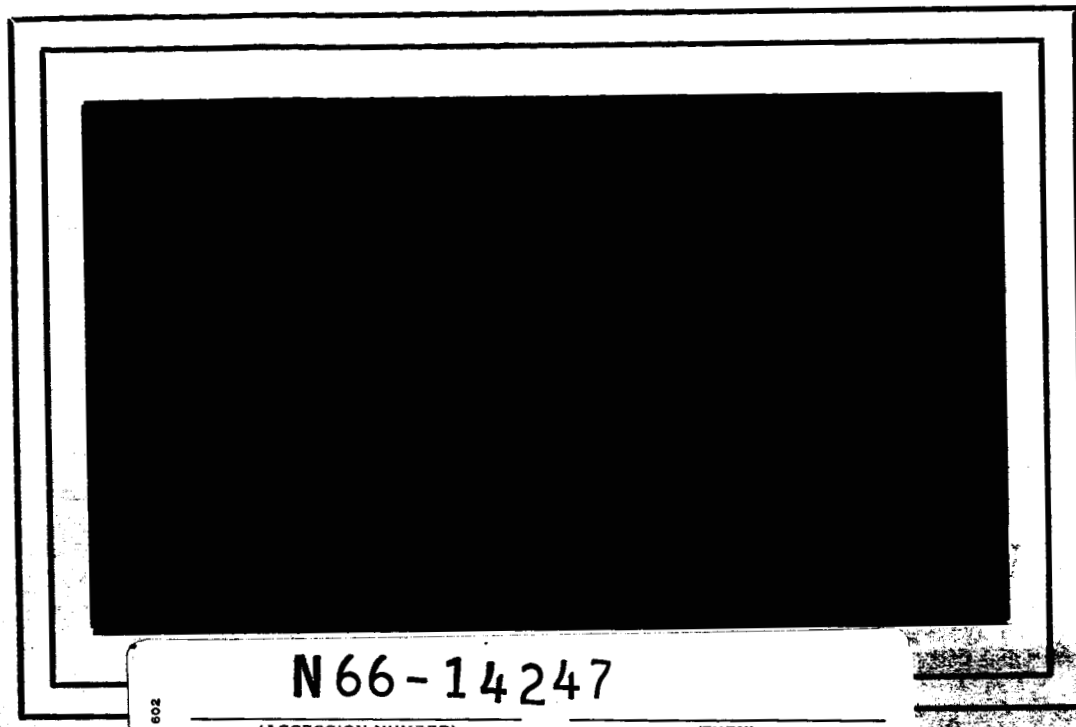

UNIVERSITY OF MARYLAND
COMPUTER SCIENCE CENTER

COLLEGE PARK, MARYLAND

Technical Report TR-65-25
NsG-398 November 1965

FORTRAN II to FORTRAN IV Translator
UOM SFT
for the
IBM 7090/7094*

by

Donald E. Eastlake III
Data Processing Systems Analyst
Computer Science Center

*All work on this program was performed at the Computer Science Center of the University of Maryland and was supported by NsG-398 of the National Aeronautics and Space Administration.

ABSTRACT

4247

This report describes a program (UOM SFT) written in SNOBOL for the IBM 7090/7094 for the translation of FORTRAN II into FORTRAN IV. The use of the basic translator and also of its many special features is described in detail. A number of examples such as deck setup, control card format, and one complete sample FORTRAN II deck and its translation are included. UOM SFT is also compared with SHARE SFT (SD 3054), a translator written in machine language.

INTRODUCTION

Many programs are or will be changed from FORTRAN II to FORTRAN IV. UOM SFT was written to aid in this change despite the existence of a SHARE SFT (SD 3054) because a need was felt for greater flexibility in translation and in ease of modifying the translators.

Although, with all its special features in use, UOM SFT is about half as fast as SHARE SFT (SD 3054), it is interesting to note that it offers more services of use to the programmer in a source deck two-thirds the size of the FMS type binary deck for SHARE SFT (SD 3054). Both this and its ease of modification are due to the fact that UOM SFT is written in SNOBOL, an interpretive character handling language.

The remainder of this report which gives detailed instructions on using UOM SFT is written in the standard SHARE format.

Identification

A. 7090/94 FORTRAN II to FORTRAN IV translator UØM SFT

B. Donald E. Eastlake III, October 8, 1965.

C. Computer Science Center, University of Maryland, College Park, Maryland.

Purpose

To translate FORTRAN II source decks into FORTRAN IV source decks, and to allow certain useful modifications during translation.

Restrictions

For a deck to be successfully translated, it must obey the following three major restrictions:

1) Comment cards are not allowed between continuation cards.
2) Use of obscure indexing on arrays is very unlikely to translate successfully. The translator does not attempt to compensate for the fact that 'EQUIVALENCE' statements do not reorder COMMON under FORTRAN IV although it may create 'EQUIVALENCE' statements in certain cases where a variable has been dimensioned zero. Under FORTRAN IV, an array may only be used with as many indices as it is dimensioned to have. Double precision and complex variables in an input program must be declared in 'DIMENSION' statements with the appropriate letters in column one and used only as double precision or complex entities.
3) There is a limit to the size of each program to be translated. In actuality the limit is on the amount

of information it is necessary for the translator to retain. If neither the statement number translation or symbol usage statistics features are used, almost any size program could be translated. If the symbol usage statistics but not the statement number translation features are used (the initial mode) the maximum successful program size corresponds approximately to the maximum size that FORTRAN IV will compile. With both of these features in use there is a greater restriction in program size (to perhaps 300 executable statements).

Method

UOM SFT is written in SNOBOL, a high level, interpretive, character string handling language which runs under the MAMOS subsystem under IBSYS. It produces punched cards which constitute a FORTRAN IV deck or decks that are supposed to perform the same actions as the input programs. Also a listing of the translation and/or the input is produced (see M-control-field under SFT CONTROL CARDS below).

While the use of SNOBOL for this translator may cause some loss in speed and capacity this is partially compensated for by the ease in making and debugging modifications for particular translation problems as they arise.

Usage

The user should precede the SFT deck by a '$EXECUTE MAMOS' card and whatever cards are required at the installation he is using (such as '$ID' or '$JOB' cards. See example below). The SFT deck should be followed by a FORTRAN II deck or decks and SFT control cards appropriate to the output wanted by the

user. The last input card should be a control card. Control cards are described in the following section.

For the basic purpose of making it easier for the user to perform certain types of optimization of his program, the translator will produce, for each input deck, a list of all symbols used and three columns of numbers. The first column to the right of the list of symbols is how many times each is set by an arithmetic statement, input statement, 'DO' statement or 'ASSIGN' statement. The second column is how many times each variable's value is used in an arithmetic expression, output statement, subscript, 'DO' statement, argument to subroutine or function, or used as a subroutine name. The third column is how many times each appears in specification statements such as 'DIMENSION', 'EQUIVALENCE', etc., or as a dummy symbol in a 'SUBROUTINE' or 'FUNCTION' statement. Thus one could determine if a variable was being set but not used, or used but not set. This feature, initially turned on, can be eliminated by control cards (see J-control-field under SFT CONTROL CARDS below).

If octal or Hollerith constants are used in the input program, they will be placed in a created array by a 'DATA' statement which, along with a 'REAL' statement to declare the array, is produced after the 'END' statement in the translation. They must be moved to near the beginning of the deck.

If complex Boolean statements are used in the source deck then a MAP subroutine, supplied with the translator, called 'BINSIM' will be called by the output FORTRAN IV program.

Although certain types of garbage will make the translator loop or cause other errors, it tries to be transparent to erroneous cards. They are marked in the listing and punched out exactly as read.

No line of listing produced by SFT is longer than 120

characters including carriage control.

It would be fairly easy to adapt SFT to other versions of SNOBOL.

Example of deck makeup:

```
$EXECUTE        MAMOS
$ID    JOE DOE*987/65/432*10M*100C*40P$
┌──────────────┐
│ UOM SFT DECK │
└──────────────┘
· control card
· control card
┌──────────────┐
│ FORTRAN II DECK │
└──────────────┘
· control card
┌──────────────┐
│ FORTRAN II DECK │
└──────────────┘
┌──────────────┐
│ FORTRAN II DECK │
└──────────────┘
etc.
· control card (last card)
```

UOM SFT CONTROL CARDS

In order to control the translation process and special features of SFT, control cards are used that may be inserted anywhere in the decks to be translated except between continuation cards. These control cards are distinguished by a period in card column one. The rest of the card (card columns 2 through 72) is scanned for fields enclosed in parentheses. All blanks and all characters not enclosed in parentheses are ignored (a string of blanks is the same as a null string). Each control card read causes a one page printout of various control information as it stands after the card is processed. the first non-blank character of a parenthetic control field. which is followed by arguments in some cases, determines the type of control action to be taken. All valid first characters and their effects with appropriate arguments are listed below, in alphabetic order, after a few sample control cards.
Examples of control cards:

```
.      (M2)   (N99,1)     (IABCDEF,5)     (DZORO)
.        (END)  (M1)      (N)    (IXYZ,20)
.    (Q(5,7) (7,5))  (J)  (I)   (N)   (M3)
.              (X3)            (EMPTY QTAB)
```

D-control-field

From 1 to 5 non-blank characters following the 'D' become the name of the dummy array into which all Hollerith and octal constants are put. It is initially 'MQZK'. This symbol with an 'X' concatenated on the end is also used in the translation of certain types of non-arithmetic FORTRAN II 'IF' statements.
Examples:

```
(DFOO)
(DDUMMY)
```

E-control-field

The Q-table is emptied and the end counter is set to zero. Characters after the 'E' are ignored. (see Q-control-field)
Examples:

```
(E          )
(     E     N     D)
(EMPTY QTB)
```

I-control-field

The 'I' may be followed by one argument or by two arguments separated by a comma. If the first argument is not null, the translator will change the identification (card columns 73 through 80) of translated instructions to a series of eight character sequences the first one to seven characters of which are the first argument and the rest of which are numeric and, starting at zero, are increased by the value of the second argument which must be numeric. If the second argument is omitted this increment will remain at the last previously set value. It is initially zero. At the start of execution and whenever the first argument is null, the translator is put in a mode where the original identification of the first card of each statement is used for all lines of output produced by that instruction.
Examples:

```
(IMN3A, 10)
(IBOND)
( I )
```

J-control-field

Initially, or when the character string after the 'J' is not null, a list of all symbols used in a translated program

and their relative frequency in different contexts is produced (see USAGE above for more details). If the 'J' is followed by a null string, this feature is turned off.

Examples:

```
(J)
( J RESTART)
```

M-control-field

One argument follows the 'M' and the translators output mode is set to it. If the mode is set to a numeric one ('1'), which it is initially, the listing of the translation is interspersed with a listing of the input. If the mode is set to a numeric two ('2'), the listing of the translation is followed by a listing of the input. If the mode is anything else, only a listing of the translation is produced.

Examples:

```
(M 1)
(MTURN OFF THE PRINTING MACHINE)
```

N-control-field

One argument or two arguments separated by a comma may follow the 'N'. If the first argument is null or if no 'N' control field has been read, the statement numbers in the original program are left unchanged. Otherwise they are translated, in the order seen, to new numbers starting at the first argument plus the second and increasing by the second. If the second argument is left out, the incrementing will be unchanged. It is initially zero. When statement number translation is used a table of correspondence between the new and old numbers is printed when an 'END' statement is encountered and also the maximum translatable program size is reduced in

proportion to the number of different statement numbers encountered.

Examples:

(N490,10)

(N9990)

(N)

Q-control-field

A list of pairs of arguments should follow the 'Q', each pair separated by commas and surrounded by parentheses. During translation, if one of the first elements of a pair is found in the tape number position of an input or output statement it will be replaced by the second element of that pair. The pairs following all 'Q's encountered are appended to a table called the Q-table. (see E-control-field)

Examples:

(Q(2,3) (8,2) (4,1) (12,8) (3,4))

(Q(INTAPE,5) (6,ZTAPE) (ATAPE,BTAPE))

(Q (5, 7) (7, 5))

X-control-field

The main use of this control field is in debugging modifications to the translator. If its argument is numeric and divisible by 2, 3, 5, or any combination, the SNOBOL functions TDUMP, DUMP, VDUMP, or a corresponding combination are called.

Examples:

(X3)

(X 3 0)

APPENDIX A

This appendix includes a sample FORTRAN II program with UOM SFT control cards in part one and its printed translation output in part two. This made up program illustrates two translation problems that SFT cannot handle. The first is the peculiar mixing of double precision and complex variables. The second is that the comments printed out by this simple program concerning timing will not generally be true because sense switch testing is done by subroutine under FORTRAN IV. Although the first of these could probably be solved by suitable expansion of the translation, this is not so clear of problems of the second type.

APPENDIX A.1

Listing of sample FORTRAN II deck to be translated, with UOM SFT control cards.

```
c (M3) (JON) (N90,10) (Q(INTAPE,5)(7,6)(6,7)) (IEXAMP,5) (DDUMMY)
*     LIST8
*     LABEL
C
C      THIS IS A SAMPLE OF SFT INPUT AND OUTPUT.
C      NOTE THE MANY BELLS AND WHISTLES USED ON THE ABOVE CONTROL CARD.
C
      SUBROUTINE EXAMPL (A, XX, IBBB, YYYY, ICCCCC, ZZZZZZ)
C
C      ARITHMETIC STATMENT FUNCTIONS
      FOOF(Q) = SINF(Q)**2+COSF(Q)**2
B     XORF(Q,U) = Q*(-U)+U*(-Q)
C
F     SUB1, SUB2, ROUSPC
C
      DIMENSION II(10), JJ(0), ZAP(20)
I     DIMENSION A(IBBB,ICCCCC), XX(1000)
D     DIMENSION YYYY(IBBB,ICCCCC,2),KK(0),ZZZZZZ(IBBB,ICCCCC)
      FUN = 3HFUN
      READ INPUT TAPE INTAPE, 4, ZZ1, ZZ2
      INTEGR = ZZ2
      IF (XORF(ZZ1, FUN)) 101, 1111, 101
1111  J =0
      I=0
            PRINT       5
      IF (SENSE SWITCH 5) 1, 2
      THIS IS A DELIBERATLY ERRONEOUS STATMENT 12345679912345GN7891234567891234 5
     1    CONTINUATION OF ERRONEOUS CARD                                      FOO
 1    IF (SENSE SWITCH 5) 11, 3
 11    I = I+1
      IF (I-32767) 1,111,69
 111 I=0
      J=J+1
      GO TO 1
 2    IF (SENSE SWITCH 5) 3, 22
 22   I =I+1
      IF (I-32767) 2,222,69
 222    J=J+1
      I=0
      GO TO 2
 3    Z=FLOATF(J)*0.832+FLOATF(I)*(0.832/32766.0)
      PRINT 6,     Z
      WRITE OUTPUT TAPE 7,6,Z
 101 DO 1010 I = 1, IBBB, INTEGR
      A(I,1) = FLOATF(I)*XX(INTEGR)
      DO 1011 J = 1, ICCCCC, I
1011   ZZZZZZ(I,J) = A(J,1)+XX(J)
      YYYY(I,1,1) = YYYY(I,1,2)+A(I,2)+A(I,3)
1010 CONTINUE
      DO 1020 I = 1, ICCCCC, INTEGR
B     A(1,I) = 77
B     A(2,I) = 1000
B     YYYY(1,I,2) = YYYY(1,I,1)*707070707070+ZZZZZZ(I,I)
1020 CONTINUE
      ZAP(13)
     1=
     213
      DO 1030 I=1, 10
1030   II(I)=JJ(I)
      RETURN
```

```
 69          PRINT7
      WRITE OUTPUT TAPE    7,7
      PAUSE
       RETURN
 7     FORMAT (18H HORRENDOUS ERROR.)
 6    FORMAT (9H YOU TOOK, F10.2,9HSECONDS.)
    5 FORMAT (14H DEAR OPERATOR/15H ARE YOU AWAKE/
     C43H IF SO CHANGE THE STATUS OF SENSE SWITCH 5.)
  4   FORMAT (A3, F8.2)
       END
         (END)
```

APPENDIX A.2

Listing of output produced by translation of deck in appendix A.1.

```
$EXECUTE        MAMOS
```

$ID DONALD E. EASTLAKE 3RD*001/65/003*5M*100C*40P*N$ 27204 11/22/65 ON 20-55-24

$SNOBOL

SNOBOL (07-16-65 VERSION) PROGRAM LISTING ... ...

```
          MODE('ANCHOR')                                                    1
          MODE = '1'                                                        2
          BKLIST = ','                                                      3
          READ = '1'                                                        4
          WRITE = '2'                                                       5
          PS = '++++++++++'                                                 6
          STARS = '**********'                                              7
          DOLLARS = '0000000000'                                            8
        $('+') = '1'                                                        9
        $('-') = '2'                                                       10
        $('*') = '3'                                                       11
          STN = '     '                                                     12
          DUMMY = 'MQZK'                                                    13
          G.O = 'F.READ'                                                    14
          G..O = 'PROG'                                                     15
          READ('TEMPIN','4')                                                16
          PRINT('TMPOUT','4')                                               17
          REWIND('TMPOUT')                                                  18
        DEFINE('DATRAN(ARG)','DATRAN')                                      19
        DEFINE('ARSF(ARG,XCOL)','ARSF','RIAH')                              20
          DEFINE('NPAGE(X)','NPAGEL')                                       21
          DEFINE('SQZ(ARG)','SQZL')                                         22
          DEFINE('BOUT(ARG)','BOUTL')                                       23
          DEFINE('TRN(ARG)','TRNL')                                         24
          DEFINE('IDF(X)','IDLE','FOO')                                     25
          DEFINE('LBEF(ARG)','LBLOC')                                       26
          DEFINE('GLUG(ARG1,ARG2)','GLUGL')                                 27
          DEFINE('QTRN(ARG)','QTRL')                                        28
          DEFINE('SYSPNT(XXX)','SISPINT')                                   29
          DEFINE('BLANK(ARG)','BLNKL')                                      30
          DEFINE('TRNLST(ARG)','TRNLST','F.K')                              31
        DEFINE('PUT(ARG,N)','PUT','I.J,F.K')                                32
        DEFINE('PUT1(ARG)','PUT1','I.J,F.K')                                33
        DEFINE('PUT2(ARG)','PUT2','I.J,F.K')                                34
        DEFINE('PUT3(ARG)','PUT3','I.J,F.K')                                35
        RPTFLG = 'RPUT2'                                                    36
*
PROG                                            /($G.O)                     37
F.READ    SYSPIT *INCARD1*                      /F(END)                     38
          $(.EQ('2',MODE) 'TMPOUT') = INCARD1   /F(RD2)                     39
          TMPREC = TMPREC + '1'                                             40
RD2       INCARD1 *F1COL/'1'* *SNMBR/'4'* *6COL/'1'* *FRTRN/'66'* *ID1*     41
        '.' F1COL                               /S(PERIOD)                  42
          $(.NUM(F1COL) 'SNMBR') = F1COL SNMBR                              43
          G.O = 'R.E.AD'                        /(E.READ)                   44
R.E.AD    INCARD1 = $('INCARD' I.R)                                         45
          F1COL = 1COL                                                      46
          SNMBR = NMBR                                                      47
          ID1 = $('ID' I.R)                     /($C.CARD)                  48
E.READ    I.R = '2'                                                         49
          STATMENT = FRTRN                                                  50
D.READ    SYSPIT *$('INCARD' I.R)*              /F(END)                     51
          $(.EQ('2',MODE) 'TMPOUT') = $('INCARD' I.R) /F(RD3)               52
          TMPREC = TMPREC + '1'                                             53
RD3       $('INCARD' I.R) *1COL/'1'* *NMBR/'4'* *6COL/'1'*                  54
.           *FRTRN/'66'* *$('ID' I.R)*                                      54
          'C$*' ** 1COL                         /S(N.READ)                  55
```

```
                '.' 1COL                                /S(CCARD)                    56
              ' 0' ** 6COL                              /S(N.READ)                   57
                STATMENT = STATMENT FRTRN                                            58
                I.R = I.R + '1'                         /(D.READ)                    59
N.READ          C.CARD = 'E.READ'                                                    60
                $(.NUM(1COL) 'NMBR') = 1COL NMBR        /(PROGPROG)                  61
CCARD           C.CARD = 'CCARD2'                                                    62
PROGPROG ' ' F1COL                                      /S(RPROG)                    63
                'C$' ** F1COL                           /S(SPIT3)                    64
              '*'  F1COL                                /S(SPIT2)                    65
              '.' F1COL                                 /S(PERIOD)F(PROG2)           66
CCARD2          G.O = 'F.READ'                                                       67
PERIOD        STATMENT = SQZ(SNMBR 6COL FRTRN)                                       68
              .NE('5',LINE) NPAGE(X)                                                 69
                SYSPOT = '+' DOLLARS INCARD1 'H CONTROL H CARD H'                    70
                SYSPNT(PS BLANK('84') 'I  /   /  I      I')                          71
CTLOOP          STATMENT ** '(' *(CTX)* ')' = /F(PROG2A)                             72
                CTX *FOO/'1'* =                         /F(CTLERR)                   73
              'MINEDJQX' ** FOO                         /S($('CTL' FOO))             74
CTLERR          SYSPOT = '+' DOLLARS ' INVALID SPECIFICATION '                       75
.                '00000(' FOO CTX ')'                                                75
                SYSPNT(PS BLANK('20') '/  +++++')       /(CTLOOP)                    76
CTLQ            QTAB = QTAB CTX                         /(CTLOOP)                    77
CTLE            QTAB =                                                               78
                END = '0'                               /(CTLOOP)                    79
CTLD            DUMMY = CTX                             /(CTLOOP)                    80
CTLM            MODE = CTX                              /(CTLOOP)                    81
CTLI            IDNO = '0'                                                           82
CTLN            CTX *$(FOO FOO)* ',' *$('DELTA' FOO)* /S(CLT)                        83
                (.NE('0',SIZE(CTX)) CTX) *$(FOO FOO)* /S(CTLOOP)                     84
                $(FOO FOO) =                            /(CTLOOP)                    85
CLT             .NUM($('DELTA' FOO))                    /S(CTLOOP)                   86
                $('DELTA' FOO) = '10'                   /(CTLERR)                    87
CTLX            .EQ('0',.REMDR(CTX,'2')) TDUMP()                                     88
                .EQ('0',.REMDR(CTX,'3')) DUMP()                                      89
                .EQ('0',.REMDR(CTX,'5')) VDUMP()        /(CTLOOP)                    90
CTLJ    $(.EQ(SIZE(CTX),'0') 'RPTFLG') = 'RETURN' /S(CTLOOP)                         91
          RPTFLG = 'RPUT2'                              /(CTLOOP)                    92
PROG2A          SYSPOT = '-MODE = ' MODE                                             93
                SYSPOT = '-DUMMY = ' DUMMY                                           94
                SYSPOT = '-END = ' END                                               95
                SYSPOT = '-IDENT = ' II                                              96
                SYSPOT = ' DELTAI = ' DELTAI                                         97
                SYSPOT = '-S-NMBR = ' NN                                             98
                SYSPOT = ' DELTAN = ' DELTAN                                         99
                SYSPOT = '-'                                                        100
                $(.EQ(SIZE(QTAB),'0') 'SYSPOT') = '-QTAB UNUSED' /S(PROG2TRM)       101
                SYSPOT = '-QTAB IS AS FOLLOWS'                                      102
                FOO2 =                                                              103
TRM             QTAB *FOO* ')' =                        /F(TRM2)                    104
                SYSPNT(' ' FOO ')')                                                 105
                FOO2 = FOO2 FOO ')'                     /(TRM)                      106
TRM2            QTAB = FOO2                                                         107
PROG2TRM        NPAGE(X)                                /(PROG)                     108
*
PROG2           'F' F1COL                               /F(RPROG)                   109
                STATMENT = 'EXTERNAL ' PUT3(SQZ(STATMENT))                          110
```

```
GOUT        STATMENT = TRIM(STATMENT)                                          111
            .EQ('1',MODE)                          /F(GOU2)                    112
            J.R = '1'                                                          113
GOUX        .GE(J.R,I.R)                           /S(GOU2)                    114
            SYSPNT(BLANK('38') '$' $('INCARD' J.R))                            115
            J.R = J.R + '1'                        /(GOUX)                     116
GOU2        STATMENT *FOO/'66'* =                  /F(POUT)                    117
            BOUT(TRN(SNMBR) ' ' FOO IDF(X))                                    118
GOUG        STATMENT *FOO/'66'* =                  /F(OOUT)                    119
            BOUT('      E' FOO IDF(X))             /(GOUG)                     120
OOUT      .EQ(SIZE(STATMENT),'0')                  /S($G..O)                   121
            BOUT('      E' STATMENT BLANK('66' - SIZE(STATMENT)) IDF(X))       122
.                                                  /($G..O)                    122
POUT BOUT(TRN(SNMBR) ' ' STATMENT BLANK('66' - SIZE(STATMENT)) IDF(X))         123
.                                                  /($G..O)                    123
*
SPIT3       INCARD1 *MUMBLE/'72'*                                              124
            BOUT(MUMBLE IDF(X))                    /(PROG)                     125
*
GAGH                                                                           126
GAGL                                                                           127
GAGN                                                                           128
GAGO                                                                           129
GAGQ                                                                           130
GAGT                                                                           131
GAGU                                                                           132
GAGV                                                                           133
GAGX                                                                           134
GAGY                                                                           135
GAGZ                                                                           136
SPIT2       J.R = '1'                                                          137
SPIT4       .GE(J.R,I.R)                           /S(PROG)                    138
            SYSPOT = '+' $('INCARD' J.R) DOLLARS                               139
            SYSPNT(BLANK('84') PS)                                             140
            SYSPPT = $('INCARD' J.R)                                           141
            J.R = J.R + '1'                        /(SPIT4)                    142
*
RPROG       LBEF(STATMENT) *(FOO2)* '=' *FOO4*     /F(RPROG3)                  143
            FOO4 *()* ','                          /S(PDO)                     144
          FOO2 *FOO3* '(' *FOO* /F(ARTH)                                       145
          FOO3 *FOO3/(SIZE(FOO3) - '1')* 'F' /F(ARTH)                          146
          FOO *(FOO)* ')' /F(ARTH)                                             147
          FOO2 = FOO3 '(' FOO ')'                                              148
ARTH        STATMENT = PUT1(FOO2) ' = ' ARSF(FOO4,F1COL) /S(GOUT)F(SPIT2)      149
RPROG3      STATMENT ' ' =                         /S(RPROG3)                  150
            STATMENT *FOO/'1'* =                   /F(SPIT2)                   151
          .NUM(FOO)                                /S(SPIT2)                   152
RPROG77     STATMENT ' ' =                 /S(RPROG77)F($('GAG' FOO))          153
GAGA        GLUG(STATMENT,'SSIGN') *FOO* 'T' *FOO2* '0' *FOO3* /F(SPIT2)       154
            $(.EQ('0',SIZE(TRIM(FOO2))) 'STATMENT') = 'ASSIGN ' TRN(FOO)       155
.             ' TO ' PUT1(TRIM(FOO3))              /S(GOUT)F(SPIT2)            155
GAGC        STATMENT 'A' =                         /S(GAGCA)                   156
            STATMENT 'O' =                         /F(SPIT2)                   157
            STATMENT = LBEF(STATMENT)                                          158
            STATMENT 'N' =                         /S(GAGCON)                  159
            STATMENT = 'COMMON ' PUT3(GLUG(STATMENT,'MMON'))                   160
.                                                  /S(GOUT)F(SPIT2)            160
```

```
GAGCON      STATMENT = 'CONTINUE' GLUG(STATMENT,'TINUE') /S(GOUT)F(SPIT2)          161
GAGCA       STATMENT = GLUG(STATMENT,'LL')          /F(SPIT2)                     162
            STATMENT *FOO* '(' *(FOO2)* ')'         /F(GAGCAS)                    163
            STATMENT = 'CALL ' SQZ(PUT2(FOO)) ' ('                                164
GAGCAL    FOO2 *(FOO)* ',' =                        /F(EGAGCA)                    165
            STATMENT = STATMENT ARSF(FOO,F1COL) ',' /(GAGCAL)                     166
EGAGCA      STATMENT = STATMENT ARSF(FOO2,F1COL) ')' /(GOUT)                      167
GAGCAS      STATMENT = 'CALL ' SQZ(PUT2(STATMENT)) /(GOUT)                        168
GAGE        STATMENT 'Q' =                          /S(GAGEQ)                     169
            STATMENT = GLUG(STATMENT, 'ND')         /F(SPIT2)                     170
            STATMENT = 'END FILE ' QTRN(GLUG(STATMENT,'FILE')) /S(GOUT)           171
GAGEND      END = END + '1'                                                       172
            G..O = 'ENDER'                                                        173
            STATMENT = 'END'                        /(GOUT)                       174
ENDER       .EQ(DATNUM,'0')                         /S(ENDER3)                    175
            G..O = 'ENDER2'                                                       176
            STATMENT = 'REAL ' DUMMY '(' DATNUM ')' /(GOUT)                       177
ENDER2      G..O = 'ENDER3'                                                       178
            DATNUM =                                                              179
            DATL *DATL/(SIZE(DATL) - '1')*                                        180
            STATMENT = 'DATA ' DUMMY '/' DATL '/' /(GOUT)                         181
ENDER3      G..O = 'PROG'                                                         182
LDEL        DATL *FOO* ',' =                        /F(LDELL)                     183
          $FOO =                                    /(LDEL)                       184
LDELL     $DATL =                                                                 185
          DATL =                                                                  186
        .NE(LINE,'5') NPAGE(X)                                                    187
            .EQ(MODE,'2') SYSPNT(STARS 'MODE TWO LISTING' STARS) /F(ECTH)         188
            REWIND('TEMPIN')                                                      189
MTL         TMPREC = TMPREC - '1'                                                 190
            .LE(TMPREC,'0')                         /S(ECTH)                      191
            SYSPNT(TEMPIN)                          /(MTL)                        192
ECTH        .NE(LINE,'5') NPAGE(X)                                                193
            .NE('0',SIZE(BNKLST))                   /F(CTLE2)                     194
            SYSPNT(STARS 'CORRESPONDENCE BETWEEN GENERATED AND '                  195
.             'ORIGINAL STATMENT NUMBERS' STARS)                                  195
CTLE3       BNKLST *FOO* ',' =                      /F(CTLE2)                     196
            SYSPNT(BLANKS $('STN' FOO) BLANK('10' - SIZE($('STN' FOO)))           197
.             FOO )                                                               197
            $('STN' FOO) =                          /(CTLE3)                      198
CTLE2       .NE(LINE,'5') NPAGE(X)                                                199
          .NE(SIZE(BKLIST),'1') SYSPNT(STARS ' SYMBOL USAGE STATISTICS '          200
.             STARS)                                /F(CTLE5)                     200
            BKLIST ',' =                                                          201
CTLE4       BKLIST *FOO* ',' =                      /F(CTLE5)                     202
            SYSPNT(STN FOO BLANK('10' - SIZE(FOO)) $('X' FOO '1')                 203
.             BLANK('9' - SIZE($('X' FOO '1'))) $('X' FOO '2')                    203
.             BLANK('9' - SIZE($('X' FOO '2'))) $('X' FOO '3'))                   203
            $('X' FOO '1') =                                                      204
            $('X' FOO '2') =                                                      205
            $('X' FOO '3') =                        /(CTLE4)                      206
CTLE5       BKLIST = ','                                                          207
            REWIND('TMPOUT')                                                      208
            .NE(LINE,'5') NPAGE(X)                  /(PROG)                       209
GAGEQ       STATMENT = 'EQUIVALENCE ' PUT3(GLUG(STATMENT,'UIVALENCE'))            210
.                                                   /S(GOUT)F(SPIT2)              210
GAGW        STATMENT = GLUG(STATMENT,'RITE')        /F(SPIT2)                     211
```

```
          CUTE = 'WRITE'                          /(CUT)                          212
GAGR      STATMENT 'E' =                          /F(SPIT2)                       213
          STATMENT = LBEF(STATMENT)                                               214
          STATMENT 'A' =                          /S(GAGREA)                      215
          STATMENT 'T' =                          /F(GAGRE)                       216
          STATMENT = 'RETURN' GLUG(STATMENT,'URN') /S(GOUT)F(SPIT2)               217
GAGRE     STATMENT = 'REWIND ' QTRN(GLUG(STATMENT,'WIND'))                        218
.                                                 /S(GOUT)F(SPIT2)                218
GAGREA    STATMENT = GLUG(STATMENT,'D')           /F(SPIT2)                       219
          CUTE = 'READ'                           /(CUT)                          220
GAGB      STATMENT = 'BACKSPACE ' QTRN(GLUG(STATMENT,'ACKSPACE'))                 221
.                                                 /S(GOUT)F(SPIT2)                221
GAGG      STATMENT = GLUG(STATMENT,'OTO')         /F(SPIT2)                       222
          STATMENT ** '(' *FOO* ')' ** ',' *FOO2* /F(GAGG2)                       223
          STATMENT = 'GO TO (' TRNLST(FOO) '), ' PUT2(FOO2) /(GOUT)               224
GAGG2     STATMENT *FOO* ',' ** '(' *FOO2* ')'    /F(GAGSM)                       225
          STATMENT = 'GO TO ' PUT2(FOO) ',(' TRNLST(FOO2) ')' /(GOUT)             226
GAGSM     STATMENT = 'GO TO ' TRN(STATMENT)       /(GOUT)                         227
GAGD      STATMENT = GLUG(STATMENT, 'IMENSION') /F(SPIT2)                         228
          ARG1 =                                                                  229
          ARG2 =                                                                  230
GAGDLOP   STATMENT *FOO* '(' *(FOO2)* ')' ** ',' *STATMENT* /F(GAGDL)             231
          .EQ(FOO2, '0')                          /S(GAGDX)                       232
        $(.EQ(X,'1') 'ARG2') = FOO '), '                                          233
        X = '0'                                                                   234
          ARG1 = ARG1 PUT3(FOO) '(' PUT2(FOO2) '), ' /S(GAGDLOP)                  235
GAGDX     ARG1 = ARG1 PUT3(FOO) '(1), '                                           236
        $(.EQ(X,'0') 'ARG2') = ARG2 '('                                           237
        X = '1'                                                                   238
          ARG2 = ARG2 FOO ','                     /(GAGDLOP)                      239
GAGDL     STATMENT *FOO* '(' *(FOO2)* ')'                                         240
        $(.EQ(X,'1') 'ARG2') = ARG2 FOO ')'                                       241
        X = '0'                                                                   242
          ARG1 = ARG1 PUT3(FOO) '(' PUT2(FOO2) ')'                                243
          $(.NE('0', SIZE(ARG2)) 'G..0') = 'GAGD.L'                               244
          $(EQUALS(F1COL, 'D') 'STATMENT') = 'DOUBLE PRECISION'                   245
.                                                 /S(GAGD..L)                     245
          $(EQUALS(F1COL, 'I') 'STATMENT') = 'COMPLEX' /S(GAGD..L)                246
          STATMENT = 'DIMENSION'                                                  247
GAGD..L   STATMENT = STATMENT ' ' ARG1            /(GOUT)                         248
GAGD.L    G..0 = 'PROG'                                                           249
          SNMBR = BLANK('6')                                                      250
          STATMENT = 'EQUIVALENCE ' ARG2          /(GOUT)                         251
GAGF      STATMENT 'O' =                          /S(GAGFO)                       252
          STATMENT 'R' =                          /S(GAGFR)                       253
          STATMENT = 'FUNCTION ' PUT3(GLUG(STATMENT,'UNCTION'))                   254
.                                                 /S(GOUT)F(SPIT2)                254
GAGFR       GLUG(STATMENT,'EQUENCY')              /S(PROG)F(SPIT2)                255
GAGFO     STATMENT = 'FORMAT ' GLUG(STATMENT,'RMAT') /S(GOUT)F(SPIT2)             256
GAGSU     STATMENT = 'SUBROUTINE ' PUT3(GLUG(STATMENT,'BROUTINE'))                257
.                                                 /S(GOUT)F(SPIT2)                257
GAGST     STATMENT = 'STOP' GLUG(STATMENT,'OP') /S(GOUT)F(SPIT2)                  258
GAGS      STATMENT 'T' =                          /S(GAGST)                       259
          STATMENT 'U' =                          /S(GAGSU)                       260
          STATMENT = 'CALL SLITE (' SQZ(GLUG(STATMENT,'ENSELIGHT'))               261
            ')'                                   /S(GOUT)F(SPIT2)                261
.                                                                                 261
GAGI      STATMENT 'F' =                          /F(SPIT2)                       262
```

```
                STATMENT = LBEF(STATMENT)                                       263
                STATMENT '(' =                          /S(GAGIF)               264
                STATMENT = SQZ(STATMENT)                                        265
                STATMENT 'ACCUMULATOROVERFLOW' =        /S(GAGIFA)              266
                STATMENT 'DIVIDECHECK'         =        /S(GAGIFD)              267
                STATMENT 'QUOTIENTOVERFLOW' =           /F(SPIT2)               268
GAGIFA          CUTE = 'OVERFL'                         /(GAGI2)                269
GAGIFD          CUTE = 'DVCHK'                                                  270
GAGI2           STATMENT *FOO* ',' *FOO1*               /F(SPIT2)               271
                G..O = 'GAGIRET'                                                272
                STATMENT = 'CALL ' CUTE '(' DUMMY 'X)' /(GOUT)                  273
GAGIRET         G..O = 'PROG'                                                   274
                SNMBR = BLANK('6')                                              275
                STATMENT = 'IF (' DUMMY 'X)' TRN(FOO) ', ' TRN(FOO) ', '        276
.                 TRN(FOO1)                             /(GOUT)                 276
GAGIF           STATMENT *(FOO)* ')' *FOO1* ',' *FOO3* /F(SPIT2)                277
                FOO2 = FOO                                                      278
                FOO = SQZ(FOO)                                                  279
                FOO 'SENSESWITCH' *FOO*                 /S(GAGIFS)              280
                FOO 'SENSELIGHT' *FOO*                  /S(GAGIFL)              281
                FOO3 *FOO4* ',' *FOO3*                  /F(SPIT2)               282
                STATMENT = 'IF (' ARSF(FOO2,F1COL) ') ' TRN(FOO1) ', '          283
.                 TRN(FOO4) ', ' TRN(FOO3)              /S(GOUT)F(SPIT2)        283
GAGIFL          CUTE = 'SLITET'                         /(GAGIF2)               284
GAGIFS          CUTE = 'SSWTCH'                                                 285
GAGIF2          G..O = 'GAGIRFT'                                                286
                STATMENT = 'CALL ' CUTE '(' FOO ',' DUMMY 'X)' /(GOUT)          287
GAGIRFT         G..O = 'PROG'                                                   288
                SNMBR = BLANK('6')                                              289
                STATMENT = 'IF (' DUMMY 'X)' TRN(FOO1) ', ' TRN(FOO1) ', '      290
.                 TRN(FOO3)                             /(GOUT)                 290
GAGP            STATMENT 'A' =                          /S(GAGPA)               291
                STATMENT 'R' =                          /S(GAGPR)               292
                STATMENT = GLUG(STATMENT,'UNCH')        /F(SPIT2)               293
                CUTE = 'PUNCH'                          /(CVT)                  294
GAGPR           STATMENT = GLUG(STATMENT,'INT')         /F(SPIT2)               295
                CUTE = 'PRINT'                          /(CVT)                  296
GAGPA           STATMENT = 'PAUSE' GLUG(STATMENT,'USE') /S(GOUT)F(SPIT2)        297
CUT             STATMENT = LBEF(STATMENT)                                       298
                STATMENT *FOO/'1'* =                                            299
                $(EQUALS(FOO,'(') 'STATMENT') *FOO* ')' *FOO3* /S(CUT3)         300
                $(EQUALS(FOO,'I') 'STATMENT') = GLUG(STATMENT,'NPUTTAPE')       301
.                                               /S(CUT5)                        301
                $(EQUALS(FOO,'O') 'STATMENT') = GLUG(STATMENT,'UTPUTTAPE')      302
.                                               /S(CUT5)                        302
                $(EQUALS(FOO,'T') 'STATMENT') = GLUG(STATMENT,'APE')            303
.                                               /S(CUT7)F(CUT8)                 303
CUT5            STATMENT *FOO* ',' *FOO2* ',' *FOO3*    /F(CUT6)                304
CUT4            STATMENT = CUTE ' (' QTRN(FOO) ', ' TRN(FOO2) ') '              305
.                 PUT(FOO3,$CUTE)                       /(GOUT)                 305
CUT6            STATMENT *FOO* ',' *FOO2*               /F(SPIT2)               306
                STATMENT = CUTE ' (' QTRN(FOO) ',' TRN(FOO2) ') ' /(GOUT)       307
CUT7            STATMENT *FOO* ',' *FOO2*               /F(CUT2)                308
                STATMENT = CUTE ' (' QTRN(FOO) ') ' PUT(FOO2,$CUTE) /(GOUT)     309
CUT2            STATMENT = CUTE ' (' QTRN(STATMENT) ')' /(GOUT)                 310
CUT3            FOO *FOO* ',' *FOO2*                    /S(CUT4)                311
                STATMENT = CUTE ' (' QTRN(FOO) ') ' PUT(FOO3,$CUTE) /(GOUT)     312
```

```
CUT8      (FOO STATMENT) *FOO* ',' *FOO2*          /F(CUT9)                    313
          STATMENT = CUTE ' ' QTRN(FOO) ',' PUT1(FOO2) /(GOUT)                 314
CUT9      STATMENT = CUTE ' ' QTRN((FOO STATMENT)) /(GOUT)                     315
*
PDO       STATMENT = GLUG(STATMENT,'DO')           /F(ARTH)                    316
          STATMENT = SQZ(STATMENT)                                             317
          I.N =                                                                318
PD2       STATMENT *FOO/'1'* =                                                 319
          $(.NUM(FOO) 'I.N') = I.N FOO             /S(PD2)                     320
          STATMENT *FOO2* '=' *FOO3* ',' =         /F(SPIT2)                   321
        PUT2(STATMENT)                                                         322
          STATMENT *T69* ',' =                     /F(PD3)                     323
          STATMENT = ', ' STATMENT                                             324
PD4       STATMENT = 'DO ' TRN(I.N) ' ' PUT1(FOO FOO2) ' = '                   325
.           PUT2(FOO3) ', ' T69 STATMENT           /(GOUT)                     325
PD3       T69 = STATMENT                                                       326
          STATMENT =                               /(PD4)                      327
*
CVT       STATMENT *FOO* ',' *FOO2*                /F(CVT2)                    328
          STATMENT = CUTE ' ' TRN(FOO) ', ' PUT2(FOO2)                         329
.                                                  /(GOUT)                     329
CVT2      STATMENT = CUTE ' ' TRN(STATMENT)        /(GOUT)                     330
*
PUT         I.J = N                                                            331
          PUT = ARG                                /($RPTFLG)                  332
PUT1        I.J = '1'                              /(RPUT)                     333
PUT2        I.J = '2'                              /(RPUT)                     334
PUT3        I.J = '3'                                                          335
RPUT      $('PUT' I.J) = ARG                       /($RPTFLG)                  336
RPUT2       ARG = SQZ(ARG)                                                     337
RPLP        F.K =                                                              338
RPLP2       ARG *FK/'1'* =                         /F(RETURN)                  339
            '.()+-*/,=' ** FK                      /S(RPLP2)F(PUTER)           340
RLOP        ARG *FK/'1'* =                         /F(PUTR)                    341
          '.' FK                                   /S(RLOP)                    342
            '+-,*/)' ** FK                         /S(PUTR)                    343
          '(' FK                                   /S(PUSH)                    344
          '=' FK                                   /S(PONE)                    345
PUTER       F.K = F.K FK                           /(RLOP)                     346
PUTR        .NUM(F.K)                              /S(RPLP)                    347
            .NE('0',SIZE(F.K))                     /F(RPLP)                    348
            $('X' F.K I.J) = $('X' F.K I.J) + '1'                              349
OH          BKLIST ** ',' F.K ','                  /S(RPLP)                    350
            BKLIST = BKLIST F.K ','                /(RPLP)                     351
PUSH        ARG *(RARG)* ')' *ARG*                 /F(RLOP)                    352
            PUT2(RARG)                             /(PUTR)                     353
PONE        $('X' F.K '1') = $('X' F.K '1') + '1'  /(OH)                       354
                                                                               355
*
TRNLST      ARG *FK* ',' =                         /F(TRNEX)                   356
            F.K = F.K TRN(FK) ','                  /(TRNLST)                   357
TRNEX       TRNLST = F.K TRN(ARG)                  /(RETURN)                   358
*
NPAGEL      PAGE = PAGE + '1'                                                  359
            SYSPOT = '1'                                                       360
            SYSPOT = '+H' BLANK('39') 'FORTRAN TRANSLATOR AND ANALYSER.'       361
.             BLANK('39') 'PAGE ' PAGE                                         361
```

```
                SYSPOT = ' I' BLANK('40') '/' BLANK('14') '/' BLANK('14')          362
     .            QUOTE                                                          362
                SYSPOT = '-'                                                      363
                LINE = '5'                          /(RETURN)                     364
     *
     SCZL       MODE('UNANCHOR')                                                  365
     SCZ2       ARG ' ' =                           /S(SQZ2)                      366
                SQZ = ARG MODE('ANCHOR')            /(RETURN)                     367
     *
     TRNL       ARG = SQZ(ARG)                                                    368
                TRN = ARG                                                         369
                .NUM(ARG)                           /F(RETURN)                    370
     TRN2       $(.EQ('0',SIZE(NN)) 'TRN') = BLANK('5' - SIZE(ARG)) ARG           371
     .                                              /S(RETURN)                    371
                $(.NE('0',SIZE($('STN' ARG))) 'TRN') = $('STN' ARG)               372
     .                                              /S(RETURN)                    372
                NN = NN + DELTAN                                                  373
                BNKLST = BNKLST ARG ','                                           374
                $('STN' ARG) = BLANK('5' - SIZE(NN)) NN /(TRN2)                   375
     *
     BOUTL      SYSPNT(ARG)                                                       376
                SYSPPT = ARG                        /(RETURN)                     377
     *
     ICLE       IDF = ID1                                                         378
                .EQ('0',SIZE(II))                   /S(RETURN)                    379
     IDEL       T69 = SIZE(II) + SIZE(IDNO)                                       380
                $(.LE(T69,'8') 'FOO') = '10' ** ('8' - T69) /F(IDLE2)             381
                FOO */'1'* *FOO*                                                  382
                IDF = II FOO IDNO                                                 383
                IDNO = IDNO + DELTAI                /(RETURN)                     384
     IDLE2      IDNO = '0'                          /(IDEL)                       385
     *
     QTRL       ARG = SQZ(ARG)                                                    386
                QTRN = ARG                                                        387
                QTAB ** '(' ARG ',' *QTRN* ')'      /(RETURN)                     388
     *
     LBLCC      ARG ' ' =                           /S(LBLOC)                     389
                LBEF = ARG                          /(RETURN)                     390
     *
     GLUGL      ARG2 *T69/'1'* =                    /F(GLUGOT)                    391
     GLUG2      ARG1 ' ' =                          /S(GLUG2)                     392
                ARG1 T69 =                          /S(GLUGL)F(FRETURN)           393
     GLUGOT     GLUG = ARG1                         /(RETURN)                     394
     *
     SISPINT    SYSPOT = ' ' XXX                                                  395
                LINE = LINE + '1'                                                 396
                .GE(LINE,'57') NPAGE(X)             /(RETURN)                     397
     *
     BLNKL      ('                                                  '            398
     .            '                              ') *BLANK/ARG* /(RETURN)        398
     *
     ARSF     'B' XCOL                             /S(ARSBOOL)                   399
              FLAG = 'RETURN'                                                     400
     ARSF2      RIAH =                                                           401
     PEEL       ARG ' ' =                           /S(PEEL)                      402
                ARG *HAIR/'1'* =                    /F($FLAG)                     403
                FLAG = 'NUM'                                                      404
```

```
          $(.NUM(HAIR) 'RIAH') = RIAH HAIR      /S(PEEL)                405
VARIB     '+-*/,' ** HAIR                       /S(OPOP)                406
          '(' HAIR                              /S(OPN)                 407
          'H' .NUM(RIAH) .NE(SIZE(SQZ(RIAH)),'0') HAIR /S(HOLLER)       408
            FLAG = 'VAR'                                                409
            RIAH = RIAH HAIR                                            410
            ARG *HAIR/'1'* =                    /S(VARIB)F($FLAG)       411
NUM       ARSF = ARSF RIAH                      /(RETURN)               412
VAR         ARSF = ARSF PUT2(RIAH)              /(RETURN)               413
OPOP      ARSF = ARSF PUT2(RIAH) HAIR           /(PLOP)                 414
OPN       .GE(SIZE(RIAH),'4')                   /S(PFUNC)               415
SUBS      ARSF = ARSF RIAH '('                  /(ARSF)                 416
PFUNC     RIAH *RIAH/(SIZE(RIAH) - '1')* 'F'    /(SUBS)                 417
HOLLER      ARG *SUBS/RIAH* =                   /F(FRETURN)             418
            XXX = RIAH                                                  419
            RIAH = RIAH 'H' SUBS                                        420
            .EQ($RIAH,'0')                      /F(HOLER)               421
            $RIAH = DATNUM + '1'                                        422
            DATNUM = DATNUM + (((XXX - '1') / '6') + '1')               423
            DATL = DATL RIAH ','                                        424
HOLER       ARSF = ARSF DUMMY '(' $RIAH ')'     /(ARSF)                 425
PLOP        FLAG = 'FRETURN'                    /(ARSF2)                426
*
ARSBOOL   ARG ** '+'                            /S(RARSBOL)             427
          ARG ** '-'                            /S(RARSBOL)             428
          ARG ** '*'                            /S(RARSBOL)             429
          ARG ** ','                            /S(RARSBOL)             430
          ARG ** '('                            /S(RARSBOL)             431
          ARSF = DATRAN(ARG)                    /(RETURN)               432
DATRAN    ARG = SQZ(ARG)                                                433
          ARG *XXX/'1'*                         /F(RETURN)              434
          DATRAN = ARG                                                  435
          .NUM(XXX)                             /F(RETURN)              436
          ARG = '0' ARG                                                 437
          .NE($ARG,'0')                         /F(DATNR)               438
DATOT     DATRAN = DUMMY '(' $ARG ')'           /(RETURN)               439
DATNR     DATNUM = DATNUM + '1'                                         440
          $ARG = DATNUM                                                 441
          DATL = DATL ARG ','                   /(DATOT)                442
RARSBOL   ARSF = 'BINSIM('                                              443
RZXQQ     ARG '+' =                             /S(RZXQQ)               444
          ARG '-'                               /F(RBOLOP)              445
          ARG = '0' ARG                                                 446
RBOLOP    RIAH =                                                        447
BOLCP2    ARG *HAIR/'1'* =                      /F(ENRBOL)              448
          '+-*' ** HAIR                         /S(BBLOP)               449
          '(' HAIR                              /S(PFUK2)               450
          ',' HAIR                              /S(FNRBOL)              451
          ' ' HAIR                              /S(BOLOP2)              452
          RIAH = RIAH HAIR                      /(BOLOP2)               453
ENRBOL    ARSF = ARSF DATRAN(RIAH) ')'          /(RETURN)               454
FNRBOL    ARSF = ARSF DATRAN(RIAH) '),('        /(RBOLOP)               455
PFUK2     .GE(SIZE(RIAH),'4')                   /S(FKU2)                456
FFKK      ARG *(DEF)* ')' =                     /F(FRETURN)             457
          .EQ(SIZE(RIAH),'0')                   /S(FKFK)                458
          ARSF = ARSF RIAH '(' ARSF(DEF,' ') ')' /(RBOLOP)              459
FKU2      RIAH *RIAH/(SIZE(RIAH) - '1')* 'F'    /F(FFKK)                460
```

```
          ARG *(DEF)* ')' =                                  /F(FRETURN)                    461
     FKFK ARSF = ARSF DATRAN(RIAH) '(' ARSF(DEF,'B') ')' /(RBOLOP)                          462
     BBLOP ARSF = ARSF DATRAN(RIAH) ',' $HAIR ','  /(RBOLOP)                                463
     END                                                                                    464
```

SUCCESSFUL COMPILATION

■■■■■■■■■■. (M3) (JON) (N90,10) (Q(INTAPE,5)(7,6)(6,7)) (IEXAMP,5) (DDUMMY) ■ CØNTRØL ■ CARD ■

MODE = 3

DUMMY = DUMMY

END =

IDENT = EXAMP
DELTAI = 5

S-NMBR = 90
DELTAN = 10

QTAB IS AS FOLLOWS
(INTAPE,5)
(7,6)
(6,7)

```
*     LIST8                                                                    ●●●●●●●●●●
*     LABEL                                                                    ●●●●●●●●●●
C                                                                    EXAMP000
C      THIS IS A SAMPLE OF SFT INPUT AND OUTPUT.                     EXAMP005
C      NOTE THE MANY BELLS AND WHISTLES USED ON THE ABOVE CONTROL CARD. EXAMP010
C                                                                    EXAMP015
      SUBROUTINE  EXAMPL (A, XX, IBBB, YYYY, ICCCCC, ZZZZZZ)         EXAMP020
C                                                                    EXAMP025
C      ARITHMETIC STATMENT FUNCTIONS                                 EXAMP030
      FOO(Q) = SIN(Q)**2+COS(Q)**2                                   EXAMP035
      XOR(Q,U) = BINSIM(Q,3,(BINSIM(DUMMY(1),2,U)),1,U,3,(BINSIM(DUMMY(1EXAMP040
     E),2,C)))                                                       EXAMP045
C                                                                    EXAMP050
      EXTERNAL SUB1,SUB2,ROUSPC                                      EXAMP055
C                                                                    EXAMP060
      DIMENSION  II(10),  JJ(1),  ZAP(20)                            EXAMP065
      EQUIVALENCE ( JJ, ZAP)                                         EXAMP070
      COMPLEX  A(IBBB,ICCCCC),  XX(1000)                             EXAMP075
      DOUBLE PRECISION  YYYY(IBBB,ICCCCC,2), KK(1), ZZZZZZ(IBBB,ICCCCC) EXAMP080
      EQUIVALENCE (KK,ZZZZZZ)                                        EXAMP085
      FUN  = DUMMY(2)                                                EXAMP090
      READ (5,   100)  ZZ1, ZZ2                                      EXAMP095
      INTEGR  = ZZ2                                                  EXAMP100
      IF (XOR(ZZ1,FUN))   110,   120,   110                          EXAMP105
  120 J  = 0                                                         EXAMP110
      I = 0                                                          EXAMP115
      PRINT   130                                                    EXAMP120
      CALL SSWTCH(5,DUMMYX)                                          EXAMP125
      IF (DUMMYX)  140,   14C,   150                                 EXAMP130
      THIS IS A DELIBERATLY ERRONEOUS STATMENT 123456799123456N789123456789123456 ●●●●●●●●●●
     1    CONTINUATION OF ERRONEOUS CARD                             FOO      ●●●●●●●●●●
  140 CALL SSWTCH(5,DUMMYX)                                          EXAMP135
      IF (DUMMYX)  160,   160,   170                                 EXAMP140
  160 I  = I+1                                                       EXAMP145
      IF (I-32767)   140,   180,   190                               EXAMP150
  180 I = 0                                                          EXAMP155
      J = J+1                                                        EXAMP160
      GO TO   140                                                    EXAMP165
  150 CALL SSWTCH(5,DUMMYX)                                          EXAMP170
      IF (DUMMYX)  170,   170,   200                                 EXAMP175
  200 I  = I+1                                                       EXAMP180
      IF (I-32767)   150,   210,   190                               EXAMP185
  210 J = J+1                                                        EXAMP190
      I = 0                                                          EXAMP195
      GO TC   150                                                    EXAMP200
  170 Z = FLOAT(J)*0.832+FLOAT(I)*(0.832/32766.0)                    EXAMP205
      PRINT   220,     Z                                             EXAMP210
      WRITE (6,   220) Z                                             EXAMP215
  110 CO   230 I = 1, IBBB, INTEGR                                   EXAMP220
      A(I,1)  = FLOAT(I)*XX(INTEGR)                                  EXAMP225
      CO   240 J = 1, ICCCCC, I                                      EXAMP230
  240 ZZZZZZ(I,J)  = A(J,1)+XX(J)                                    EXAMP235
```

```
      YYYY(I,1,1)  = YYYY(I,1,2)+A(I,2)+A(I,3)                          EXAMP240
  230 CONTINUE                                                          EXAMP245
      DO   250 I = 1, ICCCCC, INTEGR                                    EXAMP250
      A(1,I)  = DUMMY(3)                                                EXAMP255
      A(2,I)  = DUMMY(4)                                                EXAMP260
      YYYY(1,I,2)  = BINSIM(YYYY(1,I,1),3,DUMMY(5),1,ZZZZZZ(I,I))       EXAMP265
  250 CONTINUE                                                          EXAMP270
      ZAP(13)                                                           EXAMP275
     E = 13                                                             EXAMP280
      DO   260 I = 1, 10                                                EXAMP285
  260 II(I) = JJ(I)                                                     EXAMP290
      RETURN                                                            EXAMP295
  190 PRINT   270                                                       EXAMP300
      WRITE (6,  270)                                                   EXAMP305
      PAUSE                                                             EXAMP310
      RETURN                                                            EXAMP315
  270 FORMAT  (18H HORRENDOUS ERROR.)                                   EXAMP320
  220 FORMAT  (9H YOU TOOK, F10.2,9HSECONDS.)                           EXAMP325
  130 FORMAT  (14H DEAR OPERATOR/15H ARE YOU AWAKE/                     EXAMP330
     E 43H IF SO CHANGE THE STATUS OF SENSE SWITCH 5.)                  EXAMP335
  100 FORMAT  (A3, F8.2)                                                EXAMP340
      END                                                               EXAMP345
      REAL DUMMY(5)                                                     EXAMP350
      DATA DUMMY/00,3HFUN,077,01000,07070707070707/                     EXAMP355
```

**********CORRESPONDENCE BETWEEN GENERATED AND ORIGINAL STATMENT NUMBERS**********

100	4
110	101
120	1111
130	5
140	1
150	2
160	11
170	3
180	111
190	69
200	22
210	222
220	6
230	1010
240	1011
250	1020
260	1030
270	7

********** SYMBOL USAGE STATISTICS **********

A	3	1	1
XX		1	1
I888		5	
YYYY	2	1	1
ICCCCC		6	
ZZZZZZ	1	1	1
EXAMPL			1
Q		4	
FOO	1		
U		1	
XOR	1		
SUB1			1
SUB2			1
ROUSPC			1
II	1		1
JJ			1
ZAP	1		1
KK			1
FUN	1	1	
ZZ1	1	1	
ZZ2	1	1	
INTEGR	1	3	
J	4	6	
I	8	21	
Z	1	2	

**********. (END) ■ CØNTRØL ■ CARD ■

MODE = 3

DUMMY = DUMMY

END = 0

IDENT = EXAMP
DELTAI = 5

S-NMBR = 270
DELTAN = 10

QTAB UNUSED

```
 NORMAL EXIT FROM SNOBOL AT LEVEL       0

SNOBOL RUN STATISTICS , NO. OF RULES EXECUTED = 13932 NO OF SCANNER ENTRIES =  7713

STORAGE ALLOCATION STATISTICS --    3889  STRINGS STORED   20924  WORDS FOR STORED STRINGS
       6150 REFERENCE ASSIGNMENT WORDS ,      2 REFERENCE,      2   GARBAGE, AND        0 HYPER-GARBAGE COLLECTION(S)
```

$ID DONALD E. EASTLAKE 3RD*001/65/003*5M*100C*40P*N$ 27204 11/22/65 ON 20-55-24 OFF 20-56-32

TYPE OF PROCESS	COMPILER	ASSEMBLER	LOADER	EXECUTION	UTILITY	OTHER	TOTAL TIME
TIME FOR EACH	000.28	000.00	000.00	000.81	000.02	000.00	001.12