```
//DECK BCPLGD
//  DECLARATIONS TO LINK WITH HEAD

GLOBAL [ // TO LINK WITH FSP AND IO
         // GLOBALS 1-4 ARE FREE
         FLIST       :   5  // FREESPACE PACKAGE POINTER
         OUTPUT      :   6  // STANDARD OUTPUT STREAM
         C6TO7       :   7  // DISPLAY CODE ASCII CONVERSION VECTOR
         C7TO6       :   8  // ASCII DISPLAY CODE  CONVERSION VECTOR
         IOBASE      :   9  // OPEN STREAM LIST POINTER
         MONITOR     :  10
    ]

EXTERNAL [
         NEWVEC          // NEWVEC(SIZEWANTED)
         RETVEC          // RETVEC(V,SIZEOFV)
         PACKSTRING      // (VECTOR,STRING)
         UNPACKSTRING    // (STRING,VECTOR)
         INITIALIZEIO
         FINDINPUT       //  (EFNAME,USERNAMEE)
         CREATEOUTPUT    //  (EFNAME,USERNAMEE)
         READCH          // (STREAM,LV CH)
         WRITECH         // (STREAM,CH)
         WRITES          // OUTPUT:=STREAM; WRITES(STRING)
         WRITEN          // OUTPUT:=STREAM; WRITEN(NUMBER)
         WRITEO          // OUTPUT:=STREAM; WRITEO(NUMBER)
         READN           // NUMBER READER N:=READN(STREAM)
         TRACE
         TRACECALL
         TRACERETURN
         DUMP
         ENDREAD         // CLOSE INPUT STREAM
         ENDWRITE        // CLOSE OUTPUT STREAM
         IOERROR         // GENERAL IOERROR ROUTINE
         UNPACKTO        //  (STR,VEC,BREAKCH)
         WRITEVEC
         READVEC
         FLUSH           // FORCE OUT A LINE WITHOUT CR
         BCDWORD
         ASCII
         ENDOFSTREAM
         CLOSEALL
         ABORT
    ]

MANIFEST [ AMASK = 777777B  ; ENDOFSTREAMCH =144B
         BUFFERSIZE = 136
          ESCAPECH = 173B ]
    GLOBAL [GV    //  GLOBAL VARIABLES
INPUT                :  49
    FREE             :  50
    INPUTOPEN        :  51
    EXPLIMIT         :  52
// TWO HERE
    HILINE           :  54
    LOWLINE          :  55
```

```
    FREELINE        : 56
    DATA            : 57
    TMPDATA         : 58
    DATACOUNT       :59
    TMPDATACNT      :60
    RETURNS         : 61
    PROG            : 62
    LASTPROG        : 63
    EXPSTOP         :64
    S               : 65
    CHARCNT         :66
    SIG             : 67
    RUNNING         : 68
    TIME        :72
    NEXT            : 75
    NEXTINPUT       : 76
    ENDINPUT        : 77
    STNUMBER        : 78
    NUMBER          : 79
    ACCEPT          : 80
    RECURSE         : 82
    TYPE            : 84
// 88,89 RESERVED FOR MATHLIB
// HERE IS 2
    EXPTYPE         : 92
    CURSEED         : 93
    SEEDFLAG        : 94
    LIMIT           : 95
    IERROR          : 96
    RELEASE         : 97
        ]GV
EXTERNAL  [GB      // GLOBAL BUFFERS AND VECTORS
    VARIABLES
    FUNCTIONS
    ARRAYS
    LINES
    EXPCALLS
    CHARBUF
    STRINGS
    STARRAYS
    IOBUF
    INBUF
    GLOBALO
        ]GB
EXTERNAL  [GR       //  GLOBAL ROUTINES
    COMPILE
    ADDRESS
    EXPGEN
    LEXANA
    CONVERT
    WRITEF
    UNPACK
    PACK
    COMPACT
    LINUM
    FINDLINE
```

```
COPY
DATAOK
DATASTORE
DATAINIT
ALPHA
DIGIT
INT
FLOAT
FMULT
FDIV
FADD
CHKFDIV
CHKPIV
SIMPLEADD
EXECUTE
MATEX
EXPRESS
GIVESPACE  = ≡RETVEC≡
GETSPACE   = ≡NEWVEC≡
READLINE
WRITE
EERROR
CERROR
RERROR
SYSERR
EXCHARS
UNREC
FWDLINE
FILL
COPYSTRING
ELE
ELE3
MATADD
MATSUB
MATSML
INPROD
NORM
MATCMD
          ]GR
MANIFEST  [MGC    //  SOME GENERAL CONSTANTS
QUOTE             = ↓≡↓
AMASK             = 377777B
NULLCH            = 140B
NMASK             = -377777B
NAMASK            = -377777B
ZONESIZE          = 72/5
LINESIZE          = 72
ARITHEXP          = -1
LOGEXP            = -2
STRNGEXP          = -4
HIADD             = 5000
INDEF             = 60000000000000000000B
ONE               = 17204000000000000000B
TEN               = 17235000000000000000B
RNDSEED           = 113434743B
DELETE            = 3
```

```
    CHANGE              = 2
    INSERT              = 1
    DIRECT              = 4
    QUIT                = 5
    INTERUPTED = FALSE // SO THAT INTERUPT WILL BE DEFINED FOR MATEX
             ]MGC
MANIFEST    [MCMD       //  MANIFEST CONSTANTS FOR IDENTIFING LINES
    LETCMD    = 1
    GOTCMD    = 2
    IFCMD     = 3
    IFTCMD    = 3
    FORCMD    = 4
    NEXCMD    = 5
    REACMD    = 6
    INPCMD    = 7
    PRICMD    = 8
    DEFCMD    = 9
    REMCMD    = 10
    GOSCMD    = 12
    RETCMD    = 13
    ENDCMD    = 14
    STOCMD    = 15
    DATCMD    = 16
    RESCMD    = 17
    SIGCMD    = 18
    DIMCMD    = 19
    ONCMD     = 20
    MPRICMD   = 21
    MREACMD   = 22
    MTRNCMD   = 23
    MZERCMD   = 24
    MIDNCMD   = 25
    MCONCMD   = 26
    MADDCMD   = 27
    MSUBCMD   = 28
    MMLTCMD   = 29
    MSMLCMD   = 30
    MINVCMD   = 31
        ]MCMD
MANIFEST [MEC    // INDEXES INTO EXPCALLS
    ABS       = 1
    ATAN      = 2
    COSINE    = 3
    EXP       = 4
    INTEGER   = 5
    LOG       = 6
    SIGN      = 7
    SINE      = 8
    SQRT      = 9
    TAN       = 10
    RND       = 11
    ACS       = 12
    ASN       = 13
    LGT       = 14
    TIM       = 15
    ENDEXP    = 8 + 12
```

```
    ADD      = 8 + 13
    SUBTRAC  = 8 + 14
    MULT     = 8 + 15
    DIV      = 8 + 16
    NEG      = 8 + 17
    ANDCALL  = 8 + 18
    ORCALL   = 8 + 19
    EQUAL    = 8 + 20
    NEQUAL   = 8 + 21
    LESSTH   = 8 + 22
    LESS     = 8 + 22
    MORETH   = 8 + 23
    MORE     = 8 + 23
    LESSEQ   = 8 + 24
    MOREEQ   = 8 + 25
    STORE    = 8 + 26
    FETCH    = 8 + 27
    NMCNST   = 8 + 28
    ADCNST   = 8 + 29
    ONECNST  = 8 + 30
    FNCALL   = 8 + 31
    FNRET    = 8 + 32
    PARVAL   = 8 + 33
    ARYCONT  = 8 + 34
    ARYADD   = 8 + 35
    EXOPNET  = 8 + 36
    EXPO     = 8 + 36
    LSTCONT  = 8 + 37
    LSTADD   = 8 + 38
    STLSTC   = 8 + 39
    STLSTA   = 8 + 40
    STOREST  = 8 + 41
    FETCHST  = 8 + 42
    STEQUAL  = 8 + 43
    STCNST   = 8 + 44
    STNEQUAL = 8 + 45
    PICNST   = 8 + 46
    STMORETH = 8 + 47
    STMOREEQ = 8 + 48
    STLESSTH = 8 + 49
    STLESSEQ = 8 + 50
          ]MEC
MANIFEST [MPT    //PARSE TYPE CONSTANTS
    PTPLUS    = 1
    PTMULT   = 2
    PTEXPO   = 3
    PTROP    = 4
    PTCM     = 5
    PTRPREN  = 6
    PTLPREN  = 7
    PTAND    = 8
    PTOR     = 9
    PTNOT    = 10
    PTFNC    = 11
    PTID     = 12
    PTARY    = 13
```

```
    PTEND     = 14
          ]MPT
MANIFEST    [MGT     //  GENERATING TYPE CONSTANTS
    GTPLUS    = 1
    GTMINUS   = 2
    GTAOP     = 3
    GTROP     = 4
    GTLOP     = 5
    GTCM      = 6
    GTRPREN   = 7
    GTLPREN   = 8
    GTNUM     = 9
    GTFNC     = 10
    GTARY     = 11
    GTVAR     = 12
    GTEND     = 14
    GTNOT     = 15
    GTSTLST   = 16
    GTSTVAR   = 17
    GTSTNUM   = 18
    GTPI      = 21
          ]MGT
MANIFEST [ ENDMARK  = (PTEND ASHIFT 53)+(GTEND ASHIFT 46) ]
EXTERNAL [ TIMFET ]
LET BASIC() BE [BASIC // BATCH BASIC MAIN PROGRAM
FILL(ARRAYS,INDEF,26)
FILL(STRINGS,INDEF,52)
FILL(STARRAYS,INDEF,26)
FILL(FUNCTIONS,INDEF,26)
FILL(VARIABLES,INDEF,286)
IF FREE LE LINES DO PERROR(≡ FIELD LENGTH TOO SMALL*N≡)
CHARCNT := 0
INPUTOPEN := FALSE
FLIST,DATA,RETURNS,PROG,RECURSE,LIMIT:= 0,0,0,0,0,0
SIG := 7
LASTPROG := 0
SEEDFLAG := TRUE
CURSEED := RNDSEED
NEXT := NXT
INITIALIZEIO(IOBUF,2*BUFFERSIZE)
OUTPUT := CREATEOUTPUT(BCDWORD(≡OUTPUT≡))
MONITOR := OUTPUT
INPUT := FINDINPUT(BCDWORD(≡INPUT≡))
HILINE,LOWLINE := 0,0
FREELINE := LINES+1
LINES.0 := 0
WRITECH(OUTPUT,↓1↓)  // EJECT THE FIRST PAGE
// COMPILE LINES UNTIL END OF FILE
NXT: UNTIL ENDOFSTREAM(INPUT) DO [COMP
    LET I = 0
    [   I := I + 1
        READCH(INPUT,CHARBUF+I)
        WRITECH(OUTPUT,CHARBUF.I)
    ] REPEATUNTIL CHARBUF.I = ↓*N↓
    CHARBUF.0 := I
    COMPACT(CHARBUF)
```

```
    S := CHARBUF + 1
    [   LET LN = LINUM()
        LET NEWLINE = FINDLINE(LN)
        UNLESS NEWLINE = 0 v (RV NEWLINE ∧ AMASK) = 0 DO CERROR(
                =DUPLICATE LINE NUMBERS*N=)
    r   LET COM = COMPILE()
        NEWLINE := FINDLINE(LN)
        IF NEWLINE = 0 DO [ // ADD A NEW LINE
            NEWLINE := FREELINE
            FREELINE := FREELINE + 1
            IF FREELINE > FREE DO RERROR(= FIELD LENGTH TOO SMALL*N=) ]
// LINK THE LINE INTO THE PROGRAM
        TEST HILINE  = 0 DO // FIRST LINE EVER ENTERED
            HILINE,LOWLINE := NEWLINE,NEWLINE
        ELSE TEST LN > ( RV HILINE ↑ -36 ∧ 777777B) DO [1
// HIGHEST LINE IN THE PROGRAM
            RV HILINE := RV HILINE + (NEWLINE↑18)
            HILINE := NEWLINE ]1
        ELSE [W // ISSUE WARNING ABOUT ORDER
        WRITES(=WARNING - LINE NUMBER OUT OF ORDER*N=)
        TEST LN < (RV LOWLINE ↑ -36 ∧ 777777B) DO [2  //LOWEST LINE
            COM := COM + (LOWLINE↑18)
            LOWLINE := NEWLINE ]2
        ELSE [  // LINE TO BE INSERTED IN MIDDLE OF PROGRAM
            LET PRE = LOWLINE
            LET POST = FWDLINE(PRE)
            UNTIL (RV POST ↑ -36 ∧ 777777B) > LN DO [
                PRE := POST
                POST := FWDLINE(POST)  ]
            COM := COM + (POST ↑18)
            RV PRE := (RV PRE ∧ -AMASK↑18) + (NEWLINE↑18) ]W
        RV NEWLINE := COM + (LN↑36)  ]COMP
NEXT := -1 // DO NOT USE NEXT ANY MORE
// PROGRAM IS COMPILED SO EXECUTE IT
ENDREAD(INPUT)
PROG := LOWLINE
GIVESPACE(FREELINE,FREE-FREELINE)
// INITIALIZE THE TIME
RV 1 := RV TIMFET
[  IF RV 1 = 0 BREAK ] REPEAT
TIME := (RV RV TIMFET ∧ 7777B) + (RV RV TIMFET ↑ -12 ∧ 77777777B) * 1000
[   LASTPROG := PROG
    PROG := RV PROG↑-18∧ AMASK
    EXECUTE(RV LASTPROG)  ] REPEAT  ]BASIC
//DECK BCPLGD
//  DECLARATIONS TO LINK WITH HEAD

GLOBAL [ // TO LINK WITH FSP AND IO
        // GLOBALS 1-4 ARE FREE
        FLIST     :   5  // FREESPACE PACKAGE POINTER
        OUTPUT    :   6  // STANDARD OUTPUT STREAM
        C6TO7     :   7  // DISPLAY CODE ASCII CONVERSION VECTOR
        C7TO6     :   8  // ASCII DISPLAY CODE  CONVERSION VECTOR
        IOBASE    :   9  // OPEN STREAM LIST POINTER
        MONITOR   :  10
    ]
```

```
EXTERNAL [
          NEWVEC         // NEWVEC(SIZEWANTED)
          RETVEC         // RETVEC(V,SIZEOFV)
          PACKSTRING     // (VECTOR,STRING)
          UNPACKSTRING   // (STRING,VECTOR)
          INITIALIZEIO
          FINDINPUT      //  (EFNAME,USERNAMEE)
          CREATEOUTPUT   //  (EFNAME,USERNAMEE)
          READCH         // (STREAM,LV CH)
          WRITECH        // (STREAM,CH)
          WRITES         // OUTPUT:=STREAM; WRITES(STRING)
          WRITEN         // OUTPUT:=STREAM; WRITEN(NUMBER)
          WRITEO         // OUTPUT:=STREAM; WRITEO(NUMBER)
          READN          // NUMBER READER N:=READN(STREAM)
          TRACE
          TRACECALL
          TRACERETURN
          DUMP
          ENDREAD        // CLOSE INPUT STREAM
          ENDWRITE       // CLOSE OUTPUT STREAM
          IOERROR        // GENERAL IOERROR ROUTINE
          UNPACKTO       //  (STR,VEC,BREAKCH)
          WRITEVEC
          READVEC
          FLUSH          // FORCE OUT A LINE WITHOUT CR
          BCDWORD
          ASCII
          ENDOFSTREAM
          CLOSEALL
          ABORT
    ]

MANIFEST [ AMASK = 777777B  ; ENDOFSTREAMCH =144B
          BUFFERSIZE = 136
           ESCAPECH = 173B ]
    GLOBAL [GV     //  GLOBAL VARIABLES
INPUT                : 49
    FREE             : 50
    INPUTOPEN        : 51
    EXPLIMIT         : 52
// TWO HERE
    HILINE           : 54
    LOWLINE          : 55
    FREELINE         : 56
    DATA             : 57
    TMPDATA          : 58
    DATACOUNT        :59
    TMPDATACNT       :60
    RETURNS          : 61
    PROG             : 62
    LASTPROG         : 63
    EXPSTOP          :64
    S                : 65
    CHARCNT          :66
    SIG              : 67
```

```
    RUNNING         : 68
    TIME        :72
    NEXT            : 75
    NEXTINPUT       : 76
    ENDINPUT        : 77
    STNUMBER        : 78
    NUMBER          : 79
    ACCEPT          : 80
    RECURSE         : 82
    TYPE            : 84
// 88,89 RESERVED FOR MATHLIB
// HERE IS 2
    EXPTYPE         : 92
    CURSEED         : 93
    SEEDFLAG        : 94
    LIMIT           : 95
    IERROR          : 96
    RELEASE         : 97
        ]GV
EXTERNAL  [GB       // GLOBAL BUFFERS AND VECTORS
    VARIABLES
    FUNCTIONS
    ARRAYS
    LINES
    EXPCALLS
    CHARBUF
    STRINGS
    STARRAYS
    IOBUF
    INBUF
    GLOBALO
        ]GB
EXTERNAL  [GR        //  GLOBAL ROUTINES
    COMPILE
    ADDRESS
    EXPGEN
    LEXANA
    CONVERT
    WRITEF
    UNPACK
    PACK
    COMPACT
    LINUM
    FINDLINE
    COPY
    DATAOK
    DATASTORE
    DATAINIT
    ALPHA
    DIGIT
    INT
    FLOAT
    FMULT
    FDIV
    FADD
    CHKFDIV
```

```
    CHKPIV
    SIMPLEADD
    EXECUTE
    MATEX
    EXPRESS
    GIVESPACE  = FRETVECE
    GETSPACE   = ENEWVECE
    READLINE
    WRITE
    EERROR
    CERROR
    RERROR
    SYSERR
    EXCHARS
    UNREC
    FWDLINE
    FILL
    COPYSTRING
    ELE
    ELEB
    MATADD
    MATSUB
    MATSML
    INPROD
    NORM
    MATCMD
             ]GR
    MANIFEST  [MGC    //  SOME GENERAL CONSTANTS
    QUOTE              = ↓E↓
    AMASK              = 377777B
    NULLCH             = 140B
    NMASK              = -377777B
    NAMASK             = -377777B
    ZONESIZE           = 72/5
    LINESIZE           = 72
    ARITHEXP           = -1
    LOGEXP             = -2
    STRNGEXP           = -4
    HIADD              = 5000
    INDEF              = 60000000000000000000B
    CNE                = 17204000000000000000B
    TEN                = 17235000000000000000B
    RNDSEED            = 113434743B
    DELETE             = 3
    CHANGE             = 2
    INSERT             = 1
    DIRECT             = 4
    QUIT               = 5
    INTERUPTED = FALSE // SO THAT INTERUPT WILL BE DEFINED FOR MATEX
              ]MGC
MANIFEST    [MCMD      //  MANIFEST CONSTANTS FOR IDENTIFING LINES
    LETCMD   = 1
    GOTCMD   = 2
    IFCMD    = 3
    IFTCMD   = 3
    FORCMD   = 4
```

```
    NEXCMD   = 5
    REACMD   = 6
    INPCMD   = 7
    PRICMD   = 8
    DEFCMD   = 9
    REMCMD   = 10
    GOSCMD   = 12
    RETCMD   = 13
    ENDCMD   = 14
    STOCMD   = 15
    DATCMD   = 16
    RESCMD   = 17
    SIGCMD   = 18
    DIMCMD   = 19
    ONCMD    = 20
    MPRICMD  = 21
    MREACMD  = 22
    MTRNCMD  = 23
    MZERCMD  = 24
    MIDNCMD  = 25
    MCONCMD  = 26
    MADDCMD  = 27
    MSUBCMD  = 28
    MMLTCMD  = 29
    MSMLCMD  = 30
    MINVCMD  = 31
         ]MCMD
MANIFEST [MEC     // INDEXES INTO EXPCALLS
    ABS      = 1
    ATAN     = 2
    COSINE   = 3
    EXP      = 4
    INTEGER  = 5
    LOG      = 6
    SIGN     = 7
    SINE     = 8
    SQRT     = 9
    TAN      = 10
    RND      = 11
    ACS      = 12
    ASN      = 13
    LGT      = 14
    TIM      = 15
    ENDEXP   = 8 + 12
    ADD      = 8 + 13
    SUBTRAC  = 8 + 14
    MULT     = 8 + 15
    DIV      = 8 + 16
    NEG      = 8 + 17
    ANDCALL  = 8 + 18
    ORCALL   = 8 + 19
    EQUAL    = 8 + 20
    NEQUAL   = 8 + 21
    LESSTH   = 8 + 22
    LESS     = 8 + 22
    MORETH   = 8 + 23
```

```
    MORE     = 8 + 23
    LESSEQ   = 8 + 24
    MOREEQ   = 8 + 25
    STORE    = 8 + 26
    FETCH    = 8 + 27
    NMCNST   = 8 + 28
    ADCNST   = 8 + 29
    ONECNST  = 8 + 30
    FNCALL   = 8 + 31
    FRET     = 8 + 32
    PARVAL   = 8 + 33
    ARYCONT  = 8 + 34
    ARYADD   = 8 + 35
    EXOPNET  = 8 + 36
    EXPO     = 8 + 36
    LSTCONT  = 8 + 37
    LSTADD   = 8 + 38
    STLSTC   = 8 + 39
    STLSTA   = 8 + 40
    STOREST  = 8 + 41
    FETCHST  = 8 + 42
    STEQUAL  = 8 + 43
    STCNST   = 8 + 44
    STNEQUAL = 8 + 45
    PICNST   = 8 + 46
    STMORETH = 8 + 47
    STMOREEQ = 8 + 48
    STLESSTH = 8 + 49
    STLESSEQ = 8 + 50
          ]MEC
MANIFEST  [MPT    //PARSE TYPE CONSTANTS
    PTPLUS    = 1
    PTMULT   = 2
    PTEXPO   = 3
    PTROP    = 4
    PTCM     = 5
    PTRPREN  = 6
    PTLPREN  = 7
    PTAND    = 8
    PTOR     = 9
    PTNOT    = 10
    PTFNC    = 11
    PTID     = 12
    PTARY    = 13
    PTEND    = 14
          ]MPT
MANIFEST    [MGT     //  GENERATING TYPE CONSTANTS
    GTPLUS   = 1
    GTMINUS  = 2
    GTADP    = 3
    GTROP    = 4
    GTLOP    = 5
    GTCM     = 6
    GTRPREN  = 7
    GTLPREN  = 8
    GTNUM    = 9
```

```
    GTFNC     = 10
    GTARY     = 11
    GTVAR     = 12
    GTEND     = 14
    GTNOT     = 15
    GTSTLST   = 16
    GTSTVAR   = 17
    GTSTNUM   = 18
    GTPI      = 21
          ]MGT
MANIFEST [ ENDMARK  = (PTEND ASHIFT 53)+(GTEND ASHIFT 46) ]
EXTERNAL [ GETSPACE = ≡COMSPACE≡  ]
LET COMSPACE(N) = VALOF [CS
FREE := FREE-N
IF FREE < FREELINE DO RERROR(≡ FIELD LENGTH TOO SMALL*N≡)
RESULTIS FREE  ]CS
LET COMPILE() = VALOF [COM
LET V = VEC 80
LET SIZE,BLK = 1,0
LET TYPE =(S.0 * 7)+S.1
    EXPLIMIT := V + 80
    S := S + 2
    SWITCHON TYPE INTO [SW2
    CASE ↓IF↓ :  [IF
        LET PT = S
        PT := PT + 1  REPEATUNTIL PT.0 = ↓*N↓ LOGOR
            (PT.0=↓T↓∧PT.1=↓H↓∧PT.2=↓E↓∧PT.3=↓N↓) LOGOR
            (PT.0=↓G↓∧PT.1=↓O↓∧PT.2=↓T↓∧PT.3=↓O↓)
        IF PT.0 = ↓*N↓ THEN CERROR(≡THEN IS MISSING*N≡)
        SIZE := EXPGEN(V+1,PT,LOGEXP,0) +1
        S := PT + 4
        BLK := GETSPACE(SIZE)
        COPY(V,BLK,SIZE)
        TYPE := IFCMD
        GOTO LINEREF  ]IF
    CASE ↓CN↓ :  [CN
        LET PT = S
        LET CNT = 0
        PT := PT + 1 REPEATUNTIL RV PT = ↓*N↓ ∨
            (PT.0=↓G↓ ∧ PT.1=↓O↓ ∧ PT.2=↓T↓ ∧ PT.3=↓O↓)
        IF PT.0 = ↓*N↓ DO CERROR(≡GOTO MISSING*N≡)
        SIZE := EXPGEN(V+1,PT,ARITHEXP,0) +1
        S := S +4
        [1  LET LN = LINUM()
            LET LINE = FINDLINE(LN)
            UNLESS RV S = ↓,↓ ∨ RV S = ↓*N↓ DO
                CERROR(≡IMPROPER CHARACTER*N≡)
            IF LINE = 0 DO [2
                IF FREELINE GE FREE DO RERROR(
                    ≡ FIELD LENGTH TOO SMALL*N≡)
                LINE := FREELINE
                FREELINE := FREELINE +1
                RV LINE := LN * 36 ]2
            V.SIZE := LINE
            SIZE := SIZE +1
            CNT := CNT + 1
```

```
              IF PV S = ↓*N↓ BREAK
              S := S + 1   ]1 REPEAT
          V.0 := (SIZE↑ 18) + (CNT ↑ 36)
          BLK := GETSPACE(SIZE)
          COPY(V,BLK,SIZE)
          RESULTIS (ONCMD ↑ 54) +. BLK
      ]SW2
TYPE :=(TYPE ↑ 7) + S.0
S := S + 1
SWITCHON TYPE INTO [SW
CASE  ↓LET↓  :    [L
     LET PT,CT  = S,0
     LET ASNTYPE = 0
[1       PT := PT + 1 REPEATUNTIL RV PT = ↓=↓ ∨ RV PT = ↓*N↓ ∨ RV PT = ↓Ξ↓
          IF RV PT ≠ ↓=↓  THEN BREAK
          SIZE := SIZE + ADDPESS(V+SIZE) - 1
          ASNTYPE := ASNTYPE ∨ EXPTYPE
          UNLESS S = PT DO CERROR(ΞILLEGAL CHARACTER*NΞ)
          S := S + 1
          CT := CT + 1   ]1  REPEAT
     IF CT = 0 DO CERROR(ΞNO ASSIGNMENT OPERATOR*NΞ)
     UNLESS ASNTYPE = ARITHEXP ∨ ASNTYPE = STRNGEXP DO CERROR(
          ΞASSIGNED TO BOTH STRING AND NUMBER VARIABLES*NΞ)
     PT := EXPCALLS.(ASNTYPE = STRNGEXP -> STOREST,STORE)
     SIZE := SIZF + EXPGEN(V+SIZE,0,ASNTYPE,0) - 1
     FOR I = SIZE + V TO V + SIZE + CT - 1 DO
          I.0 := PT
     V.(SIZE+CT) := EXPCALLS.ENDEXP +. (ASNTYPE LSHIFT 18)
     SIZE := SIZE + CT + 1
     V.0 := SIZE ↑ 18
     BLK := GETSPACE(SIZF)
     COPY(V,BLK,SIZE)
     RESULTIS   (LETCMD ↑ 54)+.BLK    ]L
CASE ↓FOR↓ :  [F
     LET AD = SIMPLEADD()
     LET PT = S + 1
     UNLESS RV S = ↓=↓ ∧ EXPTYPE = ARITHEXP DO CERROR(
               ΞSIMPLE VARIABLE REQUIRED*NΞ)
     S := S + 1
     PT := PT + 1 REPEATUNTIL  (PT.0 = ↓T↓ ∧ PT.1 = ↓O↓)∨ PT.0 = ↓*N↓
     IF RV PT = ↓*N↓ DO CERROR(ΞTO IS MISSING*NΞ)
     SIZE := 1 + EXPGEN(V+1,PT,ARITHEXP,0)
     S:= PT + 2
     PT := PT + 1 REPEATUNTIL (PT.0 = ↓S↓ ∧ PT.1 = ↓T↓ ∧ PT.2 = ↓E↓
                    ∧ PT.3 = ↓P↓) ∨ PT.0 = ↓*N↓
     SIZE := SIZE + EXPGEN(V+SIZE,PT,ARITHEXP,0)
     TEST RV PT = ↓*N↓ DO [
          V.SIZE := EXPCALLS.ONECNST
          V.(SIZE+1) := EXPCALLS.ENDEXP
          SIZE := SIZE + 2   ]  ELSE  [
          S := PT + 4
          SIZE := SIZE + EXPGEN(V+SIZE,0,ARITHEXP,0)  ]
     V.0 :=(AD ↑ 36)+(SIZE ↑ 18)
     BLK := GETSPACE(SIZE)
     COPY(V,BLK,SIZE)
     RESULTIS(FORCMD ↑ 54)+.BLK    ]F
```

```
CASE ↓NEX↓ :    [N
    LET AD = 0
    UNREC(S.0 ≠ ↓T↓)
    S := S + 1
    AD := SIMPLEADD()
    IF RV S = ↓(↓ ∨ EXPTYPE = STRNGEXP DO CERROR(
         ≡SIMPLE VARIABLE REQUIRED*N≡)
    EXCHARS()
    BLK := GETSPACE(4)
    BLK.1,BLK.2,BLK.3 := 0,0,0
    BLK.0 :=(4 ↑ 18)+.(AD ↑ 36 )
    RESULTIS(NEXCMD ↑ 54)+.BLK     ]N
CASE ↓REA↓ :
    UNREC(S.0 ≠ ↓D↓)
RI1 :  [RI  LET COUNT = 0
//  FORM THE EXPRESSIONS IN V
  [     S := S + 1
       SIZE := ADDRESS(V+SIZE) + SIZE
       COUNT := COUNT + 1
       IF RV S = ↓*N↓ BREAK
       UNLESS RV S = ↓,↓ DO CERROR(≡ILLEGAL CHARACTER*N≡)
       ] REPEAT
    V.0 :=(COUNT ↑ 36)+(SIZE ↑ 18)
    BLK := GETSPACE(SIZE)
    COPY(V,BLK,SIZE)
    RESULTIS((TYPE = ↓REA↓ ->  REACMD,INPCMD) ↑ 54)+.BLK
CASE ↓INP↓ :
    UNREC((S.0 ↑ 7)+ S.1 ≠ ↓UT↓)
    S := S + 1
    GOTO RI1
CASE ↓DEF↓ :
    UNLESS S.0 = ↓F↓ ∧ S.1 = ↓N↓ ∧ S.3 = ↓(↓ ∧ ALPHA(S.2) DO
       CERROR(≡BAD FUNCTION NAME*N≡)
[D LET ENT = S.2 - ↓A↓ + FUNCTIONS
    S := S + 4
 [ LET AD = SIMPLEADD()
    IF EXPTYPE = STRNGEXP DO CERROR(≡PARAMETER CAN NOT BE A STRING*N≡)
    UNLESS(S.0 ↑ 7)+ S.1 = ↓)=↓ DO CERROR(≡BAD PARAMETER*N≡)
    S := S + 2
    SIZE := EXPGEN(V+1,0,ARITHEXP,AD) + 1
    V.(SIZE - 1) := EXPCALLS.FRET
    V.0 :=(ENT↑ 36)+((SIZE) ↑18)
    BLK := GETSPACE(SIZE)
    COPY(V,BLK,SIZE)
    RESULTIS(DEFCMD ↑ 54)+.BLK   ]D
CASE ↓PRI↓ :
    UNREC((S.0 ↑ 7)+ S.1 ≠ ↓NT↓)
    S := S + 2
PRI1: SWITCHON RV S INTO  [PR
    CASE ↓,↓ :
       V.SIZE := ↓*T↓
       GOTO SEP
    CASE ↓;↓ :
       V.SIZE := ↓ ↓
    SEP: SIZE := SIZE + 1
    CASE ↓:↓ :
```

```
          S := S + 1
          IF RV S = ↓*N↓ GOTO PRI2
          GOTO PRI1
     CASE QUOTE :
          V.SIZE := QUOTE
          SIZE := 1 + SIZE + COPYSTRING(V + SIZE + 1,CERROR)
          GOTO PRI1
     DEFAULT : //PRINT AN EXPRESSION OR TAB
     TEST (S.0↑21) +. (S.1↑14) +. (S.2↑7) +. S.3 = ↓TAB(↓ DO [T
          V.SIZE := ↓T↓
          S := S+4
          SIZE := EXPGEN(V+SIZE+1,-3,ARITHEXP,0) + SIZE + 1
          IF RV S ≠ ↓)↓ DO CERROR(≡RIGHT PARENTHESIS MISSING*N≡)
          S := S + 1 ]T  ELSE
     SIZE := SIZE + EXPGEN(V+SIZE,-2,ARITHEXP ∧ STRNGEXP,0)
     GOTO PRI1
     CASE ↓*N↓ :
     V.SIZE := ↓*N↓
     SIZE := SIZE + 1   ]PR
PRI2:
     V.0 := SIZE ↑ 18
     BLK := GETSPACE(SIZE)
     COPY(V,BLK,SIZE)
     RESULTIS(PRICMD ↑ 54)+.BLK
CASE ↓REM↓ :
     BLK := GETSPACE(1)
     RV BLK := (1↑18)
     RESULTIS( REMCMD ↑ 54) +.BLK
CASE ↓GOT↓ :
     UNREC(S.0 ≠ ↓O↓)
     S := S + 1
     TYPE := GOTCMD
     BLK := GETSPACE(1)
     GOTO  LINEREF
CASE ↓GOS↓ :
     UNREC((S.0 ↑ 7)+ S.1 ≠ ↓UB↓)
     S := S + 2
     TYPE := GOSCMD
     BLK := GETSPACE(1)
     GOTO  LINEREF
CASE ↓RET↓ :
   UNREC((S.0 ↑ 14)+(S.1 ↑ 7)+ S.2 ≠. ↓URN↓ )
     S := S + 3
     EXCHARS()
     BLK := GETSPACE(1)
     RV BLK := (1↑18)
     RESULTIS (RETCMD ↑ 54)+.BLK
CASE ↓STO↓ :
     UNREC(S.0≠↓P↓)
     S := S + 1
     EXCHARS()
     BLK := GETSPACE(1)
     RV BLK := (1↑18)
     RESULTIS (STOCMD ↑ 54)+.BLK
CASE ↓END↓ :
     EXCHARS()
```

```
    BLK := GETSPACE(1)
    RV BLK := (1↑18)
    RESULTIS (ENDCMD ↑ 54)+.BLK
CASE ↓RES↓ :
   UNREC((S.0 ↑ 21)+(S.1 ↑ 14)+(S.2↑7)+ S.3 ≠. ↓TORE↓)
    S := S + 4
    EXCHARS()
    BLK := GETSPACE(1)
    RV BLK := (1↑18)
    RESULTIS (RESCMD ↑ 54)+.BLK
CASE ↓DAT↓ :
    UNREC(S.0 ≠ ↓A↓)
    [3  LET TS = S + 1
        S := TS
        TEST RV S = QUOTE ∨ ALPHA(RV S) DO [1
            IF COPYSTRING(V+SIZE+1,CERROR) > 2 DO CERROR(
               ≡STRING TOO BIG*N≡)
            V.SIZE := INDEF
            SIZE := SIZE + 3  ]1
            ELSE [2  V.SIZE := CONVERT(CERROR)
            SIZE := SIZE + 1  ]2
        IF S = TS DO CERROR(≡VALUE MISSING*N≡)
             ]3 REPEATWHILE RV S = ↓,↓
    UNLESS RV S = ↓*N↓ DO CERROR(≡IMPROPER CHARACTER*N≡)
    V.0 := SIZE ↑ 18
    BLK := GETSPACE(SIZE)
    COPY(V,BLK,SIZE)
    RESULTIS (DATCMD ↑ 54)+.BLK
CASE ↓SIG↓ :
    SIZE := EXPGEN(V+1,0,ARITHEXP,0)+1
    V.0 := SIZE ↑ 18
    BLK := GETSPACE(SIZE)
    COPY(V,BLK,SIZE)
    RESULTIS (SIGCMD ↑ 54) +. BLK
CASE ↓DIM↓ :
[D LET COUNT = 0
    [1 LET BASE = SIZE
        LET STRNG = S.1 = ↓$↓
        LET DIMCNT = 0
        LET AD = S.0 - ↓A↓ +(STRNG-> STARRAYS,  ARRAYS)
        UNLESS ALPHA(S.0) ∧ S.(STRNG->2,1) = ↓(↓ DO CERROR(
          ≡BAD ARRAY NAME*N≡)
        S := S + (STRNG-> 3,2)
        SIZE := SIZE + 1
//  GET THE ARRAY FUNCTION
        [2 SIZE := EXPGEN(V+SIZE,-3,ARITHEXP,0) + SIZE
           DIMCNT := DIMCNT + 1
            IF RV S = ↓)↓ THEN BREAK
            IF RV S = ↓*N↓ DO CERROR(≡MISSING RIGHT PARENTHESIS*N≡)
            UNLESS RV S = ↓,↓ DO CERROR(≡IMPROPER CHARACTER*N≡)
            S := S + 1
        ]2 REPEAT
        IF STRNG ∧ DIMCNT > 1 DO CERROR(
              ≡STRING ARRAYS MUST BE ONE DIMENSION*N≡)
        V.BASE :=(DIMCNT ↑ 18)+.AD +. (STRNG -> 1↑59,0)
        COUNT := COUNT + 1
```

```
          IF S.1 = ↓*N↓ THEN BREAK
          UNLESS  S.1 = ↓,↓ THEN  CERROR(ΞIMPROPER CHARACTER*NΞ)
      S := S + 2
             ]1 REPEAT
      V.0 :=(COUNT ↑ 36)+(SIZE ↑ 18)
      BLK := GETSPACE(SIZE)
      COPY(V,BLK,SIZE)
      RESULTIS (DIMCMD ↑ 54)+.BLK  ]D
  CASE ↓MAT↓:
      UNLESS S.1 = ↓=↓ DO SWITCHON (S.0↑14)+(S.1↑7)+S.2 INTO [MS
      CASE ↓PRI↓ :
          UNLESS (S.3↑7) + S.4 = ↓NT↓ DO CERROR(
                  ΞUNRECOGNIZABLE MATRIX COMMAND*NΞ)
          UNLESS ALPHA(S.5) DO CERROR(ΞBAD MATRIX NAME*NΞ)
          [P  LET ARYC = S.5 - ↓A↓ + ARRAYS
          SWITCHON S.6 INTO [SP
              CASE ↓*N↓ : TYPE := 0
                  GOTO MPR1
              CASE ↓,↓ : TYPE := 1
                  GOTO MPR2
              CASE ↓;↓ : TYPE := 2
                  GOTO MPR2
              CASE ↓:↓ : TYPE := 3
              MPR2:    S := S + 7
                      EXCHARS()
                      GOTO MPR1
              DEFAULT : EXCHARS()  ]SP
      MPR1 :  BLK := GETSPACE(1)
              RV BLK := (TYPE↑54) +.(ARYC↑36) +.(1↑18)
              RESULTIS (MPRICMD↑54) +. BLK  ]P
          CASE ↓REA↓ :
              UNLESS S.3=↓D↓ DO CERROR(ΞUNRECOGNIZABLE MATRIX COMMAND*NΞ)
              UNLESS ALPHA(S.4) ∧ S.5 = ↓*N↓ DO CERROR(
                      ΞBAD MATRIX NAME*NΞ)
              BLK := GETSPACE(1)
              RV BLK := ((S.4 - ↓A↓ + ARRAYS) ↑ 36) +. (1↑18)
              RESULTIS (MREACMD ↑ 54) +. BLK
          DEFAULT : CERROR(ΞUNRECOGNIZABLE MATRIX COMMAND*NΞ) ]MS
// HANDEL A MATRIX ASSIGNMENT
      UNLESS ALPHA(S.0) DO CERROR(ΞASSIGNED TO BAD MATRIX NAME*NΞ)
      [MA LET ARYC = S.0 - ↓A↓ + ARRAYS
          IF S.2 = ↓(↓ DO [1
              S := S + 3
              SIZE := EXPGEN(V+2,-3,ARITHEXP,0) + 2
              UNLESS S.0=↓)↓ ∧ S.1=↓**↓ ∧ ALPHA(S.2) DO CERROR( VALOF[2
                  IF S.0 = ↓,↓ RESULTIS ΞIMPROPER USE OF COMMA*NΞ
                  IF S.0 = ↓*N↓RESULTIS ΞRIGHT PARENTHESIS MISSING*NΞ
                  UNLESS S.0 = ↓)↓ RESULTIS ΞIMPROPER CHARACTER*NΞ
                  IF S.2 = ↓*N↓ ∧ ALPHA(S.1) DO CERROR(
                      Ξ** MISSING*NΞ)
                  UNLESS S.3 = ↓*N↓ RESULTIS
                          ΞSCALAR EXPRESSION ENDED TOO SOON*NΞ
                  RESULTIS ΞBAD ARRAY NAME*NΞ ]2  )
              S := S+3
              EXCHARS()
              V.1 := S.-1 - ↓A↓ + ARRAYS
```

```
        V.0 := (ARYC↑36) +. (SIZE ↑ 18)
        BLK := GETSPACE(SIZE)
        COPY(V,BLK,SIZE)
        RESULTIS (MSMLCMD ↑ 54) +. BLK ]I
    SWITCHON S.3 INTO [O
    CASE ↓**↓ : TYPE := MMLTCMD; GOTO MT3
    CASE ↓+↓ : TYPE := MADDCMD; GOTO MT3
    CASE ↓-↓ : TYPE := MSUBCMD
    MT3:    UNLESS ALPHA(S.2) ∧ ALPHA(S.4) DO CERROR(
                ≡BAD ARRAY NAME*N≡)
    [   LET ARYA =S.2 - ↓A↓ + ARRAYS
        LET ARYB =S.4 - ↓A↓ + ARRAYS
        S := S + 5
        EXCHARS()
        BLK := GETSPACE(2)
        BLK.0 := (ARYC↑36) +. (2↑18)
        BLK.1 := (ARYA↑18) +. ARYB
        RESULTIS (TYPE↑54) +. BLK ]O
    UNLESS ALPHA(S.3) DO CERROR(≡BAD MATRIX OPERATION*N≡)
    SWITCHON (S.2↑14) +. (S.3↑7) +. S.4 INTO [F
    DEFAULT: CERROR(≡BAD MATRIX FUNCTION*N≡)
        CASE ↓TRN↓ : TYPE := MTRNCMD; GOTO MT2
        CASE ↓INV↓ : TYPE := MINVCMD
        MT2:    UNLESS S.5 = ↓(↓ ∧ S.7 = ↓)↓ ∧ ALPHA(S.6) DO
                    CERROR(≡BAD ARGUMENT*N≡)
            S := S+8
            EXCHARS()
            BLK := GETSPACE(2)
            BLK.0 := (ARYC↑36) +. (2↑18)
            BLK.1 := S.-2 - ↓A↓ + ARRAYS
            RESULTIS (TYPE↑54) +. BLK
        CASE ↓ZER↓ : TYPE := MZERCMD; GOTO MT1
        CASE ↓IDN↓ : TYPE := MIDNCMD; GOTO MT1
        CASE ↓CON↓ : TYPE := MCONCMD
        MT1:S := S + 5
            EXCHARS()
            BLK := GETSPACE(1)
            RV BLK := (ARYC ↑ 36) +. (1↑18)
            RESULTIS (TYPE↑54) +. BLK ]F
    ]MA
DEFAULT : UNREC(TRUE)    ]SW
LINEREF :
   [ LET LIN = LINUM()
    LET PTL = FINDLINE(LIN)
    EXCHARS()
    IF PTL = 0 DO [
        IF FREELINE GE FREE DO RERROR(≡ FIELD LENGTH TOO SMALL*N≡)
        RV FREELINE := LIN ↑ 36
        PTL := FREELINE
        FREELINE := FREELINE + 1  ]
  RV BLK := (SIZE ↑ 18) +. (PTL ↑ 36)
    RESULTIS(TYPE ↑ 54)+.BLK   ]COM
//DECK BCPLGD
//  DECLARATIONS TO LINK WITH HEAD

GLOBAL [ // TO LINK WITH FSP AND IO
```

```
        // GLOBALS 1-4 ARE FREE
        FLIST      :   5  // FREESPACE PACKAGE POINTER
        OUTPUT     :   6  // STANDARD OUTPUT STREAM
        C6TO7      :   7  // DISPLAY CODE ASCII CONVERSION VECTOR
        C7TO6      :   8  // ASCII DISPLAY CODE  CONVERSION VECTOR
        IOBASE     :   9  // OPEN STREAM LIST POINTER
        MONITOR    :  10
    ]

EXTERNAL [
        NEWVEC          // NEWVEC(SIZEWANTED)
        RETVEC          // RETVEC(V,SIZEOFV)
        PACKSTRING      // (VECTOR,STRING)
        UNPACKSTRING    // (STRING,VECTOR)
        INITIALIZEIO
        FINDINPUT       //  (EFNAME,USERNAMEE)
        CREATEOUTPUT    //  (EFNAME,USERNAMEE)
        READCH          // (STREAM,LV CH)
        WRITECH         // (STREAM,CH)
        WRITES          // OUTPUT:=STREAM; WRITES(STRING)
        WRITEN          // OUTPUT:=STREAM; WRITEN(NUMBER)
        WRITEO          // OUTPUT:=STREAM; WRITEO(NUMBER)
        READN           // NUMBER READER N:=READN(STREAM)
        TRACE
        TRACECALL
        TRACERETURN
        DUMP
        ENDREAD         // CLOSE INPUT STREAM
        ENDWRITE        // CLOSE OUTPUT STREAM
        IOERROR         // GENERAL IOERROR ROUTINE
        UNPACKTO        //  (STR,VEC,BREAKCH)
        WRITEVEC
        READVEC
        FLUSH           // FORCE OUT A LINE WITHOUT CR
        BCDWORD
        ASCII
        ENDOFSTREAM
        CLOSEALL
        ABORT
    ]

MANIFEST [ AMASK = 777777B  : ENDOFSTREAMCH =144B
        BUFFERSIZE = 136
         ESCAPECH = 173B ]
    GLOBAL [GV    //  GLOBAL VARIABLES
INPUT              : 49
    FREE           : 50
    INPUTOPEN      : 51
    EXPLIMIT       : 52
// TWO HERE
    HILINE         : 54
    LOWLINE        : 55
    FREELINE       : 56
    DATA           : 57
    TMPDATA        : 58
    DATACOUNT      :59
```

```
    TMPDATACNT        :60
    RETURNS           : 61
    PROG              : 62
    LASTPROG          : 63
    EXPSTOP           :64
    S                 : 65
    CHARCNT           :66
    SIG               : 67
    RUNNING           : 68
    TIME          :72
    NEXT              : 75
    NEXTINPUT         : 76
    ENDINPUT          : 77
    STNUMBER          : 78
    NUMBER            : 79
    ACCEPT            : 80
    RECURSE           : 82
    TYPE              : 84
// 88,89 RESERVED FOR MATHLIB
// HERE IS 2
    EXPTYPE           : 92
    CURSEED           : 93
    SEEDFLAG          : 94
    LIMIT             : 95
    IERROR            : 96
    RELEASE           : 97
          ]GV
EXTERNAL  [GB      // GLOBAL BUFFERS AND VECTORS
    VARIABLES
    FUNCTIONS
    ARRAYS
    LINES
    EXPCALLS
    CHARBUF
    STRINGS
    STARRAYS
    IOBUF
    INBUF
    GLOBALO
        ]GB
EXTERNAL  [GR       //  GLOBAL ROUTINES
    COMPILE
    ADDRESS
    EXPGEN
    LEXANA
    CONVERT
    WRITEF
    UNPACK
    PACK
    COMPACT
    LINUM
    FINDLINE
    COPY
    DATAOK
    DATASTORE
    DATAINIT
```

```
ALPHA
DIGIT
INT
FLOAT
FMULT
FDIV
FADD
CHKFDIV
CHKPIV
SIMPLEADD
EXECUTE
MATEX
EXPRESS
GIVESPACE  = ERETVECE
GETSPACE   = ENEWVECE
READLINE
WRITE
EERROR
CERROR
RERROR
SYSERR
EXCHARS
UNREC
ENDLINE
FILL
COPYSTRING
ELE
ELEB
MATADD
MATSUB
MATSML
INPROD
NORM
MATCMD
           IGR
MANIFEST   EMGC     //  SOME GENERAL CONSTANTS
QUOTE                =  ↓E↓
AMASK                =  377777B
NULLCH               =  140B
NMASK                =  -377777B
NAMASK               =  -377777B
ZONESIZE             =  72/5
LINESIZE             =  72
ARITHEXP             =  -1
LOGEXP               =  -2
STRNGEXP             =  -4
HIADD                =  5000
INDEF                =  60000000000000000000B
ONE                  =  17204000000000000000B
TEN                  =  17235000000000000000B
RNDSEED              =  113434743B
DELETE               =  3
CHANGE               =  2
INSERT               =  1
DIRECT               =  4
QUIT                 =  5
```

```
    INTERUPTED = FALSE // SO THAT INTERUPT WILL BE DEFINED FOR MATEX
                 ]MGC
MANIFEST    [MCMD       //  MANIFEST CONSTANTS FOR IDENTIFING LINES
    LETCMD   = 1
    GOTCMD   = 2
    IFCMD    = 3
    IFTCMD   = 3
    FORCMD   = 4
    NEXCMD   = 5
    REACMD   = 6
    INPCMD   = 7
    PRICMD   = 8
    DEFCMD   = 9
    REMCMD   = 10
    GOSCMD   = 12
    RETCMD   = 13
    ENDCMD   = 14
    STOCMD   = 15
    DATCMD   = 16
    RESCMD   = 17
    SIGCMD   = 18
    DIMCMD   = 19
    CHCMD    = 20
    MPRICMD  = 21
    MREACMD  = 22
    MTRNCMD  = 23
    MZERCMD  = 24
    MIDNCMD  = 25
    MCONCMD  = 26
    MADDCMD  = 27
    MSUBCMD  = 28
    MMLTCMD  = 29
    MSMLCMD  = 30
    MINVCMD  = 31
          ]MCMD
MANIFEST [MFC    // INDEXES INTO EXPCALLS
    ABS      = 1
    ATAN     = 2
    COSINE   = 3
    EXP      = 4
    INTEGER  = 5
    LOG      = 6
    SIGN     = 7
    SINE     = 8
    SQRT     = 9
    TAN      = 10
    RND      = 11
    ACS      = 12
    ASN      = 13
    LGT      = 14
    TIM      = 15
    ENDEXP   = 8 + 12
    ADD      = 8 + 13
    SUBTRAC  = 8 + 14
    MULT     = 8 + 15
    DIV      = 8 + 16
```

```
    NEG       = 8 + 17
    ANDCALL   = 8 + 18
    ORCALL    = 8 + 19
    EQUAL     = 8 + 20
    NEQUAL    = 8 + 21
    LESSTH    = 8 + 22
    LESS      = 8 + 22
    MORETH    = 8 + 23
    MORE      = 8 + 23
    LESSEQ    = 8 + 24
    MOREEQ    = 8 + 25
    STORE     = 8 + 26
    FETCH     = 8 + 27
    NMCNST    = 8 + 28
    ADCNST    = 8 + 29
    UNFCNST   = 8 + 30
    FNCALL    = 8 + 31
    FRET      = 8 + 32
    PARVAL    = 8 + 33
    ARYCONT   = 8 + 34
    ARYADD    = 8 + 35
    EXOPNET   = 8 + 36
    EXPO      = 8 + 36
    LSTCONT   = 8 + 37
    LSTADD    = 8 + 38
    STLSTC    = 8 + 39
    STLSTA    = 8 + 40
    STOREST   = 8 + 41
    FETCHST   = 8 + 42
    STEQUAL   = 8 + 43
    STCNST    = 8 + 44
    STNEQUAL  = 8 + 45
    PICNST    = 8 + 46
    STMORETH  = 8 + 47
    STMOREEQ  = 8 + 48
    STLESSTH  = 8 + 49
    STLESSEQ  = 8 + 50
          ]MEC
MANIFEST  [MPT    //PARSE TYPE CONSTANTS
    PTPLUS     = 1
    PTMULT    = 2
    PTEXPO    = 3
    PTROP     = 4
    PTCM      = 5
    PTRPREN   = 6
    PTLPREN   = 7
    PTAND     = 8
    PTOR      = 9
    PTNOT     = 10
    PTFNC     = 11
    PTID      = 12
    PTARY     = 13
    PTEND     = 14
          ]MPT
MANIFEST    [MGT    //  GENERATING TYPE CONSTANTS
    GTPLUS    = 1
```

```
    GTMINUS   = 2
    GTAOP     = 3
    GTROP     = 4
    GTLOP     = 5
    GTCM      = 6
    GTRPREN   = 7
    GTLPREN   = 8
    GTNUM     = 9
    GTFNC     = 10
    GTARY     = 11
    GTVAR     = 12
    GTEND     = 14
    GTNOT     = 15
    GTSTLST   = 16
    GTSTVAR   = 17
    GTSTNUM   = 18
    GTPI      = 21
          ]MGT
MANIFEST [ ENDMARK   = (PTEND ASHIFT 53)+(GTEND ASHIFT 46) ]
MANIFEST [MPC//PARSETABLE CONSTANTS
      L = 1        //  LESS
      G = 2        //  GREATER
      E = 3        //  EQUAL
      K = 4        //  INCREMENT DIM COUNT
      D = 5        //  DONE WITH THE EXPRESSION
      N = 6        //  IMPROPER USE OF NOT
      C = 7        //  IMPROPER USE OF COMMA
      R = 8        //  MISSING RIGHT RIGHT PREN
      M = 9        //  MISSING LEFT PREN
      O = 10       //  OPERATOR MISSING
      Z = 11       //  COMMA AT ZERO LEVEL
      B = 12       //  IMPRELOP
      P = 13       //  MORE THAN ONE PARAMETER
        ]MPC
LET EXPGEN(V,LAST,TYPE,PAR) = VALOF [EG
// V IS VECTOR TO STORE CODE IN
// GLOBAL EXPLIMIT CONTAINS THE LAST CELL THAT CAN BE WRITTEN IN
// LAST GIVES ADDRESS OF CHARACTER NOT TO INCLUDE OR STOPING CONDITION
//  LAST   TERMINATING CONDITION
//    0       *N
//   -1       *N ,
//   -2       *N  , ≡ ;
//   -3       *N  , ≡ ; )
//TYPE IS THE TYPE OF EXPRESSION REQUIRED
// PAR IS THE ADDRESS  OF THE PARAMETER OF A FUNCTION
// BEWARE IF YOU ARE EXPANDING STRING CAPABILITY - A QUOTE
//     CAN TERMINATE AN EXPRESSION
LET TL() = CERROR(≡IMPROPER OPERATOR*N≡)
LET EM() = CERROR(≡ EXPRESSION IS MISSING*N≡)
LET PARSETABLE = TABLE 0
,(G↑56) + (L↑52) + (L↑48) + (G↑44) + (G↑40) +(G↑36) + (L↑32) +(G↑28) +
 (G↑24) + (G↑20) + (L↑16) + (L↑12) + (L↑8) + (G↑4)+ (G) // PTPLUS
,(G↑56) + (G↑52) + (L↑48) + (G↑44) + (G↑40) +(G↑36) + (L↑32) +(G↑28) +
 (G↑24) + (G↑20) + (L↑16) + (L↑12) + (L↑8) + (G↑4)+ (G) // PTMULT
,(G↑56) + (G↑52) + (G↑48) + (G↑44) + (G↑40) +(G↑36) + (L↑32) +(G↑28) +
 (G↑24) + (G↑20) + (L↑16) + (L↑12) + (L↑8) + (G↑4)+ (G) // PTEXPO
```

```
,(L↑56) + (L↑52) + (L↑48) + (B↑44) + (C↑40) +(G↑36) + (L↑32) +(G↑28) +   EXPGEN
 (G↑24) + (N↑20) + (L↑16) + (L↑12) + (L↑8) + (G↑4)+ (G) // PTROP          EXPGEN
,(O↑56) + (O↑52) + (O↑48) + (O↑44) + (O↑40) +(O↑36) + (O↑32) +(O↑28) +   EXPGEN
 (O↑24) + (O↑20) + (O↑16) + (O↑12) + (C↑8) + (O↑4)+ (O) // PTCM           EXPGEN
,(O↑56) + (O↑52) + (O↑48) + (O↑44) + (O↑40) +(O↑36) + (O↑32) +(O↑28) +   EXPGEN
 (O↑24) + (O↑20) + (O↑16) + (O↑12) + (O↑8) + (O↑4)+ (O) // PTRPREN        EXPGEN
,(L↑56) + (L↑52) + (L↑48) + (L↑44) + (C↑40) +(E↑36) + (L↑32) +(L↑28) +   EXPGEN
 (L↑24) + (L↑20) + (L↑16) + (L↑12) + (L↑8) + (R↑4)+ (R) // PTLPREN        EXPGEN
,(L↑56) + (L↑52) + (L↑48) + (L↑44) + (C↑40) +(G↑36) + (L↑32) +(G↑28) +   EXPGEN
 (L↑24) + (L↑20) + (L↑16) + (L↑12) + (L↑8) + (G↑4)+ (G) // PTAND          EXPGEN
,(L↑56) + (L↑52) + (L↑48) + (L↑44) + (C↑40) +(G↑36) + (L↑32) +(G↑28) +   EXPGEN
 (G↑24) + (L↑20) + (L↑16) + (L↑12) + (L↑8) + (G↑4)+ (G) // PTOR           EXPGEN
,(L↑56) + (L↑52) + (L↑48) + (L↑44) + (C↑40) +(G↑36) + (L↑32) +(G↑28) +   EXPGEN
 (G↑24) + (N↑20) + (L↑16) + (L↑12) + (L↑8) + (G↑4)+ (G) // PTNOY          EXPGEN
,(L↑56) + (L↑52) + (L↑48) + (L↑44) + (P↑40) +(E↑36) + (L↑32) +(L↑28) +   EXPGEN
 (L↑24) + (L↑20) + (L↑16) + (L↑12) + (L↑8) + (R↑4)+ (R) // PTFNC          EXPGEN
,(G↑56) + (G↑52) + (G↑48) + (G↑44) + (G↑40) +(G↑36) + (O↑32) +(G↑28) +   EXPGEN
 (G↑24) + (N↑20) + (O↑16) + (O↑12) + (O↑8) + (G↑4)+ (G) // PTID           EXPGEN
,(L↑56) + (L↑52) + (L↑48) + (L↑44) + (K↑40) +(E↑36) + (L↑32) +(L↑28) +   EXPGEN
 (L↑24) + (L↑20) + (L↑16) + (L↑12) + (L↑8) + (R↑4)+ (R) // PTARY          EXPGEN
,(L↑56) + (L↑52) + (L↑48) + (L↑44) + (Z↑40) +(M↑36) + (L↑32) +(L↑28) +   EXPGEN
 (L↑24) + (L↑20) + (L↑16) + (L↑12) + (L↑8) + (D↑4)+ (D) // PTEND          EXPGEN
    LET I   = 0                                                           EXPGEN
    LET TS,ELEMENT,SE,J = 0,0,0,0                                         EXPGEN
    LET TEMP,GT = 0,0                                                     EXPGEN
    LET STACK = VEC 20                                                    EXPGEN
 STACK.0 := ENDMARK                                                       EXPGEN
// GET NEXT LEXICAL ELEMENT                                               EXPGEN
L1: ELEMENT := LEXANA(LAST)                                               EXPGEN
//  TRY ON NEXT UNPROCESSED ELEMENT                                       EXPGEN
//  GET POINTER TO TOP UN PROCESSED ELEMENT                               EXPGEN
L2: J := TS                                                               EXPGEN
    IF TS > 20 ∨ V+I GE EXPLIMIT DO CERROR(                               EXPGEN
        ≡EXPRESSION TOO BIG TO COMPILE-SORRY*N≡)                          EXPGEN
 IF STACK.J LS 0 DO [  J := J - 1                                         EXPGEN
        IF STACK.J LS 0 DO CERROR(≡ OPERATOR MISSING*N≡  )  ]             EXPGEN
    SE := STACK.J                                                         EXPGEN
    GT := SE ↑ -46 ∧ 177B                                                 EXPGEN
//  COMPAIR AND HANDLE THE RESULT                                         EXPGEN
    SWITCHON PARSETABLE.(SE ↑ -53) ↑ ( 4*(ELEMENT ↑-53)) ∧ 17B             EXPGEN
            INTO [S]                                                      EXPGEN
    //  ERROR CONDITIONS                                                  EXPGEN
CASE N: CERROR(≡ IMPROPER USE OF NOT*N≡)                                  EXPGEN
CASE C: CERROR(≡ IMPROPER USE OF COMMA*N≡)                                EXPGEN
CASE R: CERROR(≡ RIGHT PARENTHESIS MISSING*N≡)                            EXPGEN
CASE O: CERROR(≡ OPERATOR MISSING*N≡)                                     EXPGEN
CASE B: CERROR(≡ RELATIONAL OPERATOR USED ON TRUTH VALUE*N≡)              EXPGEN
CASE P : CERROR(≡ ONLY ONE PARAMETER ALLOWED WITH FUNCTION*N≡)            EXPGEN
CASE M: UNLESS LAST+2 DO CERROR(≡ LEFT PARENTHESIS MISSING*N≡)            EXPGEN
CASE Z:UNLESS LAST DO CERROR(≡ IMPROPER USE OF COMMA*N≡)                  EXPGEN
    S:= S - 1                                                             EXPGEN
//  EXPRESSION IS DONE                                                    EXPGEN
CASE D :                                                                  EXPGEN
        V.I := EXPCALLS.ENDEXP +(STACK.1 LSHIFT 18)                       EXPGEN
    //  TEST FOR ERRORS                                                   EXPGEN
    UNLESS TS = 1 DO EM()                                                 EXPGEN
```

```
    EXPTYPE := STACK.1
    UNLESS TYPE = (STRNGEXP∧ARITHEXP)-> (STACK.1 = STRNGEXP ∨
        STACK.1 = ARITHEXP), STACK.1 = TYPE DO CERROR(
            ≡WRONG EXPRESSION TYPE*N≡)
    RESULTIS I+1
CASE L :
    TS := TS + 1
    STACK.TS := ELEMENT
    GOTO    L1
CASE E :
    UNLESS J LS TS DO EM()
    //  GENERATE CODE IF NEEDED
    SWITCHON GT INTO [
    CASE GTSTLST:
        UNLESS (SE ⁺ -36 ∧ 1777B) = 0 DO CERROR(
            ≡STRING LIST WITH MORE THAN ONE DIMENSION*N≡)
        GOTO EGEN
    CASE GTARY :
        IF (SE⁺-36 ∧ 1777B)=0 DO SE := (SE∧NAMASK) + EXPCALLS.LSTCONT
    CASE GTFNC : EGEN:
    V.I := SE ∧ 177777777777777B
    I := I + 1
    UNLESS STACK.TS = ARITHEXP DO TL()      ]
    STACK.J := GT=GTLPREN -> STACK.TS, GT=GTSTLST -> STRNGEXP, ARITHEXP
    TS := J
    GOTO L1
CASE K :
    UNLESS J < TS DO EM()
    UNLESS STACK.TS = ARITHEXP DO TL()
    TS := J
    STACK.TS := SE +.1000000000000B
    GOTO L1
CASE G : //  GENERATE CODE
    SWITCHON GT INTO [SID
    CASE GTPI:
        V.I := SE ∧ AMASK
        I := I + 1
        STACK.TS := ARITHEXP
        GOTO L2
    CASE GTSTVAR:
        V.I :=((SE ∧ AMASK)⁺ 18) + EXPCALLS.FETCHST
        I := I + 1
        STACK.TS := STRNGEXP
        GOTO L2
    CASE GTSTNUM:
        V.I := EXPCALLS.STCNST
        V.(I+1) := STNUMBER
        V.(I+2) := NUMBER
        I := I + 3
        STACK.TS := STRNGEXP
        GOTO L2
    CASE GTVAR :
        V.I :=(SE ∧ AMASK)= PAR -> EXPCALLS.PARVAL,
              ((SE ∧ AMASK) ⁺ 18)+ EXPCALLS.FETCH
        I := I + 1
        STACK.TS := ARITHEXP
```

```
          GOTO L2
      CASE    GTNUM :
          TEST NUMBER =. ONE DO [
              V.I := EXPCALLS.ONECNST
              I := I + 1 ]  ELSE  [
              V.I := EXPCALLS.NMCNST
              V.(I+1) := NUMBER
              I := I + 2 ]
          STACK.TS := ARITHEXP
          GOTO L2  ]SID
      IF TS = J THEN EM()
      IF GT = GTNOT DO [
          UNLESS STACK.TS = LOGEXP DO TL()
          V.I := EXPCALLS.NEG
          I := I + 1
      STACK.J := LOGEXP
          TS := J
          GOTO L2     ]
      IF(GT = GTMINUS v GT = GTPLUS)^ STACK.(J-1) >. 0 DO [UOP
  //  UNARY + OR -
      UNLESS STACK.TS = ARITHEXP DO TL()
          IF GT = GTMINUS DO [
              V.I := EXPCALLS.NEG
              I := I + 1 ]
          STACK.J := ARITHEXP
      TS:= J
          GOTO L2  ]UOP
      TEMP := STACK.(J-1) ^ STACK.TS
      IF TEMP >. 0 DO EM()
      SWITCHON GT INTO [BOP
      CASE    GTROP :
          UNLESS TEMP = ARITHEXP v TEMP = STRNGEXP DO TL()

          IF TEMP = STRNGEXP DO SE := SE + -18
          STACK.(J-1) := LOGEXP
          GOTO L3
      CASE GTLOP :
          UNLESS TEMP = LOGEXP DO TL()
          GOTO     L3
      CASE GTAOP :
      CASE GTPLUS:
      CASE GTMINUS :
          UNLESS TEMP = ARITHEXP DO TL()
      L3: V.I := SE ^ AMASK
          I := I + 1
          TS := TS - 2
          GOTO L2  ]BOP
  CASE 00 : SYSERR(1)
  DEFAULT : SYSERR(2)    ]EG
  LET LEXANA(LAST) = VALOF [LA   //  LEXIAL ANALIZER
      LET      FIN(PT,GT,SP   ) = VALOF [ S:= S+1
           RESULTIS(PT * 53)+(GT *46)+ EXPCALLS.SP ]
      LET     FIN2(PT,GT,SP   ) = VALOF [ S:= S+2
           RESULTIS(PT * 53)+(GT *46)+ EXPCALLS.SP ]
      LET    FINCT(PT,GT,SP,I) = VALOF [ S:= S+I
           RESULTIS(PT * 53)+(GT *46)+ EXPCALLS.SP ]
```

```
LET TEMP,CH = 0,RV S
LET LIBFNC = TABLE 15



    ,↓ABS(↓
    ,↓ATN(↓
    ,↓COS(↓
    ,↓EXP(↓
    ,↓INT(↓
    ,↓LOG(↓
    ,↓SGN(↓
    ,↓SIN(↓
    ,↓SQR(↓
    ,↓TAN(↓
    ,↓RND(↓
    ,↓ACS(↓
    ,↓ASN(↓
    ,↓LGT(↓
    ,↓TIM(↓
  IF S = LAST DO RESULTIS  ENDMARK
    UNLESS S+1 = LAST DO
  SWITCHON (CH ↑ 7)+ S.1 INTO [S2
    CASE ↓>=↓: CASE ↓=>↓:
    RESULTIS FIN2(PTROP,GTROF,MOREEQ)+(EXPCALLS.STMOREEQ↑18)
    CASE ↓<=↓: CASE ↓=<↓:
    RESULTIS  FIN2(PTROP  ,GTROP  ,LESSEQ) + (EXPCALLS.STLESSEQ↑18)
    CASE ↓<>↓: CASE ↓><↓:
    RESULTIS  FIN2(PTROP  ,GTROP   ,NEQUAL) + (EXPCALLS.STNEQUAL↑18)
    CASE ↓PI↓:
    RESULTIS  FIN2(PTID,GTPI,PICNST)
         ]S2
  SWITCHON CH  INTO  [S1
    CASE ↓∨↓ :
    RESULTIS FIN(PTOR,GTLOP,ORCALL)
    CASE ↓∧↓ :
    RESULTIS FIN(PTAND,GTLOP,ANDCALL)
    CASE ↓+↓ :
    RESULTIS  FIN (PTPLUS ,GTPLUS ,ADD)
    CASE ↓-↓ :
    RESULTIS  FIN (PTPLUS ,GTMINUS ,SUBTRAC)
    CASE ↓**↓ :
    RESULTIS  FIN (PTMULT ,GTAOP  ,MULT)
    CASE ↓/↓ :
    RESULTIS  FIN (PTMULT ,GTAOP  ,DIV)
    CASE ↓↑↓ :
    RESULTIS  FIN (PTEXPO ,GTAOP  ,EXPO)
    CASE ↓=↓ :
    RESULTIS  FIN (PTROP  ,GTROP  ,EQUAL) + (EXPCALLS.STEQUAL↑18)
    CASE ↓<↓ :
    RESULTIS  FIN (PTROP  ,GTROP  ,LESSTH) + (EXPCALLS.STLESSTH↑18)
    CASE ↓>↓ :
    RESULTIS  FIN (PTROP  ,GTROP  ,MORETH) + (EXPCALLS.STMORETH↑18)
    CASE ↓≠↓ :
    RESULTIS  FIN (PTROP  ,GTROP   ,NEQUAL) + (EXPCALLS.STNEQUAL↑18)
    CASE ↓,↓ :
```

```
        RESULTIS  FIN (PTCM    ,GTCM    ,0)
        CASE ↓)↓ :
        RESULTIS  FIN (PTRPREN ,GTRPREN ,0)
        CASE ↓(↓ :
        RESULTIS  FIN (PTLPREN ,GTLPREN ,0)
        CASE ↓*N↓ :
        RESULTIS   ENDMARK
        CASE ↓:↓ :
        CASE ↓;↓ :
    RESULTIS LAST + 1 -> ENDMARK , CERROR(≡ IMPROPER CHARACTER*N≡)
// BEWARE OF THIS CLUDGE IF YOU WANT TO EXPAND STRINGS
        CASE QUOTE :
        IF LAST =-2 RESULTIS ENDMARK
        [Q  LET V = VEC 10
            IF COPYSTRING(V,CERROR) > 2 DO CERROR(
                    ≡STRING CONSTANT TOO BIG*N≡)
            STNUMBER,NUMBER := V.0,V.1
            RESULTIS (PTID ↑ 53)+. (GTSTNUM ↑ 46)  ]Q
            ]S1
      SWITCHON (CH ↑ 14)+(S.1 ↑ 7)+ S.2  INTO [S3
        CASE ↓NOT↓ :
        RESULTIS FINCT(PTNOT   ,GTNOT   ,NEG,3)
            ]S3
 //  TEST FOR A NUMBER
    IF  CH = ↓.↓ ∨ DIGIT(CH) DO [
        NUMBER := CONVERT(CERROR)
    RESULTIS (PTID ↑ 53) +. (GTNUM ↑ 46)    ]
  //  TEST FOR ALPHABETIC SINCE IT SHOULD BE A FUNCTION OR A VARIABLE
      UNLESS ALPHA(CH) DO CERROR(≡ IMPROPER CHARACTER*N≡)
  //  TEST FOR AN ARRAY
      IF S.1 = ↓(↓ DO
        RESULTIS FINCT(PTARY   ,GTARY   ,ARYCONT,2)
                        +((ARRAYS+CH - ↓A↓) ↑ 18 )
// TEST FOR STRING ARRAY
        IF S.1 =↓$↓ ∧ S.2 = ↓(↓ DO RESULTIS
            FINCT(PTARY,GTSTLST,STLSTC,3)+.((STARRAYS + CH -↓A↓)↑18)
  //  TEST FOR NOT FUNCTION I.E. VARIABLE
      UNLESS S+1 ≠ LAST ≠ S.2 ∧ ALPHA(S.1) ∧ ALPHA(S.2) DO
    RESULTIS SIMPLEADD()+.((EXPTYPE=STRNGEXP->GTSTVAR,GTVAR)↑46)+.
                (PTID ↑ 53)
// INSURE THIS IS A FUNCTION CALL
    UNLESS S.3 = ↓(↓ DO CERROR(≡VARIABLE TOO LONG*N≡)
  //  CHECK FOR A USER FUNCTION
      IF CH = ↓F↓ ∧ S.1 = ↓N↓ ∧ ALPHA(S.2) DO
    RESULTIS((FUNCTIONS + S.2 - ↓A↓) ↑ 18)+
                FINCT(PTFNC,GTFNC,FNCALL,4)
  //  FIND THE LIBRARY FUNCTION
      TEMP :=(CH ↑ 21)+(S.1 ↑ 14)+(S.2 ↑7)+ S.3
      FOR I = 1 TO LIBFNC.0 DO
        IF LIBFNC . I = TEMP DO
        RESULTIS FINCT(PTFNC   ,GTFNC   ,I,4)
  //  NOT IN LIBRARY
      CERROR(≡LIBRARY FUNCTION DOES NOT EXIST*N≡) ]LA
LET ADDRESS(V) = VALOF [AD
UNLESS ALPHA(RV S) DO CERROR(≡ UNRECOGNIZABLE ADDRESS*N≡)
//CHECK FOR SIMPLE ADDRESS OR ARRAY
```

```
TEST S.1 = ↓(↓ ∨ (S.1 = ↓$↓ ∧ S.2 = ↓(↓) DO [A
LET CT,SZ = 0,0
LET STRG = S.1 = ↓$↓
LET AD = RV S -↓A↓ + (STRG -> STARRAYS,ARRAYS)
    S := S + (STRG-> 3, 2)
    [1 SZ := EXPGEN(V+SZ,-3,ARITHEXP,0)+SZ-1
        IF RV S = ↓)↓ BREAK
        IF RV S ≠ ↓,↓ DO CERROR(≡ILLEGAL CHARACTER*N≡)
        CT := CT + 1
        S := S + 1 ]1 REPEAT
    S := S + 1   .
    IF STRG∧CT≠0 DO CERROR(≡MORE THAN ONE DIMENSION ON STRING LIST*N≡)
    V.SZ := (CT * 36) + (AD * 18) + EXPCALLS.(CT = 0 ->
        (STRG -> STLSTA, LSTADD), ARYADD)
    EXPTYPE := STRG-> STRNGEXP, ARITHEXP
    V.(SZ+1) := EXPCALLS.ENDEXP +.(EXPTYPE LSHIFT 18)
    RESULTIS SZ + 2 ]A ELSE [V
    V.0 := (SIMPLEADD()*18) + EXPCALLS.ADCNST
    V.1 := EXPCALLS.ENDEXP +. (EXPTYPE LSHIFT 18)
    RESULTIS 2 ]AD
//DECK BCPLGD
//  DECLARATIONS TO LINK WITH HEAD

GLOBAL [ // TO LINK WITH FSP AND IO
          // GLOBALS 1-4 ARE FREE
          FLIST     :   5  // FREESPACE PACKAGE POINTER
          OUTPUT    :   6  // STANDARD OUTPUT STREAM
          C6TO7     :   7  // DISPLAY CODE ASCII CONVERSION VECTOR
          C7TO6     :   8  // ASCII DISPLAY CODE  CONVERSION VECTOR
          IOBASE    :   9  // OPEN STREAM LIST POINTER
          MONITOR   :  10
    ]

EXTERNAL [
          NEWVEC          // NEWVEC(SIZEWANTED)
          RETVEC          // RETVEC(V,SIZEOFV)
          PACKSTRING      // (VECTOR,STRING)
          UNPACKSTRING    // (STRING,VECTOR)
          INITIALIZEIO
          FINDINPUT       //  (≡FNAME,USERNAME≡)
          CREATEOUTPUT    //  (≡FNAME,USERNAME≡)
          READCH          // (STREAM,LV CH)
          WRITECH         // (STREAM,CH)
          WRITES          // OUTPUT:=STREAM; WRITES(STRING)
          WRITEN          // OUTPUT:=STREAM; WRITEN(NUMBER)
          WRITEO          // OUTPUT:=STREAM; WRITEO(NUMBER)
          READN           // NUMBER READER N:=READN(STREAM)
          TRACE
          TRACECALL
          TRACERETURN
          DUMP
          ENDREAD         // CLOSE INPUT STREAM
          ENDWRITE        // CLOSE OUTPUT STREAM
          IOERROR         // GENERAL IOERROR ROUTINE
          UNPACKTO        //  (STR,VEC,BREAKCH)
          WRITEVEC
```

```
        READVEC
        FLUSH          // FORCE OUT A LINE WITHOUT CR
        BCDWORD
        ASCII
        ENDOFSTREAM
        CLOSEALL
        ABORT
    ]

MANIFEST [ AMASK = 777777B  ; ENDOFSTREAMCH =144B
        BUFFERSIZE = 136
         ESCAPECH = 173B ]
    GLOBAL [GV    //  GLOBAL VARIABLES
INPUT               : 49
    FREE            : 50
    INPUTOPEN       : 51
    EXPLIMIT        : 52
// TWO HERE
    HILINE          : 54
    LOWLINE         : 55
    FREELINE        : 56
    DATA            : 57
    TMPDATA         : 58
    DATACOUNT       :59
    TMPDATACNT      :60
    RETURNS         : 61
    PROG            : 62
    LASTPROG        : 63
    EXPSTOP         :64
    S               : 65
    CHARCNT         :66
    SIG             : 67
    RUNNING         : 68
    TIME        :72
    NEXT            : 75
    NEXTINPUT       : 76
    ENDINPUT        : 77
    STNUMBER        : 78
    NUMBER          : 79
    ACCEPT          : 80
    RECURSE         : 82
    TYPE            : 84
// 88,89 RESERVED FOR MATHLIB
// HERE IS 2
    EXPTYPE         : 92
    CURSEED         : 93
    SEEDFLAG        : 94
    LIMIT           : 95
    IERROR          : 96
    RELEASE         : 97
        ]GV
EXTERNAL  [GB      // GLOBAL BUFFERS AND VECTORS
    VARIABLES
    FUNCTIONS
    ARRAYS
    LINES
```

```
     EXPCALLS
     CHARBUF
     STRINGS
     STARRAYS
     IOBUF
     INBUF
     GLOBAL0
         IGB
EXTERNAL  [GR          //  GLOBAL ROUTINES
     COMPILE
     ADDRESS
     EXPGEN
     LEXANA
     CONVERT
     WRITEF
     UNPACK
     PACK
     COMPACT
     LINUM
     FINDLINE
     COPY
     DATAOK
     DATASTORE
     DATAINIT
     ALPHA
     DIGIT
     INT
     FLOAT
     FMULT
     FDIV
     FADD
     CHKFDIV
     CHKPIV
     SIMPLEADD
     EXECUTE
     MATEX
     EXPRESS
     GIVESPACE  = ERETVEC
     GETSPACE   = ENEWVEC
     READLINE
     WRITE
     EERROR
     CERROR
     RERROR
     SYSERR
     EXCHARS
     UNREC
     FWDLINE
     FILL
     COPYSTRING
     ELE
     ELE3
     MATADD
     MATSUB
     MATSML
     INPROD
```

```
    NORM
    MATCMD
              ]GR
    MANIFEST  [MGC    //  SOME GENERAL CONSTANTS
    QUOTE             = ↓≡↓
    AMASK             = 377777B
    NULLCH            = 140B
    NMASK             = -377777B
    NAMASK            = -377777B
    ZONESIZE          = 72/5
    LINESIZE          = 72
    ARITHEXP          = -1
    LOGEXP            = -2
    STRNGEXP          = -4
    HIADD             = 5000
    INDEF             = 6000000000000000000B
    ONE               = 1720400000000000000B
    TEN               = 1723500000000000000B
    RNDSEED           = 113434743B
    DELETE            = 3
    CHANGE            = 2
    INSERT            = 1
    DIRECT            = 4
    QUIT              = 5
    INTERUPTED = FALSE // SO THAT INTERUPT WILL BE DEFINED FOR MATEX
               ]MGC
MANIFEST    [MCMD      //  MANIFEST CONSTANTS FOR IDENTIFING LINES
    LETCMD   = 1
    GOTCMD   = 2
    IFCMD    = 3
    IFTCMD   = 3
    FORCMD   = 4
    NEXCMD   = 5
    REACMD   = 6
    INPCMD   = 7
    PRICMD   = 8
    DEFCMD   = 9
    REMCMD   = 10
    GOSCMD   = 12
    RETCMD   = 13
    ENDCMD   = 14
    STOCMD   = 15
    DATCMD   = 16
    RESCMD   = 17
    SIGCMD   = 18
    DIMCMD   = 19
    ONCMD    = 20
    MPRICMD  = 21
    MREACMD  = 22
    MTRNCMD  = 23
    MZERCMD  = 24
    MIDNCMD  = 25
    MCONCMD  = 26
    MADDCMD  = 27
    MSUBCMD  = 28
    MMLTCMD  = 29
```

```
    MSMLCMD  = 30
    MINVCMD  = 31
         ]MCMD
MANIFEST [MEC     // INDEXES INTO EXPCALLS
    ABS      = 1
    ATAN     = 2
    COSINE   = 3
    EXP      = 4
    INTEGER  = 5
    LOG      = 6
    SIGN     = 7
    SINE     = 8
    SQRT     = 9
    TAN      = 10
    RND      = 11
    ACS      = 12
    ASN      = 13
    LGT      = 14
    TIM      = 15
    ENDEXP   = 8 + 12
    ADD      = 8 + 13
    SUBTRAC  = 8 + 14
    MULT     = 8 + 15
    DIV      = 8 + 16
    NEG      = 8 + 17
    ANDCALL  = 8 + 18
    ORCALL   = 8 + 19
    EQUAL    = 8 + 20
    NEQUAL   = 8 + 21
    LESSTH   = 8 + 22
    LESS     = 8 + 22
    MORETH   = 8 + 23
    MORE     = 8 + 23
    LESSEQ   = 8 + 24
    MOREEQ   = 8 + 25
    STORE    = 8 + 26
    FETCH    = 8 + 27
    NMCNST   = 8 + 28
    ADCNST   = 8 + 29
    ONECNST  = 8 + 30
    FNCALL   = 8 + 31
    FRET     = 8 + 32
    PARVAL   = 8 + 33
    ARYCONT  = 8 + 34
    ARYADD   = 8 + 35
    EXOPNET  = 8 + 36
    EXPO     = 8 + 36
    LSTCONT  = 8 + 37
    LSTADD   = 8 + 38
    STLSTC   = 8 + 39
    STLSTA   = 8 + 40
    STOREST  = 8 + 41
    FETCHST  = 8 + 42
    STEQUAL  = 8 + 43
    STCNST   = 8 + 44
    STNEQUAL = 8 + 45
```

```
    PICNST   = 8 + 46
    STMORETH = 8 + 47
    STMOREEQ = 8 + 48
    STLESSTH = 8 + 49
    STLESSEQ = 8 + 50
          ]MEC
MANIFEST  [MPT    //PARSE TYPE CONSTANTS
    PTPLUS    = 1
    PTMULT   = 2
    PTEXPO   = 3
    PTROP    = 4
    PTCM     = 5
    PTRPREN  = 6
    PTLPREN  = 7
    PTAND    = 8
    PTOR     = 9
    PTNOT    = 10
    PTFNC    = 11
    PTID     = 12
    PTARY    = 13
    PTEND    = 14
          ]MPT
MANIFEST    [MGT     //  GENERATING TYPE CONSTANTS
    GTPLUS   = 1
    GTMINUS  = 2
    GTAOP    = 3
    GTROP    = 4
    GTLOP    = 5
    GTCM     = 6
    GTRPREN  = 7
    GTLPREN  = 8
    GTNUM    = 9
    GTFNC    = 10
    GTARY    = 11
    GTVAR    = 12
    GTEND    = 14
    GTNOT    = 15
    GTSTLST  = 16
    GTSTVAR  = 17
    GTSTNUM  = 18
    GTPI     = 21
          ]MGT
MANIFEST [ ENDMARK  = (PTEND ASHIFT 53)+(GTEND ASHIFT 46) ]
LET EXECUTE(LIN) BE [EXC
SWITCHON LIN ↑ -54 INTO [SW
CASE 0 : RERROR(≡ RAN OFF END OF PROGRAM*N≡)
CASE LETCMD :
    EXPRESS(LIN+1)
    RETURN
CASE  IFCMD :
    UNLESS EXPRESS(LIN+1) DO RETURN
CASE GOTCMD :
    PROG := RV LIN ↑ -36
    IF(RV PROG ∧ AMASK)= 0 THEN RERROR(≡ GOTO UNDEFINED LINE*N≡)
    RETURN
CASE ONCMD :  [ON
```

```
    LET INDEX = INT(EXPRESS(LIN+1))
    IF INDEX = 0 DO RERROR(Ξ ZERO INDEX*NΞ)
    IF INDEX < 0 DO RERROR(Ξ NEGATIVE INDEX*NΞ)
    IF INDEX > RV LIN ♦ -36 DO RERROR(Ξ INDEX TOO BIG*NΞ)
    PROG := RV ( EXPSTOP + INDEX -1)
    IF (RV PROG ∧ AMASK) = 0 DO RERROR(Ξ GOTO UNDEFINED LINE*NΞ)
    RETURN ]ON
CASE FORCMD :
 //  FIND THE MATCHING NEXT COMMAND
[F LET PT = LV LIN
    LET CT,VAL = 1,0
    [1 PT := FWDLINE(PT)
        TEST RV PT ♦ -54 = NEXCMD DO [2  CT := CT - 1
            IF CT = 0 DO BREAK  ]2
        ELSE IF RV PT ♦ -54 = FORCMD DO CT := CT + 1
        IF PT = HILINE DO RERROR(Ξ NOT ENOUGH NEXT COMMANDS*NΞ)
            ]1 REPEAT
 // CHECK TO SEE THAT THE VARIABLES MATCH
    PT := RV PT
    UNLESS RV LIN ♦ -36 = RV PT ♦ -36 DO
        RERROR(Ξ VARIABLE IN NEXT DOES NOT MATCH*NΞ)
    VAL := EXPRESS(LIN+1)
    PT.2 := EXPRESS(EXPSTOP)
    PT.3 := EXPRESS(EXPSTOP)
 //  ASSIGN VALUE AND TEST FOR DONE
    RV(RV LIN ♦ -36) :=  VAL
    TEST PT.3 -> VAL < PT.2, VAL > PT.2 DO [
        PROG := FWDLINE(LV PT)
        PT.1 := 0  ]
    ELSE PT.1 := PROG
    RETURN  ]F
CASE NEXCMD :
    IF LIN.1 = 0 DO RERROR(Ξ UNINITALIZED NEXT EXECUTED*NΞ)
[N LET VAL = FADD(LIN.3,RV(LIN.0♦ -36))
    TEST LIN.3 -> VAL < LIN.2, VAL > LIN.2 DO LIN.1 := 0
    ELSE [
        IF VAL =. FADD(VAL,LIN.3) DO RERROR(Ξ INFINITE FOR LOOP*NΞ)
        RV(LIN.0 ♦ -36) := VAL
        PROG := LIN.1 ]
    RETURN  ]N
CASE GOSCMD :
    IF(RV(RV LIN ♦ -36) ∧ AMASK)= 0 DO
        RERROR(Ξ GOSUB UNDEFINED LINE*NΞ)
[G LET V = GETSPACE(1)
    RV V := (PROG ♦ 18) + RETURNS
    RETURNS := V
    PROG := RV LIN ♦ -36
    RETURN ]G
CASE ENDCMD :
CASE STOCMD :
    UNLESS CHARCNT = 0 DO WRITE(♦*N♦)
WRITES(Ξ EXECUTION COMPLETE*NΞ)
    CLOSEALL()
    FINISH
CASE RETCMD :
    IF RETURNS = 0 DO RERROR(Ξ NOT IN A SUBROUTINE*NΞ)
```

```
    [1  LET RET = RV RETURNS
        PROG := RET ↑ -18 ∧ AMASK
        GIVESPACE(RETURNS,1)
        RETURNS := RET ∧ AMASK
    RETURN
CASE REMCMD :
CASE DATCMD :
    RETURN
CASE RESCMD :
    DATA := 0
    RETURN
CASE DEFCMD :
[D LET SZ = (RV LIN ↑ -18  ∧  AMASK) - 1
    LET ENTRY = RV LIN ↑ -36
UNLESS RV ENTRY =.INDEF DO [  GIVESPACE(RV ENTRY,RV ENTRY ↑ -18)
    RV ENTRY := INDEF ]
    [ LET SP = GETSPACE(SZ)
        COPY(LIN+1,SP,SZ)
        RV ENTRY := (SZ↑ 18) + SP
    RETURN ]D
CASE DIMCMD :
[D EXPSTOP := LIN + 1 ∧ AMASK
    FOR I = 1 TO RV LIN ↑ -36 DO [F
        LET CT = RV EXPSTOP ↑ -18∧ AMASK
        LET STRNG = RV EXPSTOP ↑-59
        LET ENTRY = RV EXPSTOP ∧ AMASK
        LET SZ,FNC = 1,0
        UNLESS RV ENTRY =.INDEF DO [RET
            LET PTFN = RV ENTRY
            LET PTSP = PTFN↑ -18
            LET CT = PTFN ↑ -36
            RV ENTRY := INDEF
            TEST CT = 0 DO GIVESPACE(PTSP,PTFN∧AMASK) ELSE [
            GIVESPACE(PTSP,INT(PTFN.CT))
            GIVESPACE(PTFN,CT+1) ]RET
    TEST CT = 1 DO [1
        FNC :=(INT(EXPRESS(EXPSTOP+1))+.1) * (STRNG -> 2,1)
        IF FNC <. 1 DO RERROR(≡ NEGATIVE LIST SIZE*N≡)
        IF FNC >. HIADD DO RERROR(≡ LIST TOO BIG*N≡)
        SZ := FNC
    ]1 ELSE [2
        FNC := GETSPACE(CT)
        EXPSTOP := EXPSTOP + 1
        FOR I = FNC TO FNC + CT - 1 DO [3
            SZ := INT(EXPRESS(EXPSTOP))*SZ
            UNLESS 0<SZ<HIADD DO [4
                IF SZ > HIADD DO RERROR(≡ ARRAY TOO BIG*N≡)
                IF SZ = 0 DO RERROR(≡ ZERO ARRAY DIMENSION*N≡)
                RERROR(≡ NEGATIVE ARRAY DIMENSION*N≡) ]4
            RV I := FLOAT(SZ) ]2
    [ LET V = GETSPACE(SZ)
    FILL(V,INDEF,SZ)
    RV ENTRY := (CT-1 ↑ 36) + (V ↑ 18) + (FNC  ∧ AMASK) ]F
    RETURN ]D
CASE SIGCMD:  [SG
    LET SZ = INT(EXPRESS(LIN+1))
```

```
    IF SZ < 1 DO RERROR(Ξ LESS THAN ONE SIGNIFICANT DIGIT REQUESTED*NΞ)   EXECUTE
    IF SZ > 13 DO RERROR(Ξ TOO MANY SIGNIFICANT DIGIT REQUESTED*NΞ)       EXECUTE
    SIG := SZ                                                              EXECUTE
    RETURN ]SG                                                             EXECUTE
CASE PRICMD :                                                              EXECUTE
    FOR I = LIN+1 ∧ AMASK TO (LIN ∧ AMASK) + (RV LIN ↑ -18) - 1 DO         EXECUTE
    SWITCHON RV I INTO [PS                                                 EXECUTE
    CASE ↓ ↓ : WRITE(↓ ↓)                                                  EXECUTE
    CASE ↓*N↓ :                                                            EXECUTE
    CASE ↓*T↓ :                                                            EXECUTE
        WRITE(RV I)                                                        EXECUTE
        GOTO DONE                                                          EXECUTE
    CASE QUOTE :                                                           EXECUTE
    [ LET V = VEC 76                                                       EXECUTE
        UNPACKSTRING(I+1,V)                                                EXECUTE
        FOR J = 1 TO V.0 DO WRITE(V.J)                                     EXECUTE
        I := I + V.0 / 8 + 1 ]                                             EXECUTE
        GOTO DONE                                                          EXECUTE
    CASE ↓T↓:  [ LET P = INT(EXPRESS(I+1))                                 EXECUTE
        I := EXPSTOP - 1                                                   EXECUTE
        IF P LE 0 ∨ P > 72 DO RERROR(Ξ BAD ARGUMENT FOR TAB*NΞ)            EXECUTE
        IF CHARCNT GE P DO WRITE(↓*N↓)                                     EXECUTE
        UNTIL CHARCNT = P-1 DO WRITE(↓ ↓)                                  EXECUTE
        GOTO DONE  ]                                                       EXECUTE
    DEFAULT:  [E // IF NOT A SPECIAL CHARACTER IT IS AN EXPRESSION         EXECUTE
        LET VAL = EXPRESS(I)                                               EXECUTE
        TEST EXPTYPE = STRNGEXP DO  [                                      EXECUTE
            LET V = VEC 16                                                 EXECUTE
            UNPACKSTRING (LV STNUMBER,V)                                   EXECUTE
            FOR I = 1 TO V.0 DO WRITE(V.I) ]                               EXECUTE
            ELSE WRITEF(VAL) ]E //ASSUMES NUMBER=STNUMBER+1                EXECUTE
    I := EXPSTOP - 1                                                       EXECUTE
DONE:         ]PS                                                          EXECUTE
    RETURN                                                                 EXECUTE
CASE READCMD : [RD                                                         EXECUTE
    EXPSTOP  := LIN + 1                                                    EXECUTE
    DATAINIT()                                                             EXECUTE
    FOR I = 1 TO RV LIN ↑ -36 DO DATASTORE(EXPRESS(EXPSTOP),EXPTYPE)       EXECUTE
    DATAOK()                                                               EXECUTE
    RETURN  ]RD                                                            EXECUTE
CASE INPCMD:  // INPUT DATA FROM FILE                                      EXECUTE
    UNLESS INPUTOPEN DO [                                                  EXECUTE
        INPUTOPEN := TRUE                                                  EXECUTE
        INPUT := FINDINPUT(RV 2 = 0 -> BCDWORD(ΞINPUTΞ),RV 2)              EXECUTE
        ENDINPUT,NEXTINPUT := INBUF,INBUF  ]                               EXECUTE
[IN LET ERR(ST) BE [ER  //PRINTT ERROR MESS AND READ NEW CARD              EXECUTE
        LET CH = CHARBUF                                                   EXECUTE
        [  CH := CH + 1                                                    EXECUTE
        WRITECH(OUTPUT,RV CH)] REPEATUNTIL RV CH = ↓*N↓                    EXECUTE
        WRITES(ΞINPUT ERROR - CARD IGNORED Ξ)                              EXECUTE
        WRITES(ST)                                                         EXECUTE
        GOTO IERROR ]ER                                                    EXECUTE
    EXPSTOP := LIN + 1                                                     EXECUTE
    FOR I = 1 TO RV LIN ↑ -36 DO [INVAL  // INPUT ONE VALUE EACH TIME      EXECUTE
        LET ADD = EXPRESS(EXPSTOP)                                         EXECUTE
        IF NEXTINPUT = ENDINPUT DO [READ //LAST CARD EXAUSTED              EXECUTE
```

```
            IERROR := PL
        PL: IF ENDOFSTREAM(INPUT) DO RERROR(≡ INPUT FILE EXAUSTED*N≡)
            READLINE(CHARBUF)
            COMPACT(CHARBUF)
            S := CHARBUF
            ENDINPUT,NEXTINPUT := INBUF,INBUF
            [1  LET TS = S + 1
                S := TS
                TEST RV S = QUOTE ∨ ALPHA(RV S) DO [S
                    IF COPYSTRING(ENDINPUT+1,ERR) > 2 DO ERR(
                        ≡ STRING TOO BIG*N≡)
                    RV ENDINPUT := INDEF
                    ENDINPUT := ENDINPUT+3
                ]S ELSE [N
                    RV ENDINPUT := CONVERT(ERR)
                    ENDINPUT := ENDINPUT+1   ]N
                IF ENDINPUT> INBUF+38 DO ERR(≡TOO MANY VALUES≡)
                IF S = TS DO ERR(≡ VALUE MISSING*N≡)
            ]1  REPEATWHILE RV S = ↓,↓
            UNLESS RV S = ↓*N↓ DO ERR(≡ ILLEGAL SEPERATOR*N≡) ]READ
        TEST EXPTYPE = ARITHEXP DO [AR
            IF RV NEXTINPUT =. INDEF DO RERROR(
                ≡ STRING INPUT FOR A NUMBER*N≡)
            RV ADD := RV NEXTINPUT
            NEXTINPUT := NEXTINPUT + 1
        ]AR  ELSE  [ST
            UNLESS RV NEXTINPUT =. INDEF DO RERROR(
                ≡ NUMBER INPUT FOR A STRING*N≡)
            ADD.0 := NEXTINPUT.1
            ADD.1 := NEXTINPUT.2
            NEXTINPUT := NEXTINPUT+3   ]INVAL
    RETURN ]IN
DEFAULT: MATEX(LIN) ;RETURN
    ]EXC     // END OF EXECUTE
//DECK BCPLGD
//  DECLARATIONS TO LINK WITH HEAD

GLOBAL [ // TO LINK WITH FSP AND IO
          // GLOBALS 1-4 ARE FREE
          FLIST      :   5  // FREESPACE PACKAGE POINTER
          OUTPUT     :   6  // STANDARD OUTPUT STREAM
          C6TO7      :   7  // DISPLAY CODE ASCII CONVERSION VECTOR
          C7TO6      :   8  // ASCII DISPLAY CODE  CONVERSION VECTOR
          IOBASE     :   9  // OPEN STREAM LIST POINTER
          MONITOR    :  10
    ]

EXTERNAL [
          NEWVEC          // NEWVEC(SIZEWANTED)
          RETVEC          // RETVEC(V,SIZEOFV)
          PACKSTRING      // (VECTOR,STRING)
          UNPACKSTRING    // (STRING,VECTOR)
          INITIALIZEIO
          FINDINPUT       //  (≡FNAME,USERNAME≡)
          CREATEOUTPUT    //  (≡FNAME,USERNAME≡)
          READCH          // (STREAM,LV CH)
```

```
        WRITECH        // (STREAM,CH)
        WRITES         // OUTPUT:=STREAM; WRITES(STRING)
        WRITEN         // OUTPUT:=STREAM; WRITEN(NUMBER)
        WRITEO         // OUTPUT:=STREAM; WRITEO(NUMBER)
        READN          // NUMBER READER N:=READN(STREAM)
        TRACE
        TRACECALL
        TRACERETURN
        DUMP
        ENDREAD        // CLOSE INPUT STREAM
        ENDWRITE       // CLOSE OUTPUT STREAM
        IOERROR        // GENERAL IOERROR ROUTINE
        UNPACKTO       //  (STR,VEC,BREAKCH)
        WRITEVEC
        READVEC
        FLUSH          // FORCE OUT A LINE WITHOUT CR
        BCDWORD
        ASCII
        ENDOFSTREAM
        CLOSEALL
        ABORT
     ]

MANIFEST [ AMASK = 777777B  ; ENDOFSTREAMCH =144B
        BUFFERSIZE = 136
         ESCAPECH = 173B ]
    GLOBAL [GV    //  GLOBAL VARIABLES
INPUT             : 49
    FREE          : 50
    INPUTOPEN     : 51
    EXPLIMIT      : 52
// TWO HERE
    FILINE        : 54
    LOWLINE       : 55
    FREELINE      : 56
    DATA          : 57
    TMPDATA       : 58
    DATACOUNT     :59
    TMPDATACNT    :60
    RETURNS       : 61
    PROG          : 62
    LASTPROG      : 63
    EXPSTOP       :64
    S             : 65
    CHARCNT       :66
    SIG           : 67
    RUNNING       : 68
    TIME      :72
    NEXT          : 75
    NEXTINPUT     : 76
    ENDINPUT      : 77
    STNUMBER      : 78
    NUMBER        : 79
    ACCEPT        : 80
    RECURSE       : 82
    TYPE          : 84
```

```
// 88,89 RESERVED FOR MATHLIB
// HERE IS ?
    EXPTYPE          : 92
    CURSEED          : 93
    SEEDFLAG         : 94
    LIMIT            : 95
    IERROR           : 96
    RELEASE          : 97
          ]GV
EXTERNAL  [GB        // GLOBAL BUFFERS AND VECTORS
    VARIABLES
    FUNCTIONS
    ARRAYS
    LINES
    EXPCALLS
    CHARBUF
    STRINGS
    STARRAYS
    IOBUF
    INBUF
    GLOBALO
        ]GB
EXTERNAL  [GR         //  GLOBAL ROUTINES
    COMPILE
    ADDRESS
    EXPGEN
    LEXANA
    CONVERT
    WRITEF
    UNPACK
    PACK
    COMPACT
    LINUM
    FINDLINE
    COPY
    DATAOK
    DATASTORE
    DATAINIT
    ALPHA
    DIGIT
    INT
    FLOAT
    FMULT
    FDIV
    FADD
    CHKFDIV
    CHKPIV
    SIMPLEADD
    EXECUTE
    MATEX
    EXPRESS
    GIVESPACE  = ERETVECE
    GETSPACE   = ENEWVECE
    READLINE
    WRITE
    EERROR
```

```
     CERROR
     RERROR
     SYSERR
     EXCHARS
     UNREC
     FWDLINE
     FILL
     COPYSTRING
     ELE
     ELE3
     MATADD
     MATSUB
     MATSML
     INPROD
     NORM
     MATCMD
               ]GR
     MANIFEST  [MGC    //  SOME GENERAL CONSTANTS
     QUOTE             = ↓E↓
     AMASK             = 377777B
     NULLCH            = 140B
     NMASK             = -377777B
     NAMASK            = -377777B
     ZONESIZE          = 72/5
     LINESIZE          = 72
     ARITHEXP          = -1
     LOGEXP            = -2
     STRNGEXP          = -4
     HIADD             = 5000
     INDEF             = 60000000000000000000B
     ONE               = 17204000000000000000B
     TEN               = 17235000000000000000B
     RNDSEED           = 11343474 3B
     DELETE            = 3
     CHANGE            = 2
     INSERT            = 1
     DIRECT            = 4
     QUIT              = 5
     INTERUPTED = FALSE // SO THAT INTERUPT WILL BE DEFINED FOR MATEX
               ]MGC
MANIFEST    [MCMD      //  MANIFEST CONSTANTS FOR IDENTIFING LINES
     LETCMD   = 1
     GOTCMD   = 2
     IFCMD    = 3
     IFTCMD   = 3
     FORCMD   = 4
     NEXCMD   = 5
     REACMD   = 6
     INPCMD   = 7
     PRICMD   = 8
     DEFCMD   = 9
     REMCMD   = 10
     GOSCMD   = 12
     RETCMD   = 13
     ENDCMD   = 14
     STOCMD   = 15
```

```
     DATCMD    = 16
     RESCMD    = 17
     SIGCMD    = 18
     DIMCMD    = 19
     ONCMD     = 20
     MPRICMD   = 21
     MREACMD   = 22
     MTRNCMD   = 23
     MZERCMD   = 24
     MIDNCMD   = 25
     MCONCMD   = 26
     MADDCMD   = 27
     MSUBCMD   = 28
     MMLTCMD   = 29
     MSMLCMD   = 30
     MINVCMD   = 31
          ]MCMD
MANIFEST [MEC    // INDEXES INTO EXPCALLS
     ABS       = 1
     ATAN      = 2
     COSINE    = 3
     EXP       = 4
     INTEGER   = 5
     LOG       = 6
     SIGN      = 7
     SINE      = 8
     SQRT      = 9
     TAN       = 10
     RND       = 11
     ACS       = 12
     ASN       = 13
     LGT       = 14
     TIM       = 15
     ENDEXP    = 8 + 12
     ADD       = 8 + 13
     SUBTRAC   = 8 + 14
     MULT      = 8 + 15
     DIV       = 8 + 16
     NEG       = 8 + 17
     ANDCALL   = 8 + 18
     ORCALL    = 8 + 19
     EQUAL     = 8 + 20
     NEQUAL    = 8 + 21
     LESSTH    = 8 + 22
     LESS      = 8 + 22
     MORETH    = 8 + 23
     MORE      = 8 + 23
     LESSEQ    = 8 + 24
     MOREEQ    = 8 + 25
     STORE     = 8 + 26
     FETCH     = 8 + 27
     NMCNST    = 8 + 28
     ADCNST    = 8 + 29
     ONECNST   = 8 + 30
     FNCALL    = 8 + 31
     FRET      = 8 + 32
```

```
    PARVAL    = 8 + 33
    ARYCONT   = 8 + 34
    ARYADD    = 8 + 35
    EXOPNET   = 8 + 36
    EXPO      = 8 + 36
    LSTCONT   = 8 + 37
    LSTADD    = 8 + 38
    STLSTC    = 8 + 39
    STLSTA    = 8 + 40
    STOREST   = 8 + 41
    FETCHST   = 8 + 42
    STEQUAL   = 8 + 43
    STCNST    = 8 + 44
    STNEQUAL  = 8 + 45
    PICNST    = 8 + 46
    STMORETH  = 8 + 47
    STMOREEQ  = 8 + 48
    STLESSTH  = 8 + 49
    STLESSEQ  = 8 + 50
          ]MEC
MANIFEST  [MPT    //PARSE TYPE CONSTANTS
    PTPLUS     = 1
    PTMULT    = 2
    PTEXPO    = 3
    PTROP     = 4
    PTCM      = 5
    PTRPREN   = 6
    PTLPREN   = 7
    PTAND     = 8
    PTOR      = 9
    PTNOT     = 10
    PTFNC     = 11
    PTID      = 12
    PTARY     = 13
    PTEND     = 14
          ]MPT
MANIFEST    [MGT     //  GENERATING TYPE CONSTANTS
    GTPLUS    = 1
    GTMINUS   = 2
    GTAOP     = 3
    GTROP     = 4
    GTLOP     = 5
    GTCM      = 6
    GTRPREN   = 7
    GTLPREN   = 8
    GTNUM     = 9
    GTFNC     = 10
    GTARY     = 11
    GTVAR     = 12
    GTEND     = 14
    GTNOT     = 15
    GTSTLST   = 16
    GTSTVAR   = 17
    GTSTNUM   = 18
    GTPI      = 21
          ]MGT
```

```
MANIFEST [ ENDMARK  = (PTEND ASHIFT 53)+(GTEND ASHIFT 46) ]
LET MATEX(LIN) BE [EXC
LET DIMMED(ENTRY) BE [ IF ENTRY =. INDEF DO RERROR(
           ≡ UNDIMENSIONED ARRAY USED*N≡) ]
SWITCHON LIN ↑ -54 INTO [SW
CASE MINVCMD: [IN
    LET ENTRYA = RV (RV LIN ↑ -36)
    LET ENTRYC = RV LIN.1
    DIMMED(ENTRYA)
    DIMMED(ENTRYC)
    UNLESS ENTRYA↑-36 = ENTRYC ↑ -36 = 1 DO RERROR(
       ≡ ARRAY NOT A MATRIX*N≡)
    UNLESS ENTRYA.0 = ENTRYC.0 ∧ ENTRYA.1 = ENTRYC.1 DO RERROR(
       ≡ MATRIX SIZES DO NOT MATCH*N≡)
    UNLESS FMULT(ENTRYA.0,ENTRYA.0) = ENTRYA.1 DO RERROR(
       ≡ NOT A SQUARE MATRIX*N≡)
    COPY(ENTRYC↑-18,ENTRYA↑-18,INT(ENTRYA.1))
 [  LET SZ = INT(ENTRYA.0)
    LET N = SZ-1
    LET DESC = (N↑18) + (ENTRYA↑-18)
    LET P = GETSPACE(2*SZ)
    LET V = P + SZ
    LET D = ONE
    RELEASE := (SZ↑19) +. P
    FOR I = 0 TO N DO V.I := NORM(DESC+I)
    FOR K = 0 TO N DO [TRI
        LET R = -ONE
        FOR I = K TO N DO [LOW
            LET S = 0
            RV ELE(I,K,DESC) := FADD(RV ELE(I,K,DESC),-INPROD(
                DESC+K*SZ,DESC+I,SZ,K))
            S := CHKFDIV(RV ELE(I,K,DESC),V.I,
                    VALOF [ RESULTIS ERET
                    ERET:RERROR(≡MATRIX VERY SINGULAR*N≡)
                        ] )
            IF S DO S := -S
            IF S >. R  DO [
                R:=S
                P.K := I   ]LOW
        V.(P.K) := V.K
        IF CHKPIV(RV ELE(K,K,DESC),V.K) DO RERROR(
            ≡MATRIX NEAR SINGULAR*N≡)
        FOR J = 0 TO N DO [UP
            R := RV ELE(K,J,DESC)
            RV ELE(K,J,DESC) := J LE K -> RV ELE(P.K,J,DESC),
                FDIV(FADD(RV ELE(P.K,J,DESC),-INPROD(DESC+J*SZ,
                DESC+K,SZ,K)), RV ELE(K,K,DESC))
            IF P.K ≠ K DO RV ELE(P.K,J,DESC) := -R ]UP
        D := FMULT(RV ELE(K,K,DESC),D)
        ]TRI
    [INV  LET K = N
    WHILE K GE 0 DO [
        FOR J = K+1 TO N DO [
            V.J := RV ELE(K,J,DESC)
            RV ELE(K,J,DESC) := 0 ]
        RV ELE(K,K,DESC) := FDIV(ONE,RV ELE(K,K,DESC))
```

```
    [1  LET J = K-1
    WHILE J GE 0 DO [
        RV ELE(K,J,DESC) := -FDIV(INPROD(DESC+J*SZ+J+1,
            (J+1)*SZ+DESC+K,SZ,K-J), RV ELE(J,J,DESC))
        J := J-1  ]1
    FOR J = 0 TO N DO RV ELE(K,J,DESC) := FADD(RV ELE(K,J,DESC)
        ,-INPROD(V+K+1,DESC+J*SZ+K+1,1,N-K))
    K := K-1   ]INV
  [EX  LET K = N
  WHILE K GE 0 DO [
    IF P.K ≠ K DO FOR I = 0 TO N DO [
        LET ADK = ELE(I,K,DESC)
        LET ADPK = ELE(I,P.K,DESC)
        LET R = RV ADK
        RV ADK := - RV ADPK
        RV ADPK := R ]
    K := K-1   ]EX
  GIVESPACE(RELEASE,RELEASE↑-18)
  RELEASE := 0
  RETURN   ]IN
CASE MZERCMD: [Z
  LET ENTRY = RV( RV LIN ↑ -36)
  DIMMED(ENTRY)
  TEST ENTRY ↑ -36 = 0 DO FILL(ENTRY↑-18,0,ENTRY∧AMASK)
  ELSE FILL(ENTRY↑-18,0,INT(ENTRY.(ENTRY↑-36)))
  RETURN   ]Z
CASE MCONCMD: [C
  LET ENTRY = RV ( RV LIN ↑ -36)
  DIMMED(ENTRY)
  TEST ENTRY ↑ -36 = 0 DO FILL(ENTRY ↑ -18,ONE,ENTRY∧ AMASK)
  ELSE FILL(ENTRY↑-18,ONE,INT(ENTRY.(ENTRY↑-36)))
  RETURN   ]C
CASE MIDNCMD: [I
  LET ENTRY = RV (RV LIN ↑ -36)
  DIMMED(ENTRY)
  UNLESS ENTRY ↑ -36 = 1 DO RERROR(≡ NOT A MATRIX*N≡)
  UNLESS FMULT(ENTRY.0,ENTRY.0) =. ENTRY.1 DO RERROR(
     ≡ MATRIX NOT SQUARE*N≡)
  FILL(ENTRY↑-18,0,INT(ENTRY.1))
  [   LET DES = (INT(ENTRY.0)↑18) +. (ENTRY ↑ -18 ∧ AMASK)
      FOR I = 0 TO INT(ENTRY.0)-1 DO RV ELE(I,I,DES) := ONE
      RETURN ]I
CASE MTRNCMD: [T
  LET ENTRYC = RV (RV LIN↑-36)
  LET ENTRYA = RV LIN.1
  DIMMED(ENTRYC)
  DIMMED(ENTRYA)
  UNLESS ENTRYC ↑ -36 = ENTRYA ↑-36 = 1 DO RERROR(
        ≡ ARRAY IS NOT A MATRIX*N≡)
  UNLESS ENTRYC.0 =. FDIV(ENTRYA.1,ENTRYA.0)∧ ENTRYA.0 =.
     FDIV(ENTRYC.1,ENTRYC.0) DO RERROR(≡ SIZES DO NOT MATCH*N≡)
  [   LET DESA = (INT(ENTRYA.0)↑18) +. (ENTRYA↑ -18 ∧ AMASK)
      LET DESC = (INT(ENTRYC.0)↑18) +. (ENTRYC↑ -18 ∧ AMASK)
      LET SZ = INT(ENTRYC.1)
      // IF C AND A ARE THE SAME ARRAY GET SOME SCRATCH SPACE
      IF ENTRYA =.ENTRYC DO [
```

```
          DESC := GETSPACE(SZ)+. (DESC ∧ NAMASK)
          RELEASE := (SZ ↑ 18) +. (DESC ∧ AMASK) ]
      FOR I = 0 TO (DESA↑-18)- 1 DO
      FOR J = 0 TO INT(FDIV(ENTRYA.1,ENTRYA.0))-1 DO
          RV ELE(J,I,DESC) := RV ELE(I,J,DESA)
// IF SCRATCH SPACE WAS USED COPY TO ARRAY AND RELEASE SPACE
      IF ENTRYA =. ENTRYC DO [
          COPY(DESC,ENTRYC↑ -18,SZ)
          RELEASE := 0
          GIVESPACE(DESC,SZ) ]
      RETURN  ]T
CASE MADDCMD: CASE MSUBCMD: [A
    LET ENTRYC = RV (RV LIN ↑ -36)
    LET ENTRYA = RV (LIN.1 ↑ -18)
    LET ENTRYB = RV LIN.1
    DIMMED(ENTRYA)
    DIMMED(ENTRYB)
    DIMMED(ENTRYC)
// INSURE THAT THE SHAPES MATCH
    UNLESS ENTRYA↑-36 = ENTRYB↑-36 = ENTRYC↑-36 DO RERROR(
        ≡ NUMBER OF DIMENSIONS DO NOT MATCH*N≡)
// INSURE THAT THE SIZES MATCH
    TEST ENTRYC↑-36 = 0 DO [ONE
        UNLESS (ENTRYC∧AMASK) = (ENTRYB∧AMASK) = (ENTRYA∧AMASK) DO
            RERROR(≡ LIST SIZES DO NOT MATCH*N≡)
        (LIN↑-54 = MADDCMD -> MATADD,MATSUB)(ENTRYA↑-18,
            ENTRYB↑-18,ENTRYC↑-18,ENTRYC∧AMASK) ]ONE
    ELSE [MULTI
        LET CT = ENTRYC↑-36
        FOR I = 0 TO CT DO UNLESS ENTRYC.I =. ENTRYB.I =. ENTRYA.I DO
            RERROR(≡ ARRAY SIZES DO NOT MATCH*N≡)
        (LIN↑-54 = MADDCMD -> MATADD,MATSUB)(ENTRYA↑-18,
            ENTRYB↑-18,ENTRYC↑-18,INT(ENTRYC.CT))]MULTI
    RETURN  ]A
CASE MSMLCMD: [SM
    LET ENTRYC = RV(RV LIN ↑-36)
    LET ENTRYB = RV LIN.1
    LET MULTIPLIER = EXPRESS(LIN+2)
    LET CT = ENTRYC ↑ -36
    DIMMED(ENTRYC)
    DIMMED(ENTRYB)
    UNLESS CT = ENTRYB ↑ -36 DO RERROR(
        ≡ NUMBER OF DIMENSIONS DO NOT MATCH*N≡)
    TEST CT = 0 DO [ONE
        UNLESS ENTRYB∧AMASK = ENTRYC∧AMASK DO RERROR(
            ≡ LIST SIZES DO NOT MATCH*N≡)
        MATSML(MULTIPLIER,ENTRYB↑-18,ENTRYC↑-18,ENTRYC∧AMASK) ]ONE
    ELSE [MULTI
        FOR I = 0 TO CT DO UNLESS ENTRYB.I =. ENTRYC.I DO RERROR(
            ≡ ARRAY SIZES DO NOT MATCH*N≡)
        MATSML(MULTIPLIER,ENTRYB↑-18,ENTRYC↑-18,INT(ENTRYC.CT)) ]MULTI
    RETURN ]SM
CASE MMLTCMD: [M
    LET ENTRYC = RV(RV LIN↑-36)
    LET ENTRYB = RV LIN.1
    LET ENTRYA = RV (LIN.1↑-18)
```

```
    DIMMED(ENTRYA)
    DIMMED(ENTRYB)
    DIMMED(ENTRYC)
    UNLESS ENTRYC↑-36 = ENTRYB↑-36 = ENTRYA↑-36 = 1 DO RERROR(
        ≡ ARRAY IS NOT A MATRIX*N≡)
  [ LET ROWCTA = INT(ENTRYA.0)
    LET ROWCTB = INT(ENTRYB.0)
    LET ROWCTC = INT(ENTRYC.0)
    LET COLCTA = INT(FDIV(ENTRYA.1,ENTRYA.0))
    LET COLCTB = INT(FDIV(ENTRYB.1,ENTRYB.0))
    LET COLCTC = INT(FDIV(ENTRYC.1,ENTRYC.0))
    LET ROWSZA = COLCTA
    LET ROWSZB = COLCTB
    LET ROWSZC = COLCTC
    LET COLSZA = ROWCTA
    LET COLSZB = ROWCTB
    LET COLSZC = ROWCTC
//  CHECK TO SEE THAT THE DIMENSIONS MATCH
    UNLESS ROWSZA = COLSZB ∧ COLSZA = COLSZC ∧ ROWSZB = ROWSZC DO
        RERROR(≡ MATRIX SIZES DO NOT MATCH*N≡)
// FORM DESCRIPTOR FOR C AND BASES FOR A NAD B
 [  LET DESC =(ENTRYC↑-18 ∧ AMASK) +. (COLSZC↑18)
    LET BASEB = ENTRYB↑-18 ∧ AMASK
    LET BASEA = ENTRYA↑-18 ∧ AMASK
// IF C IS THE SAME MATRIX AS A OR B GET SOME SCRATCH SPACE
    IF ENTRYC =. ENTRYA ∨ ENTRYC =. ENTRYB DO [
        DESC := GETSPACE(ROWSZC*COLSZC) +.(COLSZC↑18)
        RELEASE := (ROWSZC*COLSZC↑18) +. (DESC∧AMASK) ]
// GET VALUES IN C
    FOR I = 0 TO ROWCTC-1 DO
    FOR J = 0 TO COLCTC-1 DO
        RV ELE(I,J,DESC) := INPROD(BASEB+J*COLSZB,BASEA+I,COLSZA,ROWSZA)
//  IF SCRATCH SPACE WAS USED COPY TO NEW ARRAY AND RELEASE OLD
    IF ENTRYC =. ENTRYA ∨ ENTRYC =. ENTRYB DO [
        COPY(DESC,ENTRYC↑-18,RELEASE↑-18)
        GIVESPACE(RELEASE,RELEASE↑-18)
        RELEASE := 0 ]
    RETURN ]M
CASE MPRICMD: [P // PRINT A MATRIX
    LET ENTRY = RV(RV LIN ↑ -36)
    LET SEP = RV LIN ↑ -54
    DIMMED(ENTRY)
    SWITCHON ENTRY ↑ -36 INTO [S
    CASE 0 :  [O //PRINT A ONE DIMENSIONAL MATRIX
    LET PT = ENTRY ↑ -18
        FOR I = 0 TO (ENTRY ∧ AMASK) - 1 DO  [L1
            TEST PT.I =. INDEF DO WRITES(≡UNINT*N≡)
            ELSE [ WRITEF(PT.I) ; WRITE(+*N+) ]
            IF INTERUPTED DO RERROR(≡ PANIC*N≡) ]L1
        RETURN  ]O
    CASE 1 :  [1
    LET DESC = (INT(ENTRY.0)↑18) +. (ENTRY↑-18 ∧ AMASK)
    FOR I = 0 TO INT(ENTRY.0) - 1 DO [ROW
        FOR J = 0 TO INT(FDIV(ENTRY.1,ENTRY.0))-1 DO [COL
            LET VAL = RV ELE(I,J,DESC)
            TEST VAL =. INDEF DO [ CHARCNT := CHARCNT + 5
```

```
            WRITES(ΞUNINTΞ)  ] ELSE WRITEF(VAL)
        SWITCHON SEP INTO [SP
            DEFAULT: SYSERR(Ξ BAD SEP*NΞ)
        CASE 0:
        CASE 1: // COMMA
            WRITE(↓*T↓)
            GOTO MPRI1
        CASE 2: // SEMI COLON
            WRITE(↓ ↓)
            WRITE(↓ ↓)
        CASE 3: // COLON
        MPRI1: IF INTERUPTED DO RERROR(Ξ PANIC*NΞ)  ]COL
    WRITE(↓*N↓)  ]ROW
    RETURN  ]1
CASE 2 :  [2 //  A THREE DIMENSIONAL MATRIX
LET SEP = RV LIN ↑ -54
LET DESC = (INT(ENTRY.1)↑36) +. (INT(ENTRY.0)↑18) +. (ENTRY↑-18∧AMAS
    FOR K = 0 TO INT(FDIV(ENTRY.2,ENTRY.1)) - 1  DO  [L3
    FOR I = 0 TO INT(ENTRY.0) - 1 DO [L1
    FOR J = 0 TO INT(FDIV(ENTRY.1,ENTRY.0)) - 1 DO [L2
        LET VAL = RV ELE3(I,J,K,DESC)
        TEST VAL =. INDEF DO [ WRITES(ΞUNINTΞ) ; CHARCNT := CHARCNT+
        ELSE  WRITEF(VAL)
        TEST SEP < 2 DO WRITE(↓*T↓) ELSE IF SEP = 2 DO
            [ WRITE(↓ ↓) ; WRITE(↓ ↓)  ]
        IF INTERUPTED DO RERROR(Ξ PANIC*NΞ)  ]L2
    WRITE(↓*N↓) ]L1
    WRITE(↓*N↓) ; WRITE(↓*N↓)  ]L3
    RETURN  ]2
DEFAULT : RERROR(Ξ TOO MANY DIMENSIONS TO PRINT*NΞ)  ]P
CASE MREADCMD:  [R
LET ENTRY = RV (RV LIN ↑ -36)
DIMMED(ENTRY)
DATAINIT()
[ LET DESC = (INT(ENTRY.0)↑18) +. (ENTRY↑-18∧ AMASK)
SWITCHON ENTRY ↑-36 INTO [DM
CASE 0: // ONE DIMENSION
    FOR I = 0 TO (ENTRY ∧ AMASK) - 1 DO
        DATASTORE((ENTRY ↑ -18) + I, ARITHEXP)
    GOTO OK
CASE 1:  // TWO DIMENSIONS
    FOR I = 0 TO INT(ENTRY.0)-1 DO
        FOR J = 0 TO INT(FDIV(ENTRY.1,ENTRY.0))-1 DO
            DATASTORE(ELE(I,J,DESC),ARITHEXP)
    GOTO OK
CASE 2:  // THREE DIMENSIONS
    DESC := (INT(ENTRY.1)↑36) +. DESC
    FOR K = 0 TO INT(FDIV(ENTRY.2,ENTRY.1))-1 DO
        FOR I = 0 TO INT(ENTRY.0) - 1 DO
            FOR J = 0 TO INT(FDIV(ENTRY.1,ENTRY.0))-1 DO
                DATASTORE(ELE3(I,J,K,DESC),ARITHEXP)
    GOTO OK
DEFAULT:  RERROR(Ξ MORE THAN THREE DIMENSIONS*NΞ)  ]DM
OK: DATAOK()
    RETURN  ]R
DEFAULT: SYSERR(Ξ EXECUTING BAD LINE*NΞ)
```

```
    ]EXC      // END OF EXECUTE
//DECK BCPLGD
//   DECLARATIONS TO LINK WITH HEAD

GLOBAL [ // TO LINK WITH FSP AND IO
          // GLOBALS 1-4 ARE FREE
          FLIST      :   5  // FREESPACE PACKAGE POINTER
          OUTPUT     :   6  // STANDARD OUTPUT STREAM
          C6TO7      :   7  // DISPLAY CODE ASCII CONVERSION VECTOR
          C7TO6      :   8  // ASCII DISPLAY CODE  CONVERSION VECTOR
          IOBASE     :   9  // OPEN STREAM LIST POINTER
          MONITOR    :  10
     ]

EXTERNAL [
          NEWVEC           // NEWVEC(SIZEWANTED)
          RETVEC           // RETVEC(V,SIZEOFV)
          PACKSTRING       // (VECTOR,STRING)
          UNPACKSTRING     // (STRING,VECTOR)
          INITIALIZEIO
          FINDINPUT        //  (EFNAME,USERNAME=)
          CREATEOUTPUT     //  (EFNAME,USERNAME=)
          READCH           // (STREAM,LV CH)
          WRITECH          // (STREAM,CH)
          WRITES           // OUTPUT:=STREAM; WRITES(STRING)
          WRITEN           // OUTPUT:=STREAM; WRITEN(NUMBER)
          WRITEO           // OUTPUT:=STREAM; WRITEO(NUMBER)
          READN            // NUMBER READER N:=READN(STREAM)
          TRACE
          TRACECALL
          TRACERETURN
          DUMP
          ENDREAD          // CLOSE INPUT STREAM
          ENDWRITE         // CLOSE OUTPUT STREAM
          IOERROR          // GENERAL IOERROR ROUTINE
          UNPACKTO         //  (STR,VEC,BREAKCH)
          WRITEVEC
          READVEC
          FLUSH            // FORCE OUT A LINE WITHOUT CR
          BCDWORD
          ASCII
          ENDOFSTREAM
          CLOSEALL
          ABORT
     ]

MANIFEST [ AMASK = 777777B   ; ENDOFSTREAMCH =144B
          BUFFERSIZE = 136
           ESCAPECH = 173B ]
EXTERNAL [   // NECESSAR ROUTINTE FOR ERRORS
          RERROR
          SYSERR
   ]
 MANIFEST
  [ AMASK=777777B      // ADDRESS MASK
    CPENR=142B          // OPEN FOR READING
```

```
    OPENRNR=102B         // OPEN FOR READING (NO REWIND)
    OPENWNR=106B         // OPEN FOR WRITINT (NO REWIND)
    READ=12B
    WRITE=16B
    WRITER=26B           // WRITE FOR
    CLOSENR=132B         // CLOSE FILE (NO RE-WIND)
     REWIND=52B          // REWIND FILE
    ALTERNR=122B
    EORBIT=20B           // EOR BIT IN FET.0
     CIO=311172B
    OPE=1720C52B        // PP PROGRAM FOR OPEN
    MSG=15230728
   ]

 LET INITIALIZEIO(V, SIZE) BE
  [1 IOBASE := V
        C7T06:=TABLE
             00B, 00B, 00B, 00B, 00B, 00B, 00B, 00B,    //  00
             00B, 00B, 00B, 00B, 00B, 00B, 00B, 00B,    //  01
             00B, 00B, 00B, 00B, 00B, 00B, 00B, 00B,    //  02
             00B, 00B, 00B, 00B, 00B, 00B, 00B, 00B,    //  03
             55B, 66B, 60B, 64B, 53B, 63B, 67B, 71B,    //  04
             51B, 52B, 47B, 45B, 56B, 46B, 57B, 50B,    //  05
             33B, 34B, 35B, 36B, 37B, 40B, 41B, 42B,    //  06
             43B, 44B, 63B, 77B, 72B, 54B, 73B, 75B,    //  07
             74B, 01B, 02B, 03B, 04B, 05B, 06B, 07B,    //  10
             10B, 11B, 12B, 13B, 14B, 15B, 16B, 17B,    //  11
             20B, 21B, 22B, 23B, 24B, 25B, 26B, 27B,    //  12
             30B, 31B, 32B, 61B, 76B, 62B, 70B, 65B,    //  13
             71B, 01B, 02B, 03B, 04B, 05B, 06B, 07B,    //  14
             10B, 11B, 12B, 13B, 14B, 15B, 16B, 17B,    //  15
             20B, 21B, 22B, 23B, 24B, 25B, 26B, 27B,    //  16
             30B, 31B, 32B, 61B, 66B, 62B, 46B, 55B     //  17

        C6T07:=TABLE
             000B, 101B, 102B, 103B, 104B, 105B, 106B, 107B,
   [ LET I=1
     UNTIL I> SIZE-BUFFERSIZE+1  DO
       [ IOBASE.I := 0
         IOBASE.(I+1) := 1 LSHIFT 18 v LV IOBASE.(I+7)
         IOBASE.(I+4) := LV IOBASE.(I+BUFFERSIZE)
        I := I+BUFFERSIZE   ]
     IOBASE.0 := IOBASE+I      ]1

AND FINDINPUT(LFN) =
    VALOF [1 LET FET = IOBASE + 1
             [ LET I=FET
               UNTIL I>IOBASE.0-BUFFERSIZE DO
         [
               IF LFN=(I.0&-AMASK) DO
                    RERROR(= ATTEMPT TO OPEN INPUT FILE TWICE*N=)
               I := I+BUFFERSIZE   ]
          ]
             UNTIL RV FET = 0 DO
               [2 FET:=FET+BUFFERSIZE
                 IF FET GE IOBASE.0 DO
```

```
                  RERROR(≡ ATTEMPT TO OPEN TOO MANY FILES*N≡)  ]2     BCPLIO
            FET.0 := LFN                                              BCPLIO
          FET.2, FET.3 := FET.1∧AMASK, FET.1∧AMASK                    BCPLIO
        PPCALL(CIO,READ,FET)                                          BCPLIO
           IF (FET.2∧AMASK)=(FET.3∧AMASK) DO RERROR(                  BCPLIO
                  ≡ EMPTY INPUT FILE*N≡)                              BCPLIO
           FET.5 := 54                                                BCPLIO
       FET.6:=-1                                                      BCPLIO
           RESULTIS FET    ]1                                         BCPLIO
                                                                      BCPLIO
AND READVEC(FET, V, N, EORL,EORC) BE                                  BCPLIO
    [ LET I=FET.2∧AMASK                                               BCPLIO
      AND O=FET.3∧AMASK                                               BCPLIO
      AND LIM=FET.4∧AMASK                                             BCPLIO
      FOR J = 0 TO N-1 DO                                             BCPLIO
        [ IF O=I DO                                                   BCPLIO
            [ FET.3 := FET.3 ∧ ¬ AMASK ∨ O                            BCPLIO
            IF (FET.0∧EORBIT)≠0 DO [ RV EORC:=J; GOTO EORL  ]         BCPLIO
              PPCALL(CIO,READ,FET)                                    BCPLIO
              I := FET.2 ∧ AMASK  ]                                   BCPLIO
          V.J := RV O                                                 BCPLIO
          O := O+1; IF O=LIM DO O := FET.1∧AMASK  ]                   BCPLIO
      FET.3 := FET.3 ∧ ¬AMASK ∨ O                                     BCPLIO
      RETURN  ]                                                       BCPLIO
                                                                      BCPLIO
AND READCH(FET,CH) BE                                                 BCPLIO
    [1 LET INCWRD(I,FET) BE                                           BCPLIO
          [ FET.3 := FET.3 ∧ ¬AMASK ∨                                 BCPLIO
                     (I=(FET.4∧AMASK) -> FET.1∧AMASK, I )             BCPLIO
            FET.5 := 54; RETURN  ]                                    BCPLIO
       AND NO=FET.3∧AMASK                                             BCPLIO
       IF (FET.2∧AMASK) = NO DO                                       BCPLIO
         TEST (FET.0 ∧ EORBIT) ≠ 0                                    BCPLIO
           THEN [ RV CH := ENDOFSTREAMCH; RETURN ]                    BCPLIO
           OR PPCALL(CIO,READ,FET)                                    BCPLIO
      [ LET C = RV NO RSHIFT FET.5 ∧ 77B                              BCPLIO
         TEST C=0                                                     BCPLIO
           THEN [ TEST FET.5=0 THEN [ INCWRD(NO+1,FET); READCH(FET,CH)] BCPLIO
                                OR INCWRD(NO+1,FET)                   BCPLIO
                 RV CH := '*N'   ]                                    BCPLIO
         OR [ RV CH := C6TO7.C                                        BCPLIO
               FET.5 := FET.5 - 6                                     BCPLIO
                 IF FET.5 LS 0 DO INCWRD(NO+1, FET)    ]1             BCPLIO
                                                                      BCPLIO
AND ENDREAD(FET) BE                                                   BCPLIO
   [ UNTIL (FET.0 ∧ EORBIT) ≠ 0 DO                                    BCPLIO
     [ FET.3 := FET.2                                                 BCPLIO
       PPCALL(CIO,READ,FET) ]                                         BCPLIO
    FET.0 := 0   ]                                                    BCPLIO
                                                                      BCPLIO
AND CREATEOUTPUT(LFN) =                                               BCPLIO
    VALOF [1 LET FET = IOBASE+1                                       BCPLIO
           [ LET I=FET                                                BCPLIO
             UNTIL I>IOBASE.0-BUFFERSIZE DO                           BCPLIO
        [                                                             BCPLIO
              IF LFN=(I.0∧-AMASK) DO                                  BCPLIO
```

```
       TEST I.6 >C THEN [ I.6:=I.6+1; RESULTIS I ]
       OR RERROR(≡ ATTEMPT TO OPEN INPUT FILE FOR WRITING*N≡)
            I := I+BUFFERSIZE   ]
         ]
            UNTIL FET.0 = 0 DO
              [2 FET:=FET+BUFFERSIZE
                IF FET GE IOBASE.0 DO
                  RERROR(≡ ATTEMPT TO OPEN TOO MANY FILES*N≡)  ]2
            FET.0 := LFN
            FET.2,FET.3 := FET.1∧AMASK, FET.1∧AMASK
            FET.5 := 54
            RV(FET.2):=0
           FET.6:=1
            IF LFN=BCDWORD(≡DAYFILE≡) RESULTIS FET
            PPCALL(OPE,ALTERNR,FET)
            FET.0 := LFN ∨ WRITE ∨ 1
            RESULTIS FET    ]1


AND WRITECH(FET,CH) BE
     [1 LET NEWORD(I,FET) =
           VALOF [ FET.5 := 54
                 RESULTIS I=(FET.4∧AMASK) -> FET.1∧AMASK , I   ]
        AND NI = FET.2 ∧ AMASK
        CH := CH ∧ 177B
        [ LET J =
          VALOF [2 TEST CH=↓*N↓
                THEN TEST FET.0=BCDWORD(≡DAYFILE≡) THEN
                   [ LET T=FET.3↑30
                     RV (FET.2+1) := 0
                     PPCALL(MSG,0,LV T)
                     FET.2,RV(FET.2) := FET.3,0
                     FET.5 := 54
                     RETURN ]
                OR TEST FET.5=0
                       THEN [ WRITECH(FET,↓ ↓)
                              RESULTIS NEWORD((FET.2∧AMASK)+1,FET)]
                      OR RESULTIS NEWORD(NI+1,FET) OR
                TEST CH = ↓*T↓
                  THEN [ LET K=FET.5
                       UNTIL K < 0 DO
                         [ RV NI := RV NI ∨ 55B LSHIFT K
                           K:= K-6  ]
                       RESULTIS NEWORD(NI+1, FET) ]  OR
                [ RV NI := RV NI ∨ C7TO6.CH LSHIFT FET.5
                  FET.5 := FET.5 - 6
                  IF FET.5  GE  0 RETURN
                  RESULTIS NEWORD(NI+1, FET)  ]2
          IF J = (FET.3∧AMASK) DO PPCALL(CIO,WRITE,FET)
          RV J := 0; FET.2 := FET.2 ∧ -AMASK ∨ J
          IF CH = ↓*N↓ DO WRITECH(FET,↓ ↓)
          RETURN    ]]

AND ENDWRITE(FET) BE
      [1 FET.6 := FET.6 - 1
         IF FET.6 = 0 DO
```

```
          [ UNLESS FET.5 = 54 DO WRITECH(FET,**N*)
            TEST FET.0∧-AMASK=.BCDWORD(=DAYFILE=)
                 THEN UNLESS FET.2=FET.3 DO WRITECH(FET,**N*)
                 ELSE PPCALL(CIO,WRITER,FET)
            FET.0 := 0 ]1

 AND ENDOFSTREAM(FET) = (FET.2∧AMASK)=(FET.3∧AMASK)
                        ∧ (FET.0∧EORBIT)≠0

 AND CLOSEALL() BE
     [ LET I=IOBASE+1
      UNTIL I GE IOBASE.0 DO
      [ IF I.0≠0 DO
         TEST I.6<0
           THEN ENDREAD(I)
           OR ENDWRITE(I) REPEATUNTIL I.0=0
            I := I+BUFFERSIZE    ]  ]

 AND PPCALL(PROG,FUNC,FET) BE
     [ FET.0 := FET.0 ∧ -AMASK ∨ FUNC
        RV 1 := PROG LSHIFT 39 ∨ FET
        LOOP: UNLESS RV 1 = 0 GOTO LOOP ]

AND PACKSTRING(V,S) BE
    [1 LET I=49
       RV S := 0
       FOR J=1 TO V.0 DO
         [ RV S := RV S ∨ V.J LSHIFT I
           I := I-7
           IF I<0 DO
           [ S:=S+1 ; RV S:= 0; I:=49   ]1

AND UNPACKSTRING(S,V) BE
    [ LET I=0 AND J=49 AND C=0
      [ C:= RV S RSHIFT J ∧ 177B
        IF C=140B BREAK
        I:=I+1
        V.I := C
        J:=J-7
        IF J<0 DO [ S:=S+1; J:=49 ] ]
      REPEAT
      V.0 := I  ]

 AND BCDWORD(S) = VALOF
        [ LET W = 0
           LET V = VEC 10
           UNPACKSTRING(S, V)
          [ LET I=54
            AND J=1
            UNTIL I<0 ∨ J>V.0 DO
              [ W := W ∨ C7TO6.(V.J) LSHIFT I
                I,J := I-6,J+1 ]  ]
           RESULTIS W  ]
AND ASCII(W,S) BE
    [ LET V= VEC 10 AND I=54 AND J=0
      [ LET C= W RSHIFT I ∧ 77B
```

```
        IF C=0 BREAK
        J := J+1
        V.J := C6TO7.C
        I := I-6   ]
      REPEATUNTIL I<0
      V.0 := J
      PACKSTRING(V,S)   ]

 AND WRITES(S) BE
        [ LET V = VEC 80
           UNPACKSTRING(S, V)
           FOR I = 1 TO V.(0) DO WRITECH(OUTPUT, V.(I))   ]

 AND WRITEN(N) BE
     [ IF N<0 THEN [ WRITECH(OUTPUT,↓-↓); N:=-N ]
       IF N=0 THEN [ WRITECH(OUTPUT,↓0↓); RETURN ]
     [ LET TENTAB = TABLE 1,10,100,1000,10000, 100000,1000000,
                          10000000,
                          100000000,
                          1000000000,
                          10000000000,
                          100000000000,
                          1000000000000,
                          10000000000000,
                          100000000000000,
                          1000000000000000,
                          10000000000000000,
                          100000000000000000
       AND I=1
       UNTIL TENTAB.I>N ∨ I>17 DO I:=I+1
       [ I:=I-1
         [ LET J=-1 AND X=TENTAB.I
           [ N:=N-X; J:=J+1 ] REPEATUNTIL N<0
           WRITECH(OUTPUT,J+↓0↓)
           N:=N+X ]]
       REPEATUNTIL I=0   ]]
LET RETVEC(V,M) BE
   [1 LET A=FLIST
      V := V ∧ AMASK
      IF A=0 [ FLIST:=V; RV V := M↑18∨V; RETURN ]
  L: [ LET MA = RV A ↑ -18
       AND B = RV A ∧ AMASK
       LET AL = A+MA
       AND VL=V+M
       IF AL<B -> AL LE V ∧ VL LE B, AL LE V ∨ VL LE B
       DO [ TEST AL=V
              DO[ RV A := RV A +. (M↑18); V:=A ]
              OR[ RV A:= RV A ∧ NOT AMASK ∨ V
                  RV V := M↑18 ∨ B ]
            IF VL=B
             DO[ RV V := RV B +. (RV V ∧ NOT AMASK)
                 FLIST := V ] // IN CASE IT WAS B
            RETURN ]
     A:=B; UNLESS A=FLIST GOTO L; SYSERR(
          ≡FREE SPACE DOUBLY RETURNED*N≡)   ]1
LET NEWVEC(M)=
```

```
        IOERROR          // GENERAL IOERROR ROUTINE                    BCPLGD   35
        UNPACKTO         //   (STR,VEC,BREAKCH)                        BCPLGD   36
        WRITEVEC                                                       BCPLGD   37
        READVEC                                                        BCPLGD   38
        FLUSH            // FORCE OUT A LINE WITHOUT CR                BCPLGD   39
        BCDWORD                                                        BCPLGD   40
        ASCII                                                          BCPLGD   41
        ENDOFSTREAM                                                    BCPLGD   42
        CLOSEALL                                                       BCPLGD   43
        ABORT                                                          BCPLGD   44
    ]                                                                  BCPLGD   45
                                                                       BCPLGD   46
MANIFEST [ AMASK = 777777B  ; ENDOFSTREAMCH =144B                      BCPLGD   47
        BUFFERSIZE = 136                                               BCPLGD   48
         ESCAPECH = 173B  ]                                            BCPLGD   49
    GLOBAL [GV     //  GLOBAL VARIABLES                                BASICGD   2
INPUT                  : 49                                            BASICGD   3
    FREE               : 50                                            BASICGD   4
    INPUTOPEN          : 51                                            BASICGD   5
    EXPLIMIT           : 52                                            BASICGD   6
// TWO HERE                                                            BASICGD   7
    HILINE             : 54                                            BASICGD   8
    LOWLINE            : 55                                            BASICGD   9
    FREELINE           : 56                                            BASICGD  10
    DATA               : 57                                            BASICGD  11
    TMPDATA            : 58                                            BASICGD  12
    DATACOUNT          :59                                             BASICGD  13
    TMPDATACNT         :60                                             BASICGD  14
    RETURNS            : 61                                            BASICGD  15
    PROG               : 62                                            BASICGD  16
    LASTPROG           : 63                                            BASICGD  17
    EXPSTOP            :64                                             BASICGD  18
    S                  : 65                                            BASICGD  19
    CHARCNT            :66                                             BASICGD  20
    SIG                : 67                                            BASICGD  21
    RUNNING            : 68                                            BASICGD  22
    TIME          :72                                                  BASICGD  23
    NEXT               : 75                                            BASICGD  24
    NEXTINPUT          : 76                                            BASICGD  25
    ENDINPUT           : 77                                            BASICGD  26
    STNUMBER           : 78                                            BASICGD  27
    NUMBER             : 79                                            BASICGD  28
    ACCEPT             : 80                                            BASICGD  29
    RECURSE            : 82                                            BASICGD  30
    TYPE               : 84                                            BASICGD  31
// 88,89 RESERVED FOR MATHLIB                                          BASICGD  32
// HERE IS 2                                                           BASICGD  33
    EXPTYPE            : 92                                            BASICGD  34
    CURSEED            : 93                                            BASICGD  35
    SEEDFLAG           : 94                                            BASICGD  36
    LIMIT              : 95                                            BASICGD  37
    IERROR             : 96                                            BASICGD  38
    RELEASE            : 97                                            BASICGD  39
        ]GV                                                            BASICGD  40
EXTERNAL  [GB       // GLOBAL BUFFERS AND VECTORS                      BASICGD  41
    VARIABLES                                                          BASICGD  42
```

```
    FUNCTIONS                                              BASICGD   43
    ARRAYS                                                 BASICGD   44
    LINES                                                  BASICGD   45
    EXPCALLS                                               BASICGD   46
    CHARBUF                                                BASICGD   47
    STRINGS                                                BASICGD   48
    STARRAYS                                               BASICGD   49
    IOBUF                                                  BASICGD   50
    INBUF                                                  BASICGD   51
    GLOBAL 0                                               BASICGD   52
        IGB                                                BASICGD   53
EXTERNAL  IGR         //  GLOBAL ROUTINES                  BASICGD   54
    COMPILE                                                BASICGD   55
    ADDRESS                                                BASICGD   56
    EXPGEN                                                 BASICGD   57
    LEXANA                                                 BASICGD   58
    CONVERT                                                BASICGD   59
    WRITEF                                                 BASICGD   60
    UNPACK                                                 BASICGD   61
    PACK                                                   BASICGD   62
    COMPACT                                                BASICGD   63
    LINUM                                                  BASICGD   64
    FINDLINE                                               BASICGD   65
    COPY                                                   BASICGD   66
    DATAOK                                                 BASICGD   67
    DATASTORE                                              BASICGD   68
    DATAINIT                                               BASICGD   69
    ALPHA                                                  BASICGD   70
    DIGIT                                                  BASICGD   71
    INT                                                    BASICGD   72
    FLOAT                                                  BASICGD   73
    FMULT                                                  BASICGD   74
    FDIV                                                   BASICGD   75
    FADD                                                   BASICGD   76
    CHKFDIV                                                BASICGD   77
    CHKPIV                                                 BASICGD   78
    SIMPLEADD                                              BASICGD   79
    EXECUTE                                                BASICGD   80
    MATEX                                                  BASICGD   81
    EXPRESS                                                BASICGD   82
    GIVESPACE  = ERETVECE                                  BASICGD   83
    GETSPACE   = ENEWVECE                                  BASICGD   84
    READLINE                                               BASICGD   85
    WRITE                                                  BASICGD   86
    EERROR                                                 BASICGD   87
    CERROR                                                 BASICGD   88
    RERROR                                                 BASICGD   89
    SYSERR                                                 BASICGD   90
    EXCHARS                                                BASICGD   91
    UNREC                                                  BASICGD   92
    FWDLINE                                                BASICGD   93
    FILL                                                   BASICGD   94
    COPYSTRING                                             BASICGD   95
    ELE                                                    BASICGD   96
    ELEB                                                   BASICGD   97
    MATADD                                                 BASICGD   98
```

```
    MATSUB
    MATSML
    INPROD
    NORM
    MATCMD
              ]GR
    MANIFEST  [MGC    //  SOME GENERAL CONSTANTS
    QUOTE             = ↓E↓
    AMASK             = 377777B
    NULLCH            = 140B
    NMASK             = -377777B
    NAMASK            = -377777B
    ZONESIZE          = 72/5
    LINESIZE          = 72
    ARITHEXP          = -1
    LOGEXP            = -2
    STRNGEXP          = -4
    HIADD             = 5000
    INDEF             = 60000000000000000000B
    ONE               = 17204000000000000000B
    TEN               = 17235000000000000000B
    RNDSEED           = 113434743B
    DELETE            = 3
    CHANGE            = 2
    INSERT            = 1
    DIRECT            = 4
    QUIT              = 5
    INTERUPTED = FALSE // SO THAT INTERUPT WILL BE DEFINED FOR MATEX
              ]MGC
MANIFEST   [MCMD      //  MANIFEST CONSTANTS FOR IDENTIFING LINES
    LETCMD   = 1
    GOTCMD   = 2
    IFCMD    = 3
    IFTCMD   = 3
    FORCMD   = 4
    NEXCMD   = 5
    REACMD   = 6
    INPCMD   = 7
    PRICMD   = 8
    DEFCMD   = 9
    REMCMD   = 10
    GOSCMD   = 12
    RETCMD   = 13
    ENDCMD   = 14
    STOCMD   = 15
    DATCMD   = 16
    RESCMD   = 17
    SIGCMD   = 18
    DIMCMD   = 19
    ONCMD    = 20
    MPRICMD  = 21
    MREACMD  = 22
    MTRNCMD  = 23
    MZERCMD  = 24
    MIDNCMD  = 25
    MCONCMD  = 26
```

```
    MADDCMD  = 27
    MSUBCMD  = 28
    MMLTCMD  = 29
    MSMLCMD  = 30
    MINVCMD  = 31
         ]MCMD
MANIFEST [MEC     // INDEXES INTO EXPCALLS
    ABS      = 1
    ATAN     = 2
    COSINE   = 3
    EXP      = 4
    INTEGER  = 5
    LOG      = 6
    SIGN     = 7
    SINE     = 8
    SQRT     = 9
    TAN      = 10
    RND      = 11
    ACS      = 12
    ASN      = 13
    LGT      = 14
    TIM      = 15
    ENDEXP   = 8 + 12
    ADD      = 8 + 13
    SUBTRAC  = 8 + 14
    MULT     = 8 + 15
    DIV      = 8 + 16
    NEG      = 8 + 17
    ANDCALL  = 8 + 18
    ORCALL   = 8 + 19
    EQUAL    = 8 + 20
    NEQUAL   = 8 + 21
    LESSTH   = 8 + 22
    LESS     = 8 + 22
    MORETH   = 8 + 23
    MORE     = 8 + 23
    LESSEQ   = 8 + 24
    MOREEQ   = 8 + 25
    STORE    = 8 + 26
    FETCH    = 8 + 27
    NMCNST   = 8 + 28
    ADCNST   = 8 + 29
    ONECNST  = 8 + 30
    FNCALL   = 8 + 31
    FRET     = 8 + 32
    PARVAL   = 8 + 33
    ARYCONT  = 8 + 34
    ARYADD   = 8 + 35
    EXOPNET  = 8 + 36
    EXPO     = 8 + 36
    LSTCONT  = 8 + 37
    LSTADD   = 8 + 38
    STLSTC   = 8 + 39
    STLSTA   = 8 + 40
    STOREST  = 8 + 41
    FETCHST  = 8 + 42
```

```
    STEQUAL  = 8 + 43
    STCNST   = 8 + 44
    STNEQUAL = 8 + 45
    PICNST   = 8 + 46
    STMORETH = 8 + 47
    STMOREEQ = 8 + 48
    STLESSTH = 8 + 49
    STLESSEQ = 8 + 50
          ]MEC
MANIFEST [MPT    //PARSE TYPE CONSTANTS
    PTPLUS    = 1
    PTMULT   = 2
    PTEXPO   = 3
    PTROP    = 4
    PTCM     = 5
    PTRPREN  = 6
    PTLPREN  = 7
    PTAND    = 8
    PTOR     = 9
    PTNOT    = 10
    PTFNC    = 11
    PTID     = 12
    PTARY    = 13
    PTEND    = 14
          ]MPT
MANIFEST   [MGT    //  GENERATING TYPE CONSTANTS
    GTPLUS   = 1
    GTMINUS  = 2
    GTAOP    = 3
    GTROP    = 4
    GTLOP    = 5
    GTCM     = 6
    GTRPREN  = 7
    GTLPREN  = 8
    GTNUM    = 9
    GTFNC    = 10
    GTARY    = 11
    GTVAR    = 12
    GTEND    = 14
    GTNOT    = 15
    GTSTLST  = 16
    GTSTVAR  = 17
    GTSTNUM  = 18
    GTPI     = 21
          ]MGT
MANIFEST [ ENDMARK  = (PTEND ASHIFT 53)+(GTEND ASHIFT 46) ]
LET SIMPLEADD() = VALOF [SA
    LET T = S.0 - ↓A↓ + VARIABLES
    EXPTYPE := ARITHEXP
    UNLESS ALPHA(S.0) DO CERROR(≡ SIMPLEADDRESS MISSING*N≡)
    TEST DIGIT(S.1) DO [
        T := T + (S.1 - ↓0↓) * 26
        S := S + 2 ]
    ELSE TEST S.1 = ↓$↓ DO [
        EXPTYPE := STRNGEXP
        T := (S.0 - ↓A↓)*2 + STRINGS
```

```
        S := S + 2 ]
    ELSE [ S := S + 1
        T := T + 260  ]
    RESULTIS T ]SA
LET DIGIT(D) = ↓0↓LE D LE ↓9↓ -> TRUE,FALSE
LET ALPHA(D) = ↓A↓LE D LE ↓Z↓ -> TRUE,FALSE
 LET LINUM() = VALOF [LN
    LET VAL = 0
    WHILE DIGIT(RV S) DO [
    VAL := VAL * 10 +.(RV S - ↓0↓)
    S := S + 1  ]
    IF VAL = 0  DO CERROR(≡ LINE NUMBER MISSING*N≡)
    IF VAL >. 777777B DO CERROR(≡LINE NUMBER TOO BIG*N≡)
    RESULTIS VAL ]LN
LET FINDLINE(LN) = VALOF [FL
    FOR I = LINES TO FREELINE - 1 DO
    IF (RV I ↑-36 ∧ 777777B) = LN THEN RESULTIS I
    RESULTIS 0  ]FL
LET WRITE(CH) BE [W
  SWITCHON CH INTO [
    CASE ↓*T↓ :
    IF ZONESIZE*5 - CHARCNT LE ZONESIZE GOTO DONE
    WRITE(↓ ↓) REPEATUNTIL CHARCNT REM ZONESIZE = 0
    RETURN
    CASE ↓*N↓ : DONE :
    WRITECH(OUTPUT,↓*N↓)
    CHARCNT := 0
    RETURN
    DEFAULT :
    WRITECH(OUTPUT,CH)
    CHARCNT := CHARCNT + 1
    IF CHARCNT GE LINESIZE DO GOTO DONE ]W
LET CERROR(ST) BE [CE
    WRITES(≡ ERROR ≡)
    WRITES(ST)
    GOTO NEXT ]CE
LET SYSERR(N) BE [SE
    WRITES(≡SYSERR ≡)
    TEST N < 100B DO [ WRITEN(N);WRITE(↓*N↓) ]
    ELSE WRITES(N)
    RERROR(≡ BASIC BLEW IT*N≡)  ]SE
LET EXCHARS()BE[ UNLESS RV S = ↓*N↓ DO CERROR(≡ EXTRA CHARACTERS*N≡) ]
LET UNREC(T) BE [ IF T DO CERROR(≡ UNRECOGNIZABLE COMMAND*N≡)  ]
LET FILL(V,VAL,SZ) BE [FL
  FOR I = V ∧ AMASK TO V+SZ-1 DO RV I := VAL ]FL
LET EERROR(N) BE [EER
SWITCHON N INTO [
CASE 1  : RERROR(≡ OVERFLOW WHEN ADDING*N≡)
CASE 2  : RERROR(≡ OVERFLOW WHEN SUBTRACTING*N≡)
CASE 3  : RERROR(≡ OVERFLOW WHEN MULTIPLYING*N≡)
CASE 4  : RERROR(≡ OVERFLOW WHEN DIVIDING*N≡)
CASE 5  : RERROR(≡ EXPONENT OR ARGUMENT OF EXP TOO BIG*N≡)
CASE 6  : RERROR(≡ UNDEFINED FUNCTION CALLED*N≡)
CASE 8  : RERROR(≡ UNDIMENSIONED ARRAY REFERENCED*N≡)
CASE 9  : RERROR(≡ ARRAY SUBSCRIPT LESS THAN ONE*N≡)
CASE 10 : RERROR(≡ OUT OF BOUNDS ARRAY REFERENCE*N≡)
```

```
CASE 11 : RERROR(Ξ UNINITIALIZED ARRAY ELEMENT USED*NΞ)                MISC1
CASE 12 : RERROR(Ξ UNINITIALIZED VARIABLE USED*NΞ)                     MISC1
CASE 13 : RERROR(Ξ ZERO RAISED TO THE ZERO POWER*NΞ)                   MISC1
CASE 14 : RERROR(Ξ NEGATIVE NUMBER RAISED TO NON INTEGER POWER*NΞ)     MISC1
CASE 15 : RERROR(Ξ ASN OR ACS  ARGUMENT OUT OF RANGE*NΞ)               MISC1
CASE 16 : RERROR(Ξ DIVISION BY ZERO*NΞ)                                MISC1
CASE 17 : RERROR(Ξ WRONG NUMBER OF SUBSCRIPTS*NΞ)                      MISC1
CASE 18 : RERROR(Ξ UNINITIALIZED LIST ELEMENT USED*NΞ)                 MISC1
CASE 19 : RERROR(Ξ NEGATIVE LIST SUBSCRIPT*NΞ)                         MISC1
CASE 20 : RERROR(Ξ LIST SUBSCRIPT TOO BIG*NΞ)                          MISC1
CASE 22 : RERROR(Ξ RECURSIVE FUNCTION CALLED*NΞ)                       MISC1
CASE 21 : RERROR(Ξ UNIMPLEMENTED FEATURE*NΞ)                           MISC1
CASE 23 : RERROR(Ξ SIN OR COS ARGUMENT TOO BIG*NΞ)                     MISC1
CASE 24 : RERROR(Ξ TAN ARGUMENT TOO BIG*NΞ)                            MISC1
CASE 25 : RERROR(Ξ LOG ARGUMENT IS ZERO*NΞ)                            MISC1
CASE 26 : RERROR(Ξ LOG ARGUMENT IS NEGATIVE*NΞ)                        MISC1
CASE 27 : RERROR(Ξ SQR ARGUMENT IS NEGATIVE*NΞ)                        MISC1
CASE 28 : RERROR(Ξ UNINTIALIZED STRING VARIABLE USED*NΞ)               MISC1
CASE 29 : RERROR(Ξ UNINTIALIZED STRING LIST ELEMENT USED*NΞ)           MISC1
DEFAULT : RERROR(Ξ SYSTEM ERROR*NΞ) ]EER                               MISC1
LET CONVERT(ERR) = VALOF  [C                                           MISC1
    LET EXPO,VAL,BASE,PYET,VS,ES = 0,0,ONE,FALSE,TRUE,TRUE             MISC1
TEST RV S = ↓+↓ DO S := S + 1                                          MISC1
ELSE IF RV S = ↓-↓ DO [ S:= S + 1                                      MISC1
    VS := FALSE   ]                                                    MISC1
         [1                                                            MISC1
    IF RV S = ↓.↓ DO                                                   MISC1
      TEST PYET DO BREAK ELSE [ PYET := TRUE                           MISC1
          S := S+ 1 ]                                                  MISC1
      UNLESS DIGIT(RV S) DO BREAK                                      MISC1
      TEST PYET DO [                                                   MISC1
        BASE := FDIV(BASE,TEN)                                         MISC1
        VAL := FADD(VAL,FMULT(FLCAT(RV S - ↓0↓),BASE)) ]               MISC1
    ELSE  VAL := FADD(FMULT(VAL,TEN),FLOAT(RV S - ↓0↓))                MISC1
    S  := S + 1 ]1 REPEAT                                              MISC1
    IF RV S = ↓E↓ DO [2                                                MISC1
      S := S + 1                                                       MISC1
      TEST RV S = ↓-↓ DO [  ES := FALSE                                MISC1
                            S := S + 1  ]                              MISC1
      ELSE IF RV S = ↓+↓ DO S := S + 1                                 MISC1
            WHILE DIGIT(RV S) [3                                       MISC1
        EXPO := EXPO * 10 + RV S - ↓0↓                                 MISC1
        S := S + 1                                                     MISC1
        IF EXPO GE 512 GOTO OUT  ]2                                    MISC1
    RESULTIS PACK(VS-> VAL,-VAL , ES->EXPO,-EXPO , OUT)                MISC1
 OUT:  ERR(Ξ EXPCNET OUT OF RANGE*NΞ)  ]C                              MISC1
LET WRITEF(FPN) BE  [F                                                 MISC1
LET EXPO,COEF = 0,0                                                    MISC1
  LET D = VEC 14                                                       MISC1
 LET J = SIG                                                           MISC1
 // OUTPUT THE SIGN                                                    MISC1
    TEST FPN DO [ FPN := -FPN; WRITE(↓-↓) ]                            MISC1
    CR WRITE(↓ ↓)                                                      MISC1
 // IF VALUE IS ZERO HANDLE AND RETURN                                 MISC1
//  IF EXPONENT IS ZERO PRINT AS 0 SINCE IT WILL BE 0 TO DIVIDE/MULTIPLY MISC1
    IF FPN↑-48 = 0 DO  [ WRITE(↓0↓) ; RETURN ]                         MISC1
```

```
// GET DECIMAL EXPONENT AND COEFFICIENT
   UNPACK(FPN,LV EXPO)
   EXPO  := -(EXPO - 13)
   COEF := COEF + SIG.TABLE 0
       ,500000000000
       ,50000000000
       ,5000000000
       ,500000000
       ,50000000
       ,5000000
       ,500000
       ,50000
       ,5000
       ,500
       ,50
       ,5
   IF COEF LGE 1000000000000 DO [
       COEF := ( COEF ↑ -1 ) / 5
       EXPO := EXPO+1   ]
// STORE THE DECIMAL CHARACTERS FOR THE DIGITS
   FOR I = 0 TO 14 DO [
        D.(14-I) := COEF REM 10 + ↓0↓
   COEF := COEF/10 ]
// FIND LAST SIGNIFICANT NON-ZERO DIGIT
   WHILE D.J = ↓0↓ ∧(EXPO + 1 > 6 ∨ J ≠ EXPO + 1) DO J := J-1
// DECIDE PROPER FORMAT
   TEST EXPO LS -2 LOGOR EXPO GR 5 DO [EX
   LET EXPODIGIT = FALSE
// OUTPUT IN E FORMAT
     WRITE(D.1)
     WRITE(↓.↓)
   [A  LET I = 2
     [  WRITE(D.I)
          I := I + 1 ]  REPEATUNTIL I GR J  ]A
     WRITE(↓E↓)
     TEST EXPO > 0 DO WRITE(↓+↓) ELSE [ WRITE(↓-↓);EXPO := -EXPO ]
   IF EXPO GE 100 [100
       WRITE(EXPO/100 + ↓0↓)
       EXPO := EXPO REM 100
       EXPODIGIT := TRUE ]100
   IF EXPO GE 10 ∨ EXPODIGIT DO [10
       WRITE(EXPO/10 + ↓0↓)
       EXPO := EXPO REM 10 ]10
   WRITE(EXPO + ↓0↓)  ]EX
   ELSE TEST J = EXPO+1 DO
     FOR I = 1 TO J DO WRITE(D.I)
   ELSE [     // HANDLE FIXED POINT
     FOR I = 1 TO EXPO + 1 DO WRITE(D.I)
     WRITE(↓.↓)
     FOR I = EXPO+2 TO J DO WRITE(D.I)  ]F
LET COMPACT(CHARBUF) BE [CMP
   LET STRING = FALSE
   LET NC = 1
   IF CHARBUF.0 > 84 DO CERROR(≡ LINE TOO LONG*N≡)
   FOR I = 1 TO CHARBUF.0 DO
```

```
        IF CHARBUF.I ≠ ↓ ↓ ∨ STRING DO [
            CHARBUF.NC := CHARBUF.I
            NC := NC + 1
            IF CHARBUF.I = ↓≡↓ DO STRING := ¬STRING ]CMP
LET READLINE(V) BE [RL
    FOR I = V+1 TO V+100 DO [
        READCH(INPUT,I)
        IF RV I = ↓*N↓ DO [
            V.0 := I-V
            BREAK ]RL
LET FWDLINE(LIN) = RV LIN ↑ -18 ∧ AMASK
LET COPYSTRING(A,ERROR)= VALOF [CS
    LET CT = 0
    LET QREQ = RV S = QUOTE
    A.1 := 0  //INSURES ONE WORD STRINGS DO NOT HAVE GARBAGE IN THEM
    UNLESS QREQ DO [
        UNLESS ALPHA(RV S) DO ERROR(
            ≡ UNQUOTED STRING DOES NOT START WITH A LETTER*N≡)
        S := S - 1 ]
    [1 LET I = 49
        A.CT := 030140603014060301408
        [2 S := S + 1
            TEST QREQ DO [Q
                IF RV S = ↓*N↓ DO ERROR(≡ QUOTE MISSING*N≡)
                IF RV S = QUOTE DO [ S := S + 1
                    RESULTIS CT + 1 ]Q
            ELSE [NQ
                UNLESS ALPHA(RV S) ∨ DIGIT(RV S) DO
                RESULTIS CT+1  ]NQ
            A.CT := A.CT + (RV S - 140B ↑ I)
            I := I - 7  ]2 REPEATUNTIL I
        CT := CT + 1  ]1 REPEAT  ]CS
LET DATAINIT() BE [DI
    TMPDATA := DATA
    TMPDATACNT := DATACOUNT   ]DI
LET DATAOK() BE [OK
    DATA := TMPDATA
    DATACOUNT := TMPDATACNT ]OK
LET DATASTORE(ADD,TYPE) BE [DS
    IF TMPDATA = 0 DO [
        TMPDATA := LOWLINE
        TMPDATACNT := RV TMPDATA ↑ -54 = DATCMD -> 0, 377770B  ]
// FIND NEXT DATA VALUE
    TMPDATACNT := TMPDATACNT + 1
    IF TMPDATACNT GE (RV RV TMPDATA ↑ -18 ∧ AMASK) DO [F  //MOVE TO NEXT
        LET TP = 0
        [LP TMPDATA := FWDLINE(TMPDATA)
            TP := RV TMPDATA ↑ -54
            IF TP = 0  DO RERROR(≡ OUT OF DATA*N≡)
        ]LP  REPEATUNTIL TP = DATCMD
        TMPDATACNT := 1  ]F
// STORE THE DATA VALUE
    [   LET VAL = (RV TMPDATA). TMPDATACNT
        TEST TYPE = STRNGEXP DO [SE
            UNLESS VAL =. INDEF DO RERROR(≡ NUMBER READ FOR A STRING*N≡)
            COPY(RV TMPDATA + TMPDATACNT + 1 ,ADD,2)
```

```
            TMPDATACNT := TMPDATACNT + 2   ]SE
        ELSE   [AE
            IF VAL =. INDEF DO RERROR(Ξ STRING READ FOR A NUMBER*NΞ)
            RV ADD := VAL   ]DS
LET RERROR(ST ) BE [RE
    LET D = TRUE
    LET SEP = Ξ*N*NΞ
    WRITES(ΞERROR IN LINE Ξ)
    WRITEN(RV LASTPROG ↑ -36 ∧ 777777B)   ,
    WRITES(ST)
    SIG := 7
    FOR I = 0 TO 285 DO
    UNLESS VARIABLES.I =. INDEF DO [P
        IF D DO [ WRITES(SEP); D := FALSE ]
        TEST I LE 259 DO [
            WRITECH(OUTPUT,↓A↓+I REM 26)
            WRITECH(OUTPUT,↓0↓+ I/26,OUTPUT)
        ] ELSE WRITECH(OUTPUT,↓A↓ + I - 260)
        WRITES(Ξ  =  Ξ)
        WRITEF(VARIABLES.I)
        WRITE(↓*N↓) ]P
    D := TRUE
    FOR I = 0 TO 25 DO // STRING DUMP
    UNLESS STRINGS.(I*2) =.INDEF DO [SD
        IF D DO [ WRITES(SEP); D := FALSE ]
        WRITECH(OUTPUT,↓A↓ + I)
        WRITES(Ξ$  =  *ΞΞ)
        WRITES(STRINGS+(I*2))
        WRITES(Ξ*Ξ*NΞ) ]SD
    D := TRUE
    FOR I = 0 TO 25 DO
    UNLESS ARRAYS.I =. INDEF DO [AD
        LET ENTRY = ARRAYS.I
        LET T = (1↑18) +. (ARRAYS+I ↑ 36)
        IF D DO [ WRITES(SEP); D := FALSE ]
        WRITES(ΞARRAY Ξ)
        WRITECH(OUTPUT,↓A↓ + I)
        WRITES(Ξ WITH DIMENSIONS Ξ)
        TEST ENTRY ↑ -36 = 0 DO WRITEN((ENTRY∧AMASK) - 1)
        ELSE   [T
            LET J = 0
            LET DIM = ONE
            LET CT = ENTRY ↑ -36
            [D  WRITEN(INT(FDIV(ENTRY.J,DIM)))
                IF J = CT BREAK
                DIM := ENTRY.J
                J := J + 1
                WRITECH(OUTPUT,↓**↓)   ]D REPEAT   ]T
            WRITECH(OUTPUT,↓*N↓)
        TEST ENTRY ↑ -36 < 3 DO EXECUTE((MPRICMD↑54) +. LV T )
        ELSE FOR J = 0 TO INT(ENTRY.(ENTRY↑-36)) - 1  DO
        [1  LET VAL = (ENTRY↑-18).J
            TEST VAL =.INDEF DO [
                WRITES(ΞUNINTΞ)
                CHARCNT := CHARCNT + 5 ]
            ELSE WRITEF(VAL)
```

```
            WRITE(↓*T↓)  ]AD
    D := TRUE
    FOR I = 0 TO 25 DO //STRINT ARRAY DUMP
    UNLESS STARRAYS.I =. INDEF DO [SAD
        LET PT = STARRAYS.I ↑-18
        IF D DO [ WRITES(SEP); D := FALSE ]
        WRITES(≡STRING ARRAY ≡)
        WRITECH(OUTPUT,↓A↓ + I)
        WRITES(≡$ OF DIMENSION ≡)
        WRITEN((STARRAYS.I ∧ AMASK)/2 - 1)
        WRITECH(OUTPUT,↓*N↓)
        FOR J = 0 TO (STARRAYS.I ∧ AMASK)/2 - 1 DO [
            TEST PT.(2*J) =. INDEF DO WRITES(≡UNINT≡)
            ELSE WRITES(PT+2*J)
            WRITECH(OUTPUT,↓*N↓)  ]SAD
    CLOSEALL()
    FINISH  ]RE
          IDENT     START
          EXT        GLOBALO
          ENTRY     START,MASKTAB
          LCC        OVERLAY(BASICX,0,0)
 ONES     SET       60
 MASKTAB  BSS       0
          DUP       61              . GENERATE MASK TABLE
 ZEROS    SET       60-ONES
          VFD       ZEROS/0,ONES/-0
 ONES     SET       ONES-1
          ENDD
FLDFET    VFD        18/3HMEM,2/3,40/FLD
          ENTRY      FLD
FLD       DATA       0
GLOBAL    VFD       60/GLOBALO
GLBSIZE   EQU        100
STKSIZE   EQU        300
*  CODE FOR STOPING
          ENTRY      XJRET
ENDMSG    VFD        18/3HEND,2/3,40/0
XJRET     SA1        ENDMSG
          BX7        X1
          SA7        1
          DATA       0
FREE      EQU        50          THE GLOBAL FOR SPACE ALLOCATION
START     SB5        GLOBALO
          SB1       1
          SB2       -1
          MX6       0
* SET UP STACK AND FREE THE GLOBAL FOR SPACE ALLOCATION
          SA1        FLDFET
          BX7        X1
          SA7        1
LP        SA1        FLD
          ZR         X1,LP
          SA1        FLD
          AX1        30
          SX7        X1-STKSIZE
          SA7        B5+FREE
```

```
          SA7       B5                                                               START     41
          SB6      X7                                                                START     42
* CALL BASIC- NO FALL THROUGH                                                        START     43
          SX7       XJRET                                                            START     44
          JP        =XBASIC                                                          START     45
          END      START                                                             START     46
          IDENT     VECTORS                                                          START     48
          ENTRY     ERREGS,CHARBUF,VARIABL,ARRAYS,STRINGS,STARRAY                    START     49
          ENTRY     FUNCTIO,IOBUF,INBUF,LINES,GLOBALO                                START     50
GLBSIZE   EQU       100                                                              START     51
GLOBALO   BSS       GLBSIZE                                                          START     52
ERREGS    BSS       11                                                               START     53
CHARBUF   BSS       82                                                               START     54
VARIABL   BSS       26*11                                                            START     55
ARRAYS    BSS       26                                                               START     56
STRINGS   BSS       52                                                               START     57
STARRAY   BSS       26                                                               START     58
FUNCTIO  BSS       26                                                                START     59
IOBUF     BSS       2*137                                                            START     60
INBUF     BSS       40                                                               START     61
LINES     BSS       1          THIS MUST BE THE LAST DECLAIRED VECTOR                START     62
          END                                                                        START     63
         IDENT   EXPRESS                                                             EXPRESS    2
          ENTRY     EXPRESS                                                          EXPRESS    3
          EXT     GLOBALO                                                            EXPRESS    4
* MACRO TO EXECUTE NEXT INSTRUCTION                                                  EXPRESS    5
NXTINS   MACRO                                                                       EXPRESS    6
         SA5      A5+B1                                                              EXPRESS    7
         SB4      X5                                                                 EXPRESS    8
         JP       B4                                                                 EXPRESS    9
         ENDM                                                                        EXPRESS   10
* MACRO TO DELETE AND STORE ON STACK                                                 EXPRESS   11
STNXIN   MACRO                                                                       EXPRESS   12
         SB3      B3+B2                                                              EXPRESS   13
         SA7      B3                                                                 EXPRESS   14
         SA5      A5+B1                                                              EXPRESS   15
         SB4      X5                                                                 EXPRESS   16
         JP       B4                                                                 EXPRESS   17
         ENDM                                                                        EXPRESS   18
* MACRO TO ADD AN ENTRY IN THE STACK                                                 EXPRESS   19
ADNXIN   MACRO                                                                       EXPRESS   20
         SB3      B3+B1                                                              EXPRESS   21
         SA7      B3                                                                 EXPRESS   22
         SA5      A5+B1                                                              EXPRESS   23
         SB4      X5                                                                 EXPRESS   24
         JP       B4                                                                 EXPRESS   25
         ENDM                                                                        EXPRESS   26
* MACRO TO CALL GETSPACE FOR ALLOCATING UNDIMENSIONED ARRAYS                         EXPRESS   27
GETSPACE  MACRO     N                                                                EXPRESS   28
          LOCAL     RET,L1                                                           EXPRESS   29
* SAVE ALL THE RELEVANT REGISTERS                                                    EXPRESS   30
          SX6       A0                                                               EXPRESS   31
          SA6       =XERREGS                                                         EXPRESS   32
          SX6       A1                                                               EXPRESS   33
          SA6       A6+B1                                                            EXPRESS   34
          SX6       A5                                                               EXPRESS   35
```

```
         SA6       A6+B1
         SX6       B2
         SA6       A6+B1
         SX6       B3
         SA6       A6+1
         SX6       B4
         SA6       A6+B1
         SX6       B6
         SA6       A6+B1
         BX6       X0
         SA6       B1+A6
         BX6       X5
         SA6       A6+B1
         BX6       X7
         SA6       A6+B1
         MX6       0
         SB6       B3+2          CLEAR CURRENT DATA
         SX7       N             GET PARAMETER
         SA7       B6+2          STORE PARAMER
         SB2       -B1           FIX B2
         SX7       RET           GET RETURN
* GET JUMP ADDRESS
         SA1       B5+12
         SB3       X1
         JP        B3
* FILL THE SPACE WITH INDEF
RET      SA2       =60000000000000000000B
         BX7       X2
         SB7       N-1
L1       SA7       X1+B7
         SB7       B7-B1
         PL        B7,L1
         ENDM
* MACRO TO RESTORE THE REGISTERS AFTER GETSPACE HAS BEEN CALLED
RESTORE  MACRO
         SA2       =XERREGS
         SA0       X2
         SA2       A2+B1
         BX7       X1
         SA1       X2
         BX1       X7
         SA2       A2+B1
         SA5       X2
         SA2       A2+B1
         SB2       X2
         SA2       A2+B1
         SB3       X2
         SA2       A2+B1
         SB4       X2
         SA2       A2+B1
         SB6       X2
         SA2       A2+B1
         BX0       X2
         SA2       A2+B1
         BX5       X2
         SA2       A2+B1
```

```
         BX7       X2
         ENDM
* ENTRY TO EVALUATING AN EXPRESSION
EXPRESS  SA7       B6                 STORE RETURN
         SA1       B6+2               GET PARAMETER
         SX7       B6          SAVE THE STACK POINTER FOR AN ERROR
         SA7       B5+82
         SA5       X1
         SB3       B6+1
         SB4       X5
         JP        B4
*  FINAL CALL IN ALL EXPRESSIONS
* STORE THE EXPRESSION TYPE
ENDEXP   AX5       18
         BX6       X5
         SA6       B5+92
         MX6       0
* IF TYPE IS STRING STORE ITS VALUE
         SX5       X5+4
         NZ        X5,END1
* STORE STRING IN STNUMBER
         SA7       B5+79
         SA1       B3+B2
         BX6       X1
         SA6       B5+78
         MX6       0
* SET RETURN VALUE AND EXPSTOP
END1     BX1       X7
         SX7       A5+31
         SA7       B5+64
* GET RETURN ADDRESS
         SA2       B6
         SB4       X2
         JP        B4
* ARITHEMETIC OPERATIONS
ADD      SA1       B3+B2
         RX7       X1+X7
         OR        X7,INFADD
         NX7       X7
         STNXIN
SUBTRAC  SA1       B3+B2
         RX7       X1-X7
         OR        X7,INFSUB
         NX7       X7
         STNXIN
MULT     SA1       B3+B2
         RX7       X1*X7
         OR        X7,INFMULT
         STNXIN
DIV      SB4       48
         AX4       B4,X7
         ZR        X4,ZERODIV
         SA1       B3+B2
         RX7       X1/X7
         OR        X7,INFDIV
         STNXIN
```

```
NEG      BX7     -X7                                                        EXPRESS   148
         SA7     B3                                                         EXPRESS   149
         NXTINS                                                             EXPRESS   150
* LOGICAL OPERATORS                                                         EXPRESS   151
AND      SA1     B3+B2                                                      EXPRESS   152
         BX7     X1*X7                                                      EXPRESS   153
         STNXIN                                                             EXPRESS   154
OR       SA1     B3+B2                                                      EXPRESS   155
         BX7     X1+X7                                                      EXPRESS   156
         STNXIN                                                             EXPRESS   157
SA        MACRO                                                             EXPRESS   158
          SA1    B3+B2                                                      EXPRESS   159
          ENDM                                                              EXPRESS   160
* RELATIONAL OPERATORS                                                      EXPRESS   161
* A MACRO TO PREPAIR THE COMPAIRISON OF TWO STRINGS                         EXPRESS   162
SELECT    MACRO                                                             EXPRESS   163
          LOCAL      L1,L2,L3,L4,L5                                         EXPRESS   164
* CHOOSE WHICH WORD IS TO BE COMPAIRED                                      EXPRESS   165
* IF THE FIRST WORD IS THE SAME THEN TAKE THE SECOND OF EACH STRING         EXPRESS   166
          SA1        B3+B2                                                  EXPRESS   167
          SA2        B3-3                                                   EXPRESS   168
          IX3        X1-X2                                                  EXPRESS   169
          NZ         X3,L1                                                  EXPRESS   170
          SA1        A1+B1                                                  EXPRESS   171
          SA2        A2+B1                                                  EXPRESS   172
* THIS LOOP CHANGES ANY 140B≠S INTO 0                                       EXPRESS   173
L1        MX0        60-7      TO MASK OUT ONE CHAR                         EXPRESS   174
          SB7        7                                                      EXPRESS   175
          LX1        4                                                      EXPRESS   176
          LX2        4                                                      EXPRESS   177
* SHIFT TO NEXT CHARACTER AND TEST IT                                       EXPRESS   178
L4        LX1        7                                                      EXPRESS   179
          LX2        7                                                      EXPRESS   180
          BX3        -X0*X1                                                 EXPRESS   181
          SX3        X3-140B                                                EXPRESS   182
          NZ         X3,L2                                                  EXPRESS   183
          BX1        X0*X1                                                  EXPRESS   184
L2        BX3        -X0*X2                                                 EXPRESS   185
          SX3        X3-140B                                                EXPRESS   186
          NZ         X3,L3                                                  EXPRESS   187
          BX2        X0*X2                                                  EXPRESS   188
* CHECK FOR LOOP COMPLETE                                                   EXPRESS   189
L3        SB7        B7-B1                                                  EXPRESS   190
          GE         B7,B0,L4                                               EXPRESS   191
          ENDM                                                              EXPRESS   192
* STRING EQUAL                                                              EXPRESS   193
STEQUAL   SA1        B3-2                                                   EXPRESS   194
          IX3        X1-X7                                                  EXPRESS   195
          MX7        0                                                      EXPRESS   196
          NZ         X3,STEQU1                                              EXPRESS   197
          SA1        A1+B2                                                  EXPRESS   198
          SA2        B3+B2                                                  EXPRESS   199
          IX3        X1-X2                                                  EXPRESS   200
          NZ         X3,STEQU1                                              EXPRESS   201
          MX7        60                                                     EXPRESS   202
STEQU1    SB3        B3-3                                                   EXPRESS   203
```

```
         SA7       B3
         NXTINS
EQUAL    SA1       B3+B2
         IX2       X1-X7
         MX7       60
         ZR        X2,EQ1
         SX7       B0
EQ1      STNXIN
* STRING NOT EQUAL
STNEQUAL  SA1        B3-2
          IX3        X1-X7
          MX7        60
          NZ         X3,STNEQU1
          SA1        A1+B2
          SA2        B3+B2
          IX3        X1-X2
          NZ         X3,STNEQU1
          MX7        0
STNEQU1   SB3        B3-3
          SA7        B3
          NXTINS
NEQUAL   SA
          IX2     X1-X7
          SX7     B0
          ZR      X2,NE1
          MX7     60
NE1       STNXIN
* STRING LESS THAN
STLESSTH  SELECT
          IX3        X2-X1
          MX7        60
          NG         X3,STLT1
          SX7        B0
STLT1     SA7        B3-3
          SB3        A7
          NXTINS
LESSTH    SA
          IX2     X1-X7
          MX7     60
          NG      X2,LT1
          SX7     B0
LT1       STNXIN
* STRING MORE THAN
STMORETH  SELECT
          IX3        X1-X2
          MX7        60
          NG         X3,STMT1
          SX7        B0
STMT1     SA7        B3-3
          SB3        A7
          NXTINS
MORETH    SA
          IX2     X7-X1
          MX7     60
          NG      X2,MT1
          SX7     B0
```

```
MT1        STNXIN
* STRING LESS THAN OR EQUAL
STLESSEQ   SELECT
           IX3        X1-X2
           MX7        60
           PL         X3,STLE1
           SX7        B0
STLE1      SA7        B3-3
           SB3        A7
           NXTINS
LESSEQ     SA
           IX1    X7-X1
           MX7    60
           PL     X1,LE1
           SX7    B0
LE1        STNXIN
* STRING MORE THAN OR EQUAL TO
STMOREEQ   SELECT
           IX3        X2-X1
           MX7        60
           PL         X1,STME1
           SX7        B0
STME1      SA7        B3-3
           SB3        A7
           NXTINS
MOREEQ     SA
           IX1    X1-X7
           MX7    60
           PL     X1,ME1
           SX7    B0
ME1        STNXIN
* STORE A STRING
STOREST    SA1        B3-2
           SA2        B3-B1
           BX6        X2
           SA7        X1+B1
           SA6        X1
* PUT IT BACK ON THE STACK
           SA7        B3+B2
           SA6        A7+B2
           MX6        0
           SB3        B3-B1
           NXTINS
* FETCH A STRING
FETCHST    AX5        18
           SA1        X5
           ID         X1,UNINTST
           SA2        X5+B1
           BX7        X1
           SA7        B3+B1
           BX7        X2
           SA7        B3+2
           SB3        A7
           NXTINS
* STORE THE TOP OF THE STACK
STORE      SA
```

```
         SA7    X1
         STNXIN
* GET A VARIABLE VALUE
FETCH    AX5    18
         SA1    X5
         ID     X1,UNINT
         BX7    X1
         ADNXIN
* ADD A CONSTANT TO THE STACK
NMCNST   SA5    A5+B1
         BX7    X5
         ADNXIN
* STRING CONSTANT
STCNST   SA4        A5+B1
         SA5        A5+2
         BX7        X4
         SA7        B3+B1
         BX7        X5
         SA7        B3+2
         SB3        A7
         NXTINS
* ADD ADDRESS CONSTANT
ADCNST   SB4    18
         AX7    B4,X5
         ADNXIN
* ADD CONSTANT 1.0
ONECNST  SA1    =1.0
         BX7    X1
         ADNXIN
* PUT CONSTANT PI ON STACK
PICNST   SA1        =17216220773250420550B
         BX7   X1
         ADNXIN
* CALL A FUNCTION
* CHECK FOR THE STACK OVERFLOWING
FNCALL   SA1        =XFLD
         AX1        30
         SB7        X1-25
         SB7        B7-B3
         LE         B7,B0,RECURSE
* STORE OLD B6AND A5+1
         SX7    A5+B1
         SA7    B3+B1
         SX7    B6
         SA7    B3+2
*  RESET B6 AND B3
        SB6     B3
        SB3     B3+2
* GET FUNCTION ENTRY
         SB4    18
         AX2    B4,X5
         SA1    X2
         ID     X1,UNDFNC
* CONTINUE EXECUTATION
         SA5    X1
         SB4    X5
```

```
         JP      B4                                                           EXPRESS   372
* RETURN FROM A FUNCTION CALL                                                 EXPRESS   373
*  RESTORE OLD A5 AND B6 AND B3                                               EXPRESS   374
FRET    SA1     B6+B1                                                         EXPRESS   375
         SA2     B6+2                                                         EXPRESS   376
        SB3     B6                                                            EXPRESS   377
         SB6     X2                                                           EXPRESS   378
         SA5     X1                                                           EXPRESS   379
* STORE VALUE AND RETURN                                                      EXPRESS   380
         SA7     B3                                                           EXPRESS   381
         SB4     X5                                                           EXPRESS   382
         JP      B4                                                           EXPRESS   383
* GET PARAMETER VALUE                                                         EXPRESS   384
PARVAL  SA1     B6                                                            EXPRESS   385
         BX7     X1                                                           EXPRESS   386
         ADNXIN                                                               EXPRESS   387
* GET ARRAY CONTENTS OR ADDRESS                                               EXPRESS   388
ARYCCNT  SB2     B0                                                           EXPRESS   389
ARYADD   SB7     18                                                           EXPRESS   390
         AX5     B7,X5               ARRAY ADDRESS                            EXPRESS   391
*  SAVE DIMENSION COUT -1                                                     EXPRESS   392
         AX0     B7,X5                                                        EXPRESS   393
        SB4     X0                                                            EXPRESS   394
        SA0     B3-B4                                                         EXPRESS   395
* INITIALIZE LOOP TO CALCULATE ARRAY ELEMENT                                  EXPRESS   396
         SA1     X5                                                           EXPRESS   397
         DF         X1,ARY1                                                   EXPRESS   398
         EQ         B2,UNINTA                                                 EXPRESS   399
* ALLOCATE A 10 * 10 ARRAY IF 2 DIMENSIONS                                    EXPRESS   400
         NE         B4,B1,UNDFAR                                              EXPRESS   401
* GET THE SPACE                                                               EXPRESS   402
         GETSPACE  102                                                        EXPRESS   403
* GET THE REFERENCE FUNCTION                                                  EXPRESS   404
         SA2        =10.0                                                     EXPRESS   405
         SA3        =100.0                                                    EXPRESS   406
         BX7        X2                                                        EXPRESS   407
         SA7        X1                                                        EXPRESS   408
         BX7        X3                                                        EXPRESS   409
         SA7        A7+B1                                                     EXPRESS   410
* STORE THE ENTRY FOR THE TABLE OF ARRAYS INTO X1                             EXPRESS   411
         SX7        B1                                                        EXPRESS   412
         LX7        18                                                        EXPRESS   413
         SX2        X1+2                                                      EXPRESS   414
         IX7        X7+X2                                                     EXPRESS   415
         LX7        18                                                        EXPRESS   416
         IX1        X7+X1                                                     EXPRESS   417
         RESTORE                                                              EXPRESS   418
         BX7        X1                                                        EXPRESS   419
         SA7        A1                                                        EXPRESS   420
*  CHECK TO BE SURE PROPER NUMBER OF DIMENSIONS                               EXPRESS   421
ARY1     BX4     X1                                                           EXPRESS   422
         AX4     36                                                           EXPRESS   423
         SB7     X4                                                           EXPRESS   424
         NE      B4,B7,DIMCNT                                                 EXPRESS   425
*  INITIALIZE X1,X0                                                           EXPRESS   426
        SA1     X1-1                                                          EXPRESS   427
```

```
         SX1    17204B
         LX1    45                X1 INITIALIZED TO 1.0
         BX0    X1                1.0 FOR COMPARISON
         SX4    B0                CURRENT TOTAL
* GET    NEXT DIMENSION AND CONVERT TO INTEGER
AGAIN    SA2    B3-B4
         UX2     B7,X2
         GE      B7,HIDIM
         LX2     B7,X2
         PX2     B0,X2
         NX2     X2
* TEST FOR BEING LESS THAN 1
         RX2    X2-X0
         NX2     X2
         NG     X2,LOWDIM
* GET INCREMENT TO TOTAL INCREMENT
         RX3    X1*X2
         OR     X3,HIDIM
         RX4    X3+X4
         NX4     X4
* GET NEXT FUNCTION AND COMPAIR IT WITH THE CURRENT
         SA1    A1+B1
         IX3     X4-X1
         PL     X3,HIDIM
* CHECK FOR LOOP BEING FINISHED
         SB4    B4-B1
         GE     B4,AGAIN
* GET INTEGER VALUE
         UX7    B4,X4
         LX7    B4,X7
* GET POINTER TO ARRAY
         SA1    X5
         AX1    18
* GET ADDRESS
         IX7    X7+X1
         SX4    377777B
         BX7     X7*X4
         NG     B2,ADR            IF B2 = -1 ONLY WANT ADDRESS
* GET CONTENTS OF ARRAY ELEMENT
         SA1    X7
         BX7    X1
         SB2    -B1
         ID     X7,UNINTA
* STORE THE RESULT
ADR      SB3    A0
         SA7    B3
         NXTINS
*  GET LIST CONTENTS OR ADDRESS
LSTCONT  SB2    B0
LSTADD   AX5    18
         SA1    X5
*  CHECK FOR BEING DEFINED AND A LIST
         SB7    36
         AX0    B7,X1
         SB7    X0
         NE     B7,DIMCNT
```

```
* CHECK FOR THE LIST BEING DIMENSIONED
          DF        X1,LST2
          EQ        B2,UNINTL
* ALLOCATE AN 11ELEMENT ARRAY
          GETSPACE  11
* COMPUTE THE FUNCTION
          LX1       18
          SX7       11
          IX1       X7+X1
          RESTORE
* STORE THE ARRAY VALUE
          BX6       X1
          SA6       A1
          SX6       B0
*  GET INDEX AND CONVERT TO INTEGER
LST2     NG      X7,LOWIDXL
         UX7     B7,X7
         GE      B7,HIIDXL
         LX7     B7,X7
*  CHECK INDEX FOR BEING TOO GREAT
         SB7     X1
         BX7     -X7
         SB7     B7+X7
          LE        B7,B0,HIIDXL
*  GET ADDRESS IN X7
         AX1     18
         IX7     X1-X7
*   GET CONTENTS IF REQUIRED
         NE      B2,LST1
         SB2     -B1
         SA1     X7
         BX7     X1
         ID      X7,UNINTL
*  STORE RESULTS
LST1     SA7     B3
         NXTINS
* STRING LIST FETCH AND ADDRESS COMPUTATION
STLSTC   SB2     B0
STLSTA   AX5     18
         SA1     X5
*  CHECK FOR BEING DEFINED AND A LIST
         SB7     36
         AX0     B7,X1
         SB7     X0
         NE      B7,DIMCNT
* CHECK FOR THE LIST BEING DIMENSIONED
          DF        X1,STLST2
          EQ        B2,UNINTLST
* ALLOCATE AN 11ELEMENT ARRAY
          GETSPACE  22
* COMPUTE THE FUNCTION
          LX1       18
          SX7       22
          IX1       X7+X1
          RESTORE
* STORE THE ARRAY VALUE
```

```
          BX6        X1
          SA6        A1
          SX6        B0
*  GET INDEX AND CONVERT TO INTEGER
STLST2   NG      X7,LOWIDXL
         UX7     B7,X7
         GE      B7,HIIDXL
         LX7     B7,X7
         LX7     1
*  CHECK INDEX FOR BEING TOO GREAT
         SB7     X1
         BX7     -X7
         SB7     B7+X7
          LE         B7,B0,HIIDXL
*  GET ADDRESS IN X7
         AX1     18
         IX7     X1-X7
*   GET CONTENTS IF REQUIRED
         NE     B2,STLST1
         SB2     -B1
         SA1     X7
         BX7     X1
         ID      X7,UNINTLST
          SA7        B3
          SA1        A1+B1
          BX7        X1
          SB3        B3+B1
*  STORE RESULTS
STLST1   SA7     B3
         NXTINS
* FUNCTION TO TAKE INTEGER PART OF A NUMBER
INTEGER   UX1    B4,X7
          GE     B4,B0,IN1
          LX1    B4,X1
          PX1    X1
          NX7    X1
          SA7    B3
IN1       NXTINS
* GET SIGN OF A NUMBER
SIGN      ZR     X7,SG1
          SA1    =1.0
          PL     X7,SG2
          BX1    -X1
SG2       BX7    X1
SG1       SA7    B3
          NXTINS
* RANDOM NUMBER GENERATOR
SEED      EQU        GLOBAL0+93
RTIMFET   VFD        18/3HTIM,2/3,16/2,24/RTIM
RTIM      DATA       0
* IF ARG = 0 LEAVE THE SEED ALONE
RND       MX0        60-35
          ZR         X7,RND1
          SA1        B5+94
          PL         X1,RND1
* IF THE ARG IS POSITIVE USE IT TO AFFECT THE SEED
```

```
         PL        X7,RND2
* AFFECT THE SEED WITH THE CLOCK
         SA1       RTIMFET
         BX7       X1
         SA7       1
LPRND    SA1       RTIM
         ZR        X1,LPRND
         SA1       RTIM
         PX7       X1
         NX7       X7
* CHANGE THE SEED
RND2     SA1       SEED
         PX1       X1
         NX1       X1
         FX7       X1*X7
         BX7       -X0*X7
         SA7       SEED
         SA6       B5+94
* GET THE NEXT RANDOM NUMBER
RND1     SA1       SEED
         PX1       X1
         SA2       =20000000343277244615B          5↑15
         DX7       X1*X2
         BX7       -X0*X7
         SX1       13
         IX7       X7+X1
         BX7       -X0*X7
         SA7       SEED
* CONVERT TO A USEABLE FLOATING POINT NUMBER
         SB7       -35
         PX7       B7,X7
         NX7       X7
         SA7       B3
         NXTINS
* RETURN THE EXECUTION TIME
         ENTRY     TIMFET
TIMFET   VFD       18/3HTIM,2/3,40/TIMSTO
TIMSTO   DATA      0
* STORE IN RA+1
TIM      SA1       TIMFET
         BX7       X1
         SA6       TIMSTO
         SA7       1
*WAIT FOR SYSTEM
LPTIM    SA1       TIMSTO
         ZR        X1,LPTIM
*UNPACK THE TIME
         SA1       TIMSTO
         MX2       48
         BX3       -X2*X1
         PX3       X3
         NX3       X3
         SA2       =1000.0
         FX3       X3/X2
* GET THE SECONDS
         AX1       12
```

```
         MX2      36
         BX4      -X2*X1
         PX4      X4
         NX4      X4
         FX4      X4+X3
* SUBTRACT OUT STARTING TIME
         SA1      B5+72      GLOBAL CELL TIME
         PX1      X1
         NX1      X1
         SA2      =1000.0
         FX1      X1/X2
         FX7      X4-X1
         NX7      X7
         SA7      B3
         NXTINS
* GET ABSOLUTE VALUE
ABS      PL    X7,ABZ
         BX7   -X7
         SA7   B3
ABZ      NXTINS
*        EXPONENTIATION
* FIRST TEST FOR AN INTEGER LESS THAN 512
EXPONET  UX4   B7,X7
         SB4   B7+38
         PL    B4,NOTINT
* CHECK FOR BEING AN INTEGER
         LX3   B7,X4
         AX2   B7,X3
         IX2   X4-X2
         NZ    X2,NOTINT
* IT IS AN INTEGER LESS THAN 512
* CHECK FOR SIGN
         PL    X7,EX1
         SB1   B0
         BX3   -X3
* INITIALIZE LOOP TO DO MULTIPLYING
EX1      SA
         ZR    X1,ZRBAS
         SA2   =1.0
         BX7   X2
* LOOP REQUIRED NUMBER OF TIMES
EX2      ZR    X3,EX3
         RX7   X7*X1
         OR    X7,INFEXP
         SX3   X3+B2
         EQ    EX2
* LOOP FINISHED CHECK FOR INVERTING
EX3      NE    B1,B0,EX4
         RX7   X2/X7
EX4      SB1    -B2
         STNXIN
* IF BASE IS 0 CHECK FOR 0 ** /
ZRBAS    ZR    X7,ZEROZERO
         SX7   0
         EQ    EX4
* HANDEL A NON INTEGER EXPONENT
```

```
NOTINT    SA1       B3+B2
          ZR        X1,ZRBAS
          NG        X1,NEGBAS
* GET LOG OF BASE
          BX7       X1
          SB4       EX5
          JP        =XALOG
* MULTIPLY EXPONENT AND TAKE EXP
EX5       SA2       B3          GET EXPONENT
          RX7       X7*X2
          SB4       EX4
          JP        =XEXP
*EXPONETTIAL FUNCTION
EXPO      SB4       EXPO1
          JP        =XEXP
EXPO1     SA7       B3
          NXTINS
*NATURAL LOG FUNCTION
LOG       SB4       LOG1
          JP        =XALOG
LOG1      SA7       B3
          NXTINS
* TAKE LOG TO BASE 10
LGT       SB4       LGT1
          JP        =XALOG10
LGT1      SA7       B3
          NXTINS
*ERROR ROUTINE
* RESTORE B6 TO ORIGINAL VALUE
BACK      SA1       =XGLOBAL0+B2
          SB6       X1
* CALL EERROR AFTER RESTORING REST OF REGS
          SA7     B6+2
          SB1     1
          SB2     -B1
          MX6     0
          SB5     GLOBAL0
          JP      =XEERROR
ERR      MACRO    N
         SX7      N
          EQ      BACK
         ENDM
          ENTRY     INFEXP,SCRNG,TNRNG,ZEROLOG,NEGLOG,NEGSQRT,ASACARG
          ENTRY     INFMULT,INFDIV,INFADD,INFSUB
INFADD   ERR      1
INFSUB   ERR      2
INFMULT  ERR      3
INFDIV   ERR      4
INFEXP   ERR      5
UNDFNC   ERR      6
PARCNT   ERR      7
UNDFAR   ERR      8
LOWDIM   ERR      9
HIDIM    ERR      10
UNINTA   ERR      11
UNINT    ERR      12
```

```
ZEROZERO ERR        13
NEGBAS   ERR        14
ASACARG  ERR        15
ZERODIV  ERR        16
DIMCNT   ERR        17
UNINTL   ERR        18
LOWIDXL  ERR        19
HIIDXL   ERR        20
         ERR        21
RECURSE  ERR        22
SCRNG    ERR        23
TNRNG    ERR        24
ZEROLOG  ERR        25
NEGLOG   ERR        26
NEGSQRT  ERR        27
UNINTST  ERR        28
UNINTLST ERR        29
         ENTRY      EXPCALL
EXPCALLS BSS        0
EXPCALL  DATA       0
         VFD        60/ABS          1
         VFD        60/=XATAN       2
         VFD        60/=XCOS        3
         VFD        60/EXPO         4
         VFD        60/INTEGER      5
         VFD        60/LOG          6
         VFD        60/SIGN         7
         VFD        60/=XSIN        8
         VFD        60/=XSQRT       9
         VFD        60/=XTAN        10
         VFD        60/RND          11
         VFD        60/=XACOS       12
         VFD        60/=XASIN       13
         VFD        60/LGT          14
         VFD        60/FTIM         15
         ORG        EXPCALL+20
         VFD        60/ENDEXP       12
         VFD        60/ADD          13
         VFD        60/SUBTRAC      14
         VFD        60/MULT         15
         VFD        60/DIV          16
         VFD        60/NEG          17
         VFD        60/AND          18
         VFD        60/OR           19
         VFD        60/EQUAL        20
         VFD        60/NEQUAL       21
         VFD        60/LESSTH       22
         VFD        60/MORETH       23
         VFD        60/LESSEQ       24
         VFD        60/MOREEQ       25
         VFD        60/STORE        26
         VFD        60/FETCH        27
         VFD        60/NMCNST       28
         VFD        60/ADCNST       29
         VFD        60/ONECNST      30
         VFD        60/FNCALL       31
```

```
        VFD      60/FRET            32                     EXPRESS   820
        VFD      60/PARVAL          33                     EXPRESS   821
        VFD      60/ARYCCNT         34                     EXPRESS   822
        VFD      60/ARYADD          35                     EXPRESS   823
        VFD      60/EXPONET         36                     EXPRESS   824
        VFD     60/LSTCCNT          37                     EXPRESS   825
        VFD     60/LSTADD           38                     EXPRESS   826
         VFD       60/STLSTC             39                EXPRESS   827
         VFD       60/STLSTA             40                EXPRESS   828
         VFD       60/STOREST            41                EXPRESS   829
         VFD       60/FETCHST            42                EXPRESS   830
         VFD       60/STEQUAL            43                EXPRESS   831
         VFD       60/STCNST            44                 EXPRESS   832
         VFD       60/STNEQUAL          45                 EXPRESS   833
         VFD       60/PICNST            46                 EXPRESS   834
         VFD       60/STMORETH           47                EXPRESS   835
         VFD       60/STMOREEQ           48                EXPRESS   836
         VFD       60/STLESSTH           49                EXPRESS   837
         VFD       60/STLESSEQ           50                EXPRESS   838
         END                                               EXPRESS   839
         IDENT   PACK                                      MISCI       2
         ENTRY   PACK                                      MISCI       3
*  GET THE PARAMETERS                                      MISCI       4
PACK     BSS     0                                         MISCI       5
         SA1     B6+2            COEF IN X1                MISCI       6
         SA2     B6+3            EXPO IN X2                MISCI       7
         SB4     X7                                        MISCI       8
         ZR       X1,DONE                                  MISCI       9
* MULTIPLY COEF  BY THE PROPER POWER OF 10                 MISCI      10
         LX2     50                                        MISCI      11
         SB3     9                                         MISCI      12
         PL      X2,PLSEXP                                 MISCI      13
*  HANDLE  NEGATIVE EXPONENT                               MISCI      14
NEGEXP   LE      B3,DONE                                   MISCI      15
         LX2     1                                         MISCI      16
         SB3     B3-B1                                     MISCI      17
         NG      X2,NEGEXP                                 MISCI      18
         SA3     B3+FACTORS                                MISCI      19
         RX1     X1/X3                                     MISCI      20
         OR       X1,OUT                                   MISCI      21
         ZR       X1,OUT                                   MISCI      22
         EQ      NEGEXP                                    MISCI      23
* HANDLE  POSITIVE EXPONENTS                               MISCI      24
PLSEXP   LE      B3,DONE                                   MISCI      25
         LX2     1                                         MISCI      26
         SB3     B3-B1                                     MISCI      27
         PL      X2,PLSEXP                                 MISCI      28
         SA3     B3+FACTORS                                MISCI      29
         RX1     X1*X3                                     MISCI      30
         OR       X1,OUT                                   MISCI      31
         ZR       X1,OUT                                   MISCI      32
         EQ      PLSEXP          * WHEN FINISHED RETURN    MISCI      33
DONE     JP      B4                                        MISCI      34
*  ERROR RETURN                                            MISCI      35
OUT      JP       B6+4                                     MISCI      36
FACTORS  DATA     1.0E1                                    MISCI      37
```

```
         DATA      1.0E2                                                      MISCI       38
         DATA      1.0E4                                                      MISCI       39
         DATA      1.0E8                                                      MISCI       40
         DATA      1.0E16                                                     MISCI       41
         DATA      1.0E32                                                     MISCI       42
         DATA      1.0E64                                                     MISCI       43
         DATA      1.0E128                                                    MISCI       44
         DATA      1.0E256                                                    MISCI       45
PASS      END                                                                 MISCI       46
          IDENT    UNPACK                                                     MISCI       47
          ENTRY      UNPACK                                                   MISCI       48
*  FIND FACTOR TO MAKE NUMBER BETWEEN 10**15 AND 10**14                       MISCI       49
*  GET THE PARAMETERS THAT WERE PASSED                                        MISCI       50
UNPACK    SA1      B6+2                  Z                                    MISCI       51
          SA5      B6+3                                                       MISCI       52
          SB4      X7                    RETURN                               MISCI       53
*  SET UP CONSTANTS AND EXPO                                                  MISCI       54
          SA2      =1.0E+14                                                   MISCI       55
          SA3      =1.0E+13                                                   MISCI       56
          MX7      0                                                          MISCI       57
          SB7      8                                                          MISCI       58
*DECIDE WHICH WAY TO GO                                                       MISCI       59
          IX0      X1-X2                                                      MISCI       60
          PL       X0,UP1                                                     MISCI       61
          IX0      X1-X3                                                      MISCI       62
          PL       X0,DONE                                                    MISCI       63
*   Z LESS THAN 10**13                                                        MISCI       64
UP2       SA4      B7+FACTORS                                                 MISCI       65
          RX4      X4*X1                                                      MISCI       66
*  CHECK RANGE                                                                MISCI       67
          IX0      X4-X2                                                      MISCI       68
          PL       X0,UP3                                                     MISCI       69
*  ADD FACTOR TO EXPO                                                         MISCI       70
          BX1      X4                                                         MISCI       71
          SX4      B1                                                         MISCI       72
          LX4      B7,X4                                                      MISCI       73
          IX7      X7+X4                                                      MISCI       74
*  CHECK FOR BEING INRANGE 10**13 -- 10**14                                   MISCI       75
          IX0      X1-X3                                                      MISCI       76
          PL       X0,DONE                                                    MISCI       77
*  GET NEXT FACTOR AND LOOP                                                   MISCI       78
UP3       SB7      B7+B2                                                      MISCI       79
          EQ       UP2                                                        MISCI       80
*  Z GREATER THAN 10**14                                                      MISCI       81
UP1       SA4      B7+FACTORS                                                 MISCI       82
          RX4      X1/X4                                                      MISCI       83
*  CHECK RANGE                                                                MISCI       84
          IX0      X4-X3                                                      MISCI       85
          NG       X0,UP4                                                     MISCI       86
*  NE 1 AND ADD IN TO EXPO                                                    MISCI       87
          BX1      X4                                                         MISCI       88
          SX4      B1                                                         MISCI       89
          LX4      B7,X4                                                      MISCI       90
          IX7      X7-X4                                                      MISCI       91
*  CHECK TO SEE IF DONE LOOPING                                               MISCI       92
          IX0      X1-X2                                                      MISCI       93
```

```
         NG       X0,DONE                                       MISCI
*  LOOP                                                         MISCI
UP4      SB7      B7+B2                                         MISCI
         EQ       UP1                                           MISCI
*  WHEN FINISHED VALUE IS IN RANGE                              MISCI
*  STORE RETURN VALUES                                          MISCI
DONE     SA7      X5                                            MISCI
         UX7      B7,X1                                         MISCI
         LX7      B7,X7                                         MISCI
         SA7        X5+B1                                       MISCI
         JP       B4                                            MISCI
FACTORS  DATA     1.0E+1                                        MISCI
         DATA     1.0E+2                                        MISCI
         DATA     1.0E+4                                        MISCI
         DATA     1.0E+8                                        MISCI
         DATA     1.0E+16                                       MISCI
         DATA     1.0E+32                                       MISCI
         DATA     1.0E+64                                       MISCI
         DATA     1.0E+128                                      MISCI
         DATA     1.0E+256                                      MISCI
PASS     END                                                    MISCI
         IDENT    COPY                                          MISCI
         ENTRY    COPY                                          MISCI
         EQ       PASS                                          MISCI
*  FETCH VARIABLES                                              MISCI
COPY     SA1      B6+2                                          MISCI
         SA2      B6+3                                          MISCI
         SA3      B6+4                                          MISCI
*  STORE VARIABLES                                              MISCI
         SB7      X7                                            MISCI
         SB4      X3                                            MISCI
         SA1      X1                                            MISCI
         BX7      X1                                            MISCI
         SA7      X2                                            MISCI
         EQ       CP1                                           MISCI
*  GET AND STORE NEXT WORD IF OK                                MISCI
LOOP     SA1      A1+B1                                         MISCI
         BX7      X1                                            MISCI
         SA7      A7+B1                                         MISCI
*  CHECK                                                        MISCI
CP1      BSS      0                                             MISCI
         SB4      B4+B2                                         MISCI
         NE       B4,LOOP                                       MISCI
         JP       B7                                            MISCI
PASS     END                                                    MISCI
         IDENT      ELE                                         MISCI
         ENTRY      ELE,ELE3                                    MISCI
*  SUBRORTINE TO RETURN AN ADDRESS FOR A MATRIX                 MISCI
*  GET DESCRIPTOR ^ FIRST INDEX                                 MISCI
ELE      SB4        X7                                          MISCI
         SA2        B6+4                                        MISCI
         SA3        B6+2                                        MISCI
         IX1        X2+X3      ADD INDEX AND BASE               MISCI
*  USE NEXT INDEX AND MULTIPLIER                                MISCI
         SA3        B6+3                                        MISCI
         PX3        X3                                          MISCI
```

```
         AX2       18
         PX2       X2
         DX3       X3*X2
         IX1       X1+X3
         SX1       X1+B0
         JP        B4
*
*  SUBROUTINE FOR 3 DIMENSIONAL ARRAY
* GET DESCRIPTOR A FIRST INDEX
ELE3     SB4       X7
         SA2       B6+5
         SA3       B6+2
         IX1       X2+X3      ADD BASE AND FIRST INDEX
*  USE NEXT NDEX AND MULTIPLIER TO INCREMENT INDEX
         SA3       B6+3
         PX3       X3
         AX2       18
         SX4       X2
         PX4       X4
         DX3       X3*X4
         SX3       X3
         IX1       X3+X1
* USE THIRD INDEX
         SA3       B6+4
         PX3       X3
         AX2       18
         PX2       X2
         DX3       X3*X2
         IX1       X1+X3
         SX1       X1
         JP        B4
PASS     END
         IDENT   FLOATPT
         ENTRY     CHKPIV,CHKFDIV
         ENTRY     FADD,FMULT,FDIV,FLOAT,INT
*  FOLATING POINT SUBROUTINES
*  MACRO FOR RETUNRNING
RET      MACRO
         SB4     X7
         JP      B4
         ENDM
* CONVERT TO INTEGER
INT      SA1     B6+2
         UX1     B4,X1
         PL      B4,INT1
         LX1     B4,X1
INT1     RET
*  CONVERT TO FLOATING POINT
FLOAT    SA1     B6+2
         PX1     X1
         NX1     X1
         RET
*  FLOATING MULTIPLY
FMULT    SA1     B6+2
         SA2     B6+2
         RX1     X1*X2
```

```
         OR       X1,=XINFMULT
         RET
*  FLOATING DIVIDE
FDIV     SA1      B6+2
         SA2      B6+3
         RX1      X1/X2
         OR       X1,=XINFDIV
         RET
*  FLOATING ADD
FADD     SA1      B6+2
         SA2      B6+3
         RX1      X1+X2
         OR       X1,=XINFADD
         NX1      X1
         RET
CHKFDIV  SA1        B6+2
         SA2        B6+3
         RX1        X1/X2
         OR         X1,CKFD1
         RET
CKFD1    JP         B6+4
* CHECK A PIVOT POINT
CHKPIV   SA1        B6+2
         PL         X1,CHKPIV1
         BX1        -X1
CHKPIV1  BSS        0
         SA2        B6+3
         SA3        =10.0E-14
         RX2        X2*X3
         IX1        X1-X2
         RET
PASS     END
         IDENT      MAT
         ENTRY      MATADD,MATSUB,MATSML,INPROD,NORM
         EQ         PASS
* ADD TWO ARRAYS
*GET THE PARAMETERS
MATADD   SA1        B6+2
         SA2        B6+3
         SA3        B6+4
         SA4        B6+5
         SB4        X7
* SET UP LOOP WITH COUNTER IN B7
         SA1        X1+B2
         SA2        X2+B2
         SB3        X4
         SB7        0
* ADD NEXT PAIR AND STORE IF IN RANGE
AD1      SA1        A1+B1
         ID         X1,UNINT
         SA2        A2+B1
         ID         X2,UNINT
         RX7        X1+X2
         OR         X7,ADERR
         NX7        X7
         SA7        X3+B7
```

```
*CHECK FOR THE LOOP BEING DONE                                MISCI
          SB7       B7+B1                                     MISCI
          LT        B7,B3,AD1                                 MISCI
          JP        B4                                        MISCI
* HANDEL ERRORWHEN ADDING                                     MISCI
ADERR     SX7       1                                         MISCI
          SA7       B6+2                                      MISCI
          EQ        =XEERROR                                  MISCI
* SUBTRACT TWO ARRAYS                                         MISCI
*GET THE PARAMETERS                                           MISCI
MATSUB    SA1       B6+2                                      MISCI
          SA2       B6+3                                      MISCI
          SA3       B6+4                                      MISCI
          SA4       B6+5                                      MISCI
          SB4       X7                                        MISCI
* SET UP LOOP WITH COUNTER IN B7                              MISCI
          SA1       X1+B2                                     MISCI
          SA2       X2+B2                                     MISCI
          SB3       X4                                        MISCI
          SB7       0                                         MISCI
* SUB NEXT PAIR AND STORE IF IN RANGE                         MISCI
SB1       SA1       A1+B1                                     MISCI
          ID        X1,UNINT                                  MISCI
          SA2       A2+B1                                     MISCI
          ID        X2,UNINT                                  MISCI
          RX7       X1-X2                                     MISCI
          OR        X7,SBERR                                  MISCI
          NX7       X7                                        MISCI
          SA7       X3+B7                                     MISCI
*CHECK FOR THE LOOP BEING DONE                                MISCI
          SB7       B7+B1                                     MISCI
          LT        B7,B3,SB1                                 MISCI
          JP        B4                                        MISCI
* HANDEL ERRORWHEN SUBING                                     MISCI
SBERR     SX7       2                                         MISCI
          SA7       B6+2                                      MISCI
          EQ        =XEERROR                                  MISCI
* MULTIPLY A ARRAY BY A SCALAR                                MISCI
MATSML    SA1       B6+2          SCALAR                      MISCI
          SA2       B6+3          MATRIX BASE TO FETCH FROM   MISCI
          SA3       B6+4          MATRIX BASE TO STORE INTO   MISCI
          SA4       B6+5          SIZE OF ARRAY               MISCI
          SB4       X7                                        MISCI
* SET UP THE LOOP                                             MISCI
          SA2       X2+B2                                     MISCI
          SB3       X4                                        MISCI
          SB7       0                                         MISCI
* MULTIPLY NEXT ELEMENT AND STORE IF IN RANGE                 MISCI
SM1       SA2       A2+B1                                     MISCI
          ID        X2,UNINT                                  MISCI
          RX7       X2*X1                                     MISCI
          OR        X7,SMERR                                  MISCI
          SA7       X3+B7                                     MISCI
* CHECK FOR DONE                                              MISCI
          SB7       B7+B1                                     MISCI
          LT        B7,B3,SM1                                 MISCI
```

```
          JP        B4                                                MISCI
*HANDEL ERROR WHEN MULTIPLING BY A SCALAR                             MISCI
SMERR     SX7       3                                                 MISCI
          SA7       B6+2                                              MISCI
          JP        =XEERROR                                          MISCI
* FORM THE ROW NORM OF A SQUARE MATRIX                                MISCI
NORM      SA2       B6+2      PARAMETER                               MISCI
          SA3       X2        SAVE THE ADDRESS IN A3 AND FETCH FIRST  MISCI
          AX2       18        GET COUNT AND ADDRESS INCREMENT         MISCI
          SB4       X2        SAVE INCREMENT FOR A3 IN B4             MISCI
          SB7       B4        SAVE COUNT IN B7                        MISCI
          SX1       B0        CLEAR THE SUM                           MISCI
* INCLUDE NEXT NUMBER IN TOTAL                                        MISCI
NM2       ID        X3,UNINT                                          MISCI
          RX3       X3*X3     SQUARE THE NUMBER                       MISCI
          OR        X3,SMERR                                          MISCI
          RX1       X1+X3                                             MISCI
          OR        X1,ADERR                                          MISCI
          NX1       X1                                                MISCI
* CHECK FOR BEING DONE                                                MISCI
          SB7       B7+B2                                             MISCI
          LE        B7,NM1                                            MISCI
* GET NEXT VALUE                                                      MISCI
          SA3       A3+B4                                             MISCI
          EQ        NM2                                               MISCI
* TAKE THE SQUARE ROOT OF THE RESULT TO GET NORM                      MISCI
* SAVE RETURN ADDRESS                                                 MISCI
NM1       SA7       B6                                                MISCI
          SX7       B6                                                MISCI
          SA7       =XGLOBALO                                         MISCI
          SB1    1                                                    MISCI
          SB2    -48                                                  MISCI
          SA5          CA       .LOAD CA                              MISCI
          PX7          B2,X1    .W*2(-48)                             MISCI
          SA4          A5+B1    .LOAD CB                              MISCI
          FX0          X5*X7    .CA*W                                 MISCI
          UX5          B3,X1    .PICK OFF N-48 AND COEFFICIENT OF W   MISCI
          SX3          B3-B2    .N-48 -(-48) = N = 2*K + R            MISCI
          FX2          X4+X0    .CA*W + CB = INITIAL GUESS            MISCI
          AX6          X3,B1    .K = N/2                              MISCI
          IX5          X3-X6    .N-K                                  MISCI
          FX4          X2*X2    .B*B                                  MISCI
          IX3          X5-X6    .N-K-K = R                            MISCI
          SA5          X3+TWOR  LOAD 2.**(R/2)*2(-1)                  MISCI
          SB2          X6       .B2 = K                               MISCI
          FX0          X4+X7    .B*B + W                              MISCI
          FX3          X2*X0    .B*(B*B + W) = DENOM                  MISCI
          UX6          B3,X5    .                                     MISCI
          SB2          B2+B3    .INCORPORATE K INTO EXPONENT OF 2**(R/2)*2 MISCI
          PX6          B2,X6    . 2(K)*2(R/2)*2(-1)                   MISCI
          NX3       B0,X3                                             MISCI
          FX2          X6/X3    .(2(K)*2(R/2)*2(-1) )/DENOM = TERM1   MISCI
          FX5          X0*X0    .(B*B + W)**2                         MISCI
          FX7          X7+X7    .2*W                                  MISCI
          FX7          X7+X7    .4*W                                  MISCI
          FX3          X7*X4    .4*W*B*B                              MISCI
```

```
         FX7      X3+X5    .NUM = (B*B+W)**2 + 4*W*B*B           MISCI
         FX0      X7*X2    .TERM1*NUM = 2*U                      MISCI
         SA5      A4+B1    .LOAD .25                             MISCI
         NX0      B0,X0                                          MISCI
         FX2      X1/X0    .ARG/(2*U)                            MISCI
         FX3      X5*X0    . .25*(2*U)                           MISCI
         FX1      X2+X3    .SUM OF ABOVE = SQRT(ARG)             MISCI
         SA2      =XGLOBALO                                      MISCI
         SB5      =XGLOBALO                                      MISCI
         SB6      X2                                             MISCI
         SB1      1                                              MISCI
         SB2      -1                                             MISCI
         MX6      0                                              MISCI
         SA2      B6                                             MISCI
         SB4      X2                                             MISCI
         JP       B4                                             MISCI
 CA      DATA       .585786437                                   MISCI
 CB      DATA       .4204951288                                  MISCI
 QUART   DATA     17164000000000000018                           MISCI
         DATA     .353553390593                                  MISCI
 TWOR    DATA       .5                                           MISCI
         DATA       .707106781186                                MISCI
*  FORM INNER PRODUCT OF ROW AND COLUM                           MISCI
*  GET THE PARAMETERS                                            MISCI
INPROD   SB4      X7                                             MISCI
         SA2      B6+2                                           MISCI
         SA3      B6+3                                           MISCI
         SA4      B6+4                                           MISCI
         SA1      B6+5                                           MISCI
*  SET UP COUNTERS ETC.                                          MISCI
         SB3      X4                                             MISCI
         SB7      X1                                             MISCI
         SX1      B0                                             MISCI
         LE       B7,IN2                                         MISCI
         SA2      X2                                             MISCI
         SA3      X3                                             MISCI
* CHECK FOR≤ OEOVALUE BEING UNINTIALIZED                         MISCI
IN1      ID       X2,UNINT                                       MISCI
         ID       X3,UNINT                                       MISCI
*  MULTIPLY AND CHECK RANGE                                      MISCI
         RX2      X2*X3                                          MISCI
         OR       X2,SMERR                                       MISCI
*  ADD TO THE TOTAL                                              MISCI
         RX1      X1+X2                                          MISCI
         OR       X1,ADERR                                       MISCI
         NX1      X1                                             MISCI
*  CHECK FOR DONE                                                MISCI
         SB7      B7-B1                                          MISCI
         LE       B7,IN2                                         MISCI
*  GET NEXT 2 VARIABLES                                          MISCI
         SA2      A2+B1                                          MISCI
         SA3      A3+B3                                          MISCI
         EQ       IN1                                            MISCI
*  RETURN                                                        MISCI
IN2      JP       B4                                             MISCI
* HANDEL ERROR OF UNINITIALIZED ELEMENT USED                     MISCI
```

```
UNINT     SX7       11                                                MISCI    430
          SA7       B6+2                                              MISCI    431
          JP        =XEERROR                                          MISCI    432
PASS      END                                                         MISCI    433
          IDENT  SINCOS                                               MATHLIB    2
          ENTRY      SIN                                              MATHLIB    3
          ENTRY      COS                                              MATHLIB    4
*MACRO TO SAVE THE IMPORTANT REGISTERS                                MATHLIB    5
SAVE      MACRO                                                       MATHLIB    6
          SX6       B3                                                MATHLIB    7
          SA6       =XSAVREGS                                         MATHLIB    8
          SX6       B4                                                MATHLIB    9
          SA6       A6+B1                                             MATHLIB   10
          SX6       B6                                                MATHLIB   11
          SA6       A6+B1                                             MATHLIB   12
          SX6       A5                                                MATHLIB   13
          SA6       A6+B1                                             MATHLIB   14
          ENDM                                                        MATHLIB   15
* MACRO TO RESTORE THOSE REGISTERS AND THE CONSTANT REGISTERS         MATHLIB   16
RESTORE   MACRO                                                       MATHLIB   17
          SB1       1                                                 MATHLIB   18
          SB2       -B1                                               MATHLIB   19
          SB5       =XGLOBALO                                         MATHLIB   20
          MX6       0                                                 MATHLIB   21
          SA1       =XSAVREGS                                         MATHLIB   22
          SA2       A1+B1                                             MATHLIB   23
          SA3       A2+B1                                             MATHLIB   24
          SA4       A3+B1                                             MATHLIB   25
          SB3       X1                                                MATHLIB   26
          SB4       X2                                                MATHLIB   27
          SB6       X3                                                MATHLIB   28
          SA5       X4                                                MATHLIB   29
          ENDM                                                        MATHLIB   30
SIN       SB7       0                                                 MATHLIB   31
          EQ        SINCOS                                            MATHLIB   32
COS       SB7       1                                                 MATHLIB   33
SINCOS    SAVE                                                        MATHLIB   34
          BX3       X7                                                MATHLIB   35
          SA2       TWOPI                                             MATHLIB   36
          FX7         X3*X2             .N = ARG*2/PI                 MATHLIB   37
          MX0     0                                                   MATHLIB   38
          SB1         1                                               MATHLIB   39
          PX2         X0                .GET MASK = 20000000000000000008 MATHLIB   40
          FX4         X2+X7             .                             MATHLIB   41
          DX6         X2+X7             .                             MATHLIB   42
          RX0         X4+X6             .ROUNDED N IN X0              MATHLIB   43
          NX1         B2,X0             .NORMALIZE N FOR RANGE REDUCTION  MATHLIB   44
          BX6         X2-X0             .N IN INTEGER FORMAT          MATHLIB   45
          IX0         X2+X6             .MAKE INTEGER POSITIVE        MATHLIB   46
          SA2         A2+B1             .PI/2-U                       MATHLIB   47
      EQ        B2,B0,=XSCRNG                                         MATHLIB   48
          SB6         B7                                              MATHLIB   49
          FX7         X1*X2             .N*(PI/2-U)U = H1             MATHLIB   50
          SA4         A2+B1             .PI/2-L                       MATHLIB   51
          ZR          X6,NRR            .N=0, NO RANGE REDUCTION NECESSAR MATHLIB   52
          FX6         X3-X7             .(ARG-H1)U                    MATHLIB   53
```

```
         DX5      X2*X1          .N*(PI/2-U)L = H2                   MATHLIB   54
         DX2      X3-X7          .(ARG-H1)L                          MATHLIB   55
         NX6      B0,X6           .                                  MATHLIB   56
         FX7      X4*X1          .N*(PI/2-L)U                        MATHLIB   57
         FX2      X2-X5          .                                   MATHLIB   58
         SX1      B7             .K TO X1                            MATHLIB   59
         FX2      X2-X7          .LOW ORDER TERMS                    MATHLIB   60
         IX0      X0+X1          .N + K + 2**58,,, .GT. 0            MATHLIB   61
         MX5      58             .X5=-3                              MATHLIB   62
         BX7      -X5*X0         . K = (N+K)MOD4                     MATHLIB   63
         RX6      X6+X2          .FINAL REDUCED ARG= X               MATHLIB   64
         SB6      X7             .K BACK TO B6                       MATHLIB   65
         NX3      B0,X6           .   NORMALIZED                     MATHLIB   66
NRR      SX6      B6             .K TO X6                            MATHLIB   67
         FX0      X3*X3          .Z=X*X                              MATHLIB   68
         LX6      59             .K = 0,2 USE SINE EVALUATION        MATHLIB   69
         SB5      B7+B7                                              MATHLIB   70
         PL       X6,SINE        .                                   MATHLIB   71
         SA1      A4+B1          .C5                                 MATHLIB   72
         FX7      X0*X0          .Z**2                               MATHLIB   73
         SA2      A1+B1          .C6                                 MATHLIB   74
         FX4      X0*X1          .C5*Z                               MATHLIB   75
         FX5      X2*X7          .C6*Z**2                            MATHLIB   76
         SA3      A2+B1          .C4                                 MATHLIB   77
         FX6      X4+X3          .C5*Z+C4                            MATHLIB   78
         SA4      A3+B1          .C3                                 MATHLIB   79
         FX1      X4*X0          .C3*Z                               MATHLIB   80
         FX6      X6+X5          .C6*Z*Z+C5*Z+C4 = ( 1 )             MATHLIB   81
         FX2      X6*X7          .( 1 )*Z**2                         MATHLIB   82
         SA3      A4+B1          .C2                                 MATHLIB   83
         FX6      X1+X3          .C3*Z + C2                          MATHLIB   84
         SA4      A3+B1          .C1                                 MATHLIB   85
         FX6      X2+X6          .( 1 )Z**2+C3*Z+C2 = ( 2 )          MATHLIB   86
         FX0      X0*X4          .C1*Z                               MATHLIB   87
         SA5      A4+B1          .C0                                 MATHLIB   88
         FX7      X6*X7          .( 2 )*Z**2                         MATHLIB   89
         FX6      X7+X0          .( 2 )Z**2 + C1*Z                   MATHLIB   90
         RX0      X6+X5          .C3 + REST OF SERIES                MATHLIB   91
         JP       CONT           .DONE WITH COSINE SERIES            MATHLIB   92
SINE     SA1      S5             .S5                                 MATHLIB   93
         FX6      X1*X0          .S5*Z                               MATHLIB   94
         SA2      A1+B1          .S4                                 MATHLIB   95
         FX7      X0*X0          .Z**2                               MATHLIB   96
         FX5      X2+X6          .S5*Z+S4                            MATHLIB   97
         SA1      A2+B1          .S3                                 MATHLIB   98
         SA2      A1+B1          .S2                                 MATHLIB   99
         FX6      X5*X7          .(S5*Z+S4)*Z**2                     MATHLIB  100
         FX4      X1*X0          .S3*Z                               MATHLIB  101
         FX5      X6+X2          .(S5*Z+S4)Z**2 + S2                 MATHLIB  102
         SA2      A2+B1          .S1                                 MATHLIB  103
         FX5      X5+X4          .(S5*Z+S4)Z**2+S2+S3*Z = ( 3)       MATHLIB  104
         FX7      X5*X7          .( 3 )*Z**2                         MATHLIB  105
         FX0      X2*X0          .S1*Z                               MATHLIB  106
         SA1      A2+B1          .S0                                 MATHLIB  107
         FX4      X0+X7          .                                   MATHLIB  108
         RX5      X4+X1          .                                   MATHLIB  109
```

```
         RX0      X3*X5           .FINAL SINE TERM
 CONT    SX3      B1-B6           .1-K
         AX3      ?               .SIGN(1-K)
         SB7      B5+B7
         BX7      X0-X3
* FINISHED SO GET THE NEXT INSTRUCTION
         RESTORE
         SA7    B3
* GET NEXT OPERATION
         SA5    A5+B1
         SB4    X5
         JP     B4
 TWOPI   DATA      17175057460333447104B
 DCSC1   DATA      17206220773250420550B
 DCSC2   DATA      16406043230461461213B
 C5      DATA      61C53301014524016617B
 C6      DATA      16634334143344163607B
 C4      DATA      17C06400637375136457B
 C3      DATA      60712237223723423125B
 C2      DATA      17135252525252523467B
 C1      DATA      60610000000000000004B
 C0      DATA      17177777777777777777B
 S5      DATA      61111270663112476351B
 S4      DATA      16755616534201617573B
 S3      DATA      60741377140534357734B
 S2      DATA      17114210421041004142B
 S1      DATA      60622525252525255342B
 S0      DATA      17177777777777777770B
         END
         IDENT  ALNLOG
         ENTRY  ALOG,ALOG10
*MACRO TO SAVE THE IMPORTANT REGISTERS
SAVE     MACRO
         SX6    B3
         SA6    =XSAVREGS
         SX6    B4
         SA6    A6+B1
         SX6    B6
         SA6    A6+B1
         SX6    A5
         SA6    A6+B1
         ENDM
* MACRO TO RESTORE THOSE REGISTERS AND THE CONSTANT REGISTERS
RESTORE  MACRO
         SB1    1
         SB2    -B1
         SB5    =XGLOBALO
         MX6    0
         SA1    =XSAVREGS
         SA2    A1+B1
         SA3    A2+B1
         SA4    A3+B1
         SB3    X1
         SB4    X2
         SB6    X3
         SA5    X4
```

```
         ENDM
ALOG10   SAVE
         SB3     B0
         SA0     B0                  .SET ENTRY SWITCH ZERO
         EQ      LNLOG
 ALOG    SAVE
         SB3     777
         SA0     777                 .SET ENTRY SWITCH NON-ZERO
 LNLOG   BX1        X7
         SA2     SQ2                 . 1.414...*2.47
         UX7     B2,X1               .
      ZR    X1,=XZERCLOG
      NG    X1,=XNEGLOG
         SB3     -47                 .TRY K=- 47
         SB1     1
         IX6     X7-X2               .
         NG      X6,GL               .
         SB3     B3-B1               .NEED K = -48
 GL      PX7     B3,X7               .FORM W = 2.K*C
         SA5     A2+B1               .LOAD 1.0
         FX0     X7-X5               .(W-1.0)
         NX2     B0,X0
         DX0     X7-X5
         RX0     X2+X0
         RX2     X7+X5
         RX0     X0/X2
         FX7     X0*X0               .Z=T*T
         SA1     A5+B1               .D0
         SA2     A1+B1               .D1
         SA3     A2+B1               .D2
         FX6     X7*X2               .Z*D1
         FX5     X7*X7               .Z*Z
         FX1     X1+X6               .D0 + Z*D1
         FX6     X5*X3               .D2*Z*Z
         FX4     X5*X7               .Z**3
         SA2     A3+B1               .D3
         FX1     X1+X6               .D0 + D1*Z + D2*Z*Z
         FX3     X4*X2               .D3*Z**3
         FX1     X1+X3               .TOTAL DENOMINATOR
         NX4     B0,X1
         FX6     X0/X4
         SA2     A2+B1          .C1
         SA1     A2+B1          .C2
         FX7     X7*X2          .C1*Z
         FX5     X1*X5          .C2*Z**2
         FX3     X3+X3          .2*C3*Z**3
         FX0     X0+X0          .2*T
         FX4     X7+X3          .ADD C1*Z
         SX3     B2-B3
         FX4     X4+X5          .ADD C2*Z**2
         PX1     X3
         FX7     X6*X4          .FINAL TERM OF Q
         NX5     B0,X1
         SA3     A1+B1          .LOG(2.0)
         SA2     A3+B1
         FX6     X5*X3
```

```
          FX4    X0-X7
          FX1    X5*X2
          DX5    X0-X7
          NX4     B0,X4
          RX5    X4+X5
          SA2    A2+B1                .LOAD LOG10(E)
          RX5    X5+X1
          RX5    X5+X6
          SB3    A0
          NX7      B0,X5
          NZ     B3,RET
          RX7    X7*X2
* RETURN
RET       RESTORE
          JP        B4
 SQ2      DATA   5520236314774736B
 ONE      DATA   1.0
 D0       DATA   10395.0
 D1       DATA   60421030456556304033B
 D2       DATA   17344525326347004201B
 D3       DATA   -230.419130393980937
 C11      DATA   60431166777777776772B
 C2       DATA   17345152701555267627B
 LOGE2    DATA   17175427102775750000B
          DATA   16530717363257110000B
 LOG10    DATA   17166745573052233450B
          END
          IDENT  ASINCOS
          ENTRY  ASIN
          ENTRY  ACOS
*MACRO TO SAVE THE IMPORTANT REGISTERS
SAVE      MACRO
          SX6        B3
          SA6        =XSAVREGS
          SX6        B4
          SA6        A6+B1
          SX6        B6
          SA6        A6+B1
          SX6        A5
          SA6        A6+B1
          ENDM
* MACRO TO RESTORE THOSE REGISTERS AND THE CONSTANT REGISTERS
RESTORE   MACRO
          SB1        1
          SB2        -B1
          SB5        =XGLOBALO
          MX6        0
          SA1        =XSAVREGS
          SA2        A1+B1
          SA3        A2+B1
          SA4        A3+B1
          SB3        X1
          SB4        X2
          SB6        X3
          SA5        X4
          ENDM
```

```
ACOS      SB7       B1                                              MATHLIB   278
          EQ         ASINCOS                                        MATHLIB   279
ASIN      SB7       0                                               MATHLIB   280
ASINCOS   SAVE                                                      MATHLIB   281
          BX1        X7                                             MATHLIB   282
          SA2       ATA                                             MATHLIB   283
          SA3       XCRIT           FETCH CROSS-OVER POINT          MATHLIB   284
          BX6       X1                                              MATHLIB   285
          AX6       74B                                             MATHLIB   286
          SA6       =XGLOBALO+MASK                                  MATHLIB   287
          BX7       X6-X1           MAKE ARG POSTIVE                MATHLIB   288
          FX3       X7-X3                                           MATHLIB   289
          FX4       X7*X7                                           MATHLIB   290
          FX2       X2-X4         1-Z**2                            MATHLIB   291
          NG        X3,SERIES                                       MATHLIB   292
          SA7       =XGLOBALO+PARG                                  MATHLIB   293
          NX2       B0,X2                                           MATHLIB   294
          SB3       -60B                                            MATHLIB   295
          UX6       B6,X2         UNPACK                            MATHLIB   296
          SA4       SQD           START INITIAL APPROXIMATION       MATHLIB   297
          SB4       1                                               MATHLIB   298
          PX1       B3,X2         PACK F(X)  AS A FRACTION          MATHLIB   299
          SA3       A4+B4                                           MATHLIB   300
          FX5       X1*X4                                           MATHLIB   301
          AX6       74B           SET X6=0000..0                    MATHLIB   302
          FX0       X5+X3         FIRST APPROX                      MATHLIB   303
          FX3       X1/X0         F(X)/X1                           MATHLIB   304
          NG         X2,=XASACARG                                   MATHLIB   305
          ZR        X2,ASINA                                        MATHLIB   306
          SX2       B6-B3         EXPONENT + 60                     MATHLIB   307
          SX7       B4                                              MATHLIB   308
          BX4       X7*X2         1 IF ODD EXPONENT    0 IF EVEN    MATHLIB   309
          AX2       1             EXPONENT /2       2**(P/2)        MATHLIB   310
          AX7       X7,B3         PUT BIT IN POSITION 2**0 OF EXPONENT MATHLIB 311
          SB2       X4            INDEXING FOR FINAL CORRECTION     MATHLIB   312
          SB5       X2+B2         EXPONENT -60                      MATHLIB   313
          FX6       X0+X3         X1+(F(X)/X1)                      MATHLIB   314
          IX4       X6-X7         1/2(X1+(F(X)/X1)                  MATHLIB   315
          FX5       X1/X4         F(X)/X2                           MATHLIB   316
          PL        X2,SQB        SENSE ARGUMENT ) OR = U/2         MATHLIB   317
          SB2       B2-B4         SET NEW INDEXING VALUE FOR FRACTION MATHLIB 318
 SQB      SA3       B2+SQA        1/SQRT2  , 1.0  ,  OR SQRT2       MATHLIB   319
          FX6       X4+X5                                           MATHLIB   320
          IX0       X6-X7         3RD APPROXIMATION                 MATHLIB   321
          FX5       X1/X0                                           MATHLIB   322
          FX4       X0+X5                                           MATHLIB   323
          IX5       X4-X7                                           MATHLIB   324
          UX4       B6,X5                                           MATHLIB   325
          SB6       B6-B2         N IS 0 OR 1                       MATHLIB   326
          SB5       B5+B6         FINAL EXPONENT                    MATHLIB   327
          PX0       B5,X5         PACK                              MATHLIB   328
          FX6       X0*X3                                           MATHLIB   329
 ASINA    SA2       ATA           1.0                               MATHLIB   330
          SA3       =XGLOBALO+PARG                                  MATHLIB   331
          FX4       X2-X6         1-(1-Z**2)**1/2                   MATHLIB   332
          NX4       B0,X4                                           MATHLIB   333
```

```
         FX1      X4/X3                                                     MATHLIB   334
         SX6      B0                                                        MATHLIB   335
         ZR       X3,ATO                                                    MATHLIB   336
         SB3      B0                                                        MATHLIB   337
         SA3      ATB         SQ RT 2                                       MATHLIB   338
ATO      FX4      X3-X2       SQ RT 2 - 1                                   MATHLIB   339
         NX4      B0,X4                                                     MATHLIB   340
         FX5      X4-X1                                                     MATHLIB   341
         NX5      B0,X5                                                     MATHLIB   342
         PL       X5,ATP      ARG IS LESS OR = SQ RT 2 - 1                  MATHLIB   343
         FX4      X3+X2       FORM SQ RT 2 + 1                              MATHLIB   344
         NX4      B0,X4       NORMALIZE                                     MATHLIB   345
         FX5      X4-X1                                                     MATHLIB   346
         PL       X5,ATR      LESS OR = SQ RT 2 +1                          MATHLIB   347
         BX2      -X2         FORM -1                                       MATHLIB   348
         FX1      X2/X1       -1/Z                                          MATHLIB   349
         SB3      2                                                         MATHLIB   350
ATP      SA2      ATI                                                       MATHLIB   351
         FX3      X1*X1       TSQ                                           MATHLIB   352
         SA4      ATH         E3                                            MATHLIB   353
         FX2      X2+X3       T SQ + D4                                     MATHLIB   354
         BX6      X3                                                        MATHLIB   355
         SA5      ATG                                                       MATHLIB   356
         NX2      B0,X2                                                     MATHLIB   357
         FX4      X4/X2                                                     MATHLIB   358
         SA3      ATF         E2                                            MATHLIB   359
         FX5      X5+X6                                                     MATHLIB   360
         SA2      ATE                                                       MATHLIB   361
         NX5      B0,X5                                                     MATHLIB   362
         FX5      X5+X4                                                     MATHLIB   363
         NX5      B0,X5                                                     MATHLIB   364
         FX3      X3/X5                                                     MATHLIB   365
         FX2      X2+X6                                                     MATHLIB   366
         SA4      ATD         E1                                            MATHLIB   367
         NX2      B0,X2                                                     MATHLIB   368
         FX6      X2+X3                                                     MATHLIB   369
         SA5      ATC         D1                                            MATHLIB   370
         FX6      X4/X6                                                     MATHLIB   371
         SA3      B3+ATJ                                                    MATHLIB   372
         FX6     X6+X5                                                  ASI MATHLIB   373
         NX6      B0,X6                                                     MATHLIB   374
         FX6      X6*X1       *T                                            MATHLIB   375
         EQ       B3,B0,ATQ                                                 MATHLIB   376
         FX6      X6+X3                                                     MATHLIB   377
         NX6      B0,X6                                                     MATHLIB   378
ATQ      SA1      =XGLOBALO+MASK                                            MATHLIB   379
         FX6      X6+X6                                                     MATHLIB   380
         BX6      X6-X1                                                     MATHLIB   381
         SA2      PIOVER2                                                   MATHLIB   382
         ZR       B7,DONE                                                   MATHLIB   383
         FX6      X2-X6                                                     MATHLIB   384
         NX6      B0,X6                                                     MATHLIB   385
         EQ       DONE                                                      MATHLIB   386
ATR      SB3      1                                                         MATHLIB   387
         FX3      X1-X2                                                     MATHLIB   388
         FX4      X1+X2                                                     MATHLIB   389
```

```
          NX3       B0,X3
          NX4       B0,X4
          FX1       X3/X4
          EQ        ATP
 SERIES   SA1       S7             HERE TO MACLAURIN SERIES NEAR ORIGIN
          SA2       S5
          SA3       S3
          FX5       X1*X4
          FX5       X5+X2
          FX5       X5*X4
          SA1       PIOVER2
          FX5       X5+X3
          FX5       X5*X4
          FX5       X5*X7
          FX5       X5+X7          ((S7*X**2 + S5)*X**2 + S3)*X**2*X + X
          BX6       X6-X5          ATTACH SIGN
          ZR        B7,DONE
          FX6       X1-X6
          NX6       B0,X6
DONE      BX7        X6
          RESTORE
          SA7        B2
          SA5        A5+B1
          SB4        X5
          JP         B4
 XCRIT    DATA      .05
 S7       DATA      .0446428571428571
 S5       DATA      .075
 S3       DATA      .1666666666666667
 PIOVER2  DATA      17206220773250420550B
          DATA      17175520236314774740B         1/SQRT2
 SQA      DATA      17204000000000000000B
          DATA      17205520236314774735B
 SQD      DATA      17175374211460550654B
          DATA      17165374211460550654B
 MASK     EQU        88
 PARG     EQU        89
 ATA      DATA      17204000000000000000B
 ATB      DATA      17205520236314750656B
 ATC      DATA      17156342227732547734B
 ATD      DATA      17216164452160711744B
 ATE      DATA      17225317772277306204B
 ATF      DATA      60560111317142532541B
 ATG      DATA      17215337417031405016B
 ATH      DATA      60623134652277270450B
 ATI      DATA      17205265617124327034B
 ATJ      DATA      0
          DATA      17176220773250420550B
          DATA      17206220773250420550B
          END
          IDENT  ATAN
          ENTRY  ATAN
*MACRO TO SAVE THE IMPORTANT REGISTERS
SAVE      MACRO
          SX6        B3
          SA6        =XSAVREGS
```

```
         SX6       B4
         SA6       A6+B1
         SX6       B6
         SA6       A6+B1
         SX6       A5
         SA6       A6+B1
         ENDM
* MACRO TO RESTORE THOSE REGISTERS AND THE CONSTANT REGISTERS
RESTORE  MACRO
         SB1       1
         SB2       -B1
         SB5       =XGLOBALO
         MX6       0
         SA1       =XSAVREGS
         SA2       A1+B1
         SA3       A2+B1
         SA4       A3+B1
         SB3       X1
         SB4       X2
         SB6       X3
         SA5       X4
         ENDM
ATAN     SAVE
         BX2       X7
         SA1    ONE
         BX0    X2
         SA4    TANPI             LOAD TAN(PI/16)
         SB1    1
         AX2    59
         SB7    X2+777777B        SAVE SIGN IN B7
         BX3    X2-X0             X3  =  ABS(ARG)
 AT1     FX7       X3-X1             .X-1
         SA5       A4+B1             .SQ2-1= .414......
         PL        X7,LARG           .GO TO LARG IF X .GE. 1.0
         IX7       X3-X4             .X-TAN(PI/16)
         FX6       X3-X5             .X-(SQ2-1)
         SB4       B0                .FLAG = 0
         SA5       A5+B1             .LOAD TAN(3PI/16)
         NG        X7,NRR            .IF X .LT. TAN(PI/16)
         PL        X6,T2
 T1      FX7       X3*X4             .X*TAN(PI/16)
         RX6       X3-X4             .X-TAN(PI/16)
         SB4       B1                .FLAG = 1
         NX0       B0,X6
         RX7       X1+X7             . 1.0 + X*TAN(PI/16)
         RX3       X0/X7             .DIVIDE TO GET R
         JP        NRR
 T2      FX7       X3*X5             .X*TAN(3PI/16)
         RX6       X3-X5             .X-TAN(3PI/16)
         SB4       B1+B1             .FLAG=2
         NX0       B0,X6
         RX7       X1+X7             .1.0 + X*TAN(3PI/16)
         RX3       X0/X7             .DIVIDE TO GET R
         JP        NRR
 T4      FX7       X3*X4             .X*TAN(PI/16)
         FX0       X3+X4             .X+TAN(PI/16)
```

```
         SB4        B4+B4          .FLAG = 4                        MATHLIB
         FX6        X7-X1          .X*TAN(PI/16) - 1.0              MATHLIB
         NX5        B0,X6             .                             MATHLIB
         FX3        X5/X0          .R                               MATHLIB
         JP         NRR            .                                MATHLIB
LARG     SA2        A1-B1          .SQ2+1= 2.1414.....              MATHLIB
         SA5        A5+B1          .TAN(3PI/16)                     MATHLIB
         FX0        X3-X2          .X-(SQ2+1)                       MATHLIB
         SB4        B1+B1          .                                MATHLIB
T3       PL         X0,T4          .                                MATHLIB
         FX7        X3*X5          .X*TAN(3PI/16)                   MATHLIB
         RX0        X3+X5          .                                MATHLIB
         RX6        X7-X1          .X*TAN(3PI/16) - 1.0             MATHLIB
         NX5        B0,X6             .                             MATHLIB
         RX3        X5/X0          .R                               MATHLIB
         SB4        B4+B1          .FLAG = 3                        MATHLIB
NRR      RX0        X3*X3          .Z = R*R                         MATHLIB
         SA1        A5+B1          .DB3                             MATHLIB
         SA2        A1+B1          .DB2                             MATHLIB
         FX7        X0*X0          .Z**2                            MATHLIB
         FX6        X1*X0          .DB3*Z                           MATHLIB
         FX5        X6+X2          .(DB3*Z+DB2)                     MATHLIB
         SA4        A2+B1          .DB1                             MATHLIB
         FX1        X4*X0          .DB1*Z                           MATHLIB
         FX2        X5*X7          .  (    )*Z**2                   MATHLIB
         SA5        A4+B1          .DB0                             MATHLIB
         FX6        X2+X1                                           MATHLIB
         FX1        X6+X5          .TOTAL DENOMINATOR               MATHLIB
         RX6        X3/X1          .R/DENOM                         MATHLIB
         SA2        A5+B1          .DA3                             MATHLIB
         FX5        X2*X0          .DA3*Z                           MATHLIB
         SA1        A2+B1          .DA2                             MATHLIB
         SA4        A1+B1          .DA1                             MATHLIB
         FX2        X5+X1          .(DA3*Z+DA2) = ( 2)              MATHLIB
         FX7        X7*X2          .( 2)*Z**2                       MATHLIB
         FX0        X4*X0          . DA1*Z                          MATHLIB
         SA1        A4+B1          .DA0                             MATHLIB
         FX2        X7+X0          .( 2)*Z**2 + DA1*Z               MATHLIB
         FX2        X1+X2          .DA0 + ABOVE                     MATHLIB
         RX7        X6*X2          .REST OF SERIES                  MATHLIB
         RX2        X3-X7          .ATAN(R)                         MATHLIB
         SA4        CORO+B4        .LOAD CORRECTION TERM 1          MATHLIB
         SA1        A4+5           .LOAD CORRECTION TERM 2          MATHLIB
         NX2        B0,X2             .                             MATHLIB
         RX5        X2+X4          .                                MATHLIB
         NX4        B0,X5             .                             MATHLIB
         RX5        X4+X1          .                                MATHLIB
         NX3        B0,X5             .                             MATHLIB
         SX2     B7-B0                                              MATHLIB
         BX7        X3-X2          .SIGN(ARG)*ATAN(ABS(ARG)   )     MATHLIB
         RESTORE                                                    MATHLIB
         SA7       B3                                               MATHLIB
         SA5       A5+1                                             MATHLIB
         SB4       X5                                               MATHLIB
         JP        B4                                               MATHLIB
SQ2PI    DATA    2.414213                                           MATHLIB
```

```
 ONE      DATA   1.0
 TANP1    DATA   17156272765700524613B
 SQ2M1    DATA   .414213
 TAN3P    DATA   17175260670125337715B
 DB3      DATA   17355170257740451612B
 DB2      DATA   17405760164664461126B
 DB1      DATA   17416476573535613424B
 DB0      DATA   17414077370000000000B
 DA3      DATA   17354115643217650102B
 DA2      DATA   17375417632175023115B
 DA1      DATA   17375377237777777770B
 DA0      DATA   16507544514504101314B
 CORO     DATA   0.0
          DATA   16366000000000000000B
          DATA   17156220773250420550B
          DATA   17167665172122524702B
          DATA   17175376673723356473B
          DATA   0.0
          DATA   17156220773250420550B
          DATA   17166220773250420551B
          DATA   17167665172122524703B
          DATA   17175376673723356474B
          END
          IDENT  EXP
          ENTRY  EXP
*MACRO TO SAVE THE IMPORTANT REGISTERS
SAVE      MACRO
          SX6        B3
          SA6        =XSAVREGS
          SX6        B4
          SA6        A6+B1
          SX6        B6
          SA6        A6+B1
          SX6        A5
          SA6        A6+B1
          ENDM
* MACRO TO RESTORE THOSE REGISTERS AND THE CONSTANT REGISTERS
RESTORE   MACRO
          SB1        1
          SB2        -B1
          SB5        =XGLOBALO
          MX6        0
          SA1        =XSAVREGS
          SA2        A1+B1
          SA3        A2+B1
          SA4        A3+B1
          SB3        X1
          SB4        X2
          SB6        X3
          SA5        X4
          ENDM
 EXP      SAVE
          BX3        X7
          SA1    XMAX                GET XMAX
          MX0    0
          SB1    1
```

```
      SA5     A1+B1             GET XMIN                                 MATHLIB    614
      FX7     X1-X3             XMAX-X                                   MATHLIB    615
      SA2     A5+B1             GET LOG2(E)                              MATHLIB    616
      FX1     X3-X5             X-XMIN                                   MATHLIB    617
      PX4     X0                                                         MATHLIB    618
      BX7     X7-X1             SIGN OF (XMAX-X)*(X-XMIN)                MATHLIB    619
      FX6     X2*X3             X*LOG2(E)                                MATHLIB    620
     NG         X7,=XINFFXP                                              MATHLIB    621
      FX7     X4+X6                                                      MATHLIB    622
      DX6     X4+X6                                                      MATHLIB    623
      SB1     1                                                          MATHLIB    624
      RX7     X7+X6             .N IS NOW AN INTEGER WITH A 2000 EXPON   MATHLIB    625
      NX6     B2,X7             .NORMALIZE N FOR RANGE REDUCTION         MATHLIB    626
      SA4     A2+B1             .LOG(2) UPPER                            MATHLIB    627
      SA5     A4+B1             .LOG(2) LOWER                            MATHLIB    628
      FX0     X6*X4             .N*LOG(2) UPPER                          MATHLIB    629
      FX1     X6*X5             .N*LOG(2) LOWER                          MATHLIB    630
      FX6     X3-X0             .                                        MATHLIB    631
      NX2     B0,X6                 .                                    MATHLIB    632
      DX3     X3-X0             .                                        MATHLIB    633
      FX1     X3-X1             .                                        MATHLIB    634
      FX3     X2+X1                                                      MATHLIB    635
      NX0     B0,X3                 .FINAL VALUE OF X                    MATHLIB    636
      SB2     X7          .PICK UP N                                     MATHLIB    637
      RX7     X0*X0                                                      MATHLIB    638
      SA1     A5+B1       .C1=420.0                                      MATHLIB    639
      SA2     A1+B1       .C0=15120.0                                    MATHLIB    640
      FX6     X1*X7       .C1*Z                                          MATHLIB    641
      FX5     X7*X7       .Z**2                                          MATHLIB    642
      RX3     X6+X2       .C1*Z + C0                                     MATHLIB    643
      SA1     A2+B1       .C3=28.0                                       MATHLIB    644
      FX6     X1*X7       .C3*Z                                          MATHLIB    645
      RX3     X3+X5       .C1*Z + C0 + Z*Z = B                           MATHLIB    646
      SA2     A1+B1       .C4=2520.0                                     MATHLIB    647
      FX5     X0*X3       .X*B                                           MATHLIB    648
      RX2     X6+X2       .C4 + C3*Z                                     MATHLIB    649
      FX4     X7*X2       .Z*T                                           MATHLIB    650
      FX1     X3+X3       .2*B                                           MATHLIB    651
      RX6     X1-X5       .2*B - X*B                                     MATHLIB    652
      RX1     X6+X4       .Z*T + 2*B - X*B = DENOM                       MATHLIB    653
      NX1       B0,X1                                                    MATHLIB    654
      RX7     X0/X1       .TERM1 = X/DENOM                               MATHLIB    655
      RX4     X5-X4       .X*B - Z*T = TERM2                             MATHLIB    656
      RX5     X7*X4       .Q = TERM1*TERM2                               MATHLIB    657
      SA1     A2+B1       .LOAD 1.0                                      MATHLIB    658
      FX2     X1+X0                                                      MATHLIB    659
      DX3     X1+X0                                                      MATHLIB    660
      NX2     B0,X2                                                      MATHLIB    661
      FX4     X2+X5                                                      MATHLIB    662
      DX7     X2+X5                                                      MATHLIB    663
      RX7     X3+X7                                                      MATHLIB    664
      RX6     X4+X7                                                      MATHLIB    665
      UX7     B3,X6                                                      MATHLIB    666
      SB2     B2+B3                                                      MATHLIB    667
      PX7     B2,X6                                                      MATHLIB    668
      RESTORE                                                            MATHLIB    669
```

```
          JP        B4
 XMAX     DATA   741.67
 XMIN     DATA   -675.82
 LOG2E    DATA   17205612507312256030B
 LOG2U    DATA   17175427102777575000 0B
 LOG2L    DATA   16530717363257117073B
 C1       DATA   420.0
 CO       DATA   15120.0
 C3       DATA   28.C
 C4       DATA   2520.0
 ONE      DATA   1.0
          END
          IDENT  TAN
          ENTRY     TAN
*MACRO TO SAVE THE IMPORTANT REGISTERS
SAVE      MACRO
          SX6       B3
          SA6       =XSAVREGS
          SX6       B4
          SA6       A6+B1
          SX6       B6
          SA6       A6+B1
          SX6       A5
          SA6       A6+B1
          ENDM
* MACRO TO RESTORE THOSE REGISTERS AND THE CONSTANT REGISTERS
RESTORE   MACRO
          SB1       1
          SB2       -B1
          SB5       =XGLOBALO
          MX6       0
          SA1       =XSAVREGS
          SA2       A1+B1
          SA3       A2+B1
          SA4       A3+B1
          SB3       X1
          SB4       X2
          SB6       X3
          SA5       X4
          ENDM
 TAN      SAVE
          BX2        X7
          SA1       SAF             FETCH 2PI
          BX6       X2              SEND ARG TO X6
          PL        X2,TAB          JUMP IF ARG POS
          BX2       -X2             OTHERWISE COMPLEMENT
 TAB      FX3       X1-X2           TEST ARG (2PI
          SA5       SAH             FETCH PI
          PL        X3,TAC          JUMP IF ARG ( 2PI
          FX3       X2/X1           CTHERWISE SUBTRACT
          UX3       B3,X3           MULTIPLE OF
           PL       B3,=XTNRNG
          LX3       X3,B3           2PI TO REDUCE
          PX3       B0,X3           RANGE
          NX3       B0,X3
          RX3       X3*X1
```

```
         RX2     X2-X3                                              MATHLIB
         NX2     B0,X2                                              MATHLIB
TAC      FX0     X2-X5       TEST ARG ( PI                          MATHLIB
         SA3     SAL         FETCH PI/2                             MATHLIB
         NG      X0,TAD      JUMP IF ARG ( PI                       MATHLIB
         SA4     SAK         FETCH 2PU*                             MATHLIB
         RX2     X1-X2       2PI-ARG                                MATHLIB
         NX2     B0,X2       NORMALIZE                              MATHLIB
         RX2     X4+X2       2ND TERM DOUBLE PRECISION              MATHLIB
         BX6     -X6         COMPLEMENT SIGN FLAG                   MATHLIB
TAD      FX0     X2-X3       TEST ARG ( PI/2                        MATHLIB
         SA1     SAP         FETCH PI/4                             MATHLIB
         NG      X0,TAE      JUMP IF ARG (PI/2                      MATHLIB
         SA4     SAO         FETCH PI*                              MATHLIB
         RX2     X5-X2       PI-ARG                                 MATHLIB
         NX2     B0,X2       NORMALIZE                              MATHLIB
         RX2     X2+X4       IND TERM DOUBLE PRECISION              MATHLIB
         BX6     -X6         COMPLEMENT SIGN FLAG                   MATHLIB
TAE      SA6     =XGLOBALO+TAF                                      MATHLIB
         SA5     TAG         FETCH A                                MATHLIB
         FX1     X1-X2       TEST ARG ( PI/4                        MATHLIB
         NG      X1,TAR      JUMP IF ARG ) PI/4                     MATHLIB
         RX7     X2*X2       X2ND TO X7                             MATHLIB
         RX0     X2*X5       AX TO X0                               MATHLIB
         SA1     TAH         FETCH B                                MATHLIB
         SA3     TAI         FETCH C                                MATHLIB
         RX6     X7*X1       BX2ND TO X6                            MATHLIB
         RX7     X7*X2       X3RD TO X7                             MATHLIB
         RX1     X5-X6       A-BX2ND TO X1                          MATHLIB
         SA5     TAJ         FETCH D                                MATHLIB
         NX1     B0,X1       NORMALIZE                              MATHLIB
         RX6     X7*X3       CX3RD TO X6                            MATHLIB
         RX7     X7*X2       X4TH TO X7                             MATHLIB
         RX0     X0-X6       AX-CX3RD TO X0                         MATHLIB
         NX0     B0,X0       NORMALIZE                              MATHLIB
         SA3     TAK         FETCH E                                MATHLIB
         RX6     X5*X7       DX4TH TO X6                            MATHLIB
         RX7     X7*X2       X5TH TO X7                             MATHLIB
         RX1     X1+X6       A-BX2ND+DX4TH TO X1                    MATHLIB
         SA5     TAL         FETCH F                                MATHLIB
         RX6     X7*X3       EX5TH TO X6                            MATHLIB
         RX7     X7*X2       X6TH TO X7                             MATHLIB
         NX1     B0,X1       NORMALIZE                              MATHLIB
         RX0     X0+X6       AX-CX3RD+EX5TH TO X0                   MATHLIB
         SA3     TAM         FETCH G                                MATHLIB
         RX6     X7*X5       FX6TH TO X6                            MATHLIB
         RX7     X7*X2       X7TH TO X7                             MATHLIB
         RX1     X1-X6       A-BX2ND+DX4TH-FX6TH TO X1              MATHLIB
         SA5     =XGLOBALO+TAF                                      MATHLIB
         NX1     B0,X1       NORMALIZE                              MATHLIB
         RX6     X7*X3       GX7TH TO X6                            MATHLIB
         RX7     X7*X2       X8TH TO X7                             MATHLIB
         RX0     X0-X6       AX ETC -GX7TH                          MATHLIB
         NX0     B0,X0       NORMALIZE                              MATHLIB
         PL      X5,TAQ    TEST SIGN                                MATHLIB
         BX0     -X0       IF NEG COMPLEMENT                        MATHLIB
```

```
 TAQ      RX1      X1+X7        A+BXETC+X8TH                              MATHLIB
          NX1      B0,X1          NORMALIZE                               MATHLIB
          RX6      X0/X1          DIVIDE                                  MATHLIB
          EQ       B0,B0,TAA            EXIT                              MATHLIB
 TAR      SA4      SAQ            FETCH PI/2*                             MATHLIB
          SA5      TAG            FETCH A                                 MATHLIB
          FX2      X3-X2          PI/2-ARG                                MATHLIB
          NX2      B0,X2          NORMALIZE                               MATHLIB
          RX2      X2+X4          PI/2-ARG+PI/2*                          MATHLIB
          RX7      X2*X2          X2ND TO X7                              MATHLIB
          RX0      X2*X5          AX TO X0                                MATHLIB
          SA1      TAH            FETCH B                                 MATHLIB
          SA3      TAI            FETCH C                                 MATHLIB
          RX6      X7*X1          BX2ND TO X6                             MATHLIB
          RX7      X7*X2          X3RD TO X7                              MATHLIB
          RX1      X5-X6          A-BX2ND TO X1                           MATHLIB
          SA5      TAJ            FETCH D                                 MATHLIB
          NX1      B0,X1          NORMALIZE                               MATHLIB
          RX6      X7*X3          CX3RD TO X6                             MATHLIB
          RX7      X7*X2          X4TH TO X7                              MATHLIB
          RX0      X0-X6          AX-CX3RD TO X0                          MATHLIB
          NX0      B0,X0          NORMALIZE                               MATHLIB
          SA3      TAK            FETCH E                                 MATHLIB
          RX6      X5*X7          DX4TH TO X6                             MATHLIB
          RX7      X7*X2          X5TH TP X7                              MATHLIB
          RX1      X1+X6          A-BX2ND+DX4TH TO X1                     MATHLIB
          SA5      TAL            FETCH F                                 MATHLIB
          RX6      X7*X3          EX5TH TO X6                             MATHLIB
          RX7      X7*X2          X6TJ TO X7                              MATHLIB
          NX1      B0,X1          NORMALIZE                               MATHLIB
          RX0      X0+X6          AX-CX3RD+EX5TH TO X0                    MATHLIB
          SA3      TAM            FETCH G                                 MATHLIB
          RX6      X7*X5          FX6TH TO X6                             MATHLIB
          RX7      X7*X2          X7TH TO X7                              MATHLIB
          RX1      X1-X6          A-BX2ND+DX4TH-FX6TH TO X1               MATHLIB
          SA5      =XGLOBALO+TAF                                          MATHLIB
          NX1      B0,X1          NORMALIZE                               MATHLIB
          RX6      X7*X3          GX7TH TO X6                             MATHLIB
          RX7      X7*X2          X8TH TO X7                              MATHLIB
          RX0      X0-X6          AX ETC -GX7TH                           MATHLIB
          NX0      B0,X0          NORMALIZE                               MATHLIB
          PL       X5,TAO         TEST SIGN                               MATHLIB
          BX0      -X0                                                    MATHLIB
 TAO      RX1      X1+X7          A+BXETC+X8TH                            MATHLIB
          NX1      B0,X1          NORMALIZE                               MATHLIB
          RX6      X1/X0          DIVIDE                                  MATHLIB
* EXIT                                                                    MATHLIB
TAA       BX7       X6                                                    MATHLIB
          RESTORE                                                         MATHLIB
          SA7       B3                                                    MATHLIB
          SA5       A5+1                                                  MATHLIB
          SB4       X5                                                    MATHLIB
          JP        B4                                                    MATHLIB
 TAF      EQU       89                                                    MATHLIB
 TAG      DATA     17447567021000000000B                                  MATHLIB
 TAH      DATA     17437157062000000000B                                  MATHLIB
```

```
 TAI       DATA     17424077370000000000B
 TAJ       DATA     17376260340000000000B     51975
 TAK       DATA     17346611000000000000B
 TAL       DATA     17314730000000000000B
 TAM       DATA     17254400000000000000B
 SAF       DATA     17226220773250420550B
 SAH       DATA     17216220773250420550B
 SAK       DATA     16426043230461461213B
 SAL       DATA     17206220773250420550B
 SAO       DATA     16416043230461461213B
 SAP       DATA     17176220773250420550B
 SAQ       DATA     16406043230461461213B
           END
           IDENT  SQRT
           ENTRY        SQRT
*MACRO TO SAVE THE IMPORTANT REGISTERS
SAVE       MACRO
           SX6      B3
           SA6      =XSAVREGS
           SX6      B4
           SA6      A6+B1
           SX6      B6
           SA6      A6+B1
           SX6      A5
           SA6      A6+B1
           ENDM
* MACRO TO RESTORE THOSE REGISTERS AND THE CONSTANT REGISTERS
RESTORE    MACRO
           SB1      1
           SB2      -B1
           SB5      =XGLOBAL0
           MX6      0
           SA1      =XSAVREGS
           SA2      A1+B1
           SA3      A2+B1
           SA4      A3+B1
           SB3      X1
           SB4      X2
           SB6      X3
           SA5      X4
           ENDM
SQRT       ZR       X7,OK
           NG       X7,=XNEGSQRT
           SAVE
           BX1      X7
           SB1    1
           SB2    -48
           SA5        CA        .LOAD CA
           PX7        B2,X1     .W*2(-48)
           SA4        A5+B1     .LOAD CB
           FX0        X5*X7     .CA*W
           UX5        B3,X1     .PICK OFF N-48 AND COEFFICIENT OF W
           SX3        B3-B2     .N-48 -(-48) = N = 2*K + R
           FX2        X4+X0     .CA*W + CB = INITIAL GUESS
           AX6        X3,B1     .K = N/2
           IX5        X3-X6     .N-K
```

```
         FX4       X2*X2    .B*B                                           MATHLIB
         IX3       X5-X6    .N-K-K = R                                     MATHLIB
         SA5       X3+TWOR  LOAD 2.**(R/2)*2(-1)                           MATHLIB
         SB2       X6       .B2 = K                                        MATHLIB
         FX0       X4+X7    .B*B + W                                       MATHLIB
         FX3       X2*X0    .B*(B*B + W) = DENOM                           MATHLIB
         UX6       B3,X5    .                                              MATHLIB
         SB2       B2+B3    .INCORPORATE K INTO EXPONENT OF 2**(R/2)*2     MATHLIB
         PX6       B2,X6    . 2(K)*2(R/2)*2(-1)                            MATHLIB
         NX3      B0,X3                                                    MATHLIB
         FX2       X6/X3    .(2(K)*2(R/2)*2(-1) )/DENOM = TERM1            MATHLIB
         FX5       X0*X0    .(B*B + W)**2                                  MATHLIB
         FX7       X7+X7    .2*W                                           MATHLIB
         FX7       X7+X7    .4*W                                           MATHLIB
         FX3       X7*X4    .4*W*B*B                                       MATHLIB
         FX7       X3+X5    .NUM = (B*B+W)**2 + 4*W*B*B                    MATHLIB
         FX0       X7*X2    .TERM1*NUM = 2*U                               MATHLIB
         SA5       A4+B1    .LOAD .25                                      MATHLIB
         NX0      B0,X0                                                    MATHLIB
         FX2       X1/X0    .ARG/(2*U)                                     MATHLIB
         FX3       X5*X0    . .25*(2*U)                                    MATHLIB
         FX7       X2+X3    .SUM OF ABOVE = SQRT(ARG)                      MATHLIB
         RESTORE                                                           MATHLIB
OK       SA7      B3                                                       MATHLIB
         SA5      A5+1                                                     MATHLIB
         SB4      X5                                                       MATHLIB
         JP       B4                                                       MATHLIB
 CA      DATA       .585786437                                             MATHLIB
 CB      DATA       .4204951288                                            MATHLIB
 QUART   DATA     17164000000000000018                                     MATHLIB
         DATA     .353553390593                                            MATHLIB
 TWOR    DATA       .5                                                     MATHLIB
         DATA       .707106781186                                          MATHLIB
        END                                                                MATHLIB
```

BILL BRIDGE	3380 56	ZZZ
BILL BRIDGE	3380 56	ZZZ
BILL BRIDGE	3380 56	ZZZ
BILL BRIDGE	3380 56	ZZZ
BILL BRIDGE	3380 56	ZZZ

H

BILL BRIDGE	3380 56	ZZZ	01/
BILL BRIDGE	3380 56	ZZZ	01/
BILL BRIDGE	3380 56	ZZZ	01/
BILL BRIDGE	3380 56	ZZZ	01/
BILL BRIDGE	3380 56	ZZZ	01/