

AD-A131 273

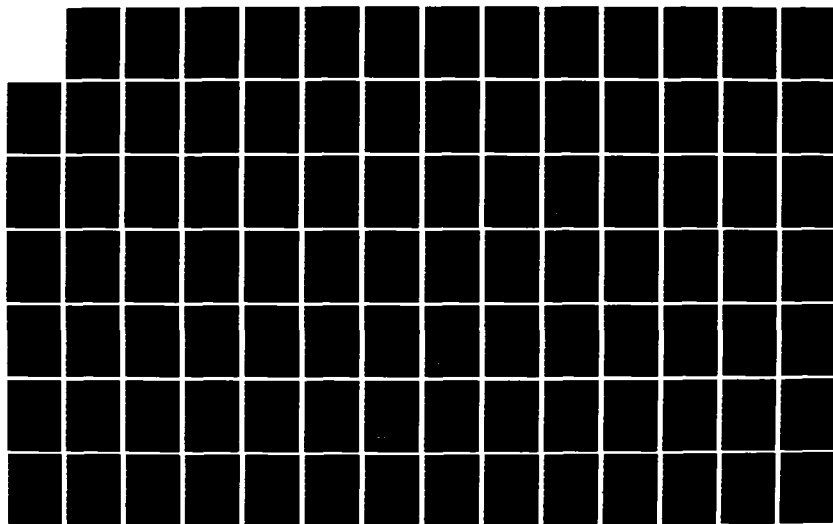
AFWL BENCHMARK COMPUTER PROGRAM(U) BOEING AEROSPACE CO
SEATTLE WA B J HENDERSON 01 MAY 82 DNA-5852F
DNA001-80-C-0056

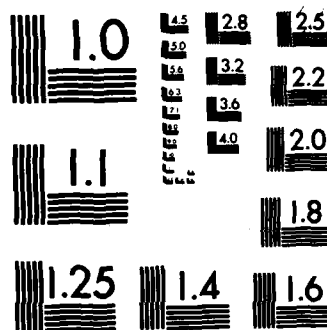
1/2

UNCLASSIFIED

F/G 9/2

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

12

DNA 5852F

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

DTIC
ELECTE
AUG 11 1983
S B

Publication of this material does not imply Defense
Nuclear Agency (DNA) indorsement of factual accuracy
or opinions, nor does DNA accept any responsibility
for maintenance or support of this software.

83 07 28 036

ADA131273

DTIC FILE COPY

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.



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER DNA 5852F	2. GOVT ACCESSION NO. A131273	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AFWL BENCHMARK COMPUTER PROGRAM		5. TYPE OF REPORT & PERIOD COVERED Final Report for Period 3 March 1980 - 1 May 1982
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) B. J. Henderson		8. CONTRACT OR GRANT NUMBER(s) DNA 001-80-C-0056
9. PERFORMING ORGANIZATION NAME AND ADDRESS Boeing Aerospace Company P.O. Box 3999 Seattle, Washington 98124		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Subtask P99DAXDB088-53
11. CONTROLLING OFFICE NAME AND ADDRESS Director Defense Nuclear Agency Washington, D. C. 20305		12. REPORT DATE 1 May 1982
		13. NUMBER OF PAGES 96
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A since UNCLASSIFIED
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES This work was sponsored by the Defense Nuclear Agency under RDT&E RMSS Code B310080464 P99DAXDB08853 H2590D.		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Computer Benchmark ANSI Fortran Translator		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) 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 fair benchmark computer programs to vendors who bid on a DNA computer procurement project.		

DD FORM 1473
1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

PREFACE

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 Millman and Dave Cruikshank made the successful completion of this contract possible.

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By _____	
Distribution/	
Availability	
Dist	Spec
A	



TABLE OF CONTENTS

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

LIST OF TABLES

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

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 Fortan 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

Initial Form	Action Taken	Translated Form
(see Table 2)		
ASSIGN s TO i	T	ASSIGN s TO i
BACKSPACE u BACKSPACE (alist)	T F	BACKSPACE u BACKSPACE (alist)
BLOCK DATA [sub]	N	BLOCK DATA [sub]
CALL sub [(a [a],...)]	T	CALL sub [(a [a],...)]
CHARACTER [*len[,]] nam [,nam]...	TI	CHARACTER [*len[,]] nam [,nam]...
CLOSE (clist)	F	CLOSE (clist)
COMMON [/[cb]/]nlist[,[,]/[cb]/nlist]...	T	COMMON [/[cb]/]nlist[,[,]/[cb]/nlist]...
COMPLEX v [,v]... COMPLEX *len[,] v [,v]...	T RT	COMPLEX [,v]... COMPLEX v [,v]...
CONTINUE	N	CONTINUE
DATA nlist/clist/ [[,]nlist/clist]	T	DATA nlist/clist/ [[,]nlist/clist]
DIMENSION a(d) [,a(d)]...	T	DIMENSION a(d) [,a(d)]...
DO s [,] i=e ₁ ,e ₂ [,e ₃]	T	DO s [,] i=e ₁ ,e ₂ [,e ₃]
DOUBLE PRECISION v[,v]...	RT	REAL v [,v]
ELSE	NI	ELSE
ELSE IF (e) THEN	TI	ELSE IF (e) THEN
END	N	END
END IF	NI	END IF
ENDFILE u ENDFILE (alist)	T F	ENDFILE u ENDFILE (alist)
ENTRY en [(d [,d],...)]	TF	ENTRY en [(d [,d],...)]
EQUIVALENCE (nlist) [,(nlist)]...	T	EQUIVALENCE (nlist) [,(nlist)]...
EXTERNAL proc [,proc]...	T	EXTERNAL proc [,proc]...

TABLE 1. (Continued)

Initial Form	Action Taken	Translated Form
FORMAT is	T	FORMAT is
fun ([d]...) = e	T	fun ([d]...)= e
[typ [*len]] FUNCTION fun([d [,d]...])	RT	[type] FUNCTION fun ([d [,d]...])
GO TO i [[,](s [,s]...)]	N	GO TO i [[,](s [,s]...)]
GO TO s	N	GO TO s
GO TO (s [,s]...)[,] i	T	GO TO (s [,s]...)[,] i
IF (e) st	T	IF (e) st
IF (e) s ₁ , s ₂ , s ₃	T	IF (e) s ₁ , s ₂ , s ₃
IF (e) s ₁ , s ₂	F	IF (e) s ₁ , s ₂
IF (e) THEN	TI	IF (e) THEN
IMPLICIT type (a [,a]...)	R	Specific Type Declaration Statements Generated As Required
INQUIRE (iflist)	F	INQUIRE (iflist)
INQUIRE (iulist)	F	INQUIRE (iulist)
INTEGER v [,v]...	T	INTEGER v [,v]...
INTEGER *len v [,v]...	RT	INTEGER v [,v]...
INTRINSIC fun [,fun]...	F	INTRINSIC fun [,fun]...
LOGICAL v [,v]...	T	LOGICAL v[,v]...
LOGICAL *len v[,v]...	RT	LOGICAL v[,v]...
NAMelist	TF	NAMelist
OPEN (olist)	I	OPEN (olist)
PARAMETER (p=e[,p=e]...)	F	PARAMETER (p=e[,p=e]...)
PAUSE [n]	N	PAUSE [n]
PRINT f[, iolist]	T	PRINT f [, iolist]
PROGRAM pgm	T	PROGRAM pgm
READ (cilst) [iolist]	T	READ (cilst) [iolist]
READ f [, iolist]	F	READ f [, iolist]

TABLE 1. (Continued)

Initial Form	Action Taken	Translated Form
REAL v [,v]...	T	REAL v [,v]...
REAL *len v [,v]	RT	REAL v [,v]...
RETURN [e]	N	RETURN [e]
REWIND u	T	REWIND u
REWIND (alist)	F	REWIND (alist)
SAVE [a [,a]...]	F	SAVE [a [,a]...]
STOP [n]	N	STOP [n]
SUBROUTINE sub [[d [,d]...]]	RT	SUBROUTINE sub [[d [,d]...]]
WRITE (cilst) [iolist]	T	WRITE (cilst) [iolist]
v = e	T	v = e Arithmetic Assignment Statement
v = e	T	v = e Logical Assignment Statement
v = e	I	v = e Character Assignment Statement
v = v [=v...] = e	RT	v = e Multiple Arithmetic v = v Assignment Statement [v = v] . . .
st \$ st [\$st...]	RT	st Multiple Statements per st coding line [st] . . .
v(e(e(e)...))	RT	v = e(e) Subscripted Subscripts [v = e (v)] . . . v(v)
td v/clist/[v,clist]...	RT	td v [,v]... Data in Type Declaration DATA v/clist/[v/clist]

Table 2. Actions taken by SC OFF

Action	Description
R	Restructured by SC OFF to statement acceptable in ANSI subset
T	Variable names translated (truncated with ambiguity resolution) by SC OFF if longer than six characters
F	Flagged as not ANSI subset. Not translated by SC OFF unless TF appears
N	Translation not needed
I	Flagged as not acceptable to pre-1981 IBM compilers

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 elipsis, ... , 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:

CALL sub([(a [, a]..)])

the following forms are allowed:

CALL sub
CALL sub ()
CALL sub (a)
CALL sub (a, a)
CALL sub (a, a, a)
etc

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

CALL ABCD (X, 1.0)

Table 4. Job control language required to execute SC OFF

```
// 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
//FT06F001 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 FT06F001 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 = expression

A Multiple Assignment Statement is expanded into two (or more) single assignment Fortran statements:

C = 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:

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:

COMPLEX*len[,] v [,v] ...

This statement is both Restructured and Translated (RT). The restructuring consists of removing the *len[,] 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.

SECTION 5

REFERENCES

1. American National Standards Institute, "ANSI X3.9-1978 Fortran 77," New York (1978).
2. Lyon, G., Ed., "Using ANS FORTRAN," NBS Handbook 131, U.S. Department of Commerce/National Bureau of Standards, U.S. Gov. Printing Office, Washington (1980).
3. Conti, D. M., "Findings of the Standard Benchmark Library Study Group," Institute for Computer Science and Technology, U.S. Department of Commerce/National Bureau of Standards, U.S. Gov. Printing Office, Washington (1979).
4. Thompson, S. L., "CSQ--A Two-Dimensional Hydrodynamic Program With Energy Flow and Material Strength," Sandia Laboratories (SAND74-0122) Albuquerque, New Mexico (1975).

APPENDIX A
SAMPLE COMPUTER RUN OF SCOFF

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 ANS,YES
EQUIVALENCE (ARRAYONE,ARRAYTHREE)
DATA YES/'Y'/
PRINT 5
5 FORMAT(* THIS PROGRAM DOESNT DO ANYTHING!*)
DO 500 VARIABLE1=1,IDIM
VARIABLE3=(-1)**VARIABLE1
ARRAYTWO(VARIABLE1)=ARRAYTHREE(VARIABLE1)+VARIABLE2*VARIABLE3
*
*VARIABLE1
* IF (ARRAYTWO(VARIABLE1))100,100,500
100 ARRAYTWO(VARIABLE1)=VARIABLE1*VARIABLE2
500 CONTINUE
PRINT 6
6 FORMAT(* DO YOU WANT A PRINT?<Y OR N> *S)
READ *,ANS
IF (ANS.EQ.YES)PRINT 7,(ARRAYTWO(R),R=1,IDIM)
7 FORMAT(* ALL NUMBERS SHOULD BE POSITIVE COUNTING BY 2-->*,
- (/1X,1P8E10.3))
DO 600 VARIABLE2=1,IDIM
ARRAYONE(VARIABLE2)=DIVIDEITUP(ARRAYTWO(VARIABLE2))
600 CONTINUE
PRINT 8
READ *,ANS
IF (ANS.EQ.YES)PRINT 8,(ARRAYTHREE(S),S=1,IDIM)
8 FORMAT(* ALL NUMBER SHOULD BE POSITIVE SEQUENTIAL-->*,
- (/1X,1P8E10.3))
CALL FLIPAROUND(ARRAYONE,IDIM,ARRAYTHREE)
PRINT 9
READ *,ANS
IF (ANS.EQ.YES)PRINT 9,ARRAYONE
9 FORMAT(* ALL NUMBERS SHOULD BE SEQUENTIAL ALTERNATING SIGN-->*,
- (/1X,1P8E10.3))
PRINT 10
10 FORMAT(* ALLLLLLLDONE!!!*)
STOP
END
REAL*4 FUNCTION DIVIDEITUP(RAY)
DIVIDEITUP=RAY/2
RETURN
END
SUBROUTINE FLIPAROUND(IIIIII1,IIIIII2,IIIIII3)
REAL IIIII1(IIIIII1),IIIIII2(IIIIII2)
DO 5000 J=1,IIIIII1
K=(-1)**J
IIIIII1(J)=IIIIII2(J)*K
5000 CONTINUE
RETURN
END

```

OUTPUT FROM SCOFF

```

C=  PROGRAM TESTSCOFF
    PARAMETER (IDIM=100)
    INTEGER R, S
C=  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 VARBL2 / 2 /
    DATA YES/ 'HY'/
    PRINT 5
    5 FORMAT(33H THIS PROGRAM DOESNT DO ANYTHING!)
    DO 500 VARBL1=1,IDIM
    VARBL3=(-1)**VARBL1
    ARRATW(VARBL1)=ARRTHR(VARBL1)+VARBL2*VARBL3
    A      *VARBL1
    IF(ARRATW(VARBL1))100,100,500
100 ARRATW(VARBL1)=VARBL1*VARBL2
500 CONTINUE
    PRINT 6
    6 FORMAT(30H DO YOU WANT A PRINT?)Y OR N> S)
    READ *,ANS
    IF(ANS.EQ.YES)PRINT 7,(ARRATW(R),R=1,IDIM)
    7 FORMAT(48H ALL NUMBERS SHOULD BE POSITIVE COUNTING BY 2-->,(/1X,1P
    -8E10.3))
    DO 600 VARBL2=1,IDIM
    ARRAYN(VARBL2)=DIVDTP(ARRATW(VARBL2))
600 CONTINUE
    PRINT 8
    READ *,ANS
    IF(ANS.EQ.YES)PRINT 8,(ARRTHR(S),S=1,IDIM)
    8 FORMAT(44H ALL NUMBER SHOULD BE POSITIVE SEQUENTIAL-->,(/1X,1P8E10
    -.3))
    CALL FLPRND(ARRAYN,IDIM,ARRTHR)
    PRINT 9
    READ *,ANS
    IF(ANS.EQ.YES)PRINT 9,ARRAYN
    9 FORMAT(53H ALL NUMBERS SHOULD BE SEQUENTIAL ALTERNATING SIGN-->,(/
    -1X,1P8E10.3))
    PRINT 10
10  FORMAT(15H ALLLLLLDONE!!!!)
    STOP
    END
    REAL*4 FUNCTION DIVDTP(RAY)
    DIVDTP=RAY/2
    RETURN
    END
    SUBROUTINE FLPRND(I1,I11,I111,I1111,I11111,I111111,I1111111,I11111111)
    REAL I11111(I11111),I111111(I111111),I1111111(I11111111)
    DO 5000 J=1,I11111
    K=(-1)**J
    I11111(J)=I111111(J)*K
5000 CONTINUE
    RETURN
    END

```

SCOFF TRANSLATION MESSAGES

```

---          PROGRAM TESTSCOFF          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 ANS,YES          ---
CHARACTER STATEMENT NOT ALLOWED IN ALL FORTRAN LANGUAGES          ---
---          6 FORMAT(= DO YOU WANT A PRINT?)Y OR N> =S)          ---
CHARACTER (S) IS NOT A VALID FORMAT TYPE.          ---
          INTEGER  R          S
END OF TRANSLATION
00000010

```

APPENDIX B
LISTING OF SCOFF

```

100 LOGICAL ENDSD
200 LOGICAL OUPDAT, NUMLY
300 REAL*8 ERRMSG(9), NAME(2)
400 COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K
500 1:RSX,LIST(191),LOCN,MINA,MAXA,NCHAR,NEXT,MNT,MREC,NROUT,NSYM
600 2: NERR,NERO,MEMKEY
700 COMMON /ALPH/ KBL,KABC(26),KDIG(10),KSPK(13)
800 COMMON /BETA/ MXCH,MXLI,N15B,LINE,NPAGE
900 COMMON /DELTA/ MOVE, JCARD(80), ICARD(80), INBX(1520), IOL, NCARD
1000 COMMON /KAPPA/ KAP(11)
1100 COMMON /OMEGA/ KLAST(4),KSTOP(4)
1200 COMMON /SIGMA/ KSTIJ(11,60)
1300 COMMON /HASH/ ILOC, LOC, NAME, IVAL, NUMLY
1400 COMMON /TAPES/ N1, N2, N3, N4, N5, N6
1500 COMMON /XUPDAT/ OUPDAT, ICDCI(127)
1600 DATA ERRMSG / 8HCHARCTR , 5HCLOSE, 4HELSE, 5HENDIF, 7HINQUIRE, /
1700 8HINTRNSIC, 4HOPEN, 8HPARAMETR, 4HSAVE
1800
1900
2000
2100
2200
2300
2400
2500
2600
2700
2800
2900
3000
3100
3200
3300
3400
3500
3600
3700
3800
3900
4000
4100
4200
4300
4400
4500
4600
4700
4800
4900
5000
5100
5200
5300
5400
5500

C
1
2
3
C
C
C
4
5
6
6001
7

LOGICAL ENDSD
LOGICAL OUPDAT, NUMLY
REAL*8 ERRMSG(9), NAME(2)
COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K
1:RSX,LIST(191),LOCN,MINA,MAXA,NCHAR,NEXT,MNT,MREC,NROUT,NSYM
2: NERR,NERO,MEMKEY
COMMON /ALPH/ KBL,KABC(26),KDIG(10),KSPK(13)
COMMON /BETA/ MXCH,MXLI,N15B,LINE,NPAGE
COMMON /DELTA/ MOVE, JCARD(80), ICARD(80), INBX(1520), IOL, NCARD
COMMON /KAPPA/ KAP(11)
COMMON /OMEGA/ KLAST(4),KSTOP(4)
COMMON /SIGMA/ KSTIJ(11,60)
COMMON /HASH/ ILOC, LOC, NAME, IVAL, NUMLY
COMMON /TAPES/ N1, N2, N3, N4, N5, N6
COMMON /XUPDAT/ OUPDAT, ICDCI(127)
DATA ERRMSG / 8HCHARCTR , 5HCLOSE, 4HELSE, 5HENDIF, 7HINQUIRE, /
8HINTRNSIC, 4HOPEN, 8HPARAMETR, 4HSAVE

IMPLGO = 0
N1 = 8
N2 = 11
N3 = 12
N4 = 6
N5 = 5
N6 = 6
MXCH = 7
MINA = 37*2
MAXA = 27*MINA-1
NUMLY = .FALSE.
NROUT = 0
NERR = 0
NERO = 0
MEMKEY = 0
DO 2 I=1,8
KLEAR(I)=KBL
NEXT=MXLI
NSYM=0
NREC=0
CALL HASHIN

IDENTIFY FORTRAN STATEMENT BY TYPE.

CALL INPUT
KOL = 6
DO 8 J=1,69
NIST=KSTIJ(11,J)
LAST = KOL
DO 7 I=1,NIST
LAST = LAST + 1
IF (KARD(LAST).EQ.KSTIJ(I,J)) GO TO 7
IF (KARD(LAST) .NE. KBL ) GO TO 8
GO TO 6001
CONTINUE

```

7000	IF (KARD(LAST+1) .NE. KBL) GO TO 7001	5600
	LAST = LAST + 1	5700
	GO TO 7000	5800
C		5900
C	GO TO FORTRAN STATEMENT PROCESSING BY TYPE.	6000
C		6100
7001	CONTINUE	6200
	GO TO (90, 4, 15, 16, 17, 17, 21, 24, 28, 4, 29, 91, 32, 33, 34, 35, 30, 16,	6300
1	40, 41, 42, 43, 44, 45, 4, 18, 48, 48, 49, 54, 54, 54, 54, 58, 64, 65, 92,	6400
2	88, 66, 67, 18, 70, 71, 75, 78, 82, 83, 660, 16, 18, 4, 66, 18, 84, 85, 86, 87,	6500
3	6100, 6200, 6200, 6200, 6200, 6600, 6200, 6800, 6200, 6800, 6200	6600
4	), J	6700
C		6800
C	CONTINUE	6900
C		7000
C	NOT IN STATEMENT LIST. ASSUME ARITHMETIC AND LOOK FOR =.	7100
C		7200
9009	KOL = 6	7300
9	LAST = 3	7400
10	ASSIGN 13 TO NOEQ	7500
	K = KMAX - 1	7600
	DO 12 I=LAST, K	7700
12	IF (KARD(I) .EQ. KSPK(1)) GO TO 14	7800
	CONTINUE	7900
13	GO TO NOEQ, (4, 13, 205, 440, 460, 68)	8000
	CALL ERROR	8100
	ITYPE=1	8200
	GO TO 11	8300
14	ITYPE = 2	8400
	CALL MULTEQ(I)	8500
	GO TO 4	8600
11	CALL SUBSUB	8700
	GO TO 4	8800
C		8900
C		9000
C		9100
C		9200
C		9300
C		9400
C	ASSIGN 12345 TO N.	9500
C		9600
15	KOL=LAST	9700
	CALL SYMBOL	9800
	IF (I CODE .EQ. MINA) CALL ERROR	9900
	ITYPE=1	10000
	CALL UPDATE	10100
	KOL=KOL+1	10200
	CALL SYMBOL	10300
	IF (I CODE .NE. MINA) CALL ERROR	10400
	ITYPE=2	10500
	CALL UPDATE	10600
	GO TO 4	10700
C		10800
C	BACKSPACE, ENDFILE, REWIND, ETC.	10900
C		11000

18	KOL=LAST	1100
	ITYPE=1	11200
	CALL SYMBOL	11300
	IF (CODE .EQ. MINA) CALL UPDATE	11400
	GO TO 4	11500
C		11600
C	17	11700
	18	11800
	19	11900
	20	12000
	21	12100
	22	12200
	23	12300
	24	12400
	25	12500
	26	12600
	27	12700
	28	12800
	29	12900
	30	13000
	31	13100
	32	13200
	33	13300
	34	13400
	35	13500
	36	13600
	37	13700
	38	13800
	39	13900
	40	14000
	41	14100
	42	14200
	43	14300
	44	14400
	45	14500
	46	14600
	47	14700
	48	14800
	49	14900
	50	15000
	51	15100
	52	15200
	53	15300
	54	15400
	55	15500
	56	15600
	57	15700
	58	15800
	59	15900
	60	16000
	61	16100
	62	16200
	63	16300
	64	16400
	65	16500

24	ITYPE = 5	16500
	CODE = 0	16700
C-----	IF (KARD(LAST+1) .NE. KSPK(4)) GO TO 25	16800
	----- LABELLED COMMON. SKIP LABEL	16900
	KOL = LAST+1	17000
	CALL SYMBOL	17100
	GO TO 26	17200
25	KOL=LAST	17300
26	CALL SYMBOL	17400
	IF (CODE) 4,26,27	17500
27	CONTINUE	17600
	CALL UPDATE	17700
	GO TO 26	17800
C		17900
C	COMPLEX STATEMENT.	18000
C		18100
28	ITYPE=6	18200
	CALL TDTEST(495, LAST, IMPLGO, 8, 16)	18300
C-----	----- TYPE DECLARATION STATEMENT. SEE IF FOLLOWED BY "FUNCTION"	18400
285	IF = LAST	18500
	N = KSTIJ(11,28)	18600
	DO 286 I=1,N	18700
2850	IF = 11 + 1	18800
	IF (KARD(11) .EQ. KSTIJ(1,28)) GO TO 286	18900
	IF (KARD(11) .NE. KBL) GO TO 288	19000
286	GO TO 2850	19100
	CONTINUE	19200
	LAST = 11	19300
	GO TO 18	19400
288	CALL SLASHS(LAST)	19500
	GO TO 4	19600
C		19700
C	DATA STATEMENT.	19800
C		19900
29	ITYPE=7	20000
	ASSIGN 25 TO NOEQ	20100
C-----	----- LOOK FOR = OR /	20200
290	DO 292 I=LAST,KMAX	20300
	IF (KARD(1) .EQ. KSPK(1)) GO TO 14	20400
	IF (KARD(1) .EQ. KSPK(4)) GO TO 294	20500
292	CONTINUE	20600
294	GO TO NOEQ, (25, 830)	20700
C		20800
C	DECODE, ENCODE.	20900
C		21000
30	KOL=LAST	21100
	ITYPE=1	21200
	CALL SYMBOL	21300
	IF (CODE .LT. 0) GO TO 4	21400
31	CALL SYMBOL	21500
	CALL UPDATE	21600
	GO TO 26	21700
C		21800
C	DIMENSION STATEMENT.	21900
C		22000

32	ITYPE=9	22100
C	GO TO 23	22200
C		22300
C	DOUBLE PRECISION STATEMENT.	22400
		22500
33	KOL = 0	22600
	CALL MOVER(4,4HREAL,1,15)	22700
	LAST = KOL	22800
	KMAX = KMAX - KOL + 10	22900
	ITYPE = 18	23000
	GO TO 285	23100
C		23200
C	DOUBLE STATEMENT.	23300
C		23400
34	ITYPE=9	23500
C	GO TO 25	23600
C		23700
C	DO STATEMENT.	23800
C		23900
35	DO 36 1=LAST,KMAX	24000
	IF (KARD(1).EQ.KSPK(1)) GO TO 37	24100
	IF (KARD(1).EQ.KSPK(3)) GO TO 9	24200
	CONTINUE	24300
36	CALL ERROR	24400
	GO TO 4	24500
37	IB=1	24600
	DO 38 1=1B,KMAX	24700
	IF (KARD(1).EQ.KSPK(2)) GO TO 39	24800
38	CONTINUE	24900
	GO TO 14	25000
39	KOL=LAST	25100
	ITYPE=20	25200
	IF (KARD(1).EQ.KSPK(3)) GO TO 11	25300
	CALL SYMBOL	25400
C		25500
	CALL UPDATE	25600
	KOL=KOL-1	25700
	ITYPE = 21	25800
	GO TO 26	25900
C		26000
C	*****	26100
C	END STATEMENT	26200
C	EXECUTE SORT AND PRINT INDEX	26300
C	*****	26400
C		26500
40	IF (KMAX.GT.LAST) GO TO 9	26600
	CALL SORT(1)	26700
	GO TO 3	26800
C		26900
C	ENTRY STATEMENT.	27000
C		27100
41	ASSIGN 205 TO NOEQ	27200
	GO TO 10	27300
C		27400
C	EQUIVALENCE STATEMENT.	27500

C 42	ITYPE=10	27600
	KOL=LAST+1	27700
	GO TO 26	27800
C		27900
C	EXTERNAL STATEMENT.	28000
C 43		28100
	ITYPE=11	28200
	KOL = LAST	28300
	CALL XTRNAL	28400
	GO TO 4	28500
C		28600
C	----- FIND STATEMENT	28700
C 44	ASSIGN 440 TO NOEQ	28800
	GO TO 10	28900
C 440	CALL ERROR	29000
	GO TO 96	29100
C		29200
C	FORTRAN OR TYPE STATEMENT.	29300
C		29400
C 45	KOL=LAST	29500
	GO TO 6	29600
C		29700
C	GO TO (1,2,3,4,5), N /// GO TO 1 /// GO TO N /// ETC.	29800
C		29900
C 46	ASSIGN 460 TO NOEQ	30000
	GO TO 10	30100
C 460	KOL=LAST	30200
	ITYPE=1	30300
C 47	CALL SYMBOL	30400
	IF (CODE) 4,47,48	30500
C 48	CALL UPDATE	30600
	GO TO 47	30700
C		30800
C	IF ACCUMULATOR OVERFLOW STATEMENT.	30900
C		31000
C 49	DO 50 I=LAST,KMAX	31100
	IF (KARD(I).EQ.KABC(23)) GO TO 51	31200
	CONTINUE	31300
C 50	CALL ERROR	31400
	GO TO 4	31500
C 51	KOL=1	31600
	ITYPE=1	31700
C 52	CALL SYMBOL	31800
	IF (CODE) 4,52,53	31900
C 53	IF (CODE.GE.MINA.AND.CODE.LE.MAXA) CALL ERROR	32000
	CALL UPDATE	32100
	GO TO 52	32200
C		32300
C	IF (DIVIDE CHECK) /// IF (END FILE) /// IF (SENSE LIGHT).	32400
C	IF (SENSE SWITCH) 1,2	32500
C		32600
C 54	DO 55 I=LAST,KMAX	32700
	IF (KARD(I).EQ.KSPK(5)) GO TO 51	32800
C 55	CONTINUE	32900
		33000

C	CALL ERROR	33100
C	GO TO 4	33200
C	IF (ARITH) 1,2,3 /// IF (LOGICAL) STATEMENT.	33300
C	LOOK FOR PARENTHESIS COUNT OF ZERO.	33400
C	NPARG=0	33500
56	DO 56 I=LAST,KMAX	33600
	IF (KARD(I).NE.KSPK(3)) GO TO 57	33700
	NPARG=NPARG+1	33800
	GO TO 58	33900
57	IF (KARD(I).NE.KSPK(5)) GO TO 58	34000
	NPARG=NPARG-1	34100
	IF (NPARG) 59,59,58	34200
58	CONTINUE	34300
	CALL ERROR	34400
	GO TO 4	34500
C	SCAN SYMBOLS WITHIN PARENTHESIS.	34600
C	J=KMAX	34700
C	IB=1	34800
59	KMAX=IB	34900
	ITYPE=1	35000
	KOL=LAST	35100
62	CALL SUBSUB	35200
	KOL=IB	35300
	KMAX=J	35400
C	LOOK FOR DIGIT IMMEDIATELY AFTER PARENTHESIS. IF ONE IS NOT	35500
C	FOUND, THE IF STATEMENT IS PROBABLY LOGICAL.	35600
C	DO 63 I=1,10	35700
	IF (KARD(KOL+1).EQ.KDIG(I)) GO TO 52	35800
63	CONTINUE	35900
	IF (KARD(KOL+1).NE.KBL) GO TO 6	36000
	IB = IB + 1	36100
	GO TO 62	36200
C	INTEGER STATEMENT.	36300
C	ITYPE=12	36400
64	CALL TDTEST(495, LAST, IMPLGO, 2, 4)	36500
	GO TO 285	36600
C	LOGICAL STATEMENT.	36700
C	ITYPE=13	36800
65	CALL TDTEST(495, LAST, IMPLGO, 1, 4)	36900
	GO TO 285	37000
C	PAUSE, STOP, RETURN STATEMENT.	37100
C	ASSIGN 4 TO NOEQ	37200
66		37300
		37400
		37500
		37600
		37700
		37800
		37900
		38000
		38100
		38200
		38300
		38400
		38500

C-----	GO TO 10	38600
990	----- RETURN.	38700
C>>	IF (LAST.GE. KMAX) GO TO 4	38800
	CALL RETRN(1)	38900
	GO TO 4	39000
C		39100
C	PRINT STATEMENT.	39200
67		39300
68	ITYPE=15	39400
	KOL=LAST	39500
	CALL SYMBOL	39600
	IF (CODE.LT. 0) GO TO 4	39700
C-----	----- LOOK FOR COMMA BEFORE =	39800
69	IF (KRXX.EQ. KSPK(2)) GO TO 11	39900
	IF (KRXX.EQ. KSPK(1)) GO TO 14	40000
	IF (KRXX.EQ. KSPK(3)) GO TO 9	40100
	GO TO 4	40200
C		40300
C	PUNCH STATEMENT.	40400
C		40500
70	ITYPE=16	40600
	GO TO 68	40700
C		40800
C	READ INPUT TAPE 5, 6, LIST.	40900
C		41000
71	ITYPE=17	41100
	KOL=LAST+3	41200
72	CALL SYMBOL	41300
	IF (CODE.LT. 0) GO TO 4	41400
73	IF (CODE.GE.MINA.AND.CODE.LE.MAXA) CALL UPDATE	41500
	CALL SYMBOL	41600
	IF (CODE) 4,4,74	41700
74	CALL UPDATE	41800
	GO TO 26	41900
C		42000
C	READ TAPE 5, LIST	42100
		42200
75	ITYPE=17	42300
76	KOL=LAST	42400
	CALL SYMBOL	42500
	IF (CODE.LT. 0) GO TO 4	42600
77	IF (CODE.GE.MINA.AND.CODE.LE.MAXA) CALL UPDATE	42700
	GO TO 26	42800
C		42900
C	READ (5) LIST /// READ (5,6) LIST	43000
C		43100
78	ITYPE=17	43200
79	KOL=LAST	43300
	CALL SYMBOL	43400
	IF (CODE.LT. 0) GO TO 4	43500
	IF (CODE.GE.MINA.AND.CODE.LE.MAXA) CALL UPDATE	43600
80	IF (KRXX.EQ.KSPK(5)) GO TO 11	43700
	CALL SYMBOL	43800
	IF (CODE.LT. 0) GO TO 4	43900
81	IF (KRXX.EQ. KSPK(11)) CALL ERROR	44000

	IF (KRXX .EQ. KSPK(1)) CALL ERREND	44100
	GO TO 80	44200
C		44300
C	READ 5, LIST	44400
C		44500
82	ITYPE=17	44600
	GO TO 68	44700
C		44800
C	REAL STATEMENT.	44900
C		45000
83	ASSIGN 830 TO NOEQ	45100
	GO TO 290	45200
830	ITYPE=18	45300
	CALL TDTST(895, LAST, IMPLGO, 4, 8)	45400
	GO TO 285	45500
C		45600
C	TYPE STATEMENT.	45700
C		45800
84	KOL=LAST	45900
	GO TO 6	46000
C		46100
C	WRITE OUTPUT TAPE 5, 6, LIST	46200
C		46300
85	ITYPE=19	46400
	KOL=LAST+5	46500
	GO TO 72	46600
C		46700
C	WRITE TAPE 5, LIST.	46800
C		46900
86	ITYPE=19	47000
	GO TO 76	47100
C		47200
C	WRITE 5, LIST.	47300
C		47400
87	ITYPE=19	47500
	GO TO 79	47600
C		47700
C	NAMelist STATEMENT.	47800
C		47900
88	ITYPE=14	48000
	GO TO 25	48100
C	----- NM MEMBER = NAME STATEMENT (FROM UPDATER TAPE)	48200
90	MOVE = -1	48300
	NREC = NREC-1	48400
	GO TO 4	48500
C	----- DEFINE FILE STATEMENT	48600
91	KOL = LAST	48700
	GO TO 97	48800
C		48900
C	----- IMPLICIT STATEMENT. TEST TO SEE IF FIRST STATEMENT IN PROGRAM	49000
C	----- PROCESS IMPLICIT STATEMENT	49100
92	IMPLGO = 1	49200
	KOL = LAST	49300
	GO TO 6	49400
95	KOL = LAST+2	49500

	CALL IMPSET(IDX)	49600
	IF (IDX .EQ. 1) GO TO 6	49700
	IMPLGO = 0	49800
96	KARD(1) = KABC(3)	49900
	KARD(2) = KSPK(1)	50000
97	CALL ERR(19,0,0)	50100
	GO TO 4	50200
C-----	----- FORMAT STATEMENT	50300
98	CALL FORTRN(KMAX, KARD, .TRUE.)	50400
	MOVE = 2	50500
	NCARD = NCARD-1	50600
	GO TO 4	50700
C		50800
C-----	----- CHARACTER STATEMENT	50900
6100	ITYPE = 3	51000
	CALL ERR(18,5,10H CHARACTER)	51100
	IF (KARD(LAST+1) .NE. KSPK(8)) GO TO 6120	51200
	KOL = LAST + 2	51300
	NUMLY = .TRUE.	51400
	CALL SYMBOL	51500
	LAST = KOL	51600
	NUMLY = .FALSE.	51700
6120	IF (IMPLGO .EQ. 1) GO TO 95	51800
	GO TO 285	51900
C-----	----- CLOSE	52000
6200	ITYPE = J - 40	52100
	ASSIGN 6250 TO NOEQ	52200
	GO TO 10	52300
6250	CALL ERR(18,4,ERRMSG(ITYPE-20))	52400
	GO TO 4	52500
C-----	----- INTRINSIC FUNCTION	52600
6600	CONTINUE	52700
	CALL ERR(18,5,10HINTRINSIC)	52800
	GO TO 4	52900
6800	CONTINUE	53000
	CALL ERR(18,5,10HPARAMETER)	53100
	GO TO 4	53200
	END	53300

C	BLOCK DATA	100
C	THIS BLOCK DATA CONTAINS ALL THE INDEX DATA STATEMENTS.	200
C	LOGICAL#1 OTAB(256)	300
	COMMON /TAPES/ N1, N2, N3, N4, N5, N6	400
	COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(13)	500
	COMMON /BETA/ MXCH, MXLI, N15B, LINE, NPAGE, KSB0, KCDO, KOUT(80)	600
	COMMON /GAMMA / OTAB	700
	COMMON /DELTA/ MOVE, JCARD(80), ICARD(1600), IOL, NCARD	800
	OSTMTH, FRSTSB, IC, JC, IPAR, L1(25), KLAS8(25)	900
	COMMON /KAPPA/ KAP(11)	1000
	COMMON /OMEGA/ KLAST(4), KSTOP(4)	1100
	COMMON /SIGMA/ 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), KST14(11), KST15(11), KST16(11), KST17(11), KST18(11), KST19(11), KST20(11), KST21(11), KST22(11), KST23(11), KST24(11), KST25(11), KST26(11), KST27(11), KST28(11), KST29(11), KST30(11), KST31(11), KST32(11), KST33(11), KST34(11), KST35(11), KST36(11), KST37(11), KST38(11), KST39(11), KST40(11), KST41(11), KST42(11), KST43(11), KST44(11), KST45(11), KST46(11), KST47(11), KST48(11), KST49(11), KST50(11), KST51(11), KST52(11), KST53(11), KST54(11), KST55(11), KST56(11), KST57(11), KST58(11), KST59(11), KST60(11)	1200
	A, KST61(11), KST62(11), KST63(11), KST64(11), KST65(11)	1300
	A, KST66(11), KST67(11), KST68(11), KST69(11)	1400
C	DATA KBL, KDIG/1H, 1H0, 1H1, 1H2, 1H3, 1H4, 1H5, 1H6, 1H7, 1H8, 1H9/ DATA KABC/1HA, 1HB, 1HC, 1HD, 1HE, 1HF, 1HG, 1HH, 1HI, 1HJ, 1HK, 1HL, 1HM, 1HN, 1HO, 1HP, 1HQ, 1HR, 1HS, 1HT, 1HU, 1HV, 1HW, 1HX, 1HY, 1HZ/ DATA KSPK/1H=, 1H, 1H1, 1H/ 1H, 1H+, 1H-, 1H*, 1H., 1H\$, 1H', 1H., 1H"/ DATA MXCH MXLI, N15B, LINE, NPAGE/6, 8191, 8192, 51, 0/ DATA KAP/1NS, 1HU, 1HP, 1HE, 1HR, 1H, 1HI, 1HN, 1HD, 1HE, 1HX/ DATA KLAST, KSTOP/1HL, 1HA, 1HS, 1HT, 1HS, 1HT, 1HO, 1HP/ DATA KST1/1H*, 1H*, 1HM, 1HE, 1HM, 1HB, 1HE, 1HR, 1H=, 1H, 9/ DATA KST2/1H/, 1H/, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 2/ DATA KST3/1HA, 1HS, 1HI, 1HG, 1HN, 1H, 1H, 1H, 1H, 6/ DATA KST4/1HB, 1HA, 1HC, 1HK, 1HS, 1HP, 1HA, 1HC, 1HE, 1H, 9/ DATA KST5/1HB, 1HL, 1HO, 1HC, 1HK, 1HD, 1HA, 1HT, 1HA, 1H, 9/ DATA KST6/1HB, 1HU, 1HF, 1HF, 1HE, 1HR, 1HI, 1HN, 1H, 9/ DATA KST7/1HB, 1HU, 1HF, 1HF, 1HE, 1HR, 1HO, 1HU, 1HT, 1H, 10/ DATA KST8/1HC, 1HA, 1HL, 1HL, 1H, 1H, 1H, 1H, 1H, 1H, 4/ DATA KST9/1HC, 1HO, 1HM, 1HM, 1HO, 1HN, 1H, 1H, 1H, 1H, 6/ DATA KST10/1HC, 1HO, 1HM, 1HP, 1HL, 1HE, 1HX, 1H, 1H, 1H, 7/ DATA KST11/1HC, 1HO, 1HN, 1HT, 1HI, 1HN, 1HU, 1HE, 1H, 1H, 8/ DATA KST12/1HD, 1HA, 1HT, 1HA, 1H, 1H, 1H, 1H, 1H, 1H, 4/ DATA KST13/1HD, 1HE, 1HF, 1HI, 1HN, 1HE, 1HF, 1HI, 1HL, 1HE, 10/ DATA KST14/1HD, 1HI, 1HM, 1HE, 1HN, 1HS, 1HI, 1HO, 1HN, 1H, 9/ DATA KST15/1HD, 1HO, 1HU, 1HB, 1HL, 1HE, 1HP, 1HR, 1HE, 1HC, 10/ DATA KST16/1HD, 1HO, 1HU, 1HB, 1HL, 1HE, 1H, 1H, 1H, 1H, 6/ DATA KST17/1HD, 1HO, 1H, 1H, 1H, 1H, 1H, 1H, 1H, 2/ DATA KST18/1HE, 1HN, 1HC, 1HO, 1HD, 1HE, 1H, 1H, 1H, 7/ DATA KST19/1HE, 1HN, 1HD, 1HF, 1HI, 1HL, 1HE, 1H, 1H, 7/ DATA KST20/1HE, 1HN, 1HD, 1H, 1H, 1H, 1H, 1H, 1H, 3/ DATA KST21/1HE, 1HN, 1HT, 1HR, 1HY, 1H, 1H, 1H, 1H, 5/	1500

DATA KST22/1NE	1HQ	1HU	1HI	1HV	1HA	1HL	1HE	1HN	1HC	10/	
DATA KST23/1NE	1HX	1HT	1HE	1HR	1HN	1HA	1HL	1H	1H	8/	
DATA KST24/1NF	1HI	1HN	1ND	1H	1H	1H	1H	1H	1H	4/	
DATA KST25/1NF	1HO	1HR	1HM	1HA	1HT	1HI	1H	1H	1H	7/	
DATA KST26/1NF	1HO	1HR	1HT	1HR	1HA	1HN	1H	1H	1H	7/	
DATA KST27/1NF	1HR	1HE	1HQ	1HU	1HE	1HN	1HC	1HY	1H	9/	
DATA KST28/1HF	1HU	1HN	1HC	1HT	1HI	1HO	1HN	1H	1H	8/	
DATA KST29/1HG	1HO	1HT	1HO	1HI	1H	1H	1H	1H	1H	5/	
DATA KST30/1HG	1HO	1HT	1HO	1H	1H	1H	1H	1H	1H	4/	
DATA KST31/1HI	1HF	1HA	1HC	1HC	1HU	1HM	1HU	1HL	1HA	10/	
DATA KST32/1HI	1HF	1HQ	1HU	1HO	1HT	1HI	1HE	1HN	1HT	10/	
DATA KST33/1HI	1HF	1HI	1HD	1HI	1HV	1HI	1HD	1HE	1HC	10/	
DATA KST34/1HI	1HF	1HI	1HE	1HN	1HD	1HF	1HI	1HL	1HE	10/	
DATA KST35/1HI	1HF	1HI	1HS	1HE	1HN	1HS	1HE	1HL	1HI	10/	
DATA KST36/1HI	1HF	1HI	1HS	1HE	1HN	1HS	1HE	1HS	1HW	10/	
DATA KST37/1HI	1HF	1HI	1H	1H	1H	1H	1H	1H	1H	3/	
DATA KST38/1HI	1HN	1HT	1HE	1HG	1HE	1HR	1H	1H	1H	7/	
DATA KST39/1HI	1HO	1HG	1HI	1HC	1HA	1HL	1H	1H	1H	7/	
DATA KST40/1HI	1HM	1HP	1HL	1HI	1HC	1HI	1HT	1H	1H	8/	
DATA KST41/1HN	1HA	1HM	1HE	1HL	1HI	1HS	1HT	1H	1H	8/	
DATA KST42/1HP	1HA	1HU	1HS	1HE	1H	1H	1H	1H	1H	5/	
DATA KST43/1HP	1HR	1HI	1HN	1HT	1H	1H	1H	1H	1H	5/	
DATA KST44/1HP	1HR	1HO	1HG	1HR	1HA	1HM	1H	1H	1H	7/	
DATA KST45/1HP	1HU	1HN	1HC	1HH	1H	1H	1H	1H	1H	5/	
DATA KST46/1HR	1HE	1HA	1HD	1HI	1HN	1HP	1HU	1HT	1HT	10/	
DATA KST47/1HR	1HE	1HA	1HD	1HT	1HA	1HP	1HE	1H	1H	8/	
DATA KST48/1HR	1HE	1HA	1HD	1HI	1H	1H	1H	1H	1H	5/	
DATA KST49/1HR	1HE	1HA	1HD	1H	1H	1H	1H	1H	1H	4/	
DATA KST50/1HR	1HE	1HA	1HL	1H	1H	1H	1H	1H	1H	4/	
DATA KST51/1HR	1HE	1HT	1HU	1HR	1HN	1H	1H	1H	1H	6/	
DATA KST52/1HR	1HE	1HW	1HI	1HN	1HD	1H	1H	1H	1H	6/	
DATA KST53/1HS	1HE	1HG	1HM	1HE	1HN	1HT	1H	1H	1H	7/	
DATA KST54/1HS	1HE	1HN	1HS	1HE	1HL	1HI	1HG	1HH	1HT	10/	
DATA KST55/1HS	1HT	1HO	1HP	1H	1H	1H	1H	1H	1H	4/	
DATA KST56/1HS	1HU	1HB	1HR	1HO	1HU	1HT	1HI	1HN	1HE	10/	
DATA KST57/1HT	1HY	1HP	1HE	1H	1H	1H	1H	1H	1H	4/	
DATA KST58/1HW	1HR	1HI	1HT	1HE	1HO	1HU	1HT	1HP	1HU	10/	
DATA KST59/1HW	1HR	1HI	1HT	1HE	1HT	1HA	1HP	1HE	1H	9/	
DATA KST60/1HW	1HR	1HI	1HT	1HE	1HC	1H	1H	1H	1H	6/	
DATA KST61/1HC	1HH	1HA	1HR	1HA	1HC	1HT	1HE	1HR	1H	9/	
DATA KST62/1HC	1HL	1HO	1HS	1HE	1H	1H	1H	1H	1H	5/	
DATA KST63/1HE	1HL	1HS	1HE	1H	1H	1H	1H	1H	1H	6/	
DATA KST64/1HE	1HN	1HD	1HI	1HF	1H	1H	1H	1H	1H	5/	
DATA KST65/1HI	1HN	1HQ	1HU	1HI	1HR	1HE	1H	1H	1H	7/	
DATA KST66/1HI	1HN	1HT	1HR	1HI	1HN	1HS	1HI	1HC	1H	9/	
DATA KST67/1HO	1HP	1HE	1HN	1H	1H	1H	1H	1H	1H	4/	
DATA KST68/1HP	1HA	1HR	1HA	1HM	1HE	1HT	1HE	1HR	1H	9/	
DATA KST69/1HS	1HA	1HV	1HE	1H	1H	1H	1H	1H	1H	4/	
DATA IPAR	ICARD	L1	KLASS	/	1	1650*1H	/				
DATA OTAB	/	63*Z26	Z25	128*Z26							
C	Z01	Z02	Z03	Z04	Z05	Z06	Z07	Z08	Z09	7*Z25	
D	Z0A	Z0B	Z0C	Z0D	Z0E	Z0F	Z10	Z11	Z12	7*Z25	
E	Z25	Z13	Z14	Z15	Z16	Z17	Z18	Z19	Z1A	6*Z25	
F	Z1B	Z1C	Z1D	Z1E	Z1F	Z20	Z21	Z22	Z23	Z24	7*Z25
											/
											END

C	SUBROUTINE CENTOR	100:
C	THIS ROUTINE CENTERS THE PAGE CAPTIONS.	200:
C		300:
	COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K	400:
1	1: RX, LIST(8191), LOCH, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM	500:
2	2: NERR, NERO, NEWKEY	600:
	COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(11)	700:
	COMMON /BETA/ MXCH, MXLI, N15B, LINE, NPAGE	800:
	COMMON /IOTA/ KEY(24)	900:
	COMMON /KAPPA/ KAP(11)	1000:
	COMMON /OMEGA/ KLAST(4), KSTOP(4)	1100:
	COMMON /SIGMA/ KSTIJ(11,60)	1200:
C	COMPRESS STATEMENT BY ELIMINATING MULTIPLE BLANKS.	1300:
C		1400:
C		1500:
1	DO 2 I=1,66	1600:
2	IF (JOB(I).NE.KBL) GO TO 3	1700:
	CONTINUE	1800:
	RETURN	1900:
C		2000:
3	JOB(1)=JOB(1)	2100:
	J=1	2200:
35	IB=I+1	2300:
	DO 4 I=IB,66	2400:
	IF (JOB(I).EQ.KBL.AND.JOB(I-1).EQ.KBL) GO TO 4	2500:
	J=J+1	2600:
	JOB(J)=JOB(I)	2700:
	CONTINUE	2800:
4		2900:
C	IB=J+1	3000:
	DO 5 I=IB,66	3100:
5	JOB(I)=KBL	3200:
C		3300:
C	CENTER HEADING.	3400:
C		3500:
6	IB=(66-J)/2	3600:
	I=J+IB	3700:
	JOB(I)=JOB(J)	3800:
	J=J-1	3900:
	IF (J) 7,7,6	4000:
C		4100:
C	ELIMINATE REMAINING NON-BLANKS.	4200:
C		4300:
7	IF (I.EQ.1) RETURN	4400:
	IB=I-1	4500:
	DO 8 I=1,IB	4600:
8	JOB(I)=KBL	4700:
	RETURN	4800:
	END	4900:
		5000:

```

SUBROUTINE ERREND
COMMON I,TYPE,JOB(66),KARD(1326),KBUFF(80),KLEAR(8),KMAX,KODE,KOL
COMMON /ALPH/ KBL,KASC(26),KDIB(10),KSPK(11)
CALL ERROR
DO 15 I=KOL,KMAX
  IF ( KARD(I) .EQ. KSPK(2) ) GO TO 25
  IF ( KARD(I) .EQ. KSPK(5) ) GO TO 25
  CONTINUE
15: RETURN
      KOL = KOL-5
      N = I-KOL-1
      CALL MOVER( 0,DIMY,DIMY,N )
      RETURN
      END

```

15:

25:

100:
200:
300:
400:
500:
600:
700:
800:
900:
1000:
1100:
1200:
1300:
1400:

	100	200	300	400	500	600	700	800	900	1000	1100
C	SUBROUTINE ERROR										
1	COMMON (S9685(9685), NERR, NERO, MEMKEY										
	COMMON /TAPES/ N1, N2, N3, N4, N5, N6										
C	CALL PAG (1)										
	WRITE (N6, 2)										
	NERR = NERR+1										
	RETURN										
2	FORMAT(' ** SYNTAX **')										
	END										

9005	FORMAT(' TWO SUCCESSIVE LETTERS.')	5600
9006	FORMAT(' P MULTIPLIER USED ON (' ,A1,') FORMAT.')	5700
9007	FORMAT('X1,') FORMAT NOT IN ANSI SUBSET LANGUAGE.')	5800
9008	FORMAT('X1,') FORMAT NOT ANSI STANDARD.')	5900
9009	FORMAT(' NO LEADING COUNT ON (X) FORMAT.')	6000
9010	FORMAT(' Z FORMAT NOT ASA STANDARD.')	6100
9012	FORMAT(' NO LEADING COUNT ON L PAREN IN FORMAT.')	6200
9013	FORMAT(' MORE THAN ONE PERIOD IN A STRING.')	6300
9014	FORMAT(' TROUBLE IN AN APOST. HOLLERITH FORMAT.')	6400
9015	FORMAT(' A STRING ENDING WITH (' ,A1,') DID NOT HAVE ALL THE RIGHT STUFF.')	6500
9016	FORMAT(' ARGUMENTS ON AN ENTRY STATEMENT ARE NOT ALLOWED IN THE CD C 6600 FORTRAN LANGUAGE.')	6600
9017	FORMAT('X42,') IS NOT A FORTRAN-SUPPLIED SUBPROGRAM ON THE CDC 6600 ,,)	6700
9018	FORMAT(' ** STATEMENT TYPE NOT ALLOWED IN IBM FORTRAN: ',5A2)	6800
9019	FORMAT(' , HEX')	6900
	END	7000
		7100
		7200
		7300

C----	RETURN	----- MATCH FOUND. IF UNTRANSLATABLE, PRINT OUT ERROR MESSAGE	5600
21	IF (KEY .GT. 0) GO TO 25		5700
C----	----- INTRINSIC FUNCTION NAME FOUND IN EXTERNAL STATEMENT. FLAG		5800
C	IN TEXT1 SO THAT NO CONVERSION TAKES PLACE.		5900
	TEXT1(1,1) = STAR		6000
	RETURN		6100
25	IF (I .GT. NTR) GO TO 41		6200
	IF (I .GT. NDT) GO TO 31		6300
	NAME = TEXT2(I)		6400
	CALL MOVER(COUNT2(I)+1, NAME, 1, COUNT1(I))		6500
	IF (SYMTAB(ILOC) .NE. TEXT1(I)) RETURN		6600
C----	----- REMOVE OLD NAME FROM SYMBOL TABLE		6700
	ILOC = ILOC-1		6800
	CALL UPDATE		6900
	RETURN		7000
C----	----- ELIMINATE FUNCTION NAME DBLE OR SNGL.		7100
31	K = KOL		7200
	KOL = K-5		7300
	CALL MOVER(0, DUMMY, DUMMY, COUNT1(I))		7400
	KOL = K		7500
	IF (SYMTAB(ILOC) .EQ. TEXT1(I)) ILOC = ILOC-1		7600
	RETURN		7700
C----	----- UNTRANSLATABLE NAME		7800
41	CALL ERR(14, DUMMY, NAME)		7900
	RETURN		8000
C----	----- INITIALIZE		8100
101	DO 111 I=1, MFMS		8200
111	TEXT1(I,1) = BLANK		8300
	RETURN		8400
	END		8500
			8600

C	100	SUBROUTINE FORTN (KMAX, IFMT, DTEFLG)
C	200	----- 360 TO 6600 FORMAT TRANSLATOR.
C	300	INTEGER*2 IFMT(2,1), LEE, LHH, KBL, INO, IANS, LPAREN, RPAREN
	400	INTEGER*2 KEYCHR, JCARD
	500	LOGICAL*4 DTEFLG
	600	LOGICAL*1 LLEAD, LPMLT, LLETR, LPERD, LCOMA, EFGFLG
	700	LOGICAL*1 JHCOM
	800	DIMENSION IANS(4), INO(10)
	900	COMMON / DELTA / MOVE, JCARD(2,90), ICARD(1600), IOL, NCARD
	1000	DATA LEE / 1HE / , OSTMTN, FRSTSB, IC, JC
	1100	DATA LHH / 1HH / , KBL / 1H /
	1200	DATA INO / 1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9 /
C	1300	----- INITIALIZATION.
C	1400	CALL ICHAR(IFMT, KMAX)
	1500	K = 0
	1600	L = 0
	1700	ASSIGN 120 TO MGO
	1800	ASSIGN 20 TO NGO
	1900	LPAREN = 0
	2000	RPAREN = 0
	2100	EFGFLG = .FALSE.
	2200	N6 = 6
C	2300	----- MOVE FORMAT STATEMENT NO. (ASSUME NO. EXISTS.)
C	2400	DO 10 I=1,6
	2500	K = K + 1
	2600	L = L + 1
	2700	JCARD(1,K) = IFMT(1,L)
10	2800	----- ASSUME THE STRING "FORMATI" EXISTS.
C	2900	L = L + 1
C	3000	IF (IFMT(2,L) .EQ. 48) GO TO 20
	3100	IF (IFMT(2,L) .EQ. 39) ASSIGN 50 TO NGO
	3200	ASSIGN 40 TO MGO
	3300	GO TO 700
	3400	GO TO NGO, (20,50)
40	3500	LPAREN = LPAREN + 1
	3600	GO TO 126
C	3700	----- A STRING OF SOMETHING HAS BEEN COMPLETED. DID IT CONTAIN
C	3800	----- ALL THE RIGHT THINGS.
C	3900	IF (JHCOM .AND. LCOMA) GO TO 126
	4000	IF (.NOT. LLEAD) GO TO 190
120	4100	IF (.NOT. LPMLT) GO TO 190
	4200	IF (.NOT. LLETR) GO TO 190
	4300	IF (.NOT. LPERD) GO TO 190
	4400	IF (.NOT. LCOMA) GO TO 190
126	4500	CONTINUE

C	130	ASSIGN 140 TO MGO	5600
C		KOUNT = 0	5700
C		----- PREPARE FOR A STRING OF SOMETHING. WE WILL WANT TO CHECK	5800
C		----- AT THE END OF THE STRING TO SEE THAT NO ILLEGAL STRING WAS	5900
C		----- ENCOUNTERED.	6000
C			6100
	130	LLEAD = .FALSE.	6200
		LPMLT = .FALSE.	6300
		LLETR = .FALSE.	6400
		LPERD = .FALSE.	6500
		LCONA = .FALSE.	6600
		JHCOM = .TRUE.	6700
		GO TO 150	6800
C			6900
C		----- PICK UP NEXT CHARACTER.	7000
C			7100
C	140	CONTINUE	7200
		JHCOM = .FALSE.	7300
	150	L = L + 1	7400
		IF (L .GT. KMAX) GO TO 730	7500
	151	KOG = IFMT(2,L)	7600
C		GO TO (210,170,160,240,238,240,230,280,210,160,160,210,160,	7700
C		160,178,360,160,160,250,260,160,160,160,440,160,178,	7800
C	1	0 1 2 3 4 5 6 7 8 9 =	7900
C	2	470,470,470,470,470,470,470,470,470,470,160,580,590,	8000
C	3	/) + - # S BL NONE	8100
C		600,610,160,160,670,650,160,670,150,160) , KOG	8200
C		----- B,C,J,K,M,N,O,Q,R,S,T,U,V,W,Y,Z,+, -,*,\$,NO MATCH, OR ERROR.	8300
C			8400
160		KERR = .1	8500
164		GO TO 195	8600
165		KERR = 2	8700
166		GO TO 195	8800
168		KERR = 3	8900
		GO TO 195	9000
		KERR = 4	9100
		GO TO 195	9200
170		IF (IFMT(2,L+1) .EQ. 14 .OR. IFMT(2,L+1) .EQ. 26) GO TO 220	9300
173		KERR = 6	9400
		GO TO 195	9500
175		KERR = 7	9600
		ASSIGN 126 TO MGO	9700
		GO TO 195	9800
178		KERR = 8	9900
183		GO TO 195	10000
		KERR = 9	10100
185		GO TO 195	10200
		KERR = 10	10300
		GO TO 195	10400
187		KERR = 11	10500
		GO TO 195	10600
			10700
			10800
			10900
			11000

190	KERR = 12	11100
	CALL ERR (KERR,L,IFMT)	11200
	GO TO 126	11300
C 195		11400
	CALL ERR (KERR,L,IFMT)	11500
	LLEAD = .TRUE.	11600
	LPMLT = .TRUE.	11700
	LLETR = .TRUE.	11800
	LPERD = .TRUE.	11900
	LCOMA = .TRUE.	12000
	GO TO 700	12100
C-----		12200
C	----- A FORMAT. LEADING NO. OK. LAGGING NO. OK. LEAD LETTER NOT.	12300
210	IF (LLETR) GO TO 165	12400
	IF (LPMLT) GO TO 168	12500
	LLEAD = .TRUE.	12600
	LPMLT = .TRUE.	12700
	LLETR = .TRUE.	12800
	LPERD = .TRUE.	12900
	GO TO 700	13000
C-----		13100
220	----- TEST FOR BN OR BZ	13200
	ASSIGN 225 TO MGO	13300
225	GO TO 700	13400
	ASSIGN 126 TO MGO	13500
	GO TO 700	13600
C-----		13700
C	----- G FORMAT: NOT IN ANSI SUBSET	13800
230	IF (LLETR) GO TO 165	13900
	KERR = 5	14000
	CALL ERR (KERR,L,IFMT)	14100
	IFMT(1,L) = LEE	14200
C-----		14300
238	----- E. TEST IF OF FORM EW.DEE	14400
	IF (.NOT. LLETR) GO TO 242	14500
	IF (KE.EQ.4 .OR. KE.EQ.5 .OR. KE.EQ.7) GO TO 239	14600
	GO TO 178	14700
239	IF (.NOT. LPERD .OR. NE .GT. 1) GO TO 178	14800
	NE = NE + 1	14900
	GO TO 700	15000
C-----		15100
C	----- D FORMAT. LEADING P OK. LEADING NO. OK. LAGGING NO. OK.	15200
	LATER PERIOD REQUIRED. LEADING LETTER NOT OK.	15300
240	IF (LLETR) GO TO 165	15400
242	NE = 0	15500
	KE = KOG	15600
	LPMLT = .TRUE.	15700
	LLEAD = .TRUE.	15800
	LLETR = .TRUE.	15900
	LPERD = .FALSE.	16000
	LCOMA = .FALSE.	16100
	GO TO 700	16200
250	CONTINUE	16300
260	CONTINUE	16400
	GO TO 160	16500
C		

C-----	----- H FORMAT. MOVE 'KOUNT' CHARACTERS.	16600:
C		16700:
280	IF (LLETR) GO TO 165	16800:
	IF (.NOT. LLEAD) GO TO 164	16900:
	IF (LPMLT) GO TO 168	17000:
	LCOMA = .TRUE.	17100:
	ASSIGN 261 TO MGO	17200:
2810	GO TO 700	17300:
281	K = K - 1	17400:
	CONTINUE	17500:
	K = K + 1	17600:
	L = L + 1	17700:
282	IF (IFMT(1,L) .EQ. 47) GO TO 2810	17800:
283	JCARD(1,K) = IFMT(1,L)	17900:
	KOUNT = KOUNT - 1	18000:
	IF (KOUNT .GT. 0) GO TO 281	18100:
	GO TO 126	18200:
C-----	----- P FORMAT. LEADING NO. OK. A D,E,F, OR Q MUST EVENTUALLY FOLLOW.	18300:
C		18400:
C		18500:
360	LPMLT = .TRUE.	18600:
	LCOMA = .FALSE.	18700:
	GO TO 700	18800:
C-----	----- X FORMAT. LEADING NUMBER MUST EXIST. LEADING LETTER NOT OK.	19000:
C		19100:
440	IF (LLETR) GO TO 165	19200:
	IF (.NOT. LLEAD) GO TO 175	19300:
	IF (LPMLT) GO TO 168	19400:
	LLETR = .TRUE.	19500:
	LCOMA = .TRUE.	19600:
	ASSIGN 126 TO MGO	19700:
	GO TO 700	19800:
C-----	----- A NUMBER. CAN I HAVE A NUMBER NOW.	19900:
C		20000:
470	IF (LLEAD) GO TO 474	20100:
472	LLEAD = .TRUE.	20200:
	KOUNT = IFMT(2,L) - 27	20300:
	GO TO 700	20400:
474	IF (LLETR) GO TO 700	20500:
	IF (LPMLT) GO TO 472	20600:
	KOUNT = 10*KOUNT + IFMT(2,L) - 27	20700:
	GO TO 700	20800:
C-----	----- A COMMA. CAN WE HAVE A COMMA HERE.	20900:
C		21000:
580	CONTINUE	21100:
583	LCOMA = .TRUE.	21200:
	ASSIGN 120 TO MGO	21300:
	GO TO 700	21400:
C-----	----- L. PAREN. MUST BE PRECEDED BY A NUMBER.	21500:
C		21600:
C		21700:
		21800:
		21900:
		22000:

590	CONTINUE	LPAREN = LPAREN + 1	22100
		LLEAD = .TRUE.	22200
		LPMLT = .TRUE.	22300
		LLETR = .TRUE.	22400
		LPERD = .TRUE.	22500
		GO TO 593	22600
C			22700
C	----	A SLASH. SLASHES CAN COME ANYWHERE COMMAS CAN.	22800
			22900
600	CONTINUE		23000
		GO TO 593	23100
C			23200
C	----	R. PAREN. LIKE A SLASH OR A COMMA.	23300
			23400
610		RPAREN = RPAREN + 1	23500
		GO TO 593	23600
C			23700
C	----	A PERIOD. CAN I HAVE A PERIOD HERE.	23800
			23900
650		IF (LPERD) GO TO 185	24000
		LPERD = .TRUE.	24100
		GO TO 700	24200
C			24300
C	----	AN APOST. SHIFT INTO HOLLERITH.	24400
			24500
670		IF (LLEAD) GO TO 187	24600
		IF (LPMLT) GO TO 187	24700
		IF (LLETR) GO TO 187	24800
		IF (LPERD) GO TO 187	24900
		KEYCHR = IFMT(1,L)	25000
671	CONTINUE		25100
		KOUNT = 0	25200
		KK = L	25300
673		KOUNT = KOUNT + 1	25400
		IF (KOUNT .GT. 1680) GO TO 677	25500
		KK = KK + 1	25600
674		IF (IFMT(1,KK) .NE. KEYCHR) GO TO 673	25700
C	----	SECOND APOST. ENCOUNTERED.	25800
		KSAV = KK	25900
		KK = KK + 1	26000
675		IF (IFMT(1,KK) .NE. KEYCHR) GO TO 676	26100
C	----	DOUBLE APOST.	26200
		IFMT(1,KK) = 47	26300
		IFMT(2,KK) = 48	26400
		GO TO 673	26500
676		KK = KSAV	26600
		IFMT(1,KK) = KBL	26700
		IFMT(2,KK) = 48	26800
		GO TO 676	26900
677		CALL PAG (1)	27000
		WRITE (N6,9037)	27100
678		KOUNT = KOUNT - 1	27200
		NO = KOUNT	27300
C			27400
			27500

C-----	----- BUILD '35H' IF KOUNT IS 35.	27600
C	DO 680 I=1,4	27700
	MPY = 10** (4-I)	27800
	MMOD = NO / MPY	27900
	IANS(I) = INO(MMOD+1)	28000
	NO = NO - MMOD*MPY	28100
680	DO 682 I=1,4	28200
	IF (IANS(I) .NE. INO(I)) GO TO 683	28300
682	IANS(I) = KBL	28400
683	DO 686 I=1,4	28500
	IF (IANS(I) .EQ. KBL) GO TO 686	28600
	K = K + 1	28700
684	JCARD(I,K) = IANS(I)	28800
686	CONTINUE	28900
	K = K + 1	29000
	JCARD(I,K) = LHH	29100
	GO TO 281	29200
C-----	----- BUMP K. STORE A CHARACTER. GO GET THE NEXT ONE.	29300
C		29400
698	EFGLG = .FALSE.	29500
700	K = K + 1	29600
	JCARD(I,K) = IFMT(I,L)	29700
	GO TO M80, (120,126,140,40,281)	29800
C-----	----- ALL CARDS HAVE BEEN SCANNED. WRAP IT UP AND GO HOME.	29900
C		30000
730	IF (LPAREN .EQ. RPAREN) GO TO 740	30100
	CALL PAG (3)	30200
	WRITE (N6,9030) LPAREN, RPAREN	30300
740	CONTINUE	30400
	JC = K + 1	30500
	MOVE = 1	30600
	RETURN	30700
9030	FORMAT(19HONO. OF L. PARENS =12 / 19HONO. OF R. PARENS =12)	30800
9037	FORMAT(' SOMETHING WRONG WITH APOST. HOLLERITH FMT.')	30900
	END	31000
		31100
		31200
		31300

	SUBROUTINE IMPSET(IDX)	100
	INTEGER COMMA, NPAREN	200
	LOGICAL OUPDAT	300
	COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K	400
1	IRSX, LIST(8191), LOCN, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM	500
2	NERO, NERO, NEMKEY	600
	COMMON /ALPH/ KBL, KABC(28), KDIG(10), KSPK(11)	700
	COMMON /HASH/ ILOC, LOC, NAME, IVAL	800
	COMMON /XUPDAT/ OUPDAT, ICDICT(27)	900
	DATA COMMA, NPAREN / 1H, 1H /	1000
	OUPDAT = .TRUE.	1100
	IFLAG = IABS(ITYPE)	1200
	IF (IFLAG .GT. 18) GO TO 25	1300
	GO TO (25, 25, 3, 25, 5, 6, 25, 8, 25, 25, 25, 12, 13, 25, 25, 25, 25, 18),	1400
	IFLAG	1500
3	KEY = IVAL + 5	1600
	GO TO 50	1700
6	KEY = 2	1800
	GO TO 50	1900
12	KEY = 3	2000
	GO TO 50	2100
13	KEY = 4	2200
	GO TO 50	2300
18	KEY = 5	2400
	GO TO 50	2500
5	CONTINUE	2600
8	CONTINUE	2700
25	KEY = 0	2800
50	CONTINUE	2900
C----	----- FIND FIRST CHARACTER	3000
95	DO 96 I=1, 26	3100
	IF (KARD(KOL) .EQ. KABC(I)) GO TO 97	3200
96	CONTINUE	3300
	GO TO 105	3400
C----	----- FIND SECOND CHARACTER	3500
97	IF (KARD(KOL+1) .EQ. KSPK(7)) GO TO 970	3600
	J = I	3700
	GO TO 99	3800
970	KOL = KOL+2	3900
	DO 98 J=1, 26	4000
	IF (KARD(KOL) .EQ. KABC(J)) GO TO 99	4100
98	CONTINUE	4200
	GO TO 105	4300
99	DO 100 K=1, J	4400
100	ICDICT(K) = KEY	4500
C----	----- TEST FOR END OF STATEMENT	4600
	IF (KARD(KOL+1) .NE. COMMA) GO TO 101	4700
	KOL = KOL+2	4800
	GO TO 95	4900
101	IF (KARD(KOL+1) .NE. NPAREN) GO TO 105	5000
	IF (KOL+2 .GE. KMAX) GO TO 120	5100
	KOL = KOL+2	5200
	IDX = 1	5300
	GO TO 150	5400
C----	----- ERRORS	5500

105	!CALL ERROR	5600
	!OUPDAT = .FALSE.	5700
120	!IDX = 2	5800
150	CONTINUE	5900
	!IF (OUPDAT) CALL SETTMP	6000
	RETURN	6100
	END	6200

SUBROUTINE INPUT										100
C	C	C	C	C	C	C	C	C	C	200
C	C	C	C	C	C	C	C	C	C	300
C	C	C	C	C	C	C	C	C	C	400
C	C	C	C	C	C	C	C	C	C	500
C	C	C	C	C	C	C	C	C	C	600
C	C	C	C	C	C	C	C	C	C	700
C	C	C	C	C	C	C	C	C	C	800
C	C	C	C	C	C	C	C	C	C	900
C	C	C	C	C	C	C	C	C	C	1000
C	C	C	C	C	C	C	C	C	C	1100
C	C	C	C	C	C	C	C	C	C	1200
C	C	C	C	C	C	C	C	C	C	1300
C	C	C	C	C	C	C	C	C	C	1400
C	C	C	C	C	C	C	C	C	C	1500
C	C	C	C	C	C	C	C	C	C	1600
C	C	C	C	C	C	C	C	C	C	1700
C	C	C	C	C	C	C	C	C	C	1800
C	C	C	C	C	C	C	C	C	C	1900
C	C	C	C	C	C	C	C	C	C	2000
C	C	C	C	C	C	C	C	C	C	2100
C	C	C	C	C	C	C	C	C	C	2200
C	C	C	C	C	C	C	C	C	C	2300
C	C	C	C	C	C	C	C	C	C	2400
C	C	C	C	C	C	C	C	C	C	2500
C	C	C	C	C	C	C	C	C	C	2600
C	C	C	C	C	C	C	C	C	C	2700
C	C	C	C	C	C	C	C	C	C	2800
C	C	C	C	C	C	C	C	C	C	2900
C	C	C	C	C	C	C	C	C	C	3000
C	C	C	C	C	C	C	C	C	C	3100
C	C	C	C	C	C	C	C	C	C	3200
C	C	C	C	C	C	C	C	C	C	3300
C	C	C	C	C	C	C	C	C	C	3400
C	C	C	C	C	C	C	C	C	C	3500
C	C	C	C	C	C	C	C	C	C	3600
C	C	C	C	C	C	C	C	C	C	3700
C	C	C	C	C	C	C	C	C	C	3800
C	C	C	C	C	C	C	C	C	C	3900
C	C	C	C	C	C	C	C	C	C	4000
C	C	C	C	C	C	C	C	C	C	4100
C	C	C	C	C	C	C	C	C	C	4200
C	C	C	C	C	C	C	C	C	C	4300
C	C	C	C	C	C	C	C	C	C	4400
C	C	C	C	C	C	C	C	C	C	4500
C	C	C	C	C	C	C	C	C	C	4600
C	C	C	C	C	C	C	C	C	C	4700
C	C	C	C	C	C	C	C	C	C	4800
C	C	C	C	C	C	C	C	C	C	4900
C	C	C	C	C	C	C	C	C	C	5000
C	C	C	C	C	C	C	C	C	C	5100
C	C	C	C	C	C	C	C	C	C	5200
C	C	C	C	C	C	C	C	C	C	5300
C	C	C	C	C	C	C	C	C	C	5400
C	C	C	C	C	C	C	C	C	C	5500

	KOL = MIN0(KOL, KMAX)	5600
	IF (MOVE.EQ. 1) CALL MOVER(0,DUMMY,DUMMY,0)	5700
	JC = JC - 1	5800
	CALL TRANCE(JCARD,JC)	5900
	GO TO 44	6000
411	KOL = KOL - 1	6100
	IF (KRSX.NE. KSPK(10)) KOL = KMAX	6200
	CALL TRANCE(KARD,KOL)	6300
44	MOVE = 0	6400
	JC = 0	6500
	NROUT = 0	6600
	FRSTSB = .TRUE.	6700
	IF (KRSX.EQ. KSPK(10)) GO TO 42	6800
	IF (NREC) 5,5,8	6900
	SET UP PAGE CAPTION.	7000
		7100
		7200
	IF (KBUFF(1).EQ.KSPK(8)) GO TO 360	7300
55	CALL EQUAT4(66,JOB,KBUFF(7))	7400
	CALL CENTON	7500
	GO TO 8	7600
		7700
	COPY THROUGH COMMENTS, LOOKING FOR FORTRAN STATEMENTS.	7800
		7900
7	CONTINUE	8000
	CALL TRANCE(KBUFF,72)	8100
	CALL READ(KBUFF, 461)	8200
	GO TO 62	8300
61	CALL STARL (KBUFF)	8400
62	CONTINUE	8500
	IF (KB1.EQ.KABC(3).OR.KB1.EQ.KSPK(10)) GO TO 7	8600
	IF (KB1.EQ.KSPK(8)) GO TO 37	8700
	---- FLAG CONTINUATION CARDS INTERSPERSED WITH COMMENTS.	8800
	IF (KB6.EQ. KBL .OR. KB6.EQ. KDIG(1)) GO TO 80	8900
	CALL ERROR	9000
	HAVE FORTRAN STATEMENT. START PACKING KARD.	9100
		9200
		9300
80	NREC=NREC+1	9400
	NCARD = NCARD+1	9500
	CALL EQUAT4(80,KOUT,KBUFF)	9600
	CALL EQUAT4(72,KARD,KBUFF)	9700
	K7=7	9800
	K72=72	9900
	LOOK FOR CONTINUATION CARDS AND PACK IN KARD.	10000
		10100
		10200
		10300
10	DO 13 J=2,20	10400
	CALL READ(KBUFF, 471)	10500
	GO TO 72	10600
71	CALL STARL (KBUFF)	10700
72	CONTINUE	10800
	IF (KBUFF(1).NE. KABC(3)) GO TO 105	10900
	---- ELIMINATE COMMENT CARDS INTERSPERSED WITH CONTINUATIONS.	11000
	CALL TRANCE(KBUFF,72)	

105	GO TO 10	11100
	DO 11 I=1,72	11200
11	IF (KBUFF(1).NE.KBL) GO TO 12	11300
	CONTINUE	11400
	GO TO 10	11500
C		11600
12	IF (KB1.EQ.KABC(3).OR.KB1.EQ.KSPK(4).OR.KB1.EQ.KSPK(8).OR.KB6.EQ.	11700
	KBL.OR.KB6.EQ.KDIG(1)) GO TO 16	11800
	K7=K7+66	11900
	CALL EQUAT4(66,KARD(K7),KBUFF(7))	12000
	L = 72	12100
	DO 13 I=1,8	12200
	L = L+1	12300
	ICHAR = KBUFF(L)	12400
	OSEQ2(1) = OCHAR	12500
13		12600
C		12700
C	NINETEEN CONTINUATION CARDS. LOAD EMPTY BUFFER.	12800
C		12900
14	CALL READ(KBUFF, &81)	13000
	GO TO 82	13100
81	CALL STARL (KBUFF)	13200
82	CONTINUE	13300
	DO 15 I=1,72	13400
15	IF (KBUFF(1).NE.KBL) GO TO 16	13500
	CONTINUE	13600
	GO TO 14	13700
C		13800
C	GET STATEMENT NUMBER, IF ANY.	13900
C		14000
16	KODE=0	14100
	OSTMTN = .FALSE.	14200
	DO 19 I=1,5	14300
	IF (KARD(1).EQ.KBL) GO TO 19	14400
	DO 17 J=1,10	14500
	IF (KARD(1).EQ.KDIG(J)) GO TO 18	14600
17	CONTINUE	14700
	IF (KARD(1) .NE. KSPK(4)) GO TO 20	14800
C	----- JCL. SKIP CARD	14900
	CALL PAG(1)	15000
	WRITE (6,170)	15100
170	FORMAT(10X25H** JCL CARD DELETED. **)	15200
	GO TO 44	15300
18	IF (J.EQ.1.AND.KODE.EQ.0) GO TO 19	15400
	OSTMTN = .TRUE.	15500
	GO TO 20	15600
19	CONTINUE	15700
C		15800
C	SQUEEZE OUT ALL BLANKS. PUT BLANK IN KARD(1), AND START SQUEEZE	15900
C	IN KARD(2).	16000
20	KMAX = K7 + 65	16100
	KARD(6)=KBL	16200
36	CONTINUE	16300
C	RETURN	16400
		16500

Line	Code	Statement	Address
360	C	NEWSUB = 1	16600
370	C	GO TO 370	16700
370	C	NEWSUB = 0	16800
370	C	121	16900
38	C	DO 38 J=2,72	17000
38	C	IF (KBUFF(J).EQ.KBL) GO TO 38	17100
38	C	IF (KBUFF(J).NE.KLAST(I)) GO TO 39	17200
38	C	I=I+1	17300
38	C	CONTINUE	17400
38	C	GO TO 400	17500
39	C	I=1	17600
39	C	DO 40 J=2,72	17700
39	C	IF (KBUFF(J).EQ.KBL) GO TO 40	17800
39	C	IF (KBUFF(J).NE.KSTOP(I)) GO TO 400	17900
39	C	I=I+1	18000
39	C	CONTINUE	18100
39	C	IF (I.EQ.5) GO TO 41	18200
39	C	CONTINUE	18300
39	C	IF (NEWSUB) 7,7,55	18400
40	C	CONTINUE	18500
40	C	CALL CLOSE	18600
40	C	WRITE (N6,46)	18700
40	C	STOP	18800
41	C	CONTINUE	18900
41	C	CALL CLOSE	19000
41	C	WRITE (N6,46)	19100
41	C	STOP	19200
42	C	----- MULTIPLE STATEMENTS ON ONE EXTENDED CARD IMAGE	19300
42	C	KOL = KOL + 2	19400
42	C	KMAX = KMAX - KOL + 1	19500
42	C	KARD(1) = KBL	19600
42	C	CALL EQUAT4(5, KARD(2), KARD(1))	19700
42	C	CALL EQUAT4(KMAX, KARD(7), KARD(KOL))	19800
42	C	GO TO 36	19900
42	C	FORMAT('O END OF TRANSLATION')	20000
42	C	END	20100
46	C	END	20200
46	C	END	20300

```

SUBROUTINE INSRTR( L1,TEXT,NCHAR,KEY )
  INTEGER BLANK, EQUAL, JCARD(80), PCARD, PMAX
  LOGICAL OSTMN, FRSTSB
  LOGICAL*1 TEXT(1), OCHAR, OSEQ1(8)
  REAL*8 SEQ1, SEQ2, SEQ3
  COMMON /SEQ/ SEQ1, SEQ2, SEQ3
  COMMON ITYPE JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K
  1 RX, LIST(8191), LOCN, MINA, MAXA, NCHXX, NEXT, NRT, NREC, NROUT, NSYM
  2 , NERR, NERO, NEMKEY
  COMMON /ALPH/ KBL, KABC(28), KDIG(10), KSPK(11)
  COMMON / DELTA/ MOVE, PCARD(80), ICARD(1600), IOL, NCARD
  1 OSTMN, FRSTSB, IC, JC
  COMMON /TAPES/ N1, N2, N3, N4, N5, N6
  EQUIVALENCE ( OCHAR, ICHAR ), ( BLANK, KBL ), ( OSEQ1(1), 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.
  IF ( OSTMN .AND. FRSTSB ) GO TO 20
  DO 10 I=1,6
    JCARD(I) = BLANK
  10 GO TO 30
  20 FRSTSB = .FALSE.
  CALL EQUAT4( 6, JCARD, KARD )
  DO 25 I=1,6
    PCARD(I) = BLANK
  25 J = 7
  30 IF ( NCHAR .LE. 0 ) GO TO 42
  DO 40 I=1, NCHAR
    OCHAR = TEXT(I)
    JCARD(J) = ICHAR
    J = J+1
  40 JCARD(J) = BLANK
  JCARD(J+1) = KSPK(KEY)
  JCARD(J+2) = BLANK
  J = J+3
  C----- MOVE DELETED TEXT TO NEW CARD IMAGE AND BLANK OUT REMAINDER
  JMAX = 72
  IF ( L1 .GE. 0 ) GO TO 41
  JCS = -L1
  GO TO 42
  41 K = KOL
  KOL = L1
  CALL MOVER( 0, DMMY, DMMY, 0 )
  JCS = JC
  KOL = K
  42 CALL MOVER( 0, DMMY, DMMY, 0 )
  K = JCS
  JCARD(J) = KARD(K)
  420 K = K+1
  J = J+1
  43 IF ( J .LE. JMAX ) GO TO 50
  CALL PAG( 1 )
  WRITE (N4,80) JCARD

```

46	CALL TRANCE(JCARD,JMAX)	5600
	DO 46 L=1,5	5700
	JCARD(L) = BLANK	5800
	J = 7	5900
50	JCARD(J-1) = KSPK(8)	6000
	IF (K.LT. JC) GO TO 420	6100
	JC = JCS	6200
	IF (J.EQ. 7) GO TO 100	6300
70	DO 70 L=J,72	6400
	JCARD(L) = BLANK	6500
	CALL PAG(1)	6600
	WRITE (N4,90) JCARD	6700
	CALL TRANCE(JCARD,JMAX)	6800
80	FORMAT(8X90A1)	6900
	----- MOVE TEXT INTO OLD IMAGE AREA	7000
100	IF (KEY.GT. 1) RETURN	7100
	CALL MOVER(NCHAR,TEXT,1,0)	7200
	RETURN	7300
	END	7400

```

SUBROUTINE MOVER( NT, TEXT, INV, MT)
INTEGER BLANK
LOGICAL OSTMTN, FRSTB
LOGICAL#1 TEXT(1), ICHAR, OSEQG(8)
REAL#8 SEQ1, SEQ2, SEQ3
COMMON /SEQ/ SEQ1, SEQ2, SEQ3
COMMON / DELTA/ MOVE, JCARD(80), ICARD(1600), IOL, NCARD
1. OSTMTN, FRSTB, IC, JC
COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K
1: NSX, LIST(8191), LOCN, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM
2: NERR, NERO, NEMKEY
COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(11)
COMMON /NU/ NCHM
EQUIVALENCE ( ICHAR, ICHAR ), ( OSEQG(1), SEQ2 )
DATA BLANK / 1H /
----- TEST 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 TEXT = MOVED TO THE NEW CARD IMAGE.
C TEXT = CHARACTERS TO BE MOVED
C INV = INCREMENT IN TEXT FIELD ( 1=LOGICAL#1, 4=INTEGER)
C MT = NUMBER OF NON-BLANK CHARACTERS TO SKIP AFTER THE MOVE
C IF ( JC.GT. 1 ) GO TO 10
C----- MOVE STATEMENT NUMBER FIELD
1 CALL EQUAT4( 6, JCARD, KARD )
C IC = 7
C JC = 7
C MOVE = 1
C N = NT
C M = IABS( MT )
C NN = KOL-IC+1
C
C IF ( NN.LE. 0 ) GO TO 30
C----- MOVE KOL-IOL CHARACTERS FROM OLD CARD TO NEW CARD
C IF ( N.LT. 0 ) GO TO 20
C CALL EQUAT4( NN, JCARD(JC), KARD(IC) )
C JC = JC + NN
C IC = KOL + 1
C IF ( N.LE. 0 ) GO TO 50
C----- MOVE N CHARACTERS FROM TEXT FIELD TO NEW CARD.
C IT = 1
C DO 40 I=1, N
C OCHAR = TEXT(IT)
C IT = IT+INV
C JCARD(JC) = ICHAR
C JC = JC+1
C CONTINUE
C IF ( M.LE. 0 ) GO TO 55
C----- COUNT PAST M NONBLANKS
C NNB = 0
C DO 52 I=1, KMAX
C IF ( KARD(I) .EQ. KBL ) GO TO 52

```

5600
5700
5800
5900
6000
6100
6200
6300
6400
6500

NMB = NMB + 1
IF (NMB .GE. M) GO TO 54
CONTINUE
KOL = I
IC = I + 1
CONTINUE
RETURN
END
C 160
C

J518,'55 REVOM '(TAMR
CJ,C1,LOK,TM,TN ,5555 TNI

100	100	100	100
200	200	200	200
300	300	300	300
400	400	400	400
500	500	500	500
600	600	600	600
700	700	700	700
800	800	800	800
900	900	900	900
1000	1000	1000	1000
1100	1100	1100	1100
1200	1200	1200	1200
1300	1300	1300	1300
1400	1400	1400	1400
1500	1500	1500	1500
1600	1600	1600	1600
1700	1700	1700	1700
1800	1800	1800	1800
1900	1900	1900	1900
2000	2000	2000	2000
2100	2100	2100	2100
2200	2200	2200	2200
2300	2300	2300	2300
2400	2400	2400	2400
2500	2500	2500	2500
2600	2600	2600	2600
2700	2700	2700	2700
2800	2800	2800	2800
2900	2900	2900	2900
3000	3000	3000	3000
3100	3100	3100	3100
3200	3200	3200	3200
3300	3300	3300	3300
3400	3400	3400	3400
3500	3500	3500	3500
3600	3600	3600	3600
3700	3700	3700	3700
3800	3800	3800	3800
3900	3900	3900	3900
4000	4000	4000	4000
4100	4100	4100	4100
4200	4200	4200	4200
4300	4300	4300	4300

```

SUBROUTINE MULTEQ( KEQ )
DIMENSION LOCEQ(80)
COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K
1 RX
COMMON /ALPH/ KBL, KABC(26), KDIQ(10), KSPK(11)
COMMON /DELTA/ MOVE, JCARD(80)
KER = KEQ
LOCEQ(2) = KER
LOCEQ(1) = 6
NEQ = 2
10 KER = KER + 1
IF ( KARD(KER) .EQ. KBL ) GO TO 10
IF ( KARD(KER) .GE. ZERO .AND. KARD(KER) .LE. NINE ) GO TO 80
DO 50 I=KER, KMAX
IF ( KARD(I) .EQ. KSPK(1) ) GO TO 100
IF ( KARD(I) .EQ. KSPK(10) ) GO TO 80
50 CONTINUE
80 CALL SUBSUB
GO TO 200
C----- MULTIPLE REPLACEMENT STATEMENT
100 NEQ = NEQ + 1
KER = I
LOCEQ(NEQ) = KER
GO TO 10
C----- WRITE NEW CARDS FROM OLD
200 CONTINUE
CALL EQUAT4( 6, JCARD, KARD )
LAST = KMAX
IF ( KARD(1) .EQ. KSPK(10) ) LAST = 1 - 1
NEQX = NEQ - 1
J = NEQX
DO 250 I=1, NEQX
KK = LOCEQ(I)
N = LAST - KK
CALL EQUAT4( N, JCARD(7), KARD(KK+1) )
CALL TRANCE( JCARD, N+6 )
JCARD(1) = KBL
CALL EQUAT4( 5, JCARD(2), JCARD(1) )
LAST = LOCEQ(J+1) - 1
J = J - 1
250 MOVE = -1
RETURN
END

```

	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000	2100	2200	2300	2400	2500	2600	2700	2800	2900	3000	3100
C	SUBROUTINE PAG (N)																														
C	THIS SUBROUTINE DOES THE GENERAL PAGE COUNTING FOR INDEX.																														
	LOGICAL KSBO, KCDO																														
	COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K																														
	1	RSX, LIST(819), LOCH, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM																													
	2	NERR, NERO, MENKEY																													
		COMMON /ALPH/ KRL, KABC(26), KDIG(10), KSPK(11)																													
		COMMON /BETA/ MACH, MXLI, N15B, LINE, NPAGE, KSBO, KCDO, KOUT(80)																													
		COMMON /KAPPA/ KAP(11)																													
		COMMON /OMEGA/ KLAST(4), KSTOP(4)																													
		COMMON /SIGMA/ KSTIJ(11,60)																													
		COMMON /TAPES/ N1, N2, N3, N4, N5, N6																													
		K = IABS(N)																													
		IF (N .LE. 0) KCDO = .FALSE.																													
		IF (KSBO) K = K+2																													
		IF (KCDO) K = K+2																													
		IF (LINE+K .LE. MXLI) GO TO 10																													
		WRITE (N6,100)																													
	10	LINE = 0																													
		IF (KSBO) WRITE (N6,110) JOB																													
		IF (KCDO) WRITE (N6,120) KOUT																													
		LINE = LINE+K																													
		KSBO = .FALSE.																													
		KCDO = .FALSE.																													
		RETURN																													
	100	FORMAT(1H1)																													
	110	FORMAT(1H013X66A1)																													
	120	FORMAT(8H0 --- 80A1,5H ---)																													
		END																													

```

SUBROUTINE RETRNI( MODE )
LOGICAL*1 OTAB(256), ODX(4), OCHAR
LOGICAL FIRST
COMMON ITYPE, JOB(66), KARD(1326), KBUFF(60), KLEAR(8), KMAX, KODE, KOL, K
1:RSX, LIST(8191), LOCH, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM
2: NERR, NERO, MEMKEY
COMMON /ALPH/ KBL, KABC(26), KDIB(10), KSPK(11)
COMMON /GAMMA/ OTAB
EQUIVALENCE ( ODX(1), ), ( OCHAR, KRSX )
FIRST = .TRUE.
SEQ2 = SEQ2
NVRSMT = 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 = .FALSE.
ITYPE = 1
IF ( KARD(KOL) .NE. KSPK(8) ) GO TO 13
----- AN ASTERISK IN THIS POSITION INDICATES AN EXPLICIT
C TYPE DECLARATION SIZE SPECIFICATION.
C 1: E... INTEGER FUNCTION CALC#2( ... )
C LAST = KOL + 3
I = KOL
DO 31 J=KOL, LAST
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 MOVER( 0, 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) = OCHAR
ODX(4) = OTAB(1)
IF ( I .LE. 36 ) GO TO 5
41: KOL = KOL + 1
IF ( KOL .GT. KMAX ) RETURN
GO TO 13
100: CONTINUE
RETURN
END

```

	SUBROUTINE SIXFR7	100
C	THIS SUBROUTINE SHORTENS 7 CHARACTER VARIABLE NAMES TO 6	200
C	CHARACTERS AND CHECKS FOR UNIQUENESS THEN ADDS THE NEW	300
C	NAMES TO THE SYMBOL TABLE AND CROSSREFERENCES OLD AND NEW	400
C	NAMES.	500
C		600
C		700
	DIMENSION ONM1(10), VOWEL(6), DOLLAR(2)	800
	REALS NAME(2), SYMTAB(2,800), NEWNM(800)	900
	INTEGER#2 IREF(800)	1000
	LOGICAL#1 ONM(16)	1100
	LOGICAL VAXFLG, DBFLAG, CHGFLG	1200
C		1300
	COMMON ITYPE, JOB(100), KARD(1320), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K	1400
	1 RSX, LIST(8191), LOCN, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM	1500
	2 MERR, NERO, MEMKEY	1600
	COMMON /HASH/ ILOC, LOC, NAME	1700
	COMMON /SIX7/ NEWNM, IREF	1800
	COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(11)	1900
	COMMON /FLAGS/ DBFLAG, VAXFLG	2000
C		2100
	EQUIVALENCE (NAME(1), ONM(1)), (ONM(1), ONM1(1))	2200
C	EQUIVALENCE (SYMTAB(1,1), LIST(4192))	2300
		2400
	DATA VOWEL /1HA, 1HE, 1HI, 1HO, 1HU, 1HY/	2500
	DATA CHGFLG /FALSE/	2600
	DATA DOLLAR /1HS, 1H_/	2700
	DO 5 I=1,800	2800
	IREF(I)=0	2900
	CONTINUE	3000
5	N4=1	3100
C		3200
C	FIND 6 OR LESS CHARACTER NAMES IN SYMTAB AND MOVE TO NEWNM	3300
C		3400
	DO 200 I=1, ILOC	3500
	NAME(I)=SYMTAB(1, I)	3600
	IF (ONM1(I).NE.KBL) GO TO 200	3700
	NEWNM(I)=NAME(I)	3800
	CONTINUE	3900
200		4000
C		4100
C	SEARCH FOR NAMES TO BE CHANGED	4200
C		4300
	N7=2	4400
7	N6=N7	4500
10	N1=N4	4600
	DO 20 I=N1, ILOC	4700
	IF (.NOT.VAXFLG) GO TO 15	4800
	IF (CHGFLG) GO TO 14	4900
	NAME(1)=SYMTAB(1, I)	5000
	NAME(2)=SYMTAB(2, I)	5100
	DO 240 J=N6, 16	5200
	IF (ONM1(J).EQ.KBL) GO TO 16	5300
	IF (ONM1(J).NE.DOLLAR(1).AND.ONM1(J).NE.DOLLAR(2)) GO TO 240	5400
14	CHGFLG=.TRUE.	5500
	J1=J	

240	N4=1 N7=J GO TO 50 CONTINUE	5600 5700 5800 5900 6000 6100 6200 6300 6400 6500 6600 6700 6800 6900 7000 7100 7200 7300 7400 7500 7600 7700 7800 7900 8000 8100 8200 8300 8400 8500 8600 8700 8800 8900 9000 9100 9200 9300 9400 9500 9600 9700 9800 9900 10000 10100 10200 10300 10400 10500 10600 10700 10800 10900 11000
15	GO TO 16 NAME(1)=SYMTAB(1,1) NAME(2)=SYMTAB(2,1) IF(OMM1(8).EQ.KBL)GO TO 17 CHGFLG=.FALSE. N4=1 N2=1 GO TO 30	
16	IF(.NOT.CHGFLG)GO TO 20 CHGFLG=.FALSE. J1=8 N4=1 N2=1 GO TO 75	
20	CONTINUE	
25	RETURN	
30	LOOK FOR LAST VOWEL IN NAME DO 40 J=N2,14 IF(OMM1(17-J).EQ.KBL)GO TO 40 DO 40 K=1,6 IF(OMM1(17-J).NE.VOWEL(K))GO TO 40 J1=17-J GO TO 50 CONTINUE GO TO 90	
40	VOWEL FOUND - DELETE AND CREATE NEW NAME 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 CONTINUE OMM1(16)=KBL IF(CHGFLG)GO TO 10 IF(OMM1(8).EQ.KBL)GO TO 75 IF(J1.EQ.3)GO TO 90 N2=18-J1 GO TO 30	
50	LOOK FOR DUPLICATE DO 80 I=1,ILOC IF(NEWMM(I).NE.NAME(1))GO TO 80 IF(J1.EQ.3)GO TO 90 IF(J1.EQ.2)GO TO 100 IF(J1.EQ.1)GO TO 115 N2=18-J1 GO TO 30	
60		
70		
71		
75		

80	CONTINUE	11100
C	NAME IS UNIQUE - ADD NAME TO NEWNM AND FLAG IREF WITH THE	11200
C	NUMBER OF CHARACTERS IN THE NEW NAME	11300
C		11400
	NEWNM(N4)=NAME(1)	11500
	DO 85 I=3,8	11600
	IF(ONM1(I).NE.KBL)GO TO 85	11700
	IREF(N4)=I-2	11800
	GO TO 86	11900
85	CONTINUE	12000
86	N4=N4+1	12100
	IF(N4.LE.ILOC)GO TO 7	12200
	GO TO 25	12300
C		12400
C	NO MORE VOWELS - DELETE LAST CHARACTER	12500
C		12600
90	J1=2	12700
	M3=0	12800
	IF(ONM1(8).EQ.KBL)GO TO 97	12900
	DO 85 I=8,16	13000
	ONM1(I)=KBL	13100
95	CONTINUE	13200
	ISAVE=7	13300
	GO TO 75	13400
97	DO 98 I=3,7	13500
	IF(ONM1(10-I).EQ.KBL)GO TO 98	13600
	ISAVE=10-I	13700
	GO TO 100	13800
98	CONTINUE	13900
	ISAVE=3	14000
C		14100
C	STILL NOT UNIQUE, CHANGE LAST CHARACTER TO NUMBER	14200
C		14300
100	IF(N3.EQ.10)GO TO 110	14400
	M3=M3+1	14500
	ONM1(ISAVE)=KDIG(M3)	14600
	GO TO 75	14700
C		14800
C	STILL NOT UNIQUE, CHANGE LAST CHARACTER TO LETTER	14900
C		15000
110	M5=0	15100
	J1 = 1	15200
115	IF(N5.EQ.26)GO TO 120	15300
	M5=M5+1	15400
	ONM1(ISAVE)=KABC(M5)	15500
	GO TO 75	15600
120	NAME(1)=SYMTAB(1,N4)	15700
	NAME(2)=SYMTAB(2,N4)	15800
	WRITE(6,140) NAME	15900
	N4=N4+1	16000
	IF(N4.LE.ILOC)GO TO 7	16100
	GO TO 25	16200
140	FORMAT(27)HOCAN'T FIND UNIQUE NAME FOR ,A8,A8)	16300
	END	16400
		16500

C	SUBROUTINE SUBS(KOL1,KOL2)	100
C	THIS SUBROUTINE SUBSTITUTES THE 6 CHARACTER NAME FOR THE 7	200
C	CHARACTER NAME.	300
C		400
	LOGICAL DFLAG,VAXFLG	500
	LOGICAL#1 ONM(1),ONS	600
	COMPLEX#16 SYNTAB(800)	700
	REAL#8 NEWNM(800),NME	800
	INTEGER#2 IREF(800)	900
C		1000
	COMMON ITYPE,JOB(66),KARD(1326),KBUFF(80),KLEAR(8),KMAX,KODE,KOL,K	1100
	1 RSX,LIST(8191),LOCH,MINA,MAXA,NCHAR,NEXT,NRT,NREC,NROUT,NSYM	1200
	2 ,NERR,NERO,MEMKEY	1300
	COMMON /ALPH/ KBL,KABC(26),KDIG(10),KSPK(11)	1400
	COMMON /HASH/ ILOC,LOC,NAME	1500
	COMMON /FLAGS/ DFLAG,VAXFLG	1600
	COMMON /SIX7/ NEWNM,IREF	1700
C		1800
	EQUIVALENCE (ONM(1),NME)	1900
	EQUIVALENCE (ONS,KSAVE)	2000
C	EQUIVALENCE (SYNTAB(1),LIST(5192))	2100
		2200
	KEND=KMAX	2300
	ITYPE=1	2400
	KOL=KOL1	2500
	KMAX=KOL2	2600
	CALL SYMBOL	2700
	CALL UPDATE	2800
	IF(IREF(LOC).EQ.0)GO TO 70	2900
	NME=NEWMN(LOC)	3000
	K=KOL2-KOL1	3100
	L=K+1	3200
	DEL=L-IREF(LOC)	3300
	IF(DEL.EQ.0.OR.DEL.EQ.1)GO TO 30	3400
	IF(KOL2.EQ.KEND)GO TO 30	3500
	IF(DEL.LT.0)GO TO 20	3600
	DO 15 I=KOL2,KEND	3700
	KARD(I-DEL)=KARD(I)	3800
15	CONTINUE	3900
	KOL2=KOL2-DEL	4000
	KEND=KEND-DEL	4100
	GO TO 30	4200
20	DO 25 I=KOL2,KEND	4300
	COUNT=KEND-KOL2-I	4400
	KARD(COUNT-DEL)=KARD(COUNT)	4500
25	CONTINUE	4600
	KOL2=KOL2-DEL	4700
	KEND=KEND-DEL	4800
30	DO 40 I=KOL1,KOL2	4900
	ONS=ONM(1+2-KOL1)	5000
	KARD(I)=KSAVE	5100
40	CONTINUE	5200
	KMAX=KEND	5300
60	IF(.NOT.DFLAG)GO TO 70	5400
		5500

70	WRITE(6,80)SYMTAB(LOC),NEWNM(LOC)	5600
80	RETURN	5700
	FORMAT(1X,A16,13H REPLACED BY ,A8.59H IN SUBROUTINE, ENTRY, FUNCTI	5800
	C: ON, OR NAMED COMMON STATEMENT.)	5900
	END	6000

100	SUBROUTINE SLASHS(LAST)	
200	LOGICAL N1 OCHAR, ONAME(8)	
300	INTEGER IDATA(4)	
400	REAL N1 NAME(2), N1 NAME(3)	
500	COMMON I TYPE JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K	
600	1 RSX, LIST(191), LOCN, MINA, MAXA, NCHAR, NEXT, NRT, NREG, NROUT, NSYM	
700	2 NERR, NERO, MENKEY	
800	COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(11)	
900	COMMON / DELTA/ MOVE, JCARD(80), I CARD(1600), IOL, NCARD	
1000	1 OSTMTN, FRSTB, IC, JC	
1100	COMMON /HASH/ ILOC, LOC, NAME	
1200	EQUIVALENCE (IDATA(1), N1 NAME(1))	
1300	DATA N1 NAME / 9H DATA, 9H /	
1400	1M = LAST-KOL	
1500	IF (M .LE. 0) GO TO 11	
1600	CALL MOVER(0, DUMMY, DUMMY, M)	
1700	KOL = LAST	
1800	CALL SYMBOL	
1900	16LOK, LOK=XPI,)XPI(DRAK (, XSRK, COL, XAMK, LOK, TNCPL, RAPI, 116 TNI	
2000	17AX2, 1AX2, 515, 11 SHSALS, (TAMR	
2100	6+LOK = 6L	
2200	IF (KOL .GT. KMAX) GO TO 100	
2300	IF (KODE .LT. MINA .OR. KODE .GT. MAXA) GO TO 12	
2400	N1 NAME(2) = NAME(1)	
2500	N1 NAME(3) = NAME(2)	
2600	CALL UPDATE	
2700	IF (KRSX .NE. KSPK(3)) GO TO 13	
2800	---- SYMBOLS WITHIN PARENTHESES ARE SKIPPED.	
2900	IF (KOL .GT. KMAX) GO TO 21	
3000	KOL = KOL+1	
3100	KRSX = KARD(KOL)	
3200	IF (KRSX .NE. KSPK(5)) GO TO 12	
3300	KOL = KOL + 1	
3400	KRSX = KARD(KOL)	
3500	IF (KRSX .EQ. KBL) GO TO 1200	
3600	---- TEST FOR DATA IN TYPE DEFINITION STATEMENTS	
3700	13 IF (KRSX .NE. KSPK(4)) GO TO 11	
3800	L1 = KOL-1	
3900	KOL = L1	
4000	1515, 31 SHSALS, (TAMR	
4100	CJ, CI, LOK, 316 TNI	
4200	CALL MOVER(0, DUMMY, DUMMY, 1)	
4300	JCS = JC	
4400	CALL SYMBOL	
4500	16LOK, LOK=XPI,)XPI(DRAK (, XSRK	
4600	17AX2, 1AX2, 515, 41 SHSALS, (TAMR	
4700	, SCJ, EDOK, CJ, CI, LOK, 416 TNI	
4800	6+LOK = 6L	
4900	IF (KODE .LT. 0) GO TO 21	
5000	IF (KRSX .NE. KSPK(4)) GO TO 14	
5100	IF (JC .GT. JCS) L1 = -JCS	
5200	CALL INSRT(L1, IDATA(2), 12, 4)	
5300	KOL = KOL+1	
5400	GO TO 11	
5500	CALL ERROR	
	CONTINUE	

RETURN
END

5600
5700

PAGE 2 OF SLASHS

```

SUBROUTINE SORT( I60 )
REAL*8 NAME(2), SYMTAB(2,900), BLANK, TDNAME(4), OUTBF(10), STAR
REAL*8 SEQ1, SEQ2, SEQ3
LOGICAL*1 SYML(10,1)
LOGICAL OUPDAT
LOGICAL*1 MORE, ONAME(8), ODX(4), OIDX, OTAB(256), OCOMMA, OBLANK
LOGICAL TYTL / .FALSE. /
1 OOB(1)
DIMENSION JCDICT(127), NX(5), JLOC(1)
COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLFAR(8), KMAX, KODE, KOL, K
1 RSX, LIST(8191)
COMMON /SEQ/ SEQ1, SEQ2, SEQ3
COMMON /HASH/ ILOC, LOC, NAME
COMMON /XUPDAT/ OUPDAT, ICDICT(127)
COMMON /GAMMA/ OTAB
EQUIVALENCE ( ONAME(1), NAME ), ( ODX(1), IDX ), ( ODX(4), OIDX ),
( OCOMMA, COMMA ), ( OBLANK, BLANK ), ( OOB(1), OUTBF(1) )
EQUIVALENCE ( SYMTAB(1), LIST(4192) ), ( JLOC(1), LIST(7392) )
EQUIVALENCE ( SYMTAB(1), SYML(1,1) )
DATA TDNAME / SHCOMPLEX, SHINTEGER, SHLOGICAL, SHREAL /
DATA JCDICT / 8#5, 6#3, 12#5, 0 /, COMMA, BLANK / 1#,, 1# /
DATA STAR / SH##### /
IF ( I60 .EQ. 0 ) GO TO 200
TYTL = .FALSE.
IF ( ILOC .LE. 0 ) RETURN
DO 50 I=1, ILOC
NAME(I) = SYMTAB(1, I)
IF ( ONAME(8) .EQ. OBLANK ) GO TO 50
IF ( TYTL ) GO TO 40
TYTL = .TRUE.
PRINT 70, JOB
PRINT 60, NAME(1)
50 CONTINUE
200 RETURN
70 FORMAT('0'ILLEGAL VARIABLE NAMES IN ', 66A1/)
80 FORMAT(2XA8)
END

```

C

40:
50:
200:
70:
80:

```

C-----
SUBROUTINE STARL (K)
SUBROUTINE STARL FILLS (KE'UFF) WITH (#LAST).
DIMENSION K(80), L(80)
DATA L / 1H#, 1HL, 1HA, 1HS, 1HT, 75#1H /
DO 9 I=1,80
K(I) = L(I)
RETURN
END

```

```

100
200
300
400
500
600
700
800

```

```

SUBROUTINE SUBSUB
REAL*8 S$NAME(25)
COMMON /ALPH/ KBL,KABC(26),KDIG(10),KSPK(11)
COMMON /DELTA/ MOVE, JCARD(80), ICARD(1600), IOL, NCARD
COMMON /NU/ NH, KOL$X
COMMON /HASH/ ILOC, LOC
DATA AMP$ND / 1Ha /
DATA S$NAME /
1 SH1$B$B17 : SH1$B$B27 : SH1$B$B37 : SH1$B$B47 : SH1$B$B57 :
2 SH1$B$B67 : SH1$B$B77 : SH1$B$B87 : SH1$B$B97 : SH1$B$BA7 :
3 SH1$B$BB7 : SH1$B$BC7 : SH1$B$BD7 : SH1$B$BE7 : SH1$B$BF7 :
4 SH1$B$BG7 : SH1$B$BH7 : SH1$B$BI7 : SH1$B$BJ7 : SH1$B$BK7 :
5 SH1$B$BL7 : SH1$B$BM7 : SH1$B$BN7 : SH1$B$BO7 : SH1$B$BP7 : /
C-----
IPAR = 1
LPCNT = 0
IF ( KARD(KOL+1) .EQ. AMP$ND ) GO TO 40
CALL SYMBOL
KOL6 = KOL+6
PRINT 611, IPAR, LPCNT, KOL, KMAX, LOC, KRSX, ( KARD(IPX), IPX=KOL, KOL6)
FORMAT( ' SUBSUB 11', 5I5, 2XA1, 2XTA1)
IF ( KOL .GT. KMAX ) GO TO 100
IF ( KRSX .EQ. KSPK(10) ) GO TO 100
L1(IPAR) = KOL$X
CALL UPDATE
IF ( KRSX .NE. KSPK(13) ) GO TO 20
IF ( KRSX .EQ. KSPK(13) ) GO TO 20
C-----
K = KOL
KOL = L1(IPAR-1)
CALL FNCHK( 1 )
PRINT 613, IPAR, JC
FORMAT( ' SUBSUB 13', 6I5)
GO TO 10
C-----
K = KOL
KOL = L1(IPAR-1)
CALL FNCHK( 1 )
PRINT 615, IPAR, K, KOL
FORMAT( ' SUBSUB 15', 6I5)
KOL = K
GO TO 10
CONTINUE
PRINT 620
FORMAT( ' SUBSUB 20', 6I5)
IF ( KRSX .EQ. KSPK(5) ) GO TO 25

```

C-----	IF (KRSX .NE. KSPK(2)) GO TO 10	5600:
22	----- COMMA. BRANCH IF INSIDE FUNCTION	5700:
	CONTINUE	5800:
622	PRINT 622, IPAR, L1(IPAR-1), ILPG, LPCNT, KOL	5900:
	FORMAT(' SUBSUB 22', 615)	6000:
	IF (IPAR .LE. 1) GO TO 10	6100:
	IF (L1(IPAR-1) .GE. 0) GO TO 10	6200:
	IF (IPAR .GT. ILPG) GO TO 13	6300:
	ILPG = ILPG-1	6400:
	LPCNT = LPCNT+1	6500:
	KOL = KOL-1	6600:
	CALL INSRTR(L1(IPAR-1), SSNAME(LPCNT), 7, 1)	6700:
	KOL = KOL+1	6800:
	GO TO 13	6900:
C-----	----- RIGHT PAREN. IF FIRST LEVEL, SKIP.	7000:
25	CONTINUE	7100:
	IPAR = IPAR-1	7200:
625	PRINT 625, IPAR, L1(IPAR), ILPG, LPCNT, KOL	7300:
	FORMAT(' SUBSUB 25', 615)	7400:
28	IF (L1(IPAR) .GE. 0) GO TO 30	7500:
	IF (IPAR .GE. ILPG) GO TO 30	7600:
	ILPG = ILPG-1	7700:
	LPCNT = LPCNT+1	7800:
	KOL = KOL-1	7900:
	CALL INSRTR(L1(IPAR), SSNAME(LPCNT), 7, 1)	8000:
30	KOL = KOL+1	8100:
	KOL = KOL+1	8200:
	IF (KOL .GT. KMAX) GO TO 100	8300:
	KRSX = KARD(KOL)	8400:
	IF (KRSX .EQ. KSPK(5)) GO TO 25	8500:
	IF (KRSX .EQ. KSPK(2)) GO TO 22	8600:
	IF (KRSX .EQ. KBL) GO TO 30	8700:
	IF (KRSX .EQ. KSPK(10)) GO TO 100	8800:
	GO TO 10	8900:
40	CONTINUE	9000:
	IF (KOL .GE. KMAX) RETURN	9100:
C-----	----- LOOK FOR AXX WHERE XX IS A STATEMENT NUMBER	9200:
C-----	----- NOT A STATEMENT NUMBER UNLESS PRECEDED BY A (OR	9300:
	IF (KARD(KOL) .NE. KSPK(2) .AND. KARD(KOL) .NE. KSPK(3)) GO TO 11	9400:
	KOL = KOL+1	9500:
	GO TO 10	9600:
100	CONTINUE	9700:
	RETURN	9800:
200	CONTINUE	9900:
	CALL ERR(17, DUMMY, DUMMY)	10000:
	WRITE (6, 222) KARD	10100:
222	FORMAT(10X\$0A1)	10200:
	STOP	10300:
	END	10400:

C	SUBROUTINE SYMBOL	100
C	THIS SUBROUTINE INSPECTS KARD STARTING AT KOL+1 FOR THE PRESENCE	200
C	OF FORTRAN VARIABLES OR STATEMENT NUMBERS. IF FOUND, THE SYMBOL	300
C	IS PACKED IN CODE AS A NUMBER WHOSE BASE IS 37.	400
C		500
C		600
C		700
C	LOGICAL=1 ONM(8), ONAME(7), OCHAR, OTAB(256), ODX(4), OCHAL	800
C	LOGICAL DTOE, NUMLY	900
C	REALS NAME, BLANK	1000
C	COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL, K	1100
C	1 RSX, LIST(8191), LOGN, MINA, MAXA, NCHAR, NEXT, NRT, NREC, NROUT, NSYM	1200
C	2 NERR, NERO, MEMKEY	1300
C	COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(11)	1400
C	COMMON /BETA/ MACH, MXLI, N15B, LINE, NPAGE, KSBO, KCDO	1500
C	COMMON /GAMMA/ OTAB	1600
C	COMMON /DELTA/ MOVE, JCARD(80), ICARD(1600), IOL, NCARD	1700
C	1 OSTMTN, FRSTSB, IC, JC, IPAR, LI(25), KLAS(25)	1800
C	COMMON /KAPPA/ KAP(11)	1900
C	COMMON /NU/ NH, KOLSX	2000
C	COMMON /SIGMA/ KSTIJ(11,60)	2100
C	COMMON /OMEGA/ KLAST(4), KSTOP(4)	2200
C	COMMON /TAPES/ N1, N2, N3, N4, N5, N6	2300
C	COMMON /HASH/ ILOC, LOC, NAME(2), IVAL, NUMLY	2400
C		2500
C	DIMENSION KSYM(37)	2600
C	EQUIVALENCE (KBL, KSYM(1))	2700
C	EQUIVALENCE (ONM(1), NAME(1)), (ONM(2), ONAME(1)), (OCHAR, KRSX),	2800
C	* (ODX(1), I), (OCHAL, KLSX)	2900
C	DATA BLANK / 1H /	3000
C	A B C D E F G H I J K L M	3100
C	1 2 3 4 5 6 7 8 9 10 11 12 13	3200
C	N O P Q R S T U V W X Y Z	3300
C	14 15 16 17 18 19 20 21 22 23 24 25 26	3400
C	= 1 2 3 4 5 6 7 8 9 10 11	3500
C	START. SET UP INITIAL CONDITIONS.	3600
C	KODE=0	3700
C	NCHAR=0	3800
C	I = 0	3900
C	NAME(1) = BLANK	4000
C	NAME(2) = BLANK	4100
C	SEARCH FOR FIRST SYMBOL.	4200
C	KLSX=KARD(KOL)	4300
C	KOLSX = KOL	4400
C	KOL=KOL+1	4500
C	IF (KOL.GT.KMAX) GO TO 23	4600
C	KRSX = KARD(KOL)	4700
C	ODX(4) = OCHAR	4800
C		4900
C		5000
C		5100
C		5200
C		5300
C		5400
C		5500

C-----	3000	ODX(4) = OTAB(1) IF (1 - 37) 4, 3, 3000 ----- BRANCH IF APOSTROPHE IF (KRSX.EQ. KSPK(11)) GO TO 24 IF (KRSX.EQ. KSPK(10)) GO TO 230 IF (KRSX.NE. KSPK(3)) GO TO 2 L1(IPAR) = 0 KCLASS(IPAR) = 0 IPAR = IPAR+1 L1(IPAR) = KOL GO TO 2	5600 5700 5800 5900 6000 6100 6200 6300 6400 6500 6600 6700 6800 6900 7000 7100 7200 7300 7400 7500 7600 7700 7800 7900 8000 8100 8200 8300 8400 8500 8600 8700 8800 8900 9000 9100 9200 9300 9400 9500 9600 9700 9800 9900 10000 10100 10200 10300 10400 10500 10600 10700 10800 10900 11000
C		TEST IF FIRST CHARACTER IS NUMERIC.	
C		IF (1 .GE. 27) GO TO 8	
C	4	FIRST CHARACTER IS ALPHA. TEST FOR LEADING PERIOD.	
C		IF FIRST ALPHA SYMBOL IS Z, CHECK FOR HEX. CONSTANT.	
C		IF (KLSX.EQ.KSPK(9)) GO TO 18	
C		IF (1.EQ.26) GO TO 19	
C		START ALPHA SYMBOL ASSEMBLY.	
C		KODE = MINA	
C	5	NCHAR=NCHAR+1	
		ONAME(NCHAR) = OCHAR	
		IF (NCHAR.GT.MXCH) GO TO 15	
	6	KOL=KOL+1	
		IF (KOL.GT.KMAX) GO TO 7	
		KRSX=KARD(KOL)	
		ODX(4) = OCHAR	
		ODX(4) = OTAB(1)	
		IF (1 - 37) 5, 6, 7	
C		PACK OUT KODE WITH ANY NECESSARY BLANKS AND EXIT.	
C			
C	7	CONTINUE	
		RETURN	
C		START NUMBER ASSEMBLY.	
C			
C		CONTINUE	
C	8	IVAL = 0	
C-----	9	----- SKIP LEADING ZEROES IF (1.EQ.27.AND.NCHAR.EQ.0) GO TO 10 IVAL = 10*IVAL + 1 - 27 NCHAR=NCHAR+1 KOL=KOL+1 IF (KOL.GT.KMAX) RETURN KRSX=KARD(KOL) ODX(4) = OCHAR ODX(4) = OTAB(1) IF (1 .GE. 27 .AND. 1 .LE. 36) GO TO 9 IF (KRSX.EQ.KSPK(9)) GO TO 11	
	10		

C	FIRST SYMBOL IS Z. SCAN LINE FOR POSSIBLE DEL. CONSTANT.	16600
C		16700
C		16800
C	----- TEST FOR DATA STATEMENT	16900
19	CONTINUE	17000
	IF (I TYPE .EQ. 7) GO TO 190	17100
	IF (KARD(KOL+1) .NE. KSPK(11)) GO TO 5	17200
190	I VAL = 0	17300
	KB1 = KOL+1	17400
	DO 21 IB=KB1,KMAX	17500
	DO 20 J=1,10	17600
	IF (KARD(IB).NE.KDIG(J)) GO TO 20	17700
	I VAL = 16*I VAL + J - 1	17800
	GO TO 21	17900
20	CONTINUE	18000
	DO 200 J=1,6	18100
	IF (KARD(IB) .NE. KABC(J)) GO TO 200	18200
	I VAL = 16*I VAL + J + 9	18300
	GO TO 21	18400
200	CONTINUE	18500
	IF (KARD(IB) .NE. KBL) GO TO 22	18600
21	CONTINUE	18700
	GO TO 23	18800
22	KB2 = IB-1	18900
	IB=KARD(IB)	19000
	IF (IB.EQ.KSPK(2) .OR. IB.EQ.KSPK(4) .OR. IB.EQ.KSPK(5) .OR. IB.EQ.KSPK(	19100
1	6) .OR. IB.EQ.KSPK(7) .OR. IB.EQ.KSPK(8) .OR. IB.EQ.KSPK(11)) GO TO 220	19200
	GO TO 5	19300
220	IF (KB2 .GE. KB1) CALL ERR(15,DUMMY,DUMMY)	19400
	GO TO 1	19500
C	END OF LINE EXIT.	19600
C		19700
C		19800
23	KRSX=KBL	19900
230	KODE=-1	20000
	NCHAR=0	20100
	KLSX=KBL	20200
	RETURN	20300
C	----- TEXT BETWEEN APOSTROPHES. LEAVE SPACE FOR MNH. LOOK FOR	20400
C	TRAILING APOSTROPHE AND MOVE TEXT TO JCARD FIELD.	20500
24	KOL = KOL - 1	20600
	CALL MOVER(4,4H H,1,1)	20700
	KH = JC - 1	20800
C	----- LOOP UNTIL TRAILING APOSTROPHE FOUND	20900
25	KOL = KOL + 1	21000
	IF (KOL .GT. KMAX) GO TO 29	21100
	IF (KARD(KOL) .NE. KSPK(11)) GO TO 25	21200
	IF (KARD(KOL+1) .NE. KSPK(11)) GO TO 26	21300
	CALL MOVER(0,DUMMY,DUMMY,-1)	21400
	GO TO 25	21500
26	KOL = KOL - 1	21600
	CALL MOVER(0,DUMMY,DUMMY,-1)	21700
	MH = JC - KH - 1	21800
	KODE = 1	21900
	DO 27 J=1,3	22000

	K1 = NH/10		22100
	K2 = NH - 10K1		22200
	KH = KH - 1		22300
	JCARD(KH) = KDIG(K2+1)		22400
	IF (K1 .LE. 0) GO TO 28		22500
	NH = K1		22600
	KOL = KOL + 1		22700
	KRSX = KARD(KOL)		22800
	IF (KRSX .EQ. KBL) GO TO 28		22900
	RETURN		23000
	----- IMPROPER END OF APOSTROPHE STRING.		23100
	WRITE (6,30) KARD(KOL-1), KSPK(11)		23200
	FORMAT('0SECOND APOSTROPHE NOT FOUND.',2X2A5)		23300
	GO TO 28		23400
	END		23500
27			
28			
C-----			
29			
30			

100	SUBROUTINE TDTST(N, LAST, BR, I1, I2)
200	INTEGER BR
300	COMMON ITYPE, JOB(66), KARD(1326), KBUFF(80), KLEAR(8), KMAX, KODE, KOL
400	COMMON /ALPH/ KBL, KABC(26), KDIG(10), KSPK(13)
500	KOL = LAST
600	IF (KARD(LAST+1) .EQ. KSPK(8)) GO TO 50
700	IF (KARD(LAST+1) .NE. KBL) GO TO 200
800	LAST = LAST + 1
900	GO TO 10
1000	LAST = LAST+2
1100	KARD(LAST-1) = KBL
1200	IF (KARD(LAST) .EQ. KDIG(11+1)) GO TO 150
1300	IF (KARD(LAST) .EQ. KDIG(12+1)) GO TO 150
1400	IF (I2 .LT. 10) GO TO 99
1500	I1 = I2/10
1600	I2 = I2 - 10*I1
1700	GO TO 60
1800	CALL ERROR
1900	RETURN
2000	LAST = LAST + 1
2100	GO TO 60
2200	KARD(LAST) = KBL
2300	KMAX = KMAX - 2
2400	DO 170 I=KOL, KMAX
2500	KARD(I+1) = KARD(I+3)
2600	CONTINUE
2700	IF (BR .EQ. 1) RETURN 1
2800	RETURN
2900	END

C	SUBROUTINE UPDATE		100:
C	THIS SUBROUTINE UPDATES THE SYMBOL TABLE. NEW SYMBOLS ARE ADDED		200:
C	AS THEY ARE FOUND. REFERENCES TO PREVIOUSLY FOUND SYMBOLS ARE		300:
C	UPDATED.		400:
C			500:
			600:
			700:
			800:
			900:
			1000:
			1100:
			1200:
			1300:
			1400:
			1500:
			1600:
			1700:
			1800:
			1900:
			2000:
			2100:
			2200:
			2300:
			2400:
			2500:
			2600:
			2700:
			2800:
			2900:
			3000:
			3100:
			3200:
			3300:
			3400:
			3500:
			3600:
			3700:
			3800:
			3900:
			4000:
			4100:
			4200:
			4300:
			4400:
			4500:
			4600:
			4700:
			4800:
			4900:
			5000:
			5100:
			5200:
			5300:
			5400:
			5500:

30	GO TO 50	5600
40	FORCEM = .TRUE.	5700
50	IF (KRSX .EQ. KSPK(3)) FORCES = .TRUE.	5800
	CONTINUE	5900
C		6000
	IS THIS A NEW NAME	6100
	IF (T1 .NE. 0) GO TO 100	6200
	ILOC = ILOC+1	6300
	IF (ILOC .GT. NST) GO TO 500	6400
	ILOC = ILOC	6500
	T1 = LOC	6600
	HTAB(IADD+1) = TEMP	6700
	GO TO 270	6800
C		6900
100	IF (NAME(1) .NE. SYMTAB(1,T1)) GO TO 200	7000
120	IF (NAME(2) .NE. SYMTAB(2,T1)) GO TO 200	7100
	IF (FORCEM) JLOC(T1) = ISIGN(KEY, JLOC(T1))	7200
	IF (FORCES) JLOC(T1) = -MAX0(1, ABS(JLOC(T1)))	7300
	ILOC = JLOC(T1)	7400
150	RETURN	7500
C		7600
200	DO WE START A CHAIN	7700
	IF (T2 .NE. 0) GO TO 300	7800
	ICOL = ICOL+1	7900
	IF (ICOL .GT. 400) GO TO 500	8000
	T2 = ICOL	8100
	HTAB(IADD+1) = TEMP	8200
	ILOC = ILOC+1	8300
250	IF (ILOC .GT. NST) GO TO 500	8400
	LOC = ILOC	8500
	T1 = LOC	8600
	T2 = 0	8700
	CTAB(ICOL) = TEMP	8800
270	SYMTAB(1,LOC) = NAME(1)	8900
	SYMTAB(2,LOC) = NAME(2)	9000
	IF (FORCES) KEY = -KEY	9100
290	JLOC(T1) = KEY	9200
	LOC = KEY	9300
	RETURN	9400
C		9500
300	SEARCH IN CHAIN	9600
	ITEM = T2	9700
	TEMP = CTAB(ITEM)	9800
	IF (NAME(1) .NE. SYMTAB(1,T1)) GO TO 400	9900
	IF (NAME(2) .EQ. SYMTAB(2,T1)) GO TO 120	10000
C		10100
400	HAVE WE FOUND THE NAME	10200
	DO WE ADD TO THE CHAIN	10300
	IF (T2 .NE. 0) GO TO 300	10400
	ICOL = ICOL+1	10500
	IF (ICOL .GT. 400) GO TO 500	10600
	T2 = ICOL	10700
	CTAB(ITEM) = TEMP	10800
	ILOC = ILOC+1	10900
	IF (ILOC .LE. NST) GO TO 250	11000
C		
500	OVERFLOW	
501	WRITE(6,501)	
	FORMAT(16#HASH TABLE FULL)	
	STOP	

INITIALIZE OR DISPLAY

11100
11200
11300
11400
11500
11600
11700
11800
11900
12000
12100
12200
12300
12400

```
ENTRY HASHIN
CALL SORT( 0 )
OUPDAT = .FALSE.
KSBO = .TRUE.
JSYM = 0
DO 600 I=1,4191
  HTAB(I) = 0
DO 620 I=1,400
  CTAB(I) = 0
  ILOC = 0
  ICOL = 0
RETURN
END
```

600

620

IC

C-----	11	SUBROUTINE XTRNAL	100
		COMMON I,TYPE, JOB(66),KARD(1320),KBUFF(80),KLEAR(8),KMAX,KODE,KOL	200
		----- TEST SUBPROGRAMS IN AN EXTERNAL STATEMENT FOR VALIDITY IN	300
		8800 FORTRAN.	400
		KSAVE = KOL	500
		CALL SYMBOL	600
		IF (KODE .LE. 0) RETURN	700
		K = KOL	800
		KOL = KSAVE	900
		CALL FNCHK (-1)	1000
		KOL = K	1100
		GO TO 11	1200
		END	1300

ALABEL	L	MACRO	SET	TITLE	SUBROUTINE	ALABEL	
ALABEL	L	SAVE	(14,12),*				100
		LA	0,ALABEL-10	LOCAL DISPLACEMENT ----> GR60			200
		SR	15,0	POINT GR 15 BACK TO THE CSECT			300
		LR	10,13	USERS SAVE AREA ADDRESS ----> TEMP STORAGE			400
		LR	13,15	GLOBAL ADDRESSABILITY IN GR 13			500
		ST	13,0(10)	ESTABLISH TRACE BACK CHAIN			600
		ST	10,4(13)	ESTABLISH VALID BACK CHAIN			700
		MEND					800
		SPACE	3				900
IO		START	DS				1000
		DS	10F				1100
		ENTRY	READ,WRITE,TREAD,TWRITE,OPEN,CLOSE,TRANCE				1200
		ENTRY	ICHAR				1300
		ENTRY	TRNSL8,MOVTMP,SETTMP				1400
		SPACE	3				1500
		ENTRY	L4CV6029				1600
		USING	*,15				1700
L4CV6	0	29	LR	0,1			1800
		LA	1,255				1900
LOOP2	9	BCI	1,LOOP29				2000
		LR	1,0				2100
		L	15,=V(MAIN)				2200
		BR	15				2300
		SPACE	3				2400
		USING	*,15				2500
SETTMP	P	LA	1,TEMP				2600
		ST	1,DCBAD				2700
		BR	14				2800
		DROP	15				2900
		USING	10,13				3000
OPEN		SET					3100
		OPEN	(INPUT,,OUTPUT,(OUTPUT),TEMP,(OUTPUT))				3200
		LA	15,0				3300
		TM	INPUT+48,X'10'				3400
		BZ	OPNERR				3500
		TM	OUTPUT+48,X'10'				3600
		BZ	OPNERR				3700
		TM	TEMP+48,X'10'				3800
		BZ	OPNERR				3900
		LA	1,OUTPUT				4000
		ST	1,DCBAD				4100
ROPN		L	13,4(13)				4200
		RETURN	(14,12),T,RC=(15)				4300
OPNER	R	LA	15,4				4400
		B	ROPN				4500
CLOSE		SET					4600
		CLOSE	(INPUT,,OUTPUT,,TEMP,)				4700
		LA	15,0				4800
		B	ROPN				4900
READ		SET					5000
		L	11,0(1)				5100
							5200
							5300
							5400
							5500

	GET	INPUT.(11)	5800
	TR	0(80,11),INTAB	5700
	LA	1,80	5600
	LA	2,316(11)	5500
	LA	3,79(11)	5400
	L	5,=CL4	5300
	L	6,=F'4	5200
A1TOA 4	IC	4,0(13)	5100
	SRDL	4,8	5000
	ST	5,0(2)	4900
	BCTR	3,0	4800
	SR	2,6	4700
	SLL	5,8	4600
	BCT	1,A1TOA4	4500
	LA	15,0	4400
	B	ROPN	4300
WRITE	SET	11,0(1)	4200
	L	OUTPUT.(11)	4100
	PUT	15,0	4000
	LA	ROPN	3900
	B		3800
TREAD	SET	11,0(1)	3700
	L	TEMP.(11)	3600
	GET	15,0	3500
	LA	ROPN	3400
	B		3300
TWRITE	SET	11,0(1)	3200
	L	TEMP.(11)	3100
	PUT	15,0	3000
	LA	ROPN	2900
	B		2800
TRANCE	SET	2,3,0(1)	2700
	LM	12,BUF	2600
	LA	9,0(3)	2500
	L	9,9	2400
	LTR	NINEOK	2300
	BP	9,72	2200
NINEOK	LA	11,0(12)	2100
	LA	7,72	2000
	XR	3,3	1900
	AR	3,9	1800
LOOP	STC	3,MBL+3	1700
	LR	1,9	1600
	SR	9,7	1500
	BM	A4TOA1	1400
	LR	1,7	1300
A4TOA 1	IC	3,0(12)	1200
	STC	3,0(11)	1100
	LA	11,1(11)	1000
	LA	2,4(2)	900
	BCT	1,A4TOA1	800
	MVC	72(8,12),BLANKS	700
	LTR	9,9	600
	BNN	TRPUT	500

GET ARGUMENTS

GET NUMBER OF CHARACTERS TO BE OUTPUT
NUMBER MUST BE POSITIVE.
IF NOT,

SET IT TO OUTPUT 1 CARD IMAGE
ADDRESS OF AREA BEING PACKED
NUMBER OF COLUMNS TO BE PACKED (MAX)

IF NOT FULL CARD, LOCATION TO BEGIN BLANK
NUMBER OF CHARACTERS REMAINING TO BE PACKED
TEST FOR FULL CARD
BRANCH IF NOT FULL CARD
ELSE SET NUMBER OF COLUMNS TO PACK

TEST TO SEE IF BLANKS NEEDED AT END OF ST

TRPUT:	LPR	1,9	NUMBER OF BLANKS NEEDED	11100
	EX	1,DCBAD	BLANK OUT RIGHTMOST COLS OF CARD	11200
	PUT	(1), (12)		11300
	LTR	9,9		11400
	BMP	ROPN	TEST FOR TASK COMPLETE	11500
	MVC	010, 12), CONT	ALL DONE UNLESS R9 > 0	11600
	LA	11, 0(12)	SET UP FOR CONTINUATION CARDS	11700
	LA	7, 66	66 COLUMNS IN A CONTINUATION CARD	11800
	LA	3, 6		11900
	B	LOOP		12000
	MVC	010, 12), BLANKS		12100
BLANK:	DC	15C	DON'T CHANGE ORDER OF THESE TWO CARDS	12200
CONT:	DC	C	DON'T CHANGE ORDER OF THESE TWO CARDS	12300
ICHAR:	SET			12400
	LM	2, 3, 0(1)		12500
	L	3, 0(3)		12600
	SR	4, 4		12700
	IC	4, 0(2)		12800
	IC	4, CHAR(4)		12900
	STH	4, 2(2)		13000
	LA	2, 4(2)		13100
	BCT	3, NXT		13200
	LA	15, 0		13300
	B	ROPN		13400
	MOVTM:	P		13500
	SET	11, 0(1)		13600
	L	CLOSE (TEMP, DISP)		13700
	OPEN	(TEMP, (INPUT))		13800
	TM	TEMP+48, X'10		13900
	BZ	OPNERR		14000
	OP	GET TEMP, (11)		14100
	PUT	OUTPUT, (11)		14200
	B	COPYLOOP		14300
	NE	LA 1, OUTPUT		14400
	ST	1, DCBAD		14500
	CLOSE	(TEMP, DISP)		14600
	OPEN	(TEMP, (OUTPUT))		14700
	TM	TEMP+48, X'10		14800
	BZ	OPNERR		14900
	LA	15, 0		15000
	B	ROPN		15100
	TRNSL:	S		15200
	SET	11, 0(1)		15300
	L	TR 0180, 11), TAB		15400
	PUT	OUTPUT, (11)		15500
	LA	15, 0		15600
	B	ROPN		15700
	EJECT			15800
	DC	256X'5D'	UP ARROW	15900
	EQU	*		16000
	ORG	TAB+C'A'		16100
	DC	C'ABCDEFghi'		16200
	ORG	TAB+C'J'		16300
	DC	C'JKLMNopqr'		16400
				16500

	ORG	TAB+C'S	16600
	DC	C'STUVWXYZ	16700
	ORG	TAB+C'O	16800
	DC	C'O123456789	16900
	ORG	TAB+C'	17000
*	ORG	(+ ; a 360 CHAR	17100
	DC	X'4BC06C505D50	17200
	ORG	TAB+C'	17300
	DC	C'	17400
	ORG	TAB+X'5A	17500
*	DC	! \$ # % - / 360 CHAR	17600
	DC	X'5E5B5C4C8B4E8061	17700
	ORG	TAB+X'6B	17800
*	DC	X'6B5D5D5F5D 360 CHAR	17900
	ORG	TAB+X'7A	18000
*	DC	: ; @ - = 360 CHAR	18100
	DC	X'7AD06D7C7B6F	18200
	ORG	ENDTAB	18300
	SPACE 3		18400
	DC	256AL1(49)	18500
CHAR	CHAR		18600
CHAR	CHAR		18700
	ORG	CHAR+C'A	18800
*	DC	A B C D E F G H I	18900
	DC	X'010203040506070809	19000
	ORG	CHAR+C'J	19100
*	DC	J K L M N O P Q R	19200
	DC	X'0A0B0C0D0E0F101112	19300
	ORG	CHAR+C'S	19400
*	DC	S T U V W X Y Z	19500
	DC	X'131415161718191A	19600
	ORG	CHAR+C'O	19700
*	DC	0 1 2 3 4 5 6 7 8 9	19800
	DC	X'1B1C1D1E1F2021222324	19900
	ORG	CHAR+C'E	20000
	DC	AL1(37)	20100
	ORG	CHAR+C'	20200
	DC	AL1(38)	20300
	ORG	CHAR+C'('	20400
	DC	AL1(39)	20500
	ORG	CHAR+C'/'	20600
	DC	AL1(40)	20700
	ORG	CHAR+C')'	20800
	DC	AL1(41)	20900
	ORG	CHAR+C'+	21000
	DC	AL1(42)	21100
	DC	AL1(49)	21200
	DC	AL1(42)	21300
	ORG	CHAR+C'-'	21400
	DC	AL1(43)	21500
	ORG	CHAR+C'*	21600
	DC	AL1(44)	21700
	ORG	CHAR+C'.'	21800
	DC	AL1(45)	21900
	ORG	CHAR+C'S	22000

	DC	AL1(46)		22100
	ORG	CHAR+X'7D'		22200
	DC	AL1(47)		22300
	ORG	CHAR+C'		22400
	DC	AL1(48)		22500
	ORG	CHAREND		22600
	SPACE 3			22700
INTAB	DC	256X'SD'		22800
INTEN	EQU	#		22900
	ORG	INTAB+C'A'		23000
	DC	C'ABCDEFHI'		23100
	ORG	INTAB+C'J'		23200
	DC	C'JKLMNOPQR'		23300
	ORG	INTAB+C'S'		23400
	DC	C'STUVWXYZ'		23500
	ORG	INTAB+C'O'		23600
	DC	C'0123456789'		23700
	ORG	INTAB+C'		23800
	)	(+ ; A		23900
	DC	X'4B5D4D4E4F50'		24000
	ORG	INTAB+X'5A'		24100
	!	\$ #) : @ - /		24200
	DC	X'5A5B5C5D5E5F6061'		24300
	ORG	INTAB+X'6B'		24400
	!	> ?		24500
	DC	X'6B4D6D6E6F'		24600
	ORG	INTAB+X'7A'		24700
	:	= "		24800
	DC	X'7A7E7D7D7E7F'		24900
	ORG	INTAB+X'40'		25000
	DC	C'		25100
	ORG	INTEND		25200
	SPACE 3			25300
DCBAD	DS	1F		25400
BUF	DS	20F		25500
	EJECT			25600
INPUT	DCB	DDNAME=INPUT,DSORG=PS,MACRF=(GM),LRECL=80,RECFM=FB,	X	25700
	DEV=DA,EODAD=OPNERR			25800
OUTPUT	T	DCB DDNAME=OUTPUT,DSORG=PS,MACRF=(PM),LRECL=80,RECFM=FB,	X	25900
	DEV=DA			26000
TEMP	DCB	DDNAME=TEMP,DSORG=PS,MACRF=(GM,PM),LRECL=80,RECFM=FB,	X	26100
	DEV=DA,EODAD=COPYDONE			26200
	END			26300

ROUTINE NAME	ROUTINE TYPE	LOCATION
MAIN	PROGRAM	SCOFF.TMP
BLOCK DATA	BLOCK DATA	SCOFF.TMP
CENTOR	SUBROUTINE	SCOFF.TMP
ERREND	SUBROUTINE	SCOFF.TMP
ERROR	SUBROUTINE	SCOFF.TMP
ERR	SUBROUTINE	SCOFF.TMP
FNCHK	SUBROUTINE	SCOFF.TMP
FORTRN	SUBROUTINE	SCOFF.TMP
IMPSET	SUBROUTINE	SCOFF.TMP
INPUT	SUBROUTINE	SCOFF.TMP
INSRTR	SUBROUTINE	SCOFF.TMP
MOVER	SUBROUTINE	SCOFF.TMP
MULTEQ	SUBROUTINE	SCOFF.TMP
PAG	SUBROUTINE	SCOFF.TMP
RETRN1	SUBROUTINE	SCOFF.TMP
SIXFR7	SUBROUTINE	SCOFF.TMP
SUB6	SUBROUTINE	SCOFF.TMP
SLASHS	SUBROUTINE	SCOFF.TMP
SORT	SUBROUTINE	SCOFF.TMP
STARL	SUBROUTINE	SCOFF.TMP
SUBSUB	SUBROUTINE	SCOFF.TMP
SYMBOL	SUBROUTINE	SCOFF.TMP
TDTEST	SUBROUTINE	SCOFF.TMP
UPDATE	SUBROUTINE	SCOFF.TMP
XTRNAL	ENTRY POINT	UPDATE
MAIN	SUBROUTINE	SCOFF.TMP
	PROGRAM	SCOFF.TMP
	ENTRY POINT	MAIN
	ENTRY POINT	MAIN
	ENTRY POINT	MAIN
	ENTRY POINT	MAIN

ROUTINE NAME	ROUTINE TYPE	LOCATION
CENTOR	ENTRY POINT	UPDATE
ERR	ENTRY POINT	MAIN
ERREND	ENTRY POINT	MAIN
ERROR	ENTRY POINT	MAIN
FNCHEK	ENTRY POINT	MAIN
FORTRN	SUBROUTINE	SCOFF.TMP
IMPSET	SUBROUTINE	SCOFF.TMP
INPUT	SUBROUTINE	SCOFF.TMP
INSRTR	SUBROUTINE	SCOFF.TMP
MAIN	SUBROUTINE	SCOFF.TMP
MAIN	PROGRAM	SCOFF.TMP
MOVER	PROGRAM	SCOFF.TMP
MULTEQ	SUBROUTINE	SCOFF.TMP
PAG	SUBROUTINE	SCOFF.TMP
RETRN1	SUBROUTINE	SCOFF.TMP
SLAFR7	SUBROUTINE	SCOFF.TMP
SLASHS	SUBROUTINE	SCOFF.TMP
SORT	SUBROUTINE	SCOFF.TMP
STARL	SUBROUTINE	SCOFF.TMP
SUB6	SUBROUTINE	SCOFF.TMP
SUBSUB	SUBROUTINE	SCOFF.TMP
SYMBOL	SUBROUTINE	SCOFF.TMP
TDTEST	SUBROUTINE	SCOFF.TMP
UPDATE	SUBROUTINE	SCOFF.TMP
XTRNAL	SUBROUTINE	SCOFF.TMP
BLOCK DATA	BLOCK DATA	SCOFF.TMP

DISTRIBUTION LIST

DEPARTMENT OF DEFENSE

Armed Forces Radiobiology Rsch Institute
2 cy ATTN: Dir

Armed Forces Staff College
ATTN: Library

Assistant to the Secretary of Defense
Atomic Energy
ATTN: Executive Assistant

Defense Communications Engineer Ctr
ATTN: R-123

Defense Intelligence Agency
2 cy ATTN: RTS-2A

Defense Nuclear Agency
ATTN: RAAE
ATTN: SPAS
ATTN: RAEV
ATTN: NATA
ATTN: RAEE
ATTN: SPSS
ATTN: SPTD
ATTN: NATD
ATTN: NATA, J. Anderson
ATTN: COMP-1, B. Robertson
ATTN: STTI, M. Vincent
2 cy ATTN: STTI
4 cy ATTN: TTTL

Defense Tech Info Ctr
12 cy ATTN: DD

Field Command Defense Nuclear Agency
Det 1
Lawrence Livermore Lab
ATTN: FC-1, J. Crandley
ATTN: FC-1

Field Command
Defense Nuclear Agency
ATTN: FCTXE
ATTN: FCPR
ATTN: FCTT, S. Humphries
ATTN: FCTT, W. Summa
ATTN: FCTT, G. Ganong
ATTN: FCTT

Joint Chiefs of Staff
ATTN: SAGA/SFD
ATTN: SAGA/STA
ATTN: J-5, Nuclear Div/Strategy Div
ATTN: J-3

Joint Strat Tgt Planning Staff
ATTN: JLA, Threat Applications Div
2 cy ATTN: NRI-STINFO, Library

Joint Strategic Connectivity Staff
ATTN: JCS/DNA, F. Tedesco

Under Secretary of Defense for Rsch & Engrg
ATTN: Strategic & Space Sys (OS)

DEPARTMENT OF THE ARMY

Aberdeen Proving Ground
2 cy ATTN: Tech Library

Atmospheric Sciences Lab
ATTN: DELAS-AS

BMD Advanced Technology Ctr
ATTN: ATC-T

Deputy Chief of staff for Ops & Plans
ATTN: DAMO-NCN

Deputy Chief Staff for Rsch Dev & Acq
ATTN: DAMA-CSS-N

Engr Studies Ctr
ATTN: DAEN-FES

Harry Diamond Labs
ATTN: DELHD-NW-P, 20240
ATTN: DELHD-TA-L, 81100
ATTN: DELHD-NP

Multi Service Communications Systems
ATTN: DRCPM-MSCS-APB, M. Francis

US Army Ballistic Rsch Labs
ATTN: R. Reisler
2 cy ATTN: DRDAR-TSB-S

US Army Chemical School
ATTN: ATZN-CM-CC

US Army Comb Arms Combat Dev Acty
ATTN: ATZL-CAN-I

US Army Comd & General Staff College
ATTN: ACQ, Library Div

US Army Communications Command
ATTN: Tech Ref Div

US Army Concepts Analysis Agency
ATTN: Classified Documents

US Army Engr Waterway Exper Station
ATTN: J. Houston
ATTN: WESGH, P. Hadala

US Army Field Artillery School
ATTN: M. Swett, Library

US Army Forces Command
ATTN: Tech Library

US Army Intell & Sec Cmd
ATTN: Tech Library

US Army Logistics Ctr
ATTN: ATCL-OSS, S. Cockrell

US Army Material Dev & Readiness Cmd
ATTN: DRCDE-D

DEPARTMENT OF THE ARMY (Continued)

US Army Materiel Sys Analysis Actvy
ATTN: DRXS-CC

US Army Nuclear & Chemical Agency
ATTN: Library

US Army Satellite Comm Agency
ATTN: Tech Library

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

US Army TRADOC Sys Analysis Actvy
ATTN: ATAA-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 Cmd Control Planning & Prgmng 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 Cmd
ATTN: Tech Library

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

Naval Electronic Systems Cmd
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, FKA6A2, 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: AFATL/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: NTYC, 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: ENMP, D. Van Gari

Deputy Chief of Staff
Rsch, Dev, & Acq
ATTN: AFRDST
ATTN: AFRDS, Space Sys & C3 Dir
ATTN: AFRDQI

Headquarters
Electronic Systems Div
ATTN: DCK, L. Staples
ATTN: XRRT

DEPARTMENT OF THE AIR FORCE (Continued)

Foreign Tech Div
ATTN: NIIS, Library

Headquarters
NORAD
ATTN: J5YS, F. Smith

Space Div
ATTN: IND
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. Bensen

DEPARTMENT OF ENERGY CONTRACTORS

University of California
Lawrence Livermore National Lab
ATTN: L-21, D. Oakley
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

Applied Rsch Associates, Inc.
ATTN: J. Phillips

Applied Theory, Inc.
ATTN: Doc Con

APTEK, Inc.
ATTN: A. Larrabee

Astron Rsch & Engrg
ATTN: A. Levien

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

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. Percy
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/S 42-37, B. Henderson

Boeing Computer Svcs, Inc.
ATTN: M/S 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

Dynallectron
ATTN: C. Scott

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: A. Stough
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. Coceano
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. Hattiwanger 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. Brelsford

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: DASIAC
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. Rankin
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: M. 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. Sanemitter
ATTN: J. Levsque
ATTN: S. Finn
ATTN: Contracts Ofc
ATTN: H. Brode, Chairman SAGE
ATTN: Doc Con

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

Pacific 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. Larson
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. Warshaw
10 cy ATTN: Tech Info Ctr

R & D Associates
2 Cy ATTN: Doc Con

R. 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. Goplen

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

Science Applications, Inc
ATTN: R. McClean

Science Applications, Inc
ATTN: H. Dolatshahi
ATTN: R. Jarka

AD-A131 273

AFWL BENCHMARK COMPUTER PROGRAM(U) BOEING AEROSPACE CO
SEATTLE WA B J HENDERSON 01 MAY 82 DNA-5852F
DNA001-80-C-0056

2/2

UNCLASSIFIED

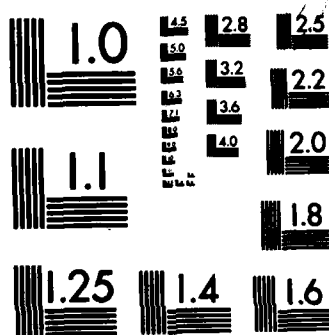
F/G 9/2

NL

END

POWER

REG



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