards) one at a time from a Fortran program which is to be standardized. It then examines the symbols of the statements one at a time, comparing the symbols with a typical set such as a blank, $, *, C, or a numeral. This identification of the symbols determines the type of statement. The program then compares strings with standard statements, and takes appropriate action in translation.

A description of all the statements (ANSI and non-ANSI) that are recognized by SCOFF appears in Table 1, along with an indication of how SCOFF deals with the statement type. There are five actions that SCOFF may take, as specified in Table 2. Any statement type which does not appear in Table 1 will be ignored (passed through, not translated) by SCOFF. SCOFF cannot translate those statements which violate the ANSI Standards denoted by F in Table 1. There are several constructs for which there is no equivalent in the ANSI subset. The file manipulation statements OPEN and CLOSE are found in virtually all vendor Fortrans, but the format of parameters for OPEN or CLOSE varies considerably. Since neither OPEN nor CLOSE is defined in the ANSI Fortran Subset Language, SCOFF does not attempt to translate these statements. Also, SCOFF cannot resolve ambiguities which require syntactical analysis. The notation used in Table 1 is that of ANSI X3.9-1978, Fortran 77; a reprint of the conventions appears in Table 3.

Of the actions listed in Table 2, the first two are described below. The last three are self-explanatory.

2-1 RESTRUCTURING (R)

SCOFF does a global reorganization of program statements. Many vendor Fortrans allow Type Declaration statements and DATA statements to appear anywhere in a program element (or anywhere prior to the first occurrence of one of the defined variable names in an expression). SCOFF collects Type Declaration statements and DATA statements and places them ahead of the executable statements as required by the ANSI Fortran Subset. Also, some Fortrans allow data values to be specified in the Declaration statements. Any
Table 1. Statement handling by SCOFF

<table>
<thead>
<tr>
<th>Initial Form</th>
<th>Action Taken</th>
<th>Translated Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>(see Table 2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASSIGN s TO i</td>
<td>T</td>
<td>ASSIGN s TO i</td>
</tr>
<tr>
<td>BACKSPACE u</td>
<td>T</td>
<td>BACKSPACE u</td>
</tr>
<tr>
<td>BACKSPACE (alist)</td>
<td>F</td>
<td>BACKSPACE (alist)</td>
</tr>
<tr>
<td>BLOCK DATA [sub]</td>
<td>N</td>
<td>BLOCK DATA [sub]</td>
</tr>
<tr>
<td>CALL sub [[(a [,a]...)]]</td>
<td>T</td>
<td>CALL sub [[(a [,a]...)]]</td>
</tr>
<tr>
<td>CHARACTER [*len[,]] nam [,nam]...</td>
<td>TI</td>
<td>CHARACTER [*len[,]] nam [,nam]</td>
</tr>
<tr>
<td>CLOSE (clist)</td>
<td>F</td>
<td>CLOSE (clist)</td>
</tr>
<tr>
<td>COMMON [[/cb]/nlist[ [,cb]/nlist]...]</td>
<td>T</td>
<td>COMMON [[/cb]/nlist[ [,cb]/nlist]...</td>
</tr>
<tr>
<td>COMPLEX v [,v]...</td>
<td>T</td>
<td>COMPLEX v [,v]...</td>
</tr>
<tr>
<td>COMPLEX *len[,] v [,v]...</td>
<td>RT</td>
<td>COMPLEX v [,v]...</td>
</tr>
<tr>
<td>CONTINUE</td>
<td>N</td>
<td>CONTINUE</td>
</tr>
<tr>
<td>DATA nlist/clist/ [[,]nlist/clist]</td>
<td>T</td>
<td>DATA nlist/clist/ [[,]nlist/clist]</td>
</tr>
<tr>
<td>DIMENSION a(d) [,a(d)]...</td>
<td>T</td>
<td>DIMENSION a(d) [,a(d)]...</td>
</tr>
<tr>
<td>DO s [,] i=e1,e2 [,e3]</td>
<td>T</td>
<td>DO s [,] i=e1,e2 [,e3]</td>
</tr>
<tr>
<td>DOUBLE PRECISION v [,v]...</td>
<td>RT</td>
<td>REAL v [,v]</td>
</tr>
<tr>
<td>ELSE</td>
<td>NI</td>
<td>ELSE</td>
</tr>
<tr>
<td>ELSE IF (e) THEN</td>
<td>TI</td>
<td>ELSE IF (e) THEN</td>
</tr>
<tr>
<td>END</td>
<td>N</td>
<td>END</td>
</tr>
<tr>
<td>END IF</td>
<td>NI</td>
<td>END IF</td>
</tr>
<tr>
<td>ENDFILE u</td>
<td>T</td>
<td>ENDFILE u</td>
</tr>
<tr>
<td>ENDFILE (alist)</td>
<td>F</td>
<td>ENDFILE (alist)</td>
</tr>
<tr>
<td>ENTRY en [[([d [,d]...]])</td>
<td>TF</td>
<td>ENTRY en [[([d [,d]...]])</td>
</tr>
<tr>
<td>EQUIVALENCE (nlist) [,,(nlist)</td>
<td>T</td>
<td>EQUIVALENCE (nlist) [,,(nlist)]</td>
</tr>
<tr>
<td>EXTERNAL proc [,proc]...</td>
<td>T</td>
<td>EXTERNAL proc [,proc]...</td>
</tr>
</tbody>
</table>
TABLE 1. (Continued)

<table>
<thead>
<tr>
<th>Initial Form</th>
<th>Action Taken</th>
<th>Translated Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT is</td>
<td>T</td>
<td>FORMAT is</td>
</tr>
<tr>
<td>fun ([d]...) = e</td>
<td>T</td>
<td>fun ([d]...)= e</td>
</tr>
<tr>
<td>[typ [*len]] FUNCTION fun([d [,d]...])</td>
<td>RT</td>
<td>[type] FUNCTION fun ([d [,d]...])</td>
</tr>
<tr>
<td>GO TO i [[,] (s [,s]...)]</td>
<td>N</td>
<td>GO TO i [[,] (s [,s]...)]</td>
</tr>
<tr>
<td>GO TO s</td>
<td>N</td>
<td>GO TO s</td>
</tr>
<tr>
<td>GO TO (s [,s]...)[,] i</td>
<td>T</td>
<td>GO TO (s [,s]...)[,] i</td>
</tr>
<tr>
<td>IF (e) st</td>
<td>T</td>
<td>IF (e) st</td>
</tr>
<tr>
<td>IF (e) s₁, s₂, s₃</td>
<td>T</td>
<td>IF (e) s₁, s₂, s₃</td>
</tr>
<tr>
<td>IF (e) s₁, s₂</td>
<td>F</td>
<td>IF (e) s₁, s₂</td>
</tr>
<tr>
<td>IF (e) THEN</td>
<td>T</td>
<td>IF (e) THEN</td>
</tr>
<tr>
<td>IMPLICIT type (a [,a]...)</td>
<td>R</td>
<td>Specific Type Declaration StatementsGenerated As Required</td>
</tr>
<tr>
<td>INQUIRE (iflist)</td>
<td>F</td>
<td>INQUIRE (iflist)</td>
</tr>
<tr>
<td>INQUIRE (iulist)</td>
<td>F</td>
<td>INQUIRE (iulist)</td>
</tr>
<tr>
<td>INTEGER v [,v]...</td>
<td>T</td>
<td>INTEGER v [,v]...</td>
</tr>
<tr>
<td>INTEGER *len v [,v]...</td>
<td>RT</td>
<td>INTEGER v [,v]...</td>
</tr>
<tr>
<td>INTRINSIC fun [,fun]...</td>
<td>F</td>
<td>INTRINSIC fun [,fun]...</td>
</tr>
<tr>
<td>LOGICAL v [,v]...</td>
<td>T</td>
<td>LOGICAL v [,v]...</td>
</tr>
<tr>
<td>LOGICAL *len v [,v]...</td>
<td>RT</td>
<td>LOGICAL v [,v]...</td>
</tr>
<tr>
<td>NAMELIST</td>
<td>TF</td>
<td>NAMELIST</td>
</tr>
<tr>
<td>OPEN (olist)</td>
<td>I</td>
<td>OPEN (olist)</td>
</tr>
<tr>
<td>PARAMETER (p=e[,p=e]...)</td>
<td>F</td>
<td>PARAMETER (p=e[,p=e]...)</td>
</tr>
<tr>
<td>PAUSE [n]</td>
<td>N</td>
<td>PAUSE [n]</td>
</tr>
<tr>
<td>PRINT f [, iolist]</td>
<td>T</td>
<td>PRINT f [, iolist]</td>
</tr>
<tr>
<td>PROGRAM pgm</td>
<td>T</td>
<td>PROGRAM pgm</td>
</tr>
<tr>
<td>READ (cilist) [iolist]</td>
<td>T</td>
<td>READ (cilist) [iolist]</td>
</tr>
<tr>
<td>READ f [, iolist]</td>
<td>F</td>
<td>READ f [, iolist]</td>
</tr>
<tr>
<td>Initial Form</td>
<td>Action Taken</td>
<td>Translated Form</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>REAL v [,v]...</td>
<td>T</td>
<td>REAL v [,v]...</td>
</tr>
<tr>
<td>REAL *len v [,v]</td>
<td>RT</td>
<td>REAL v [,v]...</td>
</tr>
<tr>
<td>RETURN [e]</td>
<td>N</td>
<td>RETURN [e]</td>
</tr>
<tr>
<td>REWIND u</td>
<td>T</td>
<td>REWIND u</td>
</tr>
<tr>
<td>REWIND (alist)</td>
<td>F</td>
<td>REWIND (alist)</td>
</tr>
<tr>
<td>SAVE [a [,a]...]</td>
<td>F</td>
<td>SAVE [a [,a]...]</td>
</tr>
<tr>
<td>STOP [n]</td>
<td>N</td>
<td>STOP [n]</td>
</tr>
<tr>
<td>SUBROUTINE sub [([d [,d]...])]</td>
<td>RT</td>
<td>SUBROUTINE sub [([d [,d]...])]</td>
</tr>
<tr>
<td>WRITE (cilist) [iolist]</td>
<td>T</td>
<td>WRITE (cilist) [iolist]</td>
</tr>
<tr>
<td>v = e</td>
<td>T</td>
<td>v = e  Arithmetic Assignment Statement</td>
</tr>
<tr>
<td>v = e</td>
<td>T</td>
<td>v = e  Logical Assignment Statement</td>
</tr>
<tr>
<td>v = e</td>
<td>I</td>
<td>v = e  Character Assignment Statement</td>
</tr>
<tr>
<td>v = v [=v...] = e</td>
<td>RT</td>
<td>v = e  Multiple Arithmetic Assignment Statement</td>
</tr>
<tr>
<td>st $ st [$st...]</td>
<td>RT</td>
<td>st $ st  Multiple Statements per coding line</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>v(e(e[(e)...])))</td>
<td>RT</td>
<td>v(e(e[(e)...])))  Subscripted Subscripts</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>td v/clist/[,vclist]...</td>
<td>RT</td>
<td>td v/clist/[,vclist]...  Data in Type Declaration DATA v/clist/[,v/clist]</td>
</tr>
</tbody>
</table>
Table 2. Actions taken by SCOFF

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Restructured by SCOFF to statement acceptable in ANSI subset</td>
</tr>
<tr>
<td>T</td>
<td>Variable names translated (truncated with ambiguity resolution) by SCOFF if longer than six characters</td>
</tr>
<tr>
<td>F</td>
<td>Flagged as not ANSI subset. Not translated by SCOFF unless TF appears</td>
</tr>
<tr>
<td>N</td>
<td>Translation not needed</td>
</tr>
<tr>
<td>I</td>
<td>Flagged as not acceptable to pre-1981 IBM compilers</td>
</tr>
</tbody>
</table>
Table 3. Notation of ANSI X3.9-1978 FORTRAN 77

In describing the form of FORTRAN statements or constructs, the following metalanguage conventions and symbols are used:

(1) Special characters from the FORTRAN character set, uppercase letters, and uppercase words are to be written as shown, except where otherwise noted.
(2) Lowercase letters and lowercase words indicate general entities for which specific entities must be substituted in actual statements. Once a given lowercase letter or word is used in a syntactic specification to represent an entity, all subsequent occurrences of that letter or word represent the same entity until that letter or word is used in a subsequent syntactic specification to represent a different entity.
(3) Brackets, [], are used to indicate optional items.
(4) An ellipsis, ..., indicates that the preceding optional items may appear one or more times in succession.
(5) Blanks are used to improve readability, but unless otherwise noted have no significance.
(6) Words or groups of words that have special significance are underlined where their meaning is described.

An example illustrates the metalanguage. Given a description of the form of a statement as:

\[
\text{CALL sub}[(\text{[a], a}...)]
\]

the following forms are allowed:

\[
\begin{align*}
\text{CALL sub} \\
\text{CALL sub ( )} \\
\text{CALL sub (a)} \\
\text{CALL sub (a, a)} \\
\text{CALL sub (a, a, a)} \\
\text{etc}
\end{align*}
\]

When an actual statement is written, specific entities are substituted for sub and each a; for example:

\[
\text{CALL ABCD (x, 1.0)}
\]
Table 4. Job control language required to execute SCOFF

```
// EXEC  PGM=SCOFF1, TIME=(0,20)
//STEPLIB  DD  DSN=ENG.BREL.PGMLIB,DISP=SHR
//OUTPUT  DD  SYSOUT=A,DCB=(BLKSIZE=400,RECFM=FB)
//TEMP  DD  UNIT=SYSDA,SPACE=(800, (200,20)), DCB=BLKSIZE=800
//FT06FOO1 DD  SYSOUT=A,DCB=BLKSIZE=1729
//INPUT  DD  *
```

The EXEC card specifies SCOFF1 to be run from library ENG.BREL. The STEPLIB statement defines where the program is in the library named ENG.BREL. The OUTPUT card image specifies the output file from SCOFF. The TEMP statement defines a temporary working file used for execution and storage of SCOFF. The FT06FOO1 statement defines FORTRAN unit six as the output unit. The INPUT statement tells the machine that the input program begins with the next statement.
such data values are removed from the Type Declaration statement and a separate
DATA statement is created for the data value assignment.

Another global action taken by SCOFF is the replacement of IMPLICIT
Type Statements. Any IMPLICIT statement is removed and individual Type
Declaration statements are generated, if necessary, to explicitly type each
variable name not already explicitly typed which does not correspond to the
default type convention of ANSI Fortran (all names represent floating point
variables unless the name begins with one of the letters I through N).

2-2 TRANSLATION (T)

Variable names that are more than six characters long are translated
by truncation to make a similar, legal, unique name. Variable name truncation
is done according to the following algorithm:

1. Drop all non-standard characters in the name.
2. If a vowel exists in the name, drop the last one; else drop the
last consonant.
3. If the name is longer than six characters, do Step 2.
4. Set I to 6; set N to 0.
5. If the name is unique, then End.
6. Replace the Ith character with the character which succeeds it in
the alphanumeric collating sequence.
7. Increment N. If N is less than 37, then do Step 5; else set I to I-
1; set N to 0; do Step 6.

Another example of direct replacement is the Multiple Assignment
Statement (permitted in CDC Fortran) which has the form

\[ A = B = C = \text{expression} \]

A Multiple Assignment Statement is expanded into two (or more) single
assignment Fortran statements:
\[ C = \text{expression} \]
\[ B = C \]
\[ A = B \]

2-3 EXAMPLES

It may be helpful to examine two statements from Table 1 in more detail. Consider the first statement:

\[ \text{ASSIGN s TO i} \]

The Action Taken is specified as T and the translated form appears to be identical with the initial form. The translated form will be identical except in those cases where the variable name i contains improper characters or is more than six characters long. SCOFF does not alter the statement segments ASSIGN, the statement number s, or TO.

Secondly, consider the statement:

\[ \text{COMPLEX*len[,] v [,v]} \ldots \]

This statement is both Restructured and Translated (RT). The restructuring consists of removing the \[ \text{*len[,] \ldots} \] while the translation ensures that every variable v is a legal name of six or fewer characters.
SECTION 3

DEMONSTRATED USES OF SCOFF

CSQ (Ref. 4) is a large two-dimensional hydrocode that was selected as a test vehicle for demonstrating the use of SCOFF. The selection was based upon the need for a large code representative of those used by the DNA community. CSQ can test the capabilities of computers proposed for purchase by DNA since it uses large amounts of both storage and CPU time.

CSQ is composed of approximately fifteen thousand cards. Most of the statements are pass-through statements with perhaps ten percent needing conversion. After application of SCOFF, a few days of hand work by a programmer were required to finish the conversion.

CSQ was first processed at Boeing via SCOFF. It was then transported to AFWL for trial runs on the DNA computer. The first runs were made under the NOS/BE system. Then when the operating system changed to NOS, some CSQ runs were made under the new operating system. The conversion was successful and made CSQ available as a benchmark code.

Appendix A presents a sample computer run of SCOFF. The input stream contains a number of statements that are non-ANSI. The output stream shows the translated statements. In addition, a set of SCOFF translation messages are presented. These messages would be used by a programmer if further translation was deemed necessary for the benchmark test.
SECTION 4

SUMMARY

SCOFF provides a positive step towards improving the development of fair, representative benchmarks for the evaluation of competing computer systems for DNA. The advantages of computer translation are apparent: the operation frees the programming staff from tedious work; it is fast, without error, and, within the limits described previously, complete. The changes which require a programmer's services are often challenging but usually such that they can be solved quickly.

Many programs already exist that are too large to translate by hand with any great confidence in the accuracy of the output. CSQ is an example of such a program. As larger computers and larger computer programs proliferate, the SCOFF technique will help reduce the cost of benchmarking and upgrading at a computer facility.


This appendix presents a sample computer run of SCOFF that demonstrates typical inputs, outputs, and translation messages. The input program was written for illustrative purposes and does not solve any particular physics problem.
INPUT TO SCOFF

PROGRAM TESTSCOFF
PARAMETER (IDIM=100)
IMPLICIT INTEGER*4 (R-S)
DIMENSION ARRAYONE(IDIM), ARRAYTWO(IDIM)
REAL*4 ARRAYTHREE(IDIM)/IDIM/0. /
INTEGER VARIABLE1, VARIABLE2/2/, VARIABLE3
CHARACTER*1 YES
EQUIVALENCE (ARRAYONE, ARRAYTHREE)
DATA YES/'Y'/
PRINT 5
5 FORMAT(* THIS PROGRAM DOESNT DO ANYTHING!*)
DO 500 VARIABLE1=1, IDIM
VARIABLE3=1-1) VARIABLE1
ARRAYTWO(VARIABLE1)ARRAYTHREE(VARIABLE1)+VARIABLE2 VARIABLE3
4 IF (ARRAYTWO(VARIABLE1)) 100, 100, 500
100 ARRAYTWO(VARIABLE1)=VARIABLE1 VARIABLE3
500 CONTINUE
PRINT 6
6 FORMAT(* DO YOU WANT A PRINT?<Y OR N> #$)
READ YES, ANS
IF (ANS.EQ.'YES') PRINT 7, (ARRAYTWO(R), R=1, IDIM)
7 FORMAT(* ALL NUMBERS SHOULD BE POSITIVE COUNTING BY 2-->
, -1/1X, 1P$10.3))
DO 600 VARIABLE2=1, IDIM
ARRAYONE(VARIABLE2)=DIVIDEITUP(ARRAYTWO(VARIABLE2))
600 CONTINUE
PRINT 6
READ YES, ANS
IF (ANS.EQ.'YES') PRINT 8, (ARRAYTHREE(S), S=1, IDIM)
8 FORMAT(* ALL NUMBERS SHOULD BE POSITIVE SEQUENTIAL-->
, -1/1X, 1P$10.3))
CALL FLIPAROUND(ARRAYONE, IDIM, ARRAYTHREE)
PRINT 6
READ YES, ANS
IF (ANS.EQ.'YES') PRINT 9, ARRAYONE
9 FORMAT(* ALL NUMBERS SHOULD BE SEQUENTIAL ALTERNATING SIGN-->
, -1/1X, 1P$10.3))
PRINT 10
10 FORMAT(* ALLLLLLDONE!!!*)
STOP
END
REAL*4 FUNCTION DIVIDEITUP(RAY)
DIVIDEITUP=RAY/2
RETURN
END
SUBROUTINE FLIPAROUND(ARRAYONE, IDIM, ARRAYTHREE)
REAL$4 ARRAYONE(IDIM)
DO 5000 J=1, IDIM
K=(-1)KJ
K=K(J)=J
5000 CONTINUE
RETURN
END
PROGRAM TESTSCOFF
PARAMETER (IDIM=100)
INTEGER R, S
IMPLICIT INTEGER*4 (R-S)
DIMENSION ARRAYN(IDIM), ARRATW(IDIM)
REAL ARRTHR(IDIM)
INTEGER VARBL1, VARBL2, VARBL3
CHARACTER*1 ANS, YES
EQUIVALENCE (ARRAYN, ARRTHR)
DATA ARRTHR / IDIM=0. /
DATA VARBL1 / 2 /
DATA YES/ 'Y'/
PRINT 5
5 FORMAT(33H THIS PROGRAM DOESN'T DO ANYTHING!)
DO 500 VARBL1=1, IDIM
VARBL3=(-1)*VARBL1
ARRATW(VARBL1)=ARRTHR(VARBL1)+VARBL2*VARBL3
4 IF(ARRATW(VARBL1))100, 100, 500
100 ARRATW(VARBL1)=VARBL1*VARBL2
500 CONTINUE
PRINT 6
6 FORMAT(33H DO YOU WANT A PRINT? Y OR N> S)
READ *, ANS
IF(ANS.EQ.YES)PRINT 7, (ARRATW(I), I=1, IDIM)
7 FORMAT(44H ALL NUMBERS SHOULD BE POSITIVE COUNTING BY 2-->, (/1X, 1P -6E10.3))
DO 600 VARBL2=1, IDIM
ARRAYN(VARBL2)=DIVDTP(ARRATW(VARBL2))
600 CONTINUE
PRINT 8
8 FORMAT(44H ALL NUMBERS SHOULDN'T BE POSITIVE SEQUENTIAL-->, (/1X, 1P8E10 -3))
CALL FLPRND(ARRAYN, IDIM, ARRTHR)
PRINT 9
9 FORMAT(33H ALL NUMBERS SHOULD BE SEQUENTIAL ALTERNATING SIGN-->, (/ -1X, 1P8E10.3))
PRINT 10
10 FORMAT(33H ALLOOLONE!!)
STOP
END
REAL*4 FUNCTION DIVDTP(RAY)
DIVDTP=RAY/2
RETURN
END
SUBROUTINE FLPRND(11111111111, 111111111112)
REAL 111111111111112
DO 5000 J=1, 11111
K=11111J
11111112(J)=K
5000 CONTINUE
RETURN
END
SCOFF TRANSLATION MESSAGES

---

PROGRAM TESTSCOFF

*** SYNTAX ***
---
PARAMETER (IDIM=100)
PARAMETER STATEMENT NOT ALLOWED IN ALL FORTRAN LANGUAGES
---
REAL*4 ARRAYTHREE(IDIM)/IDIM=0./
DATA ARRAYTHREE / IDIM=0. /
---
INTEGER VARIABLE1, VARIABLE2/2/, VARIABLE3
DATA VARIABLE2 / 2 /
---
CHARACTER#1 AMS.YES
CHARACTER STATEMENT NOT ALLOWED IN ALL FORTRAN LANGUAGES
---
6 FORMAT (N DO YOU WANT A PRINT? Y OR N> #$)
CHARACTER ($) IS NOT A VALID FORMAT TYPE.
INTEGER R S
---

END OF TRANSLATION

00000010

22
7000: IF ( KARD(LAST+1).NE. KBL ) GO TO 7001

LAST = LAST + 1
GO TO 7000

C

7001: CONTINUE
GO TO 190, 4, 15, 16, 18, 19, 21, 22, 24, 28, 41, 30, 32, 33, 34, 35, 36, 37, 38, 39, 40, 42, 43, 44, 49, 45, 48, 49, 54, 55, 56, 64, 65, 92.
3 6100, 8200, 8200, 8200, 8200, 8200, 8200, 8800, 8800, 8200
4 J, C

C

7009: KOL = 8
9 LAST = 3
10 ASSIGN 13 TO MOEQ
11 K = KMAX - 1
12 DO 13 LAST, K
13 IF ( KARD(I1).EQ. KSPK(I1) ) GO TO 14
12 CALL ERROR
13 TYPE = 1
GO TO 11
14 TYPE = 2
CALL MULTEQ(I1)
GO TO 4
11 CALL SUBSUB
GO TO 4
C

C

C

ASSIGN 12345 TO N.

C

KOL=LAST
CALL SYMBOL
IF ( KODE .EQ. MINA ) CALL ERROR
1 TYPE = 1
CALL UPDATE
KOL = KOL + 1
CALL SYMBOL
IF ( KODE .NE. MINA ) CALL ERROR
1 TYPE = 2
CALL UPDATE
GO TO 4
C
C BACKSPACE, ENDFILE, REWIND, ETC.
KOL=LAST
ITYPE=1
CALL SYMBOL
IF ( KODE .EQ. MINA ) CALL UPDATE
GO TO 4

BUFFER IN /// BUFFER OUT.

ITYPE=1
GO TO 25

BLOCK DATA, FUNCTION, PROGRAM, SUBROUTINE.

ENDCD = .TRUE.
IF ( NREC.EQ.1.OR.NSYM.LT.2 ) GO TO 19
ENDCD = .FALSE.
CALL ERR(16,DUMMY,DUMMY)
IF ( UOPDAT ) CALL SORT(1)
NREC=1
NSYM=0
NEXT=MXL

IF ( J.NE.44.OR.NROUT.EQ.0 ) GO TO 20
NROUT=0
MRT=0

UOPDAT = .TRUE.

--- SET UP PAGE CAPTION FROM PREVIOUS CARD
DO 201 I=1,LAST
201 JOB(I) = KARD(I)
DO 202 I=LAST+1,LAST+65
202 JOB(I+1) = KARD(I)

--- REPLACE THE BLANK AFTER 'SUBROUTINE' OR WHATEVER
JOB(LAST+1) = KBL
IF ( KMAX .GT. 64 ) GO TO 204
DO 203 I=KMAX,64
203 JOB(I+2) = KBL

CALL CENTOR

IF ( J .EQ. 5 ) GO TO 4
KOL = LAST
CALL FNCHEK(0)
CALL RETRN(0)
GO TO 4

CALL XYZ (ARG LIST).

KOL=LAST
ITYPE=1
CALL SYMBOL
IF ( KODE .NE. MINA ) CALL ERROR
ITYPE=4
IF ( KRSX .EQ. KSPK(1) ) GO TO 9009
CALL UPDATE
GOTO 11

COMMON STATEMENT.
24:  !TYPE = 5
25:  !KODE = 0
26:  IF ( KARD(LAST+1) .NE. KSPK(4) ) GO TO 25
27:  ---- LABELLED COMMON. SKIP LABEL
28:  !KOL = LAST+1
29:  !CALL SYMBOL
30:  !GO TO 26
31:  !KOL = LAST
32:  !CALL SYMBOL
33:  IF ( !KODE ) 4,26,27
34:  !CONTINUE
35:  !CALL UPDATE
36:  !GO TO 26
37:  !C.
38:  !C.
39:  !COMPLEX STATEMENT.
40:  !C.
41:  !TYPE=6
42:  !CALL TTEST( A95, LAST, IMPLGO, 8, 10 )
43:  !C--- TYPE DECLARATION STATEMENT. SEE IF FOLLOWED BY "FUNCTION"
44:  11 = LAST
45:  N = KSTIJ(11,28)
46:  !DO 286 !I=1,N
47:  !IF ( KARD(11) .EQ. KSTIJ(1,28) ) GO TO 286
48:  !IF ( KARD(11) .NE. KBL ) GO TO 288
49:  !GO TO 2850
50:  !CONTINUE
51:  !LAST = 1
52:  !GO TO 18
53:  289: !CALL SLACHS( LAST )
54:  !GO TO 4
55:  !C.
56:  !C.
57:  !DATA STATEMENT.
58:  !C.
59:  !TYPE=7
60:  !ASSIGN 25 TO NOEO
61:  !C--- LOOK FOR = OR / 
62:  DO 292 !I=LAST,KMAX
63:  !IF ( KARD(11) .EQ. KSPK(1) ) GO TO 14
64:  !IF ( KARD(11) .EQ. KSPK(4) ) GO TO 294
65:  !CONTINUE
66:  !GO TO NOEO, ( 25, 830 )
67:  !C.
68:  !C.
69:  !DECIDE, ENCODE.
70:  !C.
71:  !KOL = LAST
72:  !TYPE=1
73:  !CALL SYMBOL
74:  !IF ( KODE .LT. 0 ) GO TO 4
75:  !CALL SYMBOL
76:  !CALL UPDATE
77:  !GO TO 26
78:  !C.
79:  !C.
80:  !DIMENSION STATEMENT.
81:  !C.
32  I TYPE=9
33  GO TO 25
34  DO STATEMENT.
35  DO 38 I=LAST,KMAX
36     IF (KARD(I).EQ.KSPK(1)) GO TO 37
37     IF (KARD(I).EQ.KSPK(2)) GO TO 39
38     CONTINUE
39     CALL ERROR
40     GO TO 4
41  END STATEMENT
42  EXECUTE SORT AND PRINT INDEX
43  I TYPE=21
44  GO TO 28
45  CALL UPDATE
46  KOL=KOL-1
47  I TYPE=21
48  GO TO 28
49  IF (KMAX.GT.LAST) GO TO 9
50  CALL SORT(1)
51  GO TO 3
52  ENTRY STATEMENT.
53  ASSIGN 205 TO NOEQ
54  GO TO 10
55  EQUIVALENCE STATEMENT.
EXTERNAL STATEMENT.

CALL XTRMAl
GO TO 4

---- FIND STATEMENT
ASSIGN 440 TO NOEQ
GO TO 10
CALL ERROR
GO TO 90

FORTRAN OR TYPE STATEMENT.

KOL=LAST
GO TO 0

GO TO (1,2,3,4,5), N  //  GO TO 1  //  GO TO N  //  ETC.

ASSIGN 460 TO NOEQ
GO TO 10
KOL=LAST
ITYPE=1
CALL SYMBOL
IF (KODE) 4,47,48
CALL UPDATE
GO TO 47

IF ACCUMULATOR OVERFLOW STATEMENT.

DO 50 I=LAST,KMAX:
IF (KARD(I).EQ.KABC(23)) GO TO 51
CONTINUE
CALL ERROR
GO TO 4

KOL=1
ITYPE=1
CALL SYMBOL
IF (KODE) 4,52,53
IF (KODE.GE.MINA.AND.KODE.LE.MAXA) CALL ERROR
CALL UPDATE
GO TO 52

IF (IDIVIDE CHECK) // IF (END FILE) // IF (SENSE LIGHT).
IF (SENSE SWITCH) 1,2

DO 55 I=LAST,KMAX
IF (KARD(I).EQ.KSPK(5)) GO TO 51
CONTINUE
CALL ERROR
GO TO 4
C
IF (ARITH) 1,2,3 /// IF (LOGICAL) STATEMENT.
C
LOOK FOR PARENTHESIS COUNT OF ZERO.
C
MPAR=0
DO 58 I=LAST,KMAX
  IF (KARD(I).NE.KSPK(3)) GO TO 57
  MPAR=MPAR+1
  GO TO 58
57 IF (KARD(I).NE.KSPK(3)) GO TO 58
  MPAR=MPAR-1
  IF (MPAR) 59,59,58
58 CONTINUE
CALL ERROR
GO TO 4
C
SCAN SYMBOLS WITHIN PARENTHESIS.
C
J=KMAX
IB=1
KMAX=IB
I.TYPE=1
KOL=LAST
CALL SUBSUB
62 KOL=IB
KMAX=J
C
LOOK FOR DIGIT IMMEDIATELY AFTER PARENTHESIS. IF ONE IS NOT
FOUND, THE IF STATEMENT IS PROBABLY LOGICAL.
C
DO 63 I=1,10
  IF (KARD(KOL+1).EQ.KDIG(I)) GO TO 52
  CONTINUE
63 IF (KARD(KOL+1).NE.KBL) GO TO 6
  IB = IB + 1
  GO TO 62
C
INTEGER STATEMENT.
C
I.TYPE=12
CALL TDTST( &95, LAST, IMPLGO, 2, 4 )
GO TO 285
C
LOGICAL STATEMENT.
C
I.TYPE=13
CALL TDTST( &95, LAST, IMPLGO, 1, 4 )
GO TO 285
C
PAUSE, STOP, RETURN STATEMENT.
C
ASSIGN 4 TO NOEQ
PAGE 7 OF MAIN
(IF IKRES .EQ. KSPK(1) ) CALL ERREN

GO TO 60

READ 5, LIST

ITYPE=17
GO TO 68

REAL STATEMENT.

ASSIGN #30 TO NOEQ
GO TO 200
ITYPE=18
CALL TDTEST( A95, LAST, IMPLGO, 4, 8 )
GO TO 205

TYPE STATEMENT.

KOL=LAST
GO TO 6

WRITE OUTPUT TAPE 5, 6, LIST

ITYPE=19
KOL=LAST+5
GO TO 72

WRITE TAPE 5, LIST.

ITYPE=19
GO TO 78

WRITE 5, LIST.

ITYPE=19
GO TO 79

NAMELIST STATEMENT.

ITYPE=14
GO TO 25

--- M MEMBER = NAME STATEMENT ( FROM UPDATER TAPE )
MOVE = -1
MREC = MREC-1
GO TO 4

--- DEFINE FILE STATEMENT
KOL = LAST
GO TO 97

--- IMPLICIT STATEMENT. TEST TO SEE IF FIRST STATEMENT IN PROGRAM

--- PROCESS IMPLICIT STATEMENT
IMPLGO = 1
KOL = LAST
GO TO 6

KOL = LAST+2

PAGE 9 OF MAIN
CALL IMPEY( IDX )

IF ( IDX .EQ. 1 ) GO TO 6

IMPLGO = 0

90 KARD(1) = KB(3)
KARD(2) = KSPK(1)
97 CALL ERR( 19,0,0 )
GO TO 4

C------ ---- FORMAT STATEMENT
99 CALL FORTRAN( KMAX, KARD, .TRUE. )
MOVE = 2
NCARD = NCARD-1
GO TO 4

C------ ---- CHARACTER STATEMENT
8100 ITYPE = 3
CALL ERR( 19,5,10H CHARACTER )
IF ( KARD(LAST+1) .NE. KSPK(8) ) GO TO 8120
KOL = LAST + 2
NUMLY = .TRUE.
CALL SYMBOL
LAST = KOL
NUMLY = .FALSE.
8120 IF ( IMPLGO .EQ. 1 ) GO TO 95
GO TO 285
8190 GO TO 285
8200 ---- CLOSE
6200 ITYPE = J - 40
ASSIGN 0250 TO M0EQ
8250 CALL ERR( 18,4,ERRMSG(ITYPE-20) )
GO TO 4

C------ ---- INTRINSIC FUNCTION
6800 CONTINUE
CALL ERR( 18,5,10HINTRINSIC )
GO TO 4

6800 CONTINUE
CALL ERR( 18,5,10HPARAMETER )
GO TO 4
END
BLOCK DATA

THIS BLOCK DATA CONTAINS ALL THE INDEX DATA STATEMENTS.

LOGICAL/I OTAB(256)
COMMON /TAPES/ WI, W2, W3, W4, WI, W5
COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(13)
COMMON /BETA/ MXCH, MXLI, N15B, LINE, NPAGE, NSBO, KCOO, KOUT(80)
COMMON /GAMMA/ OTAB
COMMON /DELTA/ MOVE, JCARD(80), ICARD(1600), IOL, NCARD
COMMON /KAPPA/ KAP(11)
COMMON /OMEGA/ KLAST(4), KSTOP(4)
COMMON /SIGMA/ KST1(11), KST2(11), KST3(11), KST4(11), KST5(11), KST6(11)
COMMON /DIM/ KST1(11), KST2(11), KST3(11), KST4(11), KST5(11), KST6(11)
KST7(11), KST8(11), KST9(11), KST10(11), KST11(11), KST12(11), KST13(11)
COMMON /DIM/ KST14(11), KST15(11), KST16(11), KST17(11), KST18(11), KST19(11)
COMMON /DIM/ KST20(11), KST21(11), KST22(11), KST23(11), KST24(11), KST25(11)
COMMON /DIM/ KST27(11), KST28(11), KST29(11), KST30(11), KST31(11), KST32(11)
COMMON /DIM/ KST34(11), KST35(11), KST36(11), KST37(11), KST38(11), KST39(11)
COMMON /DIM/ KST40(11), KST41(11), KST42(11), KST43(11), KST44(11), KST45(11)
COMMON /DIM/ KST46(11), KST47(11), KST48(11), KST49(11), KST50(11), KST51(11)
COMMON /DIM/ KST52(11), KST53(11), KST54(11), KST55(11), KST56(11), KST57(11)
COMMON /DIM/ KST58(11), KST59(11)

DATA KBL, KDIG(10), IW1, IH1, IH2, IH3, IH4, IH5, IH6, IH7, IH8, IH9/
DATA KABC/1HA, 1HB, 1HC, 1HD, 1HE, 1HF, 1HG, 1HH, 1HI, 1HK, 1HL, 1HM, 1HN, 1HO, 1HP, 1HQ, 1HR, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ/
DATA KSPK/1HA, 1HB, 1HC, 1HD, 1HE, 1HF, 1HG, 1HH, 1HI, 1HK, 1HL, 1HM, 1HN, 1HO, 1HP /
DATA MXCH, MXLI, N15B, LINE, NPAGE, 6, 5191, 5192, 5193, 0/
DATA KAP/1HA, 1HB, 1HC, 1HD, 1HE, 1HF, 1HG, 1HH, 1HI, 1HK, 1HL, 1HM, 1HN, 1HO, 1HP /
DATA KLAST, KSTOP/1HL, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ/

DATA KST1/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/ 9/
DATA KST2/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST3/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST4/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST5/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST6/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST7/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST8/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST9/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST10/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST11/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST12/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST13/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST14/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST15/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST16/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST17/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST18/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST19/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST20/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/
DATA KST21/1HM, 1HN, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ, 1/

PAGE 1 OF BLOCK DATA
SUBROUTINE CENTOR

THIS ROUTINE CENTERS THE PAGE CAPTIONS.

COMMON /TYPE, JOB(86), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K 
RX, LIST(310), LOCN, MINA, MAXA, MCHAR, NEXT, NRT, NREC, NROUT, NSYM

COMMON /NERE, NERO, MEMKEY
COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(11)
COMMON /BETA/ MXCH, MXL1, M1SB, LINE, NPAGE
COMMON /OTA/ KEY(24)
COMMON /KAPPA/ KAP(11)
COMMON /OMEGA/ KLAST(4), KSTOP(4)
COMMON /SIGMA/ KSTJ(11,60)

COMPRESSION STATEMENT BY ELIMINATING MULTIPLE BLANKS.

1 DO 2 I=1,66
   IF (JOB(I).NE.KBL) GO TO 3
2   CONTINUE
   RETURN
335   JOB(I)=JOB(I)
        J=1
        IB=1+1
4 DO 4 I=IB,66
   IF (JOB(I).EQ.KBL.AND.JOB(I-1).EQ.KBL) GO TO 4
   J=J+1
   JOB(J)=JOB(I)
4   CONTINUE
4 IB=J+1
5 DO 5 I=IB,66
   JOB(I)=KBL
5   CENTER HEADING.
6 IB=IB-J/2
6   JB=IB
   JOB(I)=JOB(J)
   J=J-1
   IF (J) 7,7,6
7 IF (I.EQ.1) RETURN
    IB=IB-1
8 DO 8 I=1,IB
    JOB(I)=KBL
   RETURN
   END
SUBROUTINE ERREN
COMMON IYPE, JOB(60), KARD(13280), KBUFF(80), KLEAR(6), KMAX, KODE, KOL
COMMON /ALPH/ KBL,KABC(20), KBIG(10), KSPK(11)
CALL ERROR

DO 15 I=KOL,KMAX
   IF ( KARD(I) .EQ. KSPK(2) ) GO TO 25
   IF ( KARD(I) .EQ. KSPK(6) ) GO TO 25
15   CONTINUE
RETURN

25   KOL = KOL-5
     N = I-KOL-1
     CALL MOVER( 0, DUMMY, DUMMY, N )
     RETURN
END
SUBROUTINE ERROR
COMMON ISS9955(ISS9955), NERR, NERO, MEMKEY
COMMON /TAPES/ N1, N2, NS, N4, N5, N6

C
CALL PAG (1)
WRITE (NO.2)
NERR = NERR+1
RETURN

C
2 FORMATE( '### SYNTAX ###')
END

PAGE 1 OF ERROR
9005 FORMAT(' TWO SUCCESSIVE LETTERS. ')
9006 FORMAT(' P MULTIPLIER USED ON (',A1,') FORMAT. ')
9007 FORMAT(1X,A1, ' FORMAT NOT IN ANSI SUBSET LANGUAGE. ')
9008 FORMAT(1X,A1, ' FORMAT NOT ANSI STANDARD. ')
9009 FORMAT(' NO LEADING COUNT ON (X) FORMAT. ')
9010 FORMAT(' Z FORMAT NOT ASA STANDARD. ')
9011 FORMAT(' NO LEADING COUNT ON L, PAREN IN FORMAT. ')
9012 FORMAT(' MORE THAN ONE PERIOD IN A STRING. ')
9013 FORMAT(' TROUBLE IN AN APOST. HOLLERITH FORMAT. ')
9014 FORMAT(' A STRING ENDING WITH (',A1,') DID NOT HAVE ALL THE RIGHT ')
9015 FORMAT(' STUFF. ')
9016 FORMAT(' ARGUMENTS ON AN ENTRY STATEMENT ARE NOT ALLOWED IN THE CD ')
9017 FORMAT(1X,A2, 'IS NOT A FORTRAN-SUPPLIED SUBPROGRAM ON THE CDC 6600 ')
9018 FORMAT(' ## STATEMENT TYPE NOT ALLOWED IN IBM FORTRAN: ',A2)
9019 FORMAT(' HEX')
9020 END
RETURN
C--- ---- MATCH FOUND. IF UNTRANSLATABLE, PRINT OUT ERROR MESSAGE
21 IF ( KEY .GT. 0 ) GO TO 25
C--- ---- INTRINSIC FUNCTION NAME FOUND IN EXTERNAL STATEMENT. FLAG
C IN TEXT1 SO THAT NO CONVERSION TAKES PLACE.
TEXT1(1,1) = STAR
RETURN
25 IF ( I .GT. NTR ) GO TO 41
IF ( I .GT. NDT ) GO TO 31
NAME = TEXT2(I)
CALL MOVER( COUNT2(I)+1, NAME, 1, COUNT1(I) )
IF ( SYMTAB(ILOC) .NE. TEXT1(I) ) RETURN
C--- ---- REMOVE OLD NAME FROM SYMBOL TABLE
ILOC = ILOC-1
CALL UPDATE
RETURN
CREASE UPDATE
RETURN
C--- ---- ELIMINATE FUNCTION NAME DBLE OR SNGL.
31 K = KOL
KOL = K-5
CALL MOVER( 0, DMMY, DMMY, COUNT1(I) )
KOL = K
IF ( SYMTAB(ILOC) .EQ. TEXT1(I) ) ILOC = ILOC-1
RETURN
C--- ---- UNTRANSLATABLE NAME
41 CALL ERR( 14, DMMY, NAME )
RETURN
C--- ---- INITIALIZE
101 DO 111 I=1,NFNS
111 TEXT1(I,1) = BLANK
RETURN
END
SUBROUTINE FORTRAN (KMAX, IFMT, DTEFLG)

--- 360 TO 6900 FORMAT TRANSLATOR.

INTEGER M2 IFMT(2,1), LEE, LHH, KBL, INO, IAMS, LPAREN, RPAREN
INTEGER KEYCHR, JCARD
LOGICAL DTEFLG
LOGICAL LLEAD, LPMLT, LLETN, LPRED, LCOMA, EFIFLG
LOGICAL JHCOM, ONEMN
DIMENSION IAMS(4), INO(10)
COMMON DELTA, MOVE, JCARD(2,20), ICARD(1600), IOL, NCARD
COMMON OSTMN, FRSTSB, IC, JC
DATA LEE / 1HE / LHH / IHH / KBL / 1H /
DATA INO / 1HO,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9 /

--- INITIALIZATION.

CALL ICHAR (IFMT, KMAX)
K = 0
L = 0
ASSIGN 120 TO NGO
ASSIGN 20 TO NGO
LPAREN = 0
RPAREN = 0
EFIFLG = .FALSE.
N6 = 6

--- MOVE FORMAT STATEMENT NO. (ASSUME NO. EXISTS.)

DO 10 I=1,6
K = K + 1
L = L + 1
10 JCARD(1,K) = IFMT(1,L)

--- ASSUME THE STRING "FORMATI" EXISTS.

L = L + 1
20 IF (IFMT(2,L) .EQ. 48) GO TO 20
20 IF (IFMT(2,L) .EQ. 39) ASSIGN 50 TO NGO
ASSIGN 40 TO NGO
GO TO 700
40 GO TO NGO, (20,50)
50 LPAREN = LPAREN + 1
GO TO 128

--- A STRING OF SOMETHING HAS BEEN COMPLETED. DID IT CONTAIN

--- ALL THE RIGHT THINGS.

120 IF (JHCOM .AND. LCOMA) GO TO 126
120 CONTINUE
I-MOUNT = 0

Cnx--- PREPARE FOR A STRING OF SOMETHING. WE WILL WANT TO CHECK
Cnx--- AT THE END OF THE STRING TO SEE THAT NO ILLEGAL STRING WAS
Cnx--- ENCOUNTERED.
Cnx

130 1LLEAD = .FALSE.
1LPMIL = .FALSE.
1LLETR = .FALSE.
1LPERD = .FALSE.
1LCOMA = .FALSE.
1JHCOM = .TRUE.
GO TO 150

Cnx--- PICK UP NEXT CHARACTER.
Cnx

140 CONTINUE
1JHCOM = .FALSE.
1L = L + 1
1IF ( L .GT. KMAX ) GO TO 730
151 KOG = IFMT(2, L)
Cnx

1A B C D E F G H I J K L M
1N O P Q R S T U V W X Y Z
1160, 170, 180, 180, 260, 180, 180, 180, 160, 440, 160, 170,
10 1 2 3 4 5 6 7 8 9 = , ( , ) / * + - # . $ , BL NONE
Cnx


160 KERR = 1
1GO TO 195
164 KERR = 2
1GO TO 195
165 KERR = 3
1GO TO 195
168 KERR = 4
1GO TO 195
170 IF ( IFMT(2, L+1) .EQ. 14 .OR. IFMT(2, L+1) .EQ. 28 ) GO TO 220
173 KERR = 6
1GO TO 195
175 KERR = 7
1ASSIGN 126 TO MGO
1GO TO 195
178 KERR = 8
1GO TO 195
183 KERR = 9
1GO TO 195
185 KERR = 10
1GO TO 195
187 KERR = 11
1GO TO 195
190  KERR = 13
   CALL ERR ( KERR,L,IFMT )
   GO TO 120
C
195  CALL ERR ( KERR,L,IFMT )
   LLEAD = .TRUE.
   LPMLT = .TRUE.
   LLETTR = .TRUE.
   LPEDR = .TRUE.
   LCOMA = .TRUE.
   GO TO 700
C
197  -------- A FORMAT. LEADING NO. OK. LAGGING NO. OK. LEAD LETTER NOT.
210  IF ( LLETTR ) GO TO 165
   IF ( LPMLT ) GO TO 168
   LLEAD = .TRUE.
   LPMLT = .TRUE.
   LLETTR = .TRUE.
   LPEDR = .TRUE.
   GO TO 700
C
213  -------- TEST FOR BN OR BZ
220  :ASSIGN 225 TO MGO
   GO TO 700
225  :ASSIGN 126 TO MGO
   GO TO 700
C
227  -------- G FORMAT; NOT IN ANSI SUBSET
230  IF ( LLETTR ) GO TO 165
   KERR = 5
   CALL ERR ( KERR,L,IFMT )
   IFMTT ( L ) = LEE
C
235  -------- E. TEST IF OF FORM EW.DEE
238  IF ( .NOT. LLETTR ) GO TO 242
   IF ( KE EQ .4 OR KE EQ .5 OR KE EQ .7 ) GO TO 239
   GO TO 178
239  IF ( .NOT. LPEDR OR NE .GT. 1 ) GO TO 178
   NE = NE + 1
   GO TO 700
C
243  -------- D FORMAT. LEADING P OK. LEADING NO. OK. LAGGING NO. OK.
C
247  -------- LATER PERIOD REQUIRED. LEADING LETTER NOT OK.
240  IF ( LLETTR ) GO TO 165
242  NE = 0
   KE = KOG
   LPMLT = .TRUE.
   LLEAD = .TRUE.
   LLETTR = .TRUE.
   LPEDR = .TRUE.
   LCOMA = .TRUE.
   GO TO 700
250  CONTINUE
255  CONTINUE
260  CONTINUE
265  GO TO 160
C
C---- H FORMAT. MOVE 'KOUNT' CHARACTERS.
C
280:  IF ( LLETR ) GO TO 165
280:  IF ( .NOT. LLEAD ) GO TO 164
280:  IF ( LPMLT ) GO TO 166
280:    LCOMA = .TRUE.
280:    ASSIGN 281 TO MGO
280:    GO TO 700
281:    K = K - 1
281:  CONTINUE
281:    K = K + 1
281:    L = L + 1
282:  IF ( IFMT(1,L) .EQ. 47 ) GO TO 2810
283:    JCARD(1,K) = IFMT(1,L)
283:    KOUNT = KOUNT - 1
283:    IF ( KOUNT .GT. 0 ) GO TO 281
283:    GO TO 120
C
C---- P FORMAT. LEADING NO. OK. A D,E,F, OR G MUST EVENTUALLY FOLLOW.
C
380:  LPMLT = .TRUE.
380:  LCOMA = .FALSE.
380:  GO TO 700
C
C---- X FORMAT. LEADING NUMBER MUST EXIST. LEADING LETTER NOT OK.
C
440:  IF ( LLETR ) GO TO 165
440:  IF ( .NOT. LLEAD ) GO TO 175
440:    IF ( LPMLT ) GO TO 166
440:      LLETR = .TRUE.
440:      LCOMA = .TRUE.
440:      ASSIGN 120 TO MGO
440:      GO TO 700
C
C---- A NUMBER. CAN I HAVE A NUMBER NOW.
C
470:  IF ( LLEAD ) GO TO 474
472:    LLEAD = .TRUE.
472:    KOUNT = IFMT(2,L) - 27
472:    GO TO 700
474:  IF ( LLETR ) GO TO 700
474:  IF ( LPMLT ) GO TO 472
474:    KOUNT = 10*KOUNT + IFMT(2,L) - 27
474:    GO TO 700
C
C---- A COMMA. CAN WE HAVE A COMMA HERE.
C
580:  CONTINUE
583:    LCOMA = .TRUE.
583:    ASSIGN 120 TO MGO
583:    GO TO 700
C
C---- L. PAREN. MUST BE PRECEDED BY A NUMBER.
590: CONTINUE
  LPAREN = LPAREN + 1
  LLEAD = TRUE.
  LPMLT = TRUE.
  LLET = TRUE.
  LPERD = TRUE.
  GO TO 583

C----- ---- A SLASH. SLASHES CAN COME ANYWHERE COMMAS CAN.
C
  600 CONTINUE
  GO TO 583
C----- ---- R. PAREN. LIKE A SLASH OR A COMMA.
C
  610 RPAREN = RPAREN + 1
  GO TO 583
C----- ---- A PERIOD. CAN I HAVE A PERIOD HERE.
C
  650 IF ( LPERD ) GO TO 185
  LPERD = TRUE.
  GO TO 700
C----- ---- AN APOST. SHIFT INTO HOLLERTH.
C
  670 IF ( LLEAD ) GO TO 187
  IF ( LPMLT ) GO TO 187
  IF ( LLET ) GO TO 187
  IF ( LPERD ) GO TO 187
  KEYCHR = I (FMT(1,L))
  871 CONTINUE
  KOUNT = 0
  KK = L
  873: KOUNT = KOUNT + 1
  IF ( KOUNT .GT. 1680 ) GO TO 877
  KK = KK + 1
  874: IF ( I (FMT(1,KK)) .NE. KEYCHR ) GO TO 873
C----- ---- SECOND APOST. ENCOUNTERED.
C
  KSAV = KK
  KK = KK + 1
  875: IF ( I (FMT(1,KK)) .NE. KEYCHR ) GO TO 876
C----- ---- DOUBLE APOST.
C
  IFMT1(1,KK) = 47
  IFMT2(2,KK) = 48
  GO TO 873
  876: KK = KSAV
  IFMT1(1,KK) = KBL
  IFMT2(2,KK) = 48
  GO TO 876
  877: CALL PAG (1)
  WRITE (NS, 0037)
  878: KOUNT = KOUNT - 1
  NO = KOUNT

PAGE 5 OF FORTRN
C----- ---- BUILD '35H' IF KOUNT IS 35.
C
DO 680 I=1,4
  MTPY = 10**M(4-I)
  NMOD = NO / MTPY
  IANS(I) = [NO(MMOD+1)
680   NO = NO - NMOD*MTPY
DO 682 I=1,4
  IF ( IANS(I) .NE. NO(1) ) GO TO 683
682  IANS(I) = KBL
683  DO 686 I=1,4
  IF ( IANS(I) .EQ. KBL ) GO TO 686
  K = K + 1
684  JCARD(1,K) = IANS(I)
685  CONTINUE
  K = K + 1
  JCARD(1,K) = LHH
  GO TO 261
C----- ---- BUMP K. STORE A CHARACTER. GO GET THE NEXT ONE.
C
699  EFGFLG = .FALSE.
700  K = K + 1
  JCARD(1,K) = IFMT(1,L)
  GO TO 890, ( 120,126,140,40,281 )
C----- ---- ALL CARDS HAVE BEEN SCANNED. WRAP IT UP AND GO HOME.
C
730  IF ( LPALEN .EQ. RPALEN ) GO TO 740
  CALL PAG ( 3 )
  WRITE (NO.9030) LPALEN, RPALEN
740  CONTINUE
  JC = K + 1
  MOVE = 1
  RETURN
9030  FORMAT(19HNO. OF L. PARENS =12 / 19HNO. OF R. PARENS =12 )
9037  FORMAT( ' SOMETHING WRONG WITH APOST. Hollerith FMT.' )
END
SUBROUTINE IMPSET( IDX )
INTEGER COMMA, Rparen
LOGICAL OUDPAT
COMMON I TYPE, JOB ( 66 ), KARD ( 1326 ), KBuff ( 90 ), KLEAR ( 8 ), KMAX, KODE, KOL, K
LIST, NMAX, MMAX, MCHAR, MEXT, MRT, MRRC, MROUT, NSYM
2, MERR, MERO, MEMBER
COMMON / ALPH / KBL.KABC ( 28 ), KDIAG ( 10 ), KSPI ( 11 )
COMMON / HASH / LOC, LOC, NAME, IVOL
COMMON / XUPDAT/ OUPDAT, ICDC ( 27 )
DATA COMMA, Rparen / 1H, 1H /
OUPDAT = . TRUE.
IFLAG = IABS ( ITYPE )
IF ( IFLAG .GT. 16 ) GO TO 25
GO TO 1 23, 25, 3, 25, 5, 6, 25, 6, 25, 6, 25, 25, 25, 12, 13, 25, 25, 25, 25, 25, 18 )
C----- ----- FIND FIRST CHARACTER
95  DO 96 I = 1, 26
   IF ( KARD ( KOL ) .EQ. KABC ( I ) ) GO TO 97
96  CONTINUE
GO TO 105
C----- ----- FIND SECOND CHARACTER
97  IF ( KARD ( KOL + 1 ) .EQ. KSPI ( 7 ) ) GO TO 970
   J = I
GO TO 99
970  KOL = KOL + 2
   DO 98 J = 1, 26
      IF ( KARD ( KOL ) .EQ. KABC ( J ) ) GO TO 99
98  CONTINUE
GO TO 105
99  DO 100 K = 1, J
   1CDC ( K ) = KEY
100  CONTINUE
C----- ----- TEST FOR END OF STATEMENT
   IF ( KARD ( KOL + 1 ) .NE. COMMA ) GO TO 101
101  KOL = KOL + 2
GO TO 95
   IF ( KARD ( KOL + 1 ) .NE. Rparen ) GO TO 105
   IF ( KOL + 2 .GE. KMAX ) GO TO 120
   KOL = KOL + 2
   IDX = 1
GO TO 150
C----- ----- ERRORS
KOL = M1NO( KOL, KMAX )
IF ( MOVE .EQ. 1 ) CALL MOVER( 0, DMIMY, DMEXT, 0 )
JC = JC - 1
CALL TRANCE( JCARD, JC )
GO TO 44
411 KOL = KOL - 1
IF ( KRSX .NE. KSPK(10) ) KOL = KMAX
CALL TRANCE( KARD, KOL )
44 MOVE = 0
JC = 0
MROUT = 0
FRSTSB = .TRUE.
IF ( KRSX .EQ. KSPK(10) ) GO TO 42
IF ( MREC ) 5, 5, 8
50 C = 1
SET UP PAGE CAPTION.
55 IF ( KBUFF(1) .EQ. KSPK(8) ) GO TO 380
CALL EQUAT4( 0, KOB, KBUFF(7) )
CALL CENTER
GO TO 8
C COPY THROUGH COMMENTS, LOOKING FOR FORTRAN STATEMENTS.
7 CONTINUE
CALL TRANCE( KBUFF, 72 )
CALL READ( KBUFF, 461 )
GO TO 62
S1 CALL STARTL( KBUFF )
S2 CONTINUE
8 IF ( KBE.EQ.KABC(3).OR.KB1.EQ.KSPK(10)) GO TO 7
IF ( KBE.EQ.KSPK(8)) GO TO 37
C---- ----FLAG CONTINUATION CARDS INTERSPERSED WITH COMMENTS.
IF ( KBB .EQ. KBL .OR. KB6 .EQ. KDI1(1) ) GO TO 50
CALL ERROR
C HAVE FORTRAN STATEMENT. START PACKING KARD.
80 MREC = MREC + 1
NCARD = NCARD + 1
CALL EQUAT4( 0, KOUT, KBUFF )
CALL EQUAT4( 72, KARD, KBUFF )
K7=7
K72=72
C LOOK FOR CONTINUATION CARDS AND PACK IN KARD.
C DO 13 J=2, 20
10 CALL READ( KBUFF, 471 )
GO TO 72
71 CALL STARTL( KBUFF )
72 CONTINUE
IF ( KBUFF(1) .NE. KABC(3) ) GO TO 105
C---- ----ELIMINATE COMMENT CARDS INTERSPERSED WITH CONTINUATIONS.
CALL TRANCE( KBUFF, 72 )

PAGE 2 OF INPUT
GO TO 10
DO 11 I=1,72
IF (KBUFF(I).NE.KBL) GO TO 12
CONTINUE
GO TO 10

IF (KB1.EQ.KABC(3).OR.KB1.EQ.KSPK(4).OR.KB1.EQ.KSPK(8).OR.KB8.EQ.
KBL.OR.KB8.EQ.KDIG(1)) GO TO 16
K7=K7+66
CALL EQUAT4(66,KARD(K7),KBUFF(7))
L = 72
DO 13 I=1,8
L = L+1
ICHAR = KBUFF(L)
OSEQ2(I) = ICHAR

NINETEEN CONTINUATION CARDS. LOAD EMPTY BUFFER.

CALL READ( KBUFF, &81)
GO TO 82
CALL STARL ( KBUFF )
CONTINUE
DO 15 I=1,72
IF (KBUFF(I).NE.KBL) GO TO 16
CONTINUE
GO TO 14
GET STATEMENT NUMBER, IF ANY.

KODE=0
OSTMTN = .FALSE.
DO 19 I=1,5
IF (KARD(I).EQ.KBL) GO TO 19
DO 17 J=1,10
IF (KARD(I).EQ.KDIG(J)) GO TO 18
CONTINUE
IF (KARD(I).EQ. KSPK(4)) GO TO 20
---- JCL SKIP CARD
CALL PAGE(1)
WRITE (6,170)
FORMAT(10X25H/ JCL CARD DELETED. */
GO TO 44
IF (J.EQ.1.AND.KODE.EQ.0) GO TO 19
OSTMTN = .TRUE.
GO TO 20
CONTINUE
SQUEEZE OUT ALL BLANKS. PUT BLANK IN KARD(1), AND START SQUEEZE
IN KARD(2).

KMAX = K7 + 65
KARD(6)=KBL
CONTINUE
RETURN
LOOK FOR #LAST OR #STOP CONTROL CARDS.

360 NEWSUB = 1
   GO TO 370
370 NEWSUB = 0

   DO 38 J=2,72
      IF (KBUFF(J).EQ.KBL) GO TO 38
      IF (KBUFF(J).NE.KLAST(I)) GO TO 39
      I=I+1
   CONTINUE

38 GO TO 400

   DO 40 J=2,72
      IF (KBUFF(J).EQ.KBL) GO TO 40
      IF (KBUFF(J).NE.KSTOP(I)) GO TO 400
      I=I+1
   CONTINUE

40 IF (NEWSUB) 7,7,55

CONTINUE

41 CALL CLOSE
   WRITE (NB,46)
   STOP

42 KOL = KOL + 2
   KMAX = KMAX + 1
   KARD(1) = KBL
   CALL EQUAT4( 5, KARD(2), KARD(1) )
   CALL EQUAT4( KMAX, KARD(7), KARD(KOL) )
   GO TO 36

46 FORMAT ('0 END OF TRANSLATION')
   END
SUBROUTINE INSRT3(L1, TEXT, NCHAR, KEY) 
INTEGER BLANK, EQUAL, JCARD(SO), PCARD, PMAX 
LOGICAL OSTMN, FRSTSB 
REAL NSEQ1(8), SEQ2, SEQ3 
COMMON /SEQ1, SEQ2, SEQ3 
COMMON (TYPE, JB(86), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K 
RSX, LIST(S191), LOCN, MINA, MAXA, NCHXX, NEXT, NRT, NREC, NROUT, NSYM 
COMMON /ALPH/ KBL, KBZ(80), KS(11) 
COMMON /DELTA/ MOVE, PCARD(SO), ICARD(1600), IOL, NCARD 
COMMON /TAPES/ M1, M2, N4, N5, N6 
EQUIVALENCE ( OCHAR, ICHAR, ( BLANK, KBL), ( OSEQ111), SEQ1 
C----- ---- GENERATE SEPARATE FORTRAN STATEMENT FOR DELETED TEXT 
C----- ---- IF THE ORIGINAL STATEMENT HAD A STATEMENT NUMBER AND IF 
C----- ---- THIS IS THE FIRST NEW STATEMENT GENERATED FROM THE ORIGINAL 
C----- ---- STATEMENT, PUT THE STATEMENT NUMBER FIELD INTO THIS CARD. 
C----- ---- OTHERWISE, BLANK OUT STATEMENT NUMBER FIELD. 
C----- ---- MOVE DELETED TEXT TO NEW CARD IMAGE AND BLANK OUT REMAINDER 
C----- ---- IF ( NCHAR .LE. 0 ) GO TO 42 
C----- ---- IF ( L1 .GE. 0 ) GO TO 41 
C----- ---- IF ( J .LE. JMAX ) GO TO 50 
C----- ---- WRITE (H4,90) JCARD
SUBROUTINE MOVER ( NT, TEXT, INV, MT)  
INTEGER BLANK  
LOGICAL OSTMN, FRSTSB  
LOGICAL TEXT(1), OCHAR, OSEQQ(8)  
REAL *8 SEQ1, SEQ2, SEQ3  
COMMON /SEQ/ SEQ1, SEQ2, SEQ3  
COMMON /DELTA/ MOVE, JCARD(80), I CARD(1600), IOL, NCARD  
1 1, OSTMN, FRSTSB, IC, JC  
COMMON IYPE, JOB(80), KARD(1326), KBUFF(80), KLEAD(80), KMAX, XODE, KOL, K  
C>R, LIST(8191), LOCN, MINA, MAXA, NCHAR, NEXT, NT, NREC, NROUT, NSYM  
2 COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(11)  
COMMON /NUM/ NCM  
EQUIVALENCE ( OCHAR, ICHAR ), ( OSEQQ(11), SEQ2 )  
DATA BLANK /1H/, BLANK /3H/  
C----- ONTEST FOR CHARACTERS MOVED TO NEW CARD  
C KOL COLUMN COUNTER IN DEBLANKED CARD  
C IC COLUMN POINTER IN OLD CARD  
C JC COLUMN POINTER IN NEW CARD  
C NT = NUMBER OF CHARACTERS FROM THE TEXT FIELD TO BE  
C MOVED TO THE NEW CARD IMAGE.  
C TEXT = CHARACTERS TO BE MOVED  
C INV = INCREMENT IN TEXT FIELD ( 1=LOGICAL, 4=INTEGER)  
C MT = NUMBER OF NON-BLANK CHARACTERS TO SKIP AFTER THE MOVE  
C-----  
C 1: IF ( JC , 8T, 1 ) GO TO 10  
C 2: IF ( JC , 8T, 1 ) GO TO 20  
C 3: IF ( JC , 8T, 1 ) GO TO 30  
C 4: IF ( JC , 8T, 1 ) GO TO 40  
C 5: IF ( JC , 8T, 1 ) GO TO 50  
C 6: IF ( JC , 8T, 1 ) GO TO 60  
C 7: IF ( JC , 8T, 1 ) GO TO 70  
C 8: IF ( JC , 8T, 1 ) GO TO 80  
C 9: IF ( JC , 8T, 1 ) GO TO 90  
C 10: IF ( JC , 8T, 1 ) GO TO 10  
C 11: IF ( JC , 8T, 1 ) GO TO 11  
C 12: IF ( JC , 8T, 1 ) GO TO 12  
C 13: IF ( JC , 8T, 1 ) GO TO 13  
C 14: IF ( JC , 8T, 1 ) GO TO 14  
C 15: IF ( JC , 8T, 1 ) GO TO 15  
C 16: IF ( JC , 8T, 1 ) GO TO 16  
C 17: IF ( JC , 8T, 1 ) GO TO 17  
C 18: IF ( JC , 8T, 1 ) GO TO 18  
C 19: IF ( JC , 8T, 1 ) GO TO 19  
C 20: IF ( JC , 8T, 1 ) GO TO 20  
C 21: IF ( JC , 8T, 1 ) GO TO 21  
C 22: IF ( JC , 8T, 1 ) GO TO 22  
C 23: IF ( JC , 8T, 1 ) GO TO 23  
C 24: IF ( JC , 8T, 1 ) GO TO 24  
C 25: IF ( JC , 8T, 1 ) GO TO 25  
C 26: IF ( JC , 8T, 1 ) GO TO 26  
C 27: IF ( JC , 8T, 1 ) GO TO 27  
C 28: IF ( JC , 8T, 1 ) GO TO 28  
C 29: IF ( JC , 8T, 1 ) GO TO 29  
C 30: IF ( JC , 8T, 1 ) GO TO 30  
C 31: IF ( JC , 8T, 1 ) GO TO 31  
C 32: IF ( JC , 8T, 1 ) GO TO 32  
C 33: IF ( JC , 8T, 1 ) GO TO 33  
C 34: IF ( JC , 8T, 1 ) GO TO 34  
C 35: IF ( JC , 8T, 1 ) GO TO 35  
C 36: IF ( JC , 8T, 1 ) GO TO 36  
C 37: IF ( JC , 8T, 1 ) GO TO 37  
C 38: IF ( JC , 8T, 1 ) GO TO 38  
C 39: IF ( JC , 8T, 1 ) GO TO 39  
C 40: IF ( JC , 8T, 1 ) GO TO 40  
C 41: IF ( JC , 8T, 1 ) GO TO 41  
C 42: IF ( JC , 8T, 1 ) GO TO 42  
C 43: IF ( JC , 8T, 1 ) GO TO 43  
C 44: IF ( JC , 8T, 1 ) GO TO 44  
C 45: IF ( JC , 8T, 1 ) GO TO 45  
C 46: IF ( JC , 8T, 1 ) GO TO 46  
C 47: IF ( JC , 8T, 1 ) GO TO 47  
C 48: IF ( JC , 8T, 1 ) GO TO 48  
C 49: IF ( JC , 8T, 1 ) GO TO 49  
C 50: IF ( JC , 8T, 1 ) GO TO 50  
C 51: IF ( JC , 8T, 1 ) GO TO 51  
C 52: IF ( JC , 8T, 1 ) GO TO 52  
C 53: IF ( JC , 8T, 1 ) GO TO 53  
C 54: IF ( JC , 8T, 1 ) GO TO 54  
C 55: IF ( JC , 8T, 1 ) GO TO 55  
C 56: IF ( JC , 8T, 1 ) GO TO 56  
C 57: IF ( JC , 8T, 1 ) GO TO 57  
C 58: IF ( JC , 8T, 1 ) GO TO 58  
C 59: IF ( JC , 8T, 1 ) GO TO 59  
C 60: IF ( JC , 8T, 1 ) GO TO 60  
C 61: IF ( JC , 8T, 1 ) GO TO 61  
C 62: IF ( JC , 8T, 1 ) GO TO 62  
C 63: IF ( JC , 8T, 1 ) GO TO 63  
C 64: IF ( JC , 8T, 1 ) GO TO 64  
C 65: IF ( JC , 8T, 1 ) GO TO 65  
C 66: IF ( JC , 8T, 1 ) GO TO 66  
C 67: IF ( JC , 8T, 1 ) GO TO 67  
C 68: IF ( JC , 8T, 1 ) GO TO 68  
C 69: IF ( JC , 8T, 1 ) GO TO 69  
C 70: IF ( JC , 8T, 1 ) GO TO 70  
C 71: IF ( JC , 8T, 1 ) GO TO 71  
C 72: IF ( JC , 8T, 1 ) GO TO 72  
C 73: IF ( JC , 8T, 1 ) GO TO 73  
C 74: IF ( JC , 8T, 1 ) GO TO 74  
C 75: IF ( JC , 8T, 1 ) GO TO 75  
C 76: IF ( JC , 8T, 1 ) GO TO 76  
C 77: IF ( JC , 8T, 1 ) GO TO 77  
C 78: IF ( JC , 8T, 1 ) GO TO 78  
C 79: IF ( JC , 8T, 1 ) GO TO 79  
C 80: IF ( JC , 8T, 1 ) GO TO 80  
C 81: IF ( JC , 8T, 1 ) GO TO 81  
C 82: IF ( JC , 8T, 1 ) GO TO 82  
C 83: IF ( JC , 8T, 1 ) GO TO 83  
C 84: IF ( JC , 8T, 1 ) GO TO 84  
C 85: IF ( JC , 8T, 1 ) GO TO 85  
C 86: IF ( JC , 8T, 1 ) GO TO 86  
C 87: IF ( JC , 8T, 1 ) GO TO 87  
C 88: IF ( JC , 8T, 1 ) GO TO 88  
C 89: IF ( JC , 8T, 1 ) GO TO 89  
C 90: IF ( JC , 8T, 1 ) GO TO 90  
C 91: IF ( JC , 8T, 1 ) GO TO 91  
C 92: IF ( JC , 8T, 1 ) GO TO 92  
C 93: IF ( JC , 8T, 1 ) GO TO 93  
C 94: IF ( JC , 8T, 1 ) GO TO 94  
C 95: IF ( JC , 8T, 1 ) GO TO 95  
C 96: IF ( JC , 8T, 1 ) GO TO 96  
C 97: IF ( JC , 8T, 1 ) GO TO 97  
C 98: IF ( JC , 8T, 1 ) GO TO 98  
C 99: IF ( JC , 8T, 1 ) GO TO 99  
C 100: IF ( JC , 8T, 1 ) GO TO 100  
C----- RECOVERY AT BEGINNING OF TOI MOVING  
}
SUBROUTINE PAQ (N)

THIS SUBROUTINE DOES THE GENERAL PAGE COUNTING FOR INDEX.

LOGICAL KSSBO, KCCDO
COMMON /TYPE,JOB(80),KARD(1228),KBUFF(80),KLEAR(8),KMAX,KODE,KOL,K
RSX,LIST(6191),LOCN,MINA,MAXA,NCHAR,NEXT,NRT,NNREV,NROUT,NSYM
COMMON /LVAR/ KLST(400),KLIST(16),KSTOP(4)
COMMON /SIGMA/ KSTIJ(11,60)
COMMON /TAPES/ M1, M2, M3, M4, M5, M6

I = IABS( N )
IF ( N .LE. 0 ) KCCDO = .FALSE.
IF ( KSSBO ) K = K+2
IF ( KCCDO ) K = K+2
IF ( LINE.EQ.K, .LE. MALL ) GO TO 10
WRITE (N6,100)
LINE = 0
10: IF ( KSSBO ) WRITE (N6,110) JOB
     IF ( KCCDO ) WRITE (N6,120) KOUT
     LINE = LINE+K
     KSSBO = .FALSE.
     KCCDO = .FALSE.
RETURN
100: FORMAT(151H)
110: FORMAT(150H13X66A1)
120: FORMAT(80H --- 80A1,5H --- )
END
SUBROUTINE RETRNI( MODE )

LOGICAL OTAB(200), ODX(14), OCHR
LOGICAL FIRST

COMMON /TYPE/, JOB(65), KARD(1320), KBUFF(180), KLEAR(8), KMAX, KODE, KOL, K
RSL, LIST(801), LOCN, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM

COMMON /ALPH/, KBL, KABC(26), KDIG(10), KSPK(11)

FIRST = .FALSE.
SEQS = SEGS
NVSMT = 0
I = 0
GO TO 11

5 KOL = KOL - 1
11 CALL SYMBOL

IF ( KODE .LE. 0 ) GO TO 100
IF ( KODE .EQ. MINA ) CALL UPDATE
IF ( .NOT. FIRST ) GO TO 13
FIRST = .TRUE.
ITYPE = 1
IF ( KARD(KOL) .NE. KSPK(8) ) GO TO 13
C----- ---- AN ASTERISK IN THIS POSITION INDICATES AN EXPRESSED
C TYPE DECLARATION SIZE SPECIFICATION.
C 1 .E. INTEGER FUNCTION CALCW21( ... )
LAST = KOL + 3
I = KOL
DO 31 J = KOL, LAST
30 IF ( KARD(I+1) .EQ. KSPK(3) ) GO TO 35
IF ( KARD(I+1) .NE. KBL ) GO TO 31
I = I + 1
GO TO 30
31 CONTINUE
GO TO 41
C----- ---- DROP SIZE SPECIFICATION
35 KOL = KOL - 1
CALL MOVOR( 0, DUMMY, DUMMY, I-KOL )
KOL = I + 1
13 CONTINUE
IF ( KARD(KOL) .EQ. KSPK(4) ) KARD(KOL) = KBL
IF ( KARD(KOL) .EQ. KBL ) GO TO 41
KRSX = KARD(KOL)
ODX(4) = OCHR
ODX(4) = OTAB(I+1)
IF ( I .LE. 36 ) GO TO 5
4 I KOL = KOL+1
IF ( KOL .GT. KMAX ) RETURN
GO TO 13
100 CONTINUE
RETURN
END
SUBROUTINE SIXFR7

DIMENSION OMM(10), VOWEL(8), DOLLAR(2)
REAL NAME(2), SYMTAB(2,800), NEWNM(800)
INTEGER IREF(800)
LOGICAL=1 OMM(1)
LOGICAL VAXFLG, DFFLAG, CHGFLG

COMMON /TYPE, JOB(8), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, KRSX, LIST(191), LOCN, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM
COMMON /RERR, RERO, RENKEY
COMMON /HASH/, ILOC, LOC, NAME
COMMON /SIX7/, NEWNM, IREF
COMMON /ALPH/, KBL, KABC(26), KDIG(10), KSPK(11)
COMMON /FLAGS/, DFFLAG, VAXFLG

EQUIVALENCE (NAME(1), ONM(1)), (OMM(1), OMM(1))
EQUIVALENCE (SYMTAB(1,1), LIST(4192))

DATA VOWEL /1HA,1HE,1HI,1HO,1HU,1HY/
DATA CHGFLG /FALSE/
DATA DOLLAR /1HS,1H_
DO 5 I=1,800
   IF (IREF(I).EQ.0) GO TO 200
5 CONTINUE
N4=1

FIND 6 OR LESS CHARACTER NAMES IN SYMTAB AND MOVE TO NEWNM
DO 200 I=1, ILOC
   NAME(I)=SYMTAB(I,1)
   IF (OMM(I).NE.KBL) GO TO 200
   NEWNM(I)=NAME(I)
200 CONTINUE

SEARCH FOR NAMES TO BE CHANGED

DO 20 = 1, N1, 1LOC
   IF (.NOT. VAXFLG) GO TO 15
   IF (CHGFLG) GO TO 14
   NAME(1)=SYMTAB(1,1)
   NAME(2)=SYMTAB(2,1)
   DO 240 J=6,16
      IF (OMM(J).EQ.KBL) GO TO 16
      IF (OMM(J).NE.DOLLAR(I).AND.OMM(J).NE.DOLLAR(2)) GO TO 240
   14 CHGFLG=.TRUE.
   J=J

N7=2
10 N6=N7
   N4=N4
   DO 20 = 1, N1, 1LOC
      IF (.NOT. VAXFLG) GO TO 15
      IF (CHGFLG) GO TO 14
      NAME(1)=SYMTAB(1,1)
      NAME(2)=SYMTAB(2,1)
      DO 240 J=6,16
         IF (OMM(J).EQ.KBL) GO TO 16
         IF (OMM(J).NE.DOLLAR(I).AND.OMM(J).NE.DOLLAR(2)) GO TO 240
      14 CHGFLG=.TRUE.
      J=J

M4=1
M7=J
GO TO 50
CONTINUE
240
GO TO 18
15 NAME(I)=SYMTAB(I,1)
NAME(2)=SYMTAB(2,1)
18 IF(OMM1(B).EQ.KBL)GO TO 17
CHFGB*.FALSE.
M4=1
M2=1
GO TO 30
17 IF(.NOT.CHFGB)GO TO 20
CHFGB*.FALSE.
J1=8
M4=1
M2=1
GO TO 75
20 CONTINUE
25 RETURN
C
LOOK FOR LAST VOWEL IN NAME
C
30 DO 40 J=M2+1
   IF(OMM1(17-J).EQ.KBL)GO TO 40
   DO 40 K=1,J
   IF(OMM1(17-J).NE.VOWEL(K))GO TO 40
J1=17-J
GO TO 50
40 CONTINUE
GO TO 90
C
VOWEL FOUND - DELETE AND CREATE NEW NAME
C
50 IF(J1.EQ.16)GO TO 70
   DO 60 J=J1,15
   OMM1(J)=OMM1(J+1)
   IF(OMM1(J+1).EQ.KBL)GO TO 71
60 CONTINUE
70 OMM1(18)=KBL
71 IF(CHFGB)GO TO 10
   IF(OMM1(8).EQ.KBL)GO TO 75
   IF(J1.EQ.3)GO TO 90
   M2=18-J1
GO TO 30
C
LOOK FOR DUPLICATE
C
75 DO 80 I=1,1LOC
   IF(NEWMM1(I).NE.NAME(1))GO TO 90
   IF(J1.EQ.3)GO TO 90
   IF(J1.EQ.2)GO TO 100
   IF(J1.EQ.1)GO TO 115
   M2=18-J1
GO TO 30
CONTINUE

NAME IS UNIQUE - ADD NAME TO NEWNM AND FLAG IREF WITH THE
NUMBER OF CHARACTERS IN THE NEW NAME

NEWNM(N4)=NAME(1)
DO 95 I=3,6
IF((ONM1(I)).NE.KBL)GO TO 95
IREF(N4)=I-2
GO TO 86
95 CONTINUE
N4=N4+1
IF(N4.LE.1LOC)GO TO 7
GO TO 25

NO MORE VOWELS - DELETE LAST CHARACTER

J1=2
N3=0
IF((ONM1(J)).EQ.KBL)GO TO 97
DO 98 J=8,10
ONM1(J)=KBL
98 CONTINUE
ISAVE=7
GO TO 75
97 DO 98 J=3,7
IF((ONM1(J-1)).EQ.KBL)GO TO 98
ISAVE=10-1
GO TO 100
98 CONTINUE
ISAVE=3

STILL NOT UNIQUE, CHANGE LAST CHARACTER TO NUMBER

IF(IN3.EQ.10)GO TO 110
N3=N3+1
ONM1(ISAVE)=KDIG(N3)
GO TO 75

STILL NOT UNIQUE, CHANGE LAST CHARACTER TO LETTER

N5=0
J1=1
IF(IN5.EQ.26)GO TO 120
N5=N5+1
ONM1(ISAVE)=KABC(N5)
GO TO 75
120 NAME(1)=SYMTAB(1,N4)
NAME(2)=SYMTAB(2,N4)
WRITE(6,140) NAME
N4=N4+1
IF(N4.LE.1LOC)GO TO 7
GO TO 25
140 FORMAT(27'HOCAN'T FIND UNIQUE NAME FOR ,A8,A8)
END
SUBROUTINE SUB8(KOL1, KOL2)

THIS SUBROUTINE SUBSTITUTES THE 6 CHARACTER NAME FOR THE 7 CHARACTER NAME.

LOGICAL DGFLAG, VAXFLG
LOGICAL=1 ONM(8), ONM
COMPLEX=10 SYMTAB(100)
REAL=9 NEWMM(900), NME
INTEGER=2 IREF(800)

COMMON / TYPE, JOB(80), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K
RSX, LIST(8191), LOCN, MIN1, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM
2, NERR, NENO, MEMKEY

COMMON / ALPH/ KBL, KABC(28), KDIG(10), KSPK(11)
COMMON / HASH/ LOC, NAME
COMMON / FLAPP/ DGFLAG, VAXFLG
COMMON / IX7/ NEWMM, IREF

EQUIVALENCE (ONM(11), NME)
EQUIVALENCE (ONS, KSAVE)
EQUIVALENCE (SYMTAB(1), LIST(5192))

KEND=KMAX
ITYPE=1
KOL=KOL1
KMAX=KOL2

CALL SYMBOL
CALL UPDATE

IF (IREF(Loc), EQ.0) GO TO 70
NME=NEWMM(LOC)
K=KOL2=KOL1
L=K+1
DEL=L-IREF(LOC)
IF (DEE, EQ.0, OR, DEL, EQ.1) GO TO 30
IF (KOL2, EQ., KEND) GO TO 30
IF (DEL, LT.0) GO TO 20
DO 15 = KOL2, KEND
15 KARD(I-DEL)=KARD(I)
CONTINUE
KOL2=KOL2-DEL
KEND=KEND-DEL
GO TO 30
DO 25 = KOL2, KEND
25 COUNT=KEND+KOL2-1
KARD(COUNT-DEL)=KARD(COUNT)
CONTINUE
KOL2=KOL2-DEL
KEND=KEND-DEL
GO TO 30
DO 40 = KOL1, KOL2
40 ONS=ONM(1+2-KOL1)
KARD(1)=KSAVE
CONTINUE
KMAX=KEND
IF (I, NOT. DGFLAG) GO TO 70
70: RETURN
80: FORMAT(1X, A10, 13H REPLACED BY , A9, 59H IN SUBROUTINE, ENTRY, FUNCTION, OR NAMED COMMON STATEMENT.)
    END
SUBROUTINE SORT (ISO)

REAL*8 NAME, X(2, 800), BLANK, TDNAME, OUTBF(10), STAR
REAL*8 SEQ1, SEQ2, SEQ3
LOGICAL SYML(1, 1)
LOGICAL OUPDAT
LOGICAL MORE, ONAME, ODX(4), OIDX, OTAB(256), OCOMMA, OBLANK
LOGICAL TYTL / .FALSE. /

DIMENSION JCDICT(27), MX(1), JLOC(1)
COMMON /TYPE/, JOB(88), KARD(1328), KBUFF(80), KLFAR(8), KMAX, KODE, KOL, K
COMMON /LIST/(810)
COMMON /SEQ/, SEQ1, SEQ2, SEQ3
COMMON /NAME/, LOC, NAME
COMMON /XUPDAT/, OUPDAT, JCDICT(27)
COMMON /NAME/, OTAB

EQUIVALENCE (ONAME, NAME), (ODX(1), IDX(1)), (ODIX(4), OIDX(1)), (OCOM, OBLANK), (OBLANK, BLANK),

EQUIVALENCE (SYTAB(1), LIST(4192)), (JLOC(1), LIST(7592))

EQUIVALENCE (SYTAB(1), SYML(1, 1))

DATA TDNAME /SHCOMPLEX, SHINT, SHREAL /
DATA JCDICT /SH5, SH3, SH5, 0/, COMMA, BLANK /1H, 1H /
DATA STAR /SH3SH3SH3/

IF (.ISO. EQ. 0) GO TO 200
TYTL = .FALSE.

IF (ILOC LE 0) RETURN
DO 50 I = 1, ILOC
  NAME(I) = SYTAB(1, I)
  IF (ONAME(I) EQ. OBLANK) GO TO 50
  IF (TYTL) GO TO 40
  TYTL = .TRUE.
  PRINT 70, JOB
  PRINT 80, NAME(I)
  CONTINUE
50 CONTINUE
RETURN
70 FORMAT('0ILLEGAL VARIABLE NAMES IN ', A)
80 FORMAT(2XAB)
END
SUBROUTINE SUBSUB

REAL*8 SSNAME(25), K0D1(25)
INTEGER AMPSND, L1(126), KCLASS(25)
COMMON TYPE, J0B(10), KARD(1225), KBUFF(80), KLEAR(8), KMAXKODE, KOL, K
CALL LIST(0101), LOCN, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM
COMMON /ALPH/, KOL, KARD(20), KD(10), KSPK(11)
COMMON /DEL*, MOVE, JCARD(80), ICARD(1000), IOL, NCARD
COMMON /H1U/, NH, KOL8X
COMMON /HASH/, LOC, I
DATA AMPSND / 00 /
DATA SSNAME /
DATA / 8H1585B17, 8H1585B27, 8H1585B37, 8H1585B47, 8H1585B57 /
1: IF ( KARD(K0D1+1) .EQ. AMPSND ) GO TO 40
1: CALL SYMBOL
2: K0L6 = K0L0
3: PRINT 611, IPAR, LP0NT, K0L, KMAX, LOC, KRSX, ( KARD(IPX), IPX=K0L, K0L6)
611: FORMAT(' SUBSUB 17, 515, 2X1, 2X7A1)
7: IF ( K0L .GT. KMAX ) GO TO 100
8: IF ( KRSX .EQ. KSPK(10) ) GO TO 100
9: L1(IPAR) = K0L6X
10: CALL UPDATE
11: IF ( KRSX .NE. KSPK(3) ) GO TO 20
C---- ARRAY NAME OR FUNCTION REFERENCE
12: KCLASS(IPAR) = LOC
13: I1PG = IPAR
14: IPAR = IPAR+1
15: IF ( IPAR .GT. 25 ) GO TO 200
16: IF ( LOC .GE. 0 ) GO TO 15
C---- ARRAY
17: CALL MOVE( 0, D0MNY, D0MNY, 0 )
18: L1(IPAR-1) = -JC
19: PRINT 613, IPAR, JC
613: FORMAT(' SUBSUB 17', 615)
20: GO TO 10
C---- FUNCTION REFERENCE
21: IF ( KRSX .EQ. KSPK(5) ) GO TO 25
22: CONTINUE
23: PRINT 620
620: FORMAT(' SUBSUB 20', 615)
24: IF ( KRSX .EQ. KSPK(5) ) GO TO 25

PAGE 1 OF SUBSUB
C-----  ---- COMMA, BRANCH IF INSIDE FUNCTION
22 CONTINUE
   PRINT 622, IPAR, L1(IPAR-1),ILPG,LPCNT,KOL
622 FORMAT (SUBSUB 26',6I5)
   IF ( IPAR .LE. 1 ) GO TO 10
   IF ( L1(IPAR-1) .GE. 0 ) GO TO 10
   IF ( IPAR .GT. ILPG ) GO TO 13
   ILPG = ILPG-1
   LPCNT = LPCNT+1
   KOL = KOL-1
   CALL INSRTR ( L1(IPAR-1),SSNAME(LPCNT),7,1 )
   KOL = KOL+1
   GO TO 13
C-----  ---- RIGHT PAREN, IF FIRST LEVEL, SKIP.
25 CONTINUE
   IPAR = IPAR-1
   PRINT 625, IPAR, L1(IPAR),ILPG, LPCNT, KOL
625 FORMAT (SUBSUB 26',6I5)
   IF ( L1(IPAR) .GE. 0 ) GO TO 30
   IF ( IPAR .GE. ILPG ) GO TO 30
   ILPG = ILPG-1
   LPCNT = LPCNT+1
   KOL = KOL-1
   CALL INSRTR ( L1(IPAR),SSNAME(LPCNT),7,1 )
   KOL = KOL+1
30 CONTINUE
   IF ( KOL .GT. KMAX ) GO TO 100
   KRSX = KARD(KOL)
   IF ( KRSX .EQ. KSPK(5) ) GO TO 25
   IF ( KRSX .EQ. KSPK(2) ) GO TO 22
   IF ( KRSX .EQ. KBL ) GO TO 30
   IF ( KRSX .EQ. KSPK(10) ) GO TO 100
   GO TO 10
40 CONTINUE
   IF ( KOL .GE. KMAX ) RETURN
C-----  ---- LOOK FOR XXX WHERE XX IS A STATEMENT NUMBER
200 CONTINUE
   CALL ERR(17,DUMMY,DMNY)
   WRITE (16,222) KARD
222 FORMAT(10X90A1)
   STOP
   END
SUBROUTINE SYMBOL

THIS SUBROUTINE INSPECTS KARD STARTING AT KOL+1 FOR THE PRESENCE
OF FORTRAN VARIABLES OR STATEMENT NUMBERS. IF FOUND, THE SYMBOL
IS PACKED IN KODA AS A NUMBER WHOSE BASE IS 37.

LOGICAL=1 OMM(8), ONAME(7), OCHAR, OTAB(256), ODX(4), OCHAL
LOGICAL DICT, NUMLY
REAL=0 NAME, BLANK
COMMON ITYPE, IOB(86), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KDE, KOL, K
RES, LIST(8191), LOC, MNA, MXA, NCHAR, NEXT, MRT, MREC, NROUT, NSYM
2: MERR, MENO, MERROR
COMMON /ALPH/ KBL, KABC(20), KDIG(10), KSPK(11)
COMMON /BETA/ MXCH, MXLI, N15B, LNE, MPAGE, KSO, KCDO
COMMON /GAMMA / OTAB
COMMON /DELTA/ MOVE, JCARD(80), JCARD(18001), JOL, NCARD
1: OSMTH, FRSTSB, IC, J, IFAR, LI(25), KLASS(25)
COMMON /KAPPA/ KAP(11)
COMMON /MU/ HH, KOLSX
COMMON /SIGMA/ KSTJ(11, 60)
COMMON /OMEGA/ KLASS(4), KSTOP(4)
COMMON /TAPES/ N1, N2, N3, N4, N5, N6
COMMON /HASH/ ILOC, LOC, NAME(2), IVAL, NUMLY

DIMENSION KSYM(37)
EQUIVALENCE (KBL, KSYM(1))
EQUIVALENCE (OMM(1), NAME(1)), (OMM(2), ONAME(1)), (OCHAR, KRSX)
N: (ODX(1), I), (OCHAL, KLSX)

DATA BLANK /IH/

START. SET UP INITIAL CONDITIONS.
1: KODE=0
NCHAR=0
I = 0
NAME(1) = BLANK
NAME(2) = BLANK

SEARCH FOR FIRST SYMBOL.
2: KLSX=KARD(KOL)
KOLSX = KOL
3: KOL=KOL+1
IF (KOLGT,KMAX) GO TO 23
KRSX = KARD(KOL)
ODX(4) = OCHAL
SUBROUTINE TTEST( N, LAST, BR, 11, 12 )
INTEGER BR
COMMON  ITYPE, JOB(60), KARD(1328), KBUFF(80), KLEAR(6), KMAX, KDE, KOL
COMMON /ALPH/ KBL,KABC(28), KDIG(10), KSPK(13)

10 KOL = LAST
   IF ( KARD(LAST+1) .EQ. KSPK(8) ) GO TO 50
   IF ( KARD(LAST+1) .NE. KBL ) GO TO 200
   LAST = LAST + 1
   GO TO 10
50 LAST = LAST+2
   KARD(LAST-1) = KBL
60 IF ( KARD(LAST) .EQ. KDIG(11+1) ) GO TO 150
   IF ( KARD(LAST) .EQ. KDIG(12+1) ) GO TO 150
   IF ( 12 .LT. 10 ) GO TO 99
   11 = 12/10
   12 = 12 - 10*11
   GO TO 80
99 CALL ERROR
   RETURN
100 LAST = LAST + 1
   GO TO 80
150 KARD(LAST) = KBL
   KMAX = KMAX - 2
   DO 170 1 = KOL, KMAX
   170 KARD(1+1) = KARD(1+3)
200 CONTINUE
   IF ( BR .EQ. 1 ) RETURN 1
   RETURN
END
SUBROUTINE UPDATE

THIS SUBROUTINE UPDATES THE SYMBOL TABLE. NEW SYMBOLS ARE ADDED
AS THEY ARE FOUND. REFERENCES TO PREVIOUSLY FOUND SYMBOLS ARE
UPDATED.

LOGICAL O1, OTYPE(4), QNAME(8), FORCEM, FORCES
LOGICAL OUPDAT, KSBO, KCDO
REAL NAME(2), SYMTAB(2, 800)
INTEGER# T1, T2, TT(2), N2, NN(4), N3
INTEGER# JLOC(400), CTAB(400), TEMP, HTAB(1)
COMMON ITYPE, JOS(80), KARD(1320), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K
COMMON LIST(8191), LOC, MINA, MAXA, HCHAR, NEXT, MRT, MREC, MROUT, NSYM
COMMON /NERM, NERO, NMEMKEY
COMMON /ALPH, KBL, KABC(20), KDIG(10), KSPK(11)
COMMON /BETA, MIXH, MXXL, MXXS, LINE, NPAGE, KSBO, KCDO
COMMON /HASH / ILOC, LOC, NAME, IVAL
COMMON /XUPDAT/ OUPDAT, LCDICT(27)

EQUIVALENCE (TEMP, TT(1), TT1, TT(2), T2)
EQUIVALENCE ( NAME(1), O1, N1, NN(1) ), ( NN(3), N2 ), ( OTYPE(1), KEY)
EQUIVALENCE ( SYMTAB(1, 1), LIST(14192) )

DATA MST / 800 /

IF ( KODE .LT. MINA .OR. KODE .GT. MAXA ) RETURN
FORCES = .FALSE.
FORCEM = .FALSE.
ADD = MOD ( IABS, N1+N2+N3 ), 4191
TEMP = HTAB(1ADD+1)
IFLAG = IABS( ITYPE )

CHARACTER 3
COMMON 5
COMPLEX 6
DIMENSION 8

IF ( IFLAG .LT. 18 ) GO TO 25
GO TO ( 25, 25, 3, 25, 5, 6, 25, 8, 25, 25, 12, 13, 25, 25, 25, 25, 18 )

KEY = IVAL + 5
GO TO 30
KEY = 1
GO TO 40
KEY = 2
GO TO 30
KEY = 1
GO TO 40
KEY = 3
GO TO 30
KEY = 5
GO TO 30
KEY = 0
GO TO 30
ENTRY HASHIN
CALL SORT(0)
GUPDAT = .FALSE.
KSB0 = .TRUE.
JSTM = 0
DO 600 I=1,4191
   600 HTAB(I) = 0
   DO 620 I=1,400
      620 CTAB(I) = 0
   ILOC = 0
   ICOL = 0
RETURN
END
SUBROUTINE XTRNAL

COMMON I TYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(9), KMAX, KODE, KOL

C---- TEST SUBPROGRAMS IN AN EXTERNAL STATEMENT FOR VALIDITY IN
C 6500 FORTRAN.

10 KSAVE = KOL
20 CALL SYMBOL
30 IF ( KODE .LE. 0 ) RETURN
40 K = KOL
50 KOL = KSAVE
60 CALL FNCHEK(-1)
70 KOL = K
80 GO TO 11
90 END
MACRO
ALABEL L
SET ,SUBROUTINE ALABEL'
ALABEL L
SAVE (14,12),,-
LA 0,ALABEL-I0 : LOCAL DISPLACEMENT ----> GR#0
SR 15,0 : POINT GR 15 BACK TO THE CSSCT
LR 10,13 : USERS SAVE AREA ADDRESS ----> TEMP STORAGE
LR 15,15 : GLOBAL ADDRESSABILITY IN GR 13
ST 15,0(10) : ESTABLISH TRACE BACK CHAIN
ST 10,4(113) : ESTABLISH VALID BACK CHAIN
MEND
I0
SPACE 3
START
13F
ENTRY READ,WRITE,TREAD,TWRITE,OPEN,CLOSE,TRANCE
ENTRY ICHAR
ENTRY TRANSL,MOVTP,SETTP
SPACE 3
ENTRY L4CV8029
USING #,15
L4CV80:29 LR 0,1
LA 1,255
LOOP:99 STC 1,TAB(1)
BCT 1,LOOP29
LR 1,0
L 15,=V(MAIN)
BR 15
SPACE 3
USING #,15
SETTM P
LA 1,TEMP
ST 1,DCBADD
BR 14
DROP 15
USING 10,13
OPEN
SET OPEN (INPUT,OUTPUT,OUTPUT),TEMP,OUTPUT)
LA 15,0
TM INPUT+48,X'10'
BZ OPNERR
TM OUTPUT+48,X'10'
BZ OPNERR
TM TEMP+48,X'10'
BZ OPNERR
LA 1,OUTPUT
ST 1,DCBADD
ROPN L 15,4(13)
RETURN (14,12),T,RC=(15)
OPNER R LA 15,4
ROPN
CLOSE:
SET
CLOSE (INPUT,OUTPUT,TEMP)
LA 15,0
ROPN
READ
SET L 11,0(1) FETCH ARRAY ADDRESS
SET INPUT,(11)
TR 0,80,(11),INTAB 5600:
LA 1,80 5700:
LA 2,316(11) 5800:
LA 3,79(11) 5900:
L 5,*CLC' 6000:
L 6,*FO',4 6100:
A1TOA IC 4,0(3) 6200:
SRDL 4,0 6300:
ST 5,(12) 6400:
BCTR 3,0 6500:
SR 2,6 6600:
SLL 5,5 6700:
BCT 1,A1TOA4 6800:
LA 15,0 6900:
B ROPN 7000:
WRITE SET 7100:
L 11,0(1) 7200:
B ROPN 7300:
TREAD SET 7400:
L 11,0(1) 7500:
B ROPN 7600:
WRITE SET 7700:
L 11,0(1) 7800:
B ROPN 7900:
TRAN SET 8000:
LM 2,3,(11) 8100:
GET ARGUMENTS 8200:
LA 12,BUF 8300:
GET NUMBER OF CHARACTERS TO BE OUTPUT 8400:
L 9,0(3) 8500:
NUMBER MUST BE POSITIVE. 8600:
LTR 9,9 9000:
IF NOT, 9100:
BP NINEOK 9200:
LA 9,72 9300:
SET IT TO OUTPUT 1 CARD IMAGE 9400:
NINEOK SET 9500:
LA 11,0(12) 9600:
ADDRESS OF AREA BEING PACKED 9700:
LA 7,72 9800:
NUMBER OF COLUMNS TO BE PACKED (MAX) 9900:
XR 3,3 10000:
LOOP 10100:
AR 3,9 10200:
IF NOT FULL CARD, LOCATION TO BEGIN BLAN 10300:
STC 3,ML+9 10400:
NUMBER OF CHARACTERS REMAINING TO BE PAK 10500:
LR 1,9 10600:
TEST FOR FULL CARD 10700:
SR 9,7 10800:
BM A4TOA1 10900:
BRANCH IF NOT FULL CARD 11000:
LR 1,7 10000:
ELSE SET NUMBER OF COLUMNS TO PACK 11100:
A4TOA IC 3,0(2) 10500:
STC 3,0(11) 10600:
LA 11,1(11) 10700:
LA 2,4(3) 10800:
BCT 1,A4TOA1 10900:
MVC 72,19,(12),BLANKS 11000:
LTR 9,9 TEST TO SEE IF BLANKS NEEDED AT END OF ST 11100:
| LGR | 1,9 | NUMBER OF BLANKS NEEDED | 11190 |
| EX  | 1,MCL | BLANK OUT RIGHTMOST COLS OF CARD | 11200 |
| TRMV | L 1,DCBDA | TEST FOR TASK COMPLETE | 11400 |
| PUT  | (1),(12) | ALL DONE UNLESS R9 > 0 | 11600 |
| ITR  | 9,9  | SET UP FOR CONTINUATION CARDS | 11660 |
| BMP ROP | 11,9(12) | 66 COLUMNS IN A CONTINUATION CARD | 11680 |
| TRMV | MVC 0(0,12),CONT | 11700 |
| LA  | 11,9(12) | 11800 |
| LA  | 7,68 | 11800 |
| LA  | 8,6  | 12000 |
| B LOOP | MVC 0(0,12),BLANKS | 12100 |
| BLANK S | DC 15C | DON'T CHANGE ORDER OF THESE TWO CARDS | 12200 |
| ICNT  | DC C | DON'T CHANGE ORDER OF THESE TWO CARDS 2 | 12300 |
| ICHAR  | SET L 2,3,0(1) | 12500 |
| | L 3,0(3) | 12550 |
| | SR 4,4 | 12600 |
| | IC 4,0(2) | 12650 |
| | IC 4,CHAR(4) | 12700 |
| | STH 4,2(2) | 12800 |
| | LA 2,4(2) | 12900 |
| | BCT 5,XT | 13000 |
| | LA 15,0 | 13100 |
| | B ROPN | 13200 |
| | MOVMP | SET L 11,0(1) | 13300 |
| | MACLOSE (TEMP,DISP) | 13400 |
| | IN OPEN (TEMP,(INPUT)) | 13500 |
| | TM TEMP+48,X '10' | 13600 |
| | BZ OPNERR | 13700 |
| | NOP | GET TEMP,(INPUT) | 13800 |
| | COPL COPI LOOP PUT OUTPUT,(INPUT) | 13900 |
| | DPCD DPC | 14000 |
| | NE LA 1,OUTPUT | 14100 |
| | ST 1,DCBDA | 14200 |
| | COPL COPI LOOP OPEN (TEMP,(OUTPUT)) | 14300 |
| | TM TEMP+48,X '10' | 14400 |
| | BZ OPNERR | 14500 |
| | LA 15,0 | 14600 |
| | B ROPN | 14700 |
| | TRMV | SET L 11,0(1) | 14800 |
| | TR 0(0,11),TAB | 14900 |
| | PUT OUTPUT,(INPUT) | 15000 |
| | LA 15,0 | 15100 |
| | B ROPN | 15200 |
| | TRMV | SET EJECT | 15300 |
| | TAB DC 256X '5D' | 15400 |
| | ENDTAB | DC 15800 |
| | DC T AB+C'A' | 15900 |
| | ORG TAB+C'A' | 16000 |
| | ORG TAB+C'J' | 16100 |
| | DC C'JKLNMOPQR' | 16200 |
| | ORG C'ABCDEFH' | 16300 |
| | ORG C'ABCDEFH' | 16400 |
| | ORG C'JLNMOPQR' | 16500 |
DC AL1(46)
ORG CHAR=X'7D'
DC AL1(47)
ORG CHAR=C'
DC AL1(48)
ORG CHAREND
SPACE 3
INTAB
DC 256X'5D'
INTER
EQU n
ORG INTAB+C'A'
DC C'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
ORG INTAB+C'S'
DC C'0123456789'
ORG INTAB+C'.'

DC X'485904D4E4F50'
ORG INTAB+X'5A'

DC X'5A5B5C5D5E5F6061'
ORG INTAB+X'6B'

DC X'6B4D6D6D6F6F'
ORG INTAB+X'7A'

DC X'7A7E7D7D7E7F'
ORG INTAB+X'40'

DC C'

ORG INTEND
SPACE 3
DCBAD DS 1F
BUF DS 20F
EJECT

INPUT
DCS DDNAME=INPUT, DSORG=PS, MACRF=(GM), LRECL=80, RECFM=FB,

OUTPUT
DCS DDNAME=OUTPUT, DSORG=PS, MACRF=(PM), LRECL=80, RECFM=FB,

TEMP
DCS DDNAME=TEMP, DSORG=PS, MACRF=(GM, PM), LRECL=80, RECFM=FB,
<table>
<thead>
<tr>
<th>Routine Name</th>
<th>Routine Type</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN BLOCK DATA</td>
<td>PROGRAM BLOCK DATA</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>CENTER ERR</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>ERROR</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>ERR</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>FNCHEK</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>FORTRAN</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>IMPSET</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>INPUT</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>INSRT</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>MOVER</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>MLTEQ</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>PAG</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>RETANI</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SIFRA7</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SUBS</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SLASHS</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SORT</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>STARL</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SUBSUB</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>TDTST</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>UPDATE</td>
<td>ENTRY POINT</td>
<td>UPDATE</td>
</tr>
<tr>
<td>XTRNAL</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>MAIN</td>
<td>PROGRAM</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>ENTRY POINT</td>
<td>MAINTOOL</td>
<td></td>
</tr>
<tr>
<td>ROUTINE NAME</td>
<td>ROUTINE TYPE</td>
<td>LOCATION</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>----------</td>
</tr>
<tr>
<td>CENTOR</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>ERR</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>ERREND</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>ERROR</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>FUNCHEX</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>FORTAN</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>IMPSET</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>INPUT</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>INSRTR</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>MAIN</td>
<td>PROGRAM</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>MAIN</td>
<td>PROGRAM</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>MOVER</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>MULTEQ</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>PAG</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>RETRUN</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SIXPR7</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SLASHS</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SORT</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>STARL</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SUBG</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SUBSUB</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>TDTEST</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>UPDATE</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>XTRNAL</td>
<td>SUBROUTINE</td>
<td>SCOFF.TMP</td>
</tr>
<tr>
<td>BLOCK DATA</td>
<td>BLOCK DATA</td>
<td>SCOFF.TMP</td>
</tr>
</tbody>
</table>
## DISTRIBUTION LIST

### DEPARTMENT OF DEFENSE

<table>
<thead>
<tr>
<th>Organization</th>
<th>Address Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Armed Forces Radiobiology Rsch Institute</td>
<td>2 cy ATTN: Dir</td>
</tr>
<tr>
<td>Armed Forces Staff College</td>
<td>ATTN: Library</td>
</tr>
<tr>
<td>Assistant to the Secretary of Defense</td>
<td>ATTN: Executive Assistant</td>
</tr>
<tr>
<td>Atomic Energy</td>
<td>ATTN:</td>
</tr>
<tr>
<td>Defense Communications Engineer Ctr</td>
<td>ATTN: R-123</td>
</tr>
<tr>
<td>Defense Intelligence Agency</td>
<td>2 cy ATTN: RTS-2A</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: RAE</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: RAEV</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: NATA</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: RAEE</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: SPSS</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: SPD</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: NATD</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: NATA, J. Anderson</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: COMD-1, B. Robertson</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: STTII</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>4 cy ATTN: STTL</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>2 cy ATTN: STTI</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>4 cy ATTN: STTL</td>
</tr>
<tr>
<td>Defense Tech Info Ctr</td>
<td>12 cy ATTN: DD</td>
</tr>
<tr>
<td>Field Command Defense Nuclear Agency</td>
<td>Det 1 ATTN: FC-1, J. Crandley</td>
</tr>
<tr>
<td>Lawrence Livermore Lab</td>
<td>ATTN: FC-1</td>
</tr>
<tr>
<td>Field Command Defense Nuclear Agency</td>
<td>ATTN: FC-1</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: FCRT</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: FCPR</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: FCRT, S. Humphries</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: FCRT, M. Summa</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: FCRT, G. Ganong</td>
</tr>
<tr>
<td>Defense Nuclear Agency</td>
<td>ATTN: FCRT</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>ATTN: SAGA/SD</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>ATTN: SAGA/STA</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>ATTN: J-S, Nuclear Div/Strategy Div</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>ATTN: J-3</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>2 cy ATTN: NRL-STINFO, Library</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>2 cy ATTN: NRI-STINFO, Library</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>2 cy ATTN: NRI-STINFO, Library</td>
</tr>
<tr>
<td>Joint Chiefs of Staff</td>
<td>2 cy ATTN: NRI-STINFO, Library</td>
</tr>
<tr>
<td>Joint Strategic Connectivity Staff</td>
<td>ATTN: JCS/DNA, F. Tedesco</td>
</tr>
<tr>
<td>Under Secretary of Defense for Rsch &amp; Engrg</td>
<td>ATTN: Strategic &amp; Space Sys (OS)</td>
</tr>
</tbody>
</table>

### DEPARTMENT OF THE ARMY

<table>
<thead>
<tr>
<th>Organization</th>
<th>Address Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aberdeen Proving Ground</td>
<td>2 cy ATTN: Tech Library</td>
</tr>
<tr>
<td>Atmospheric Sciences Lab</td>
<td>ATTN: DELAS-AS</td>
</tr>
<tr>
<td>BMD Advanced Technology Ctr</td>
<td>ATTN: ATC-T</td>
</tr>
<tr>
<td>Deputy Chief of staff for Ops &amp; Plans</td>
<td>ATTN: DAMO-NCN</td>
</tr>
<tr>
<td>Deputy Chief Staff for Rsch Dev &amp; Acq</td>
<td>ATTN: DAMA-CSS-N</td>
</tr>
<tr>
<td>Engr Studies Ctr</td>
<td>ATTN: DAEN-FES</td>
</tr>
<tr>
<td>Harry Diamond Labs</td>
<td>ATTN: DELHD-NM-P, 20240</td>
</tr>
<tr>
<td>Harry Diamond Labs</td>
<td>ATTN: DELHD-TA-L, 81100</td>
</tr>
<tr>
<td>Harry Diamond Labs</td>
<td>ATTN: DELHD-NP</td>
</tr>
<tr>
<td>Multi Service Communications Systems</td>
<td>ATTN: DRCMP-MSCS-APB, M. Francis</td>
</tr>
<tr>
<td>US Army Ballistic Rsch Labs</td>
<td>ATTN: R. Reisler</td>
</tr>
<tr>
<td>US Army Ballistic Rsch Labs</td>
<td>2 cy ATTN: DROAR-TSB-S</td>
</tr>
<tr>
<td>US Army Chemical School</td>
<td>ATTN: ATZN-CM-CC</td>
</tr>
<tr>
<td>US Army Comb Arms Combat Dev Acty</td>
<td>ATTN: ATZL-CAN-I</td>
</tr>
<tr>
<td>US Army Comb &amp; General Staff College</td>
<td>ATTN: ACQ, Library Div</td>
</tr>
<tr>
<td>US Army Communications Command</td>
<td>ATTN: Tech Ref Div</td>
</tr>
<tr>
<td>US Army Concepts Analysis Agency</td>
<td>ATTN: Classified Documents</td>
</tr>
<tr>
<td>US Army Engr Waterways Exper Station</td>
<td>ATTN: J. Houston</td>
</tr>
<tr>
<td>US Army Engr Waterways Exper Station</td>
<td>ATTN: WESGD, P. Hadala</td>
</tr>
<tr>
<td>US Army Field Artillery School</td>
<td>ATTN: M. Swett, Library</td>
</tr>
<tr>
<td>US Army Forces Command</td>
<td>ATTN: Tech Library</td>
</tr>
<tr>
<td>US Army Intell &amp; Sec Cmd</td>
<td>ATTN: ATCL-OSS, S. Cockrell</td>
</tr>
<tr>
<td>US Army Logistics Ctr</td>
<td>ATTN: ATCL-OSS, S. Cockrell</td>
</tr>
<tr>
<td>US Army Material Dev &amp; Readiness Cmd</td>
<td>ATTN: DRCDE-D</td>
</tr>
</tbody>
</table>

89
DEPARTMENT OF THE ARMY (Continued)

US Army Materiel Sys Analysis Actvy
ATTN: ORXSY-CC

US Army Nuclear & Chemical Agency
ATTN: Library

US Army Satellite Comm Agency
ATTN: Tech Library

US Army Signal Ctr & Ft Gordon
ATTN: ATZH-CG

US Army TRADOC Sys Analysis Actvy
ATTN: ATRAA-PL

US Army War College
2 cy ATTN: Library

US Army White Sands Missile Range
ATTN: Commander

USA Missile Command
ATTN: Doc Section

DEPARTMENT OF THE NAVY

Chief of Naval Operations
Dir Cnd Control Planning & Prmgng Div, OP-940
ATTN: OP-943

David Taylor Naval Ship R&D Ctr
5 cy ATTN: Code L42-3

Naval Air Development Ctr
ATTN: Doc Con

Naval Air Systems Cnd
ATTN: Tech Library

Naval Civil Engng Lab
ATTN: L-51, S. Johnson

Naval Electronic Systems Cnd
ATTN: Tech Library

Naval Ocean Systems Ctr
ATTN: Code 4471

Naval Postgraduate School
2 cy ATTN: Code 1424, Library

Naval Rsch Lab
5 cy ATTN: Code 2627

Naval Space Surveillance System
ATTN: J. Burton

Naval Surface Weapons Ctr
ATTN: Code R44, H. Glaz
ATTN: Code F31

Naval Surface Weapons Ctr
5 cy ATTN: Tech Library & Info Svcs Br

Naval Weapons Ctr
5 cy ATTN: Code 343, FKAGA2, Tech Svcs

DEPARTMENT OF THE NAVY (Continued)

Naval Weapons Evaluation Facility
ATTN: H. Struve

Naval Underwater Systems Ctr
2 cy ATTN: Code EM, J. Kalinowski

Office of Naval Rsch
ATTN: Tech Info Svcs

DEPARTMENT OF THE AIR FORCE

Air Force Propulsion Lab
ATTN: LKDH, Stop 24, E. Haberman

Air Force Radiobiological Health Lab, AFLC
ATTN: Dir

Air Force Armament Lab
ATTN: AFFL/DLY

Air Force Rocket Propulsion Lab
ATTN: LKCP

Air Force Tech Applications Ctr
2 cy ATTN: STINFO Ofc-TF

Air Force Test & Evaluation Ctr
ATTN: J. Hoge

Air Force Weapons Lab
ATTN: AD, J. Ormand
ATTN: SUL
ATTN: NTES, R. Guice
ATTN: NTCC, J. Burgio

Air Force Wright Aeronautical Lab
ATTN: Tech Library

Air Force Wright Aeronautical Lab
ATTN: LTE

Air University Library
ATTN: AUL-LSE

Headquarters
Air Weather Service, MAC
ATTN: AWSAE

Armament Dev & Test Ctr
ATTN: Doc Con

Ballistic Missile Office
ATTN: SYDT
ATTN: EN
ATTN: EMMP, D. Van Gari

Deputy Chief of Staff
Rsch, Dev, & Acq
ATTN: AFROST
ATTN: AFRODS, Space Sys & C3 Dir
ATTN: AFRODI

Headquarters
Electronic Systems Div
ATTN: DCK, L. Staples
ATTN: XRRT
DEPARTMENT OF THE AIR FORCE (Continued)
Foreign Tech Div
ATTN: NIIS, Library

Headquarters
ATTN: J5YS, F. Smith

NORAD
ATTN: IND

Space Div
ATTN: SSP-6
ATTN: YGJ
ATTN: YKM

DEPARTMENT OF ENERGY

Department of Energy
Tech Info Ctr
ATTN: Doc Con

OTHER GOVERNMENT AGENCIES

Central Intell Agency
ATTN: OSWR/NED

Department of the Interior
US Geological Survey
ATTN: J. Dietrich

Department of the Interior, US Geological Survey
ATTN: R. Regan

Federal Emergency Management Agency
ATTN: Ofc of Rsch/NP, D. Benson

DEPARTMENT OF ENERGY CONTRACTORS

University of California
Lawrence Livermore National Lab
ATTN: P. Colella
ATTN: Tech Info Dept, Library

Los Alamos National Laboratory
ATTN: Reports Library

Sandia National Labs, Livermore
ATTN: Library & Security Classification Div

DEPARTMENT OF DEFENSE CONTRACTORS

Aerospace Corp
ATTN: M. Eskijian

Agbabian Associates
ATTN: M. Agbabian

Analytic Services, Inc
2 cy ATTN: Library

ATTN: J. Phillips

Applied Theory, Inc.
ATTN: Doc Con

APTEK, Inc.
ATTN: A. Larrabee

Astron Rsch & Engrg
ATTN: A. Levien

AVCO Systems Div
ATTN: Library, A830

BDM Corp
ATTN: J. Braddock
ATTN: M. Kelly
ATTN: L. Schlipper
2 cy ATTN: Corporate Library

BDM Corp.
ATTN: P. McClousky
ATTN: Library
ATTN: D. Peercy
ATTN: F. Leech

Beers Associates, Inc.
ATTN: Contracts Ofc

Berkeley Rsch Associates, Inc
ATTN: J. Workman

Boeing Co.
ATTN: Aerospace Library
ATTN: R. Hager
ATTN: W. Cooley
ATTN: S. Sandberg
ATTN: E. Brown
4 cy ATTN: M/5 42-37, B. Henderson

Boeing Computer Svcs, Inc.
ATTN: M/5 8C-52, D. Chapman
ATTN: B. Brown

Boeing Military Airplane Co.
ATTN: D. Pierson
ATTN: R. Syring

Booz-Allen & Hamilton, Inc
ATTN: R. Hanson

California Rsch & Tech, Inc
ATTN: T. Isabelle
ATTN: K. Kreyenhagen
ATTN: Library

Calspan Corp.
ATTN: A. Davis

Calspan Corp.
ATTN: M. Dunn
ATTN: Library

66th MI Group
ATTN: RDA

Control Data Corp.
ATTN: R. Portillo

Corrales Applied Physics Co.
ATTN: N. Froula

Cushing Associates
ATTN: V. Cushing

Decision Science Consortium, Inc.
ATTN: J. Chinnis

Dikewood
ATTN: Tech Library for K. Sabisch

Dynalectron
ATTN: C. Scott

91
DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

EG&G Wash Analytical Svcs Ctr, Inc
ATTN: Library
ATTN: J. Bridges
ATTN: J. Ridge
ATTN: L. Daley

Electro-Magnetic Applications, Inc
ATTN: B. Fisher

Electro-Mech Systems, Inc
ATTN: R. Shunk

Electrospace Systems, Inc
ATTN: H. Logston

ESL, Inc
ATTN: Library

Flow Sciences, Inc
ATTN: J. Campbell

Garjak Rsch, Inc
ATTN: G. Erickson

General Electric Co
ATTN: N. Dispensiere
ATTN: D. Coccana
ATTN: Tech Info Ctr

General Research Corp
ATTN: J. Garbarino
ATTN: G. Purvis
ATTN: D. Sangster
ATTN: J. Balter
ATTN: L. Wogulis
2 cy ATTN: Doc Con
2 cy ATTN: Tech Info Ofc

Harold Rosenbaum Associates, Inc
ATTN: H. Rosenbaum

Horizons Technology, Inc
ATTN: R. Kruger

Information Science, Inc
ATTN: W. Dudziak

Institute for Defense Analyses
ATTN: Classified Library

IRT Corp
ATTN: M. Obrien
2 cy ATTN: Library

J. D. Haltiwanger Consulting Svcs
ATTN: Doc Con

JAYCOR
ATTN: W. Hobbs
ATTN: W. Radaski

JAYCOR
ATTN: E. Wenaas
ATTN: A. Woods
ATTN: J. Young
ATTN: A. J. Woods
5 cy ATTN: Library

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

Johns Hopkins University
ATTN: F. Breilsford

Kaman Avidyne
ATTN: M. Tomayko
2 cy ATTN: Library

Kaman Sciences Corp
ATTN: D. Conroy
ATTN: F. Shelton
5 cy ATTN: Library

Kaman Tempo
ATTN: K. Schwartz
ATTN: DASIA
ATTN: C. Anderson
ATTN: A. Klein
10 cy ATTN: W. Chan

Kaman Tempo
ATTN: R. Miller
ATTN: Contracts Ofc

K Tech Corp
ATTN: B. Mason
ATTN: E. Young
JAYCOR
2 cy ATTN: Library

JAYCOR
ATTN: C. Clark

Lockheed Missiles & Space Co, Inc
ATTN: J. Henley
ATTN: T. Geers
2 cy ATTN: Reports Library

Lockheed Missiles & Space Co, Inc
ATTN: C. Hankin
2 cy ATTN: TIC-Library

Martin Marietta Denver Aerospace
ATTN: Rsch Library, 6617

Mathematical Sciences Northwest, Inc
ATTN: Doc Con

Martin Marietta Corp
2 cy ATTN: H. Griffith

Maxwell Labs, Inc
ATTN: J. Smith
ATTN: A. Kolb

McDonnell Douglas Corp
ATTN: M. Gee
ATTN: W. Gee
ATTN: D. Dean
2 cy ATTN: Tech Library Svcs

McDonnell Douglas Corp
ATTN: R. Mapes

Merritt CASES, Inc
ATTN: Library
ATTN: G. Wintergerst
DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

University of Miami
ATTN: Contract Ofc, S. Wang

Mission Research Corp
ATTN: A. Gregerson
ATTN: C. Longmire
5 cy ATTN: Doc Con

Mission Research Corp
ATTN: Library
ATTN: B. Neri

Mitre Corp
ATTN: Tech Report Ctr

Mitre Corp
ATTN: A. Schneider

Northrop Corp
ATTN: A. Scott
2 cy ATTN: RAD Effects Grp

Pacific-Sierra Rsch Corp
ATTN: S. Samenitzer
ATTN: J. Levesque
ATTN: S. Finn
ATTN: Contracts Ofc
ATTN: H. Brode, Chairman SAGE
ATTN: Doc Con

Pacific-Sierra Research Corp
ATTN: D. Gormley
ATTN: Doc Con

Pacifica Technology
ATTN: G. Kent
ATTN: M. Gittings
2 cy ATTN: Tech Library

PDA Engrg
2 cy ATTN: Doc Con

Physical Dynamics, Inc
ATTN: E. Fremouw

Physical Dynamics, Inc
ATTN: Contracts Ofc

Physics International Co
ATTN: J. Larsson
ATTN: J. Shea
ATTN: L. Burr
5 cy ATTN: Tech Library

Pulse Sciences Inc
ATTN: W. Weseloh

R & D Associates
ATTN: F. Field
ATTN: G. Ivy
ATTN: M. Wright
ATTN: M. Marshall
10 cy ATTN: Tech Inform Ctr

R & D Associates
2 Cy ATTN: Doc Con

2. F. Cross Associates, Ltd
ATTN: R. Mahoney

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

Rand Corp
2 cy ATTN: Library

Rockwell International Corp
ATTN: Library

S-CUBED
ATTN: J. Glow
ATTN: M. Scarpa
ATTN: K. Pyatt
10 cy ATTN: Doc Con
10 Cy ATTN: Library

S-CUBED
ATTN: J. Murphy

S-CUBED
ATTN: J. Crepeau

Santa Fe Corp
2 cy ATTN: Sec Ofc
2 cy ATTN: D. Paolucci

Science & Engrg Associates, Inc
ATTN: J. Prentice

Science & Engrg Associates, Inc
ATTN: J. McCullough

Science Applications, Inc
ATTN: R. Reaugh

Science Applications, Inc
ATTN: C. Dougherty

Science Applications, Inc
ATTN: B. Dial

Science Applications, Inc
ATTN: C. Burgart
10 cy ATTN: Sec Ofc

Science Applications, Inc
ATTN: Sec Ofc
ATTN: Contracts Ofc

Science Applications, Inc
ATTN: Contracts Ofc
ATTN: B. Elyea, Security Ofc

Science Applications, Inc
ATTN: Security Ofc
ATTN: B. Giplen

Science Applications, Inc
ATTN: P. Setty
ATTN: W. Layson
5 cy ATTN: Security Ofc

Science Applications, Inc
ATTN: R. McLean

Science Applications, Inc
ATTN: H. Dolatshahi
ATTN: R. Jarka
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A
DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

Science Applications, Inc
ATTN: C. Bates

Scientific Rsch Assoc., Inc
ATTN: J. Kredozsky

SRI International
ATTN: P. Wong
ATTN: B. Lew
ATTN: G. Abrahamson
ATTN: P. Dolan
9 cy ATTN: Library

SRI International
2 cy ATTN: Doc Con

Sylvania Systems Grp
2 cy ATTN: Library

Strategic Systems Div
ATTN: A. Novenski

System Planning & Analysis, Inc
ATTN: Contracts Ofc
ATTN: P. Vadola

Tetra Tech, Inc
ATTN: Contracts Ofc

Tetra Tech, Inc
ATTN: J. Preston

Toyon Rsch Corp
ATTN: J. Garbarino

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

TRW Electronics & Defense Sector
ATTN: G. Teraoka
ATTN: D. Jortner
ATTN: F. Fendell
2 cy ATTN: Tech Info Ctr

TRW Electronics & Defense Sector
ATTN: F. Sterk
ATTN: F. Straw

United Technologies Rsch Ctr
ATTN: H. Michels

Visidyne, Inc
ATTN: H. Smith

Weidlinger Assoc, Consulting Engrg
ATTN: J. Wright
3 cy ATTN: M. Baron

Weidlinger Assoc, Consulting Engrg
ATTN: K. Stultz

Weidlinger Associates
ATTN: J. Isenberg
ATTN: E. Richardson

Weidlinger Associates, Inc
ATTN: E. Raney

Westinghouse Corp
ATTN: J. Kasdorf