Purdue University
Purdue e-Pubs

Department of Computer Science Technical Reports

Department of Computer Science

1975

Corrections to Some Errors in a Version of TSP: (Time Series Processor)

Robert E. Lynch
Purdue University, rel@cs.purdue.edu

Report Number:
75-153

Lynch, Robert E., "Corrections to Some Errors in a Version of TSP: (Time Series Processor)" (1975). *Department of Computer Science Technical Reports.* Paper 100.
https://docs.lib.purdue.edu/cstech/100

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.

CORRECTIONS TO SOME ERRORS IN A VERSION OF TSP (TIME SERIES PROCESSOR)

Robert E. Lynch
Department of Computer Science
Purdue University
Lafayette, Indiana 47907

CSD TR 153

CORRECTIONS TO SOME SEVERE ERRORS IN A VERSION OF TSP (Time Series Processor)

Robert E. Lynch
Department of Computer Sciences
Purdue University

1. Introduction. TSP (Time Series Processor) is a problem-oriented computer system designed to carry out automatically the computations which occur routinely in econometric research. It has some of the same procedures as SPSS and it has some which SPSS does not have. The simplicity of its language makes TSP suitable for users who have only a casual acquaintance with the use of a computer, for example, the TSP statement PLOTS$ generates informative line printer graphical displays.

Several faculty members at Purdue wanted TSP available for their research activities as well as for student use in courses. A version of TSP, written in IBM360/370 Fortran, was obtained and converted to CDC6500 Fortran.

During the conversion, results from test cases showed that there were some severe errors in the IBM version. Reasonable looking output was produced which is incorrect. No TSP error message was printed. We do not see how a user could tell that these results were incorrect unless he had correct results to compare with or unless he made a detailed examination of the output together with some hand calculation.

The purpose of this report is to bring these errors, and their corrections, to the attention of installations which support TSP.

2. Errors in TSP. Errors occur in the IBM Fortran version of the TSP subroutines PASS1, LOAD (three errors), GRAPH, and CAPITL. These errors involve the way that the TSP processor handles blank common.

If the IBM Fortran compiler handles double precision assignment statements in a way similar to that of the CDC6500 Fortran compiler, then there is an error in the TSP inner produce routine, INPROD.

Section 4 contains listings of the CDC6500 version of PASS1, LOAD, GRAPH, CAPITL, and INPROD. The locations of the errors in the IBM version, and their corrections, are indicated.

Parts of a user's TSP program statements and intermediate results are stored in blank common; blank common is also used for scratch space. Errors include the omission of an appropriate test to see how much scratch space is available as well as incorrect tests.

The errors in the test cases mentioned in Section 1 occurred because the processor used scratch space which overlapped the other part of blank common.

If [1] is used as the TSP manual for this IBM Fortran version of TSP, then there is a misprint on page 3 of [1]. In the display of the size limitation, NOBS*VARS should be replaced with 301 + NOBS*(2*VARS + 1) + 2*VARS.

Remark: The TSP manual [1] does not mention the useful TSP feature which allows a comment to be inserted into the first statement, $$NAME program name$, of a TSP program. The processor stores the name immediately following NAME and everything else up to the terminator, $, is ignored. Consequently, instead of using something like $$NAME CASE1$, the user may give a more detailed program identification, for example:

```
$$NAME CASE1, GROSS NATIONAL PRODUCT MODELS OF I.M. JONES.  JUNE 19, 1984.
                MODEL 1 GNP = CONSTANT + A*EXP(TIME)
                MODEL 2 GNP = CONSTANT + A*EXP(TIME) + B*EXP(2*TIME) $
```

Any name in the identification, such as CASE1, GROSS, etc., may be used as a variable name in the program. The only limitation is a maximum of 8 characters

for a name; the use of SEPTEMBER would result in the generation of the TSP error message "name has too many characters."

3. Some differences between the IBM and CDC Fortran versions. In order to locate and correct the errors, detailed knowledge was required about the workings and functions of about half the 119 TSP subroutine which we had. As the purposes of various subroutines became clear, numerous comments were inserted. Consequently, the CDC6500 version is better documented than the IBM Fortran version, which is almost devoid of comments.

A number of inefficiencies were eliminated. Most of the linear searches of lists were replaced by more efficient search techniques.

In the CDC6500 version, a comment can be inserted at any place that a blank or comma separator is required or can occur. /* in columns 1 and 2 of a card denote that columns 3 to 80 contain a comment.

The user can use longer than 8 character names in order to help document his TSP program, for example

```
GENR NEWCASHBALANCE = OLDCASHBALANCE*( 1 + (INTERESTRATE/100)*(MONTHS/12) )$
```

However, the processor only uses the first 8 characters internally and it prints a record of the more-than-8-character names so the user can check for uniqueness.

The input routine, INPT, was completely rewritten. The new version is much more efficient than the older one. Free format data input takes only about 30% longer than formatted data input. The new version examines each non-blank, non-comma character about twice instead of the up to 20 times of the old version.

The user has more flexibility in the choice of storage requirements. He may choose blank common sizes of 3K, 5K, 10K, 15K, 20K, 25K. The 3K size is suitable for student jobs and the 25K size is capable of doing TSP programs which require the 30K IBM Fortran version. At the end of the execution output

of each TSP program, the amount of blank common which was used by the program is printed. With this information, the user can, perhaps, decrease the blank common size request on subsequent runs in order to decrease turnaround time and cost.

The most significant improvement in efficiency involves the number of words (NWORD) used to store a TSP keyword or user defined variable name (this improvement is directly transferable to the IBM Fortran version). The IBM version uses NWORD = 2 and two 8-byte words are used for keywords and names; floating point numerical values are stored in single 8-byte words. When a name is moved, two words must be moved and when a comparison or search of a list is done, pairs of comparisons are made (there is a lot of moving and comparisons of names during the execution of a TSP program). The CDC version uses NWORD = 1 and a keyword or name is stored in a single 60-bit word. This change to NWORD = 1 accomplishes the following:

1. Execution time for the movement and comparison of names is decreased by more than a factor of 2.
2. Numerous calls of short subroutines and functions are replaced by single assignment statements or by single IF statements.
3. Memory requirement is reduced (on the CDC6500).

Remark 1: If the change to NWORD = 1 were made in the IBM version, then the increased efficiencies of 1 and 2 are achieved by declaring names to be REAL DOUBLE PRECISION (or whatever the IBM Fortran declaration is). Accuracy is improved by using REAL DOUBLE PRECISION declarations for numerical values so that 16-bytes are used for floating point values; this improvement is made at the sacrifice of memory space. It is well-known that one must use as much precision as economically feasible in matrix computations--especially in least

squares computations. Moreover, in some cases, users might not know the importance of the choice of a basis for least squares computations and, perhaps, try to do a regression with a polynomial approximator of the form $a_0 + a_1x + \ldots + a_nx^n$; use of low precision numerical values might result in meaningless results. Thus, the change to NWORD = 1 should also be accompanied by REAL DOUBLE PRECISION declaration for just about all the variables in the TSP IBM Fortran program.

Remark 2: The CDC version uses NWORD = 1 and all changes from the IBM version are noted with comments. To change the IBM version to NWORD = 1, one can make all the appropriate changes indicated on the complete listing of the CDC version plus inserting the required REAL DOUBLE PRECISION declarations (there are a lot of these).

4. Listing of PASS1, LOAD, GRAPH, CAPITL, and INPROD. Listing of the CDC versions of the subroutines in which we found errors follows. On these listings, changes from the IBM Fortran version to CDC Fortran version are bracketed by a pair of comments: CIBMTOCDC CCDCTOIBM. Locations of errors in the IBM code and their corrections are bracketed with pairs of lines of dollar signs; notes about efficiencies are also bracketed. The corrections to the errors are CDC independent.

Since we have not seen machine language code generated by the IBM Fortran compiler, we are not sure if there is an error in the inner product routine. The original IBM program contains

```
      DOUBLE PRECISION XPROD
      J = 1
      XPROD = 0.
      DO 100 I = NN, JSA
      XPROD = XPROD + A(I)*B(J)
  100 J = J + JSB
```

The CDC compiler generates code which computes the single precision product A(I)*B(J), then makes it double precision with zero lower half, and finally adds this to the double precision XPROD.

One way to carry out this calculation correctly on the CDC6500 is to introduce two more double precision variables, TEMPA, TEMPB, and to replace the DO loop with

```
      DO 100 I = NN, JSA
      TEMPA = A(I)
      TEMPB = B(J)
      XPROD = XPROD + TEMPA*TEMPB
  100 J = J + JSB
```

Remark: Some of the DATA statement on the listings use the CDC character string specification R, for example DATA NAMES/ 8RNOPRINT , .../. This causes the BCD representation of the character string to be stored right justified with zero left fill; these can be used as variables declared INTEGER in arithmetic and logical expressions.

```
CIBMT0CDC                                                                PASS1     1
      SUBR0UTINE PASS1                                                   PASS1     2
CCDCT0IBM                                                                PASS1     3
C                                                                        PASS1     4
C     THIS PR0GRAM SUPERVISES READING AND PARSING THE INPUT              PASS1     5
CIBMT0CDC                                                                PASS1     6
C                                                                        PASS1     7
C     REPLACED  JX(400) (NW0RD = 1), ADDED LIMTYP                        PASS1     8
      C0MM0N                                                             PASS1     9
     *       TYPE(200), JX(200), C0DE(600), IX(200), ITYPE(200)          PASS1    10
      INTEGER TYPE                                                       PASS1    11
      DATA LIMTYP /200 /                                                 PASS1    12
C                                                                        PASS1    13
      DIMENSI0N X(1)                                                     PASS1    14
      EQUIVALENCE (X(1),JX(1))                                           PASS1    15
C                                                                        PASS1    16
C                                                                        TSPC0M    1
      C0MM0N /TSPC0M/                                                    TSPC0M    2
     * MEMSIZ, N0B   , NSPARG, NW0RD , LENGTH,                          TSPC0M    3
     * NTYPE , IFDBUG, IFTITL, NCHAR , NSUP  ,                          TSPC0M    4
     * MEMST , N0REG , IFPL0T, IFFAST, NPAGE ,                          TSPC0M    5
     * NUMLIN, IFREPL, PR0FF , SKIP(11), JPHAS ,                        TSPC0M    6
     * LIMARG, LINE  , NJARG , NARG  , NAME ,                           TSPC0M    7
     *  JARG(4)                                                         TSPC0M    8
C             LENGTH 0F JARG SET IN MAIN 0VERLAY                         TSPC0M    9
C                                                                        TSPC0M   10
C     DELETED  NAME2  BETWEEN  NAME  AND  JARG  IN  /TSPC0M/ (NW0RD = 1)TSPC0M   11
C                                                                        TSPC0M   12
      L0GICAL IFDBUG, IFTITL, IFPL0T, IFFAST,                            TSPC0M   13
     *        IFREPL, PR0FF                                              TSPC0M   14
C                                                                        TSPC0M   15
C                                                                        MEMC0M    1
C     NEW C0MM0N BL0CK ADDED                                             MEMC0M    2
      C0MM0N /MEMC0M/ IMNSZ(7), USRNAM, PSSVL(20)                        MEMC0M    3
      INTEGER USRNAM                                                     MEMC0M    4
      EQUIVALENCE                                                        MEMC0M    5
     * (IMNSZ(1),MMMSIZ), (IMNSZ(2),LLMARG), (IMNSZ(3),LLM0DT),          MEMC0M    6
     * (IMNSZ(4),LLMBUF), (IMNSZ(5),LLMSYM), (IMNSZ(6),LLMSMP),          MEMC0M    7
     * (IMNSZ(7),LLM0UT)                                                 MEMC0M    8
C                                                                        MEMC0M    9
C     IMMSZ USED T0 KEEP TRACK 0F MEM0RY USE                             MEMC0M   10
      DIMENSI0N IPSSVL(20)                                               MEMC0M   11
      EQUIVALENCE (IPSSVL(1),PSSVL(1))                                   MEMC0M   12
C     PSSVL AND IPSSVL USED T0 PASS VARIABLES BETWEEN SUBR0UTINES        MEMC0M   13
C     IN DIFFERENT 0VERLAYS                                              MEMC0M   14
C     IPSSVL(1) IS USED T0 KEEP REC0RD 0F USE 0F C0MM0N IN L0AD          MEMC0M   15
C        AND IN GENR                                                     MEMC0M   16
C     IPSSVL(3) PASSES C0MPUTED G0 T0 INDEX FR0M EXEC T0 MATRIX          MEMC0M   17
C     IPSSVL(2) PASSES C0MPUTED G0 T0 INDEX FR0M EXEC T0 0VERLAYS        MEMC0M   18
C     IPSSVL(4) IS USED T0 KEEP REC0RD 0F USE 0F JARG-C0MM0N             MEMC0M   19
C     IPSSVL(5) IS USED T0 KEEP REC0RD 0F USE 0F D0T-C0MM0N              MEMC0M   20
C     IPSSVL(6) IS USED T0 KELP REC0RD 0F USE 0F BUFFER-C0MM0N           MEMC0M   21
C     IPSSVL(7) IS USED T0 KEEP REC0RD 0F USE 0F JSML-C0MM0N             MEMC0M   22
C     IPSSVL(8) IS USED T0 KELP REC0RD OF USE 0F 0UTBUF-C0MM0N           MEMC0M   23
      EQUIVALENCE                                                        MEMC0M   24
     * (L0DUSE, IPSSVL(1)), (JT00VL, IPSSVL(2)), (JT0MAT, IPSSVL(3))    MEMC0M   25
     *,(JARUSE, IPSSVL(4)), (D0TUSE, IPSSVL(5)), (BUFUSE,IPSSVL(6)),    MEMC0M   26
     * (SMPUSE, IPSSVL(7)), (0BFUSE, IPSSVL(8))                         MEMC0M   27
      INTEGER D0TUSE, BUFUSE, SMPUSE, 0BFUSE                            MEMC0M   28
C                                                                        MEMC0M   29
C     CHANGED DIMENSI0N 0F  NAMES  T0 5 FR0M 20                          PASS1    19
C     CHANGED DIMENSI0N 0F NNAMES  T0 4 FR0M 20                          PASS1    20
      DIMENSI0N NAMED(2), JC0D(5), NNAMES(4), NAMES(5), NNAME(2)         PASS1    21
CCDCT0IBM                                                                PASS1    22
      L0GICAL LAGGED, IFRPAR                                             PASS1    23
CIBMT0CDC                                                                PASS1    24
```

```
C                                                                       PASS1   25
C     NEXT ARE 0PERAT0R C0DES F0R TSP FUNCTI0NS                         PASS1   26
C       11 EXP, 12 ABS, 13 L0G AND AL0G                                 PASS1   27
CCDCT0IBM                                                               PASS1   28
      DATA JC0DE /0,11,12,13,13/                                        PASS1   29
CIBMT0CDC                                                               PASS1   30
C                                                                       PASS1   31
      DATA G / 1RG /                                                    PASS1   32
C                                                                       PASS1   33
C                                                                       PASS1   34
      DATA NNAMES /                                                     PASS1   35
     * 8REXP     , 8RABS     , 8RAL0G    , 8RL0(      /                PASS1   36
C                                                                       PASS1   37
      DATA NAMES /                                                      PASS1   38
     * 8REND     , 8RGENR    , 8RNAME    , 8RBCD     , 8REBCDIC   /    PASS1   39
C / / / / / / / / / / / / / / / / / / / / / / BEGINNING 0F PASS1 / / / / PASS1   40
CCDCT0IBM                                                               PASS1   41
      LAGGED = .FALSE.                                                  PASS1   42
      IFRPAR = .FALSE.                                                  PASS1   43
C                                                                       PASS1   44
CIBMT0CDC                                                               PASS1   45
C                                                                       PASS1   46
C     REPLACED 37 WITH 10 T0 PRINT THE W0RD  LINE  0N TSP PR0GRAM 0UTPUTPASS1   47
    1 CALL 0UTEDT( 10, 1, LINE+1 )                                      PASS1   48
C                                                                       PASS1   49
C     READ NEXT TSP STATEMENT                                           PASS1   50
C                                                                       PASS1   51
CCDCT0IBM                                                               PASS1   52
      CALL INPT(JX,N,TYPE)                                              PASS1   53
CIBMT0CDC                                                               PASS1   54
C                                                                       PASS1   55
C     NAMES, NUMBERS, 0PERAT0RS 0N TSP STATEMENT ARE N0W ST0RED         PASS1   56
C       IN SUCCESSIVE W0RDS IN ARRAY  JX(I), I=1,N  IN THE 0RDER THAT   PASS1   57
C       THEY APPEAR IN THE TSP STATEMENT                                PASS1   58
C       STATEMENT KEYW0RD IS IN  JX(1)                                  PASS1   59
C       TYPE 0F ELEMENT IN JX(I) IS GIVEN BY VALUE 0F TYPE(I)           PASS1   60
C          1  NUMERICAL                                                 PASS1   61
C          2  0PERAT0R                                                  PASS1   62
C          3  NAME                                                      PASS1   63
C          4  D0TNAME                                                   PASS1   64
C                                                                       PASS1   65
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ PASS1   66
C     MAJ0R ERR0R IN IBM VERSION.  INPT CAN DESTR0Y DATA IN             PASS1   67
C     BLANK C0MM0N SUCH AS PR0GRAN STATEMENTS, DATA, VALUES             PASS1   68
C     GENERATED WITH GENR                                               PASS1   69
C     A TEST WAS ADDED AT RETURN 0F CALL 0F INPUT                       PASS1   70
      IF( N .LE. LIMTYP )                                               PASS1   71
C       THEN ARRAYS ARE LARGE EN0UGH F0R INPUT                          PASS1   72
     *                                                                  PASS1   73
C       ELSE T00 MUCH DATA                                  GO T0 10040PASS1   74
          IF( LIMTYP + N .LT. MEMSIZ )                                  PASS1   75
C           THEN HAVE N0T DESTR0YED ST0RED VALUES IN BLANK C0MM0N       PASS1   76
     *                                                                  PASS1   77
C           ELSE PART 0F BLANK C0MM0N 0VER-WRITTEN          C0 T0 10020PASS1   78
            IRUIN = LIMTYP + N - MEMSIZ                                 PASS1   79
            WRITE(6,10010) IRUIN                                        PASS1   80
10010       F0RMAT( 38H0*** ERR0R.  DURING READ 0F STATEMENTS  ,        PASS1   81
     *        16H DESTR0YED LAST , I10, 14H W0RDS 0F DATA  ,            PASS1   82
     *        28H IN BLANK C0MM0N.  AB0RT RUN  )                        PASS1   83
                                                                        PASS1   84
C                                                       CALL AB0RT      PASS1   85
10020       C0NTINUE                                                    PASS1   86
      WRITE( 6, 10030)                                                  PASS1   87
10030 F0RMAT( 31H0*** WARNING.  READ IN T00 MUCH  ,                     PASS1   88
     *   27H DATA FOR ARRAYS IN PASS1.  ,                               PASS1   89
     *   30H 0PERATION CONTINUES CORRECTLY                              PASS1   90
```

```
      *      30H IF THIS FOLLOWS AN *** ERROR   ,                                  PASS1    92
      *      32H MESSAGE.  OTHERWISE, SUBSEQUENT   ,                               PASS1    93
      *      25H RESULTS ARE LIKELY TO BE  ,  10H INCORRECT )                      PASS1    94
10040     CONTINUE                                                                 PASS1    95
C     END OF CORRECTION                                                            PASS1    96
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ PASS1    97
C                                                                                  PASS1    98
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ PASS1    99
C     SLIGHT IMPROVEMENT IN EFFICIENCY OF SEARCH                                   PASS1   100
C     REPLACED  JJ = IUCOMP(JX,NAMES)                                              PASS1   101
C               GO TO (10,100,200,300,400,410) JJ                                  PASS1   102
C     WITH THE FOLLOWING                                                           PASS1   103
C                                                                                  PASS1   104
C     THE FOLLOWING USES THE COLLATING SEQUENCE OF THE CDC6500                     PASS1   105
C     THE 8-BCD CHARACTER WORDS IN NAMES(I) SATISFY                                PASS1   106
C     CHAR. =  12345678     12345678     12345678     12345678     12345678       PASS1   107
C              BCD      .LT. EBCDIC  .LT. END     .LT. GENR    .LT. NAME         PASS1   108
C     I=        4           5            1            2            3              PASS1   109
C                                                                                  PASS1   110
C     DETERMINE WHETHER TSP KEYWORD IS ONE OF                                      PASS1   111
C        BCD, EBCDIC, END, GENR, NAME                                              PASS1   112
C                                                                                  PASS1   113
C     TEST FOR KEYWORD END                                                         PASS1   114
      IF( JX(1) - NAMES(1) )                    3,100,  4                        PASS1   115
C        TEST FOR KEYWORD EBCDIC                                                   PASS1   116
    3    IF( JX(1) - NAMES(5) )                 5,410, 10                        PASS1   117
C        TEST FOR KEYWORD GENR                                                     PASS1   118
    4    IF( JX(1) - NAMES(2) )                10,200,  6                        PASS1   119
C           TEST FOR KEYWORD BCD                                                   PASS1   120
    5       IF( JX(1) - NAMES(4) )             10,400, 10                        PASS1   121
C           TEST FOR KEYWORD NAME                                                  PASS1   122
    6       IF( JX(1) - NAMES(3) )             10,300, 10                        PASS1   123
C     END OF REPLACEMENT                                                           PASS1   124
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ PASS1   125
C                                                                                  PASS1   126
C     KEYWORD IS NOT ONE OF  BCD, EBCDIC, END, GENR, NAME                          PASS1   127
C                                                                                  PASS1   128
CCDCT0IBM                                                                          PASS1   129
   10 NARG = N - 1                                                                 PASS1   130
      IF (TYPE(1) .NE. 3) CALL ERG(1,JX)                                           PASS1   131
CIBMT0CDC                                                                          PASS1   132
C                                                                                  PASS1   133
C     MOVE KEYWORD TO NAME STORAGE                                                 PASS1   134
C                                                                                  PASS1   135
C     REPLACED CALL NMOV(NAME,JX(1)) WITH FOLLOWING (NWORD = 1)                    PASS1   136
      NAME = JX(1)                                                                 PASS1   137
   35 CONTINUE                                                                     PASS1   138
CCDCT0IBM                                                                          PASS1   139
      IF (NARG .EQ. 0) GO TO 50                                                    PASS1   140
CIBMT0CDC                                                                          PASS1   141
C                                                                                  PASS1   142
C     REPLACED  KK = NWORD + 1  WITH THE FOLLOWING  (NWORD = 1)                    PASS1   143
      KK = 2                                                                       PASS1   144
C                                                                                  PASS1   145
C     PUT ARGUMENTS OF TSP STATEMENT INTO TEMPORARY STORAGE IN JARG                PASS1   146
C                                                                                  PASS1   147
CCDCT0IBM                                                                          PASS1   148
      DO 40 I=2,N                                                                  PASS1   149
      CALL ARGPUT(I-1,JX(KK),TYPE(I),0)                                            PASS1   150
      KK = KK + 1                                                                  PASS1   151
CIBMT0CDC                                                                          PASS1   152
C                                                                                  PASS1   153
C     DELETED THE FOLLOWING   (NWORD = 1)                                          PASS1   154
C        IF(TYPE(I) .EQ. 3 .OR. TYPE(I) .EQ. 4) KK = KK + NWORD - 1                PASS1   155
CCDCT0IBM                                                                          PASS1   156
   40 CONTINUE                                                                     PASS1   157
```

```
CIBMT0CDC                                                                PASS1    159
C                                                                        PASS1    160
   50 C0NTINUE                                                           PASS1    161
C                                                                        PASS1    162
C     FR0M STATEMENT F0LL0WING 35 0R 225                                 PASS1    163
C                                                                        PASS1    164
      LINE = LINE + 1                                                    PASS1    165
C                                                                        PASS1    166
C     HAVE NAMES AND ARGUMENTS 0F C0MPLETE TSP STATEMENT ST0RED          PASS1    167
C     IN  NAME,JARG(,).   M0VE THEM T0 BLANK C0MM0N                      PASS1    168
C                                                                        PASS1    169
CCDCT0IBM                                                                PASS1    170
      CALL LINPUT                                                        PASS1    171
      G0 T0 1                                                            PASS1    172
C                                                                        PASS1    173
CIBMT0CDC                                                                PASS1    174
C     HAVE TSP KEYW0RD  *****END*****                                    PASS1    175
C     FINISHED READING TSP PR0GRAM.    ST0RE END STATEMENT AND           PASS1    176
C     RETURN T0 SUPER T0 EXECUTE TSP PR0GRAM                             PASS1    177
C                                                                        PASS1    178
C     REPLACED CALL NM0V(NAME,JX(1)) WITH F0LL0WING (NW0RD = 1)          PASS1    179
  100 NAME = JX(1)                                                       PASS1    180
CCDCT0IBM                                                                PASS1    181
      NARG=0                                                             PASS1    182
      LINE=LINE+1                                                        PASS1    183
      CALL LINPUT                                                        PASS1    184
      CALL 0UTPT                                                         PASS1    185
CIBMT0CDC                                                                PASS1    186
C                                                                        PASS1    187
C     NEXT IS ENTRY T0 SUPER                                             PASS1    188
CCDCT0IBM                                                                PASS1    189
      CALL RETURN                                                        PASS1    190
CIBMT0CDC                                                                PASS1    191
C     N0 RETURN T0 THIS P0INT FR0M ENTRY RETURN 0F SUPER                 PASS1    192
CCDCT0IBM                                                                PASS1    193
C                                                                        PASS1    194
CIBMT0CDC                                                                PASS1    195
C     HAVE TSP KEYW0RD  *****GENR*****                                   PASS1    196
C     M0VE KEYW0RD T0 NAME ST0RAGE                                       PASS1    197
C                                                                        PASS1    198
C     REPLACED CALL NM0V(NAME,JX(1)) WITH F0LL0WING (NW0RD = 1)          PASS1    199
  200 NAME = JX(1)                                                       PASS1    200
C                                                                        PASS1    201
C     C0NSTRUCT GENR-LINE-IDENTIFIER AND ST0RE IN JARG(1)                PASS1    202
CCDCT0IBM                                                                PASS1    203
      CALL INVNT(NNAME,G,LINE)                                           PASS1    204
      CALL ARGPUT(1,NNAME,6,0)                                           PASS1    205
CIBMT0CDC                                                                PASS1    206
C                                                                        PASS1    207
C     PUT LEFT SIDE 0F = INT0 TEMP0RARY ST0RAGE IN JARG                  PASS1    208
C                                                                        PASS1    209
C     REPLACED  JX(NW0RD + 1)  WITH  JX(2)   (NW0RD = 1)                 PASS1    210
      CALL ARGPUT( 2, JX(2), TYPE(2), 0 )                                PASS1    211
CCDCT0IBM                                                                PASS1    212
C                                                                        PASS1    213
C     JJ IS L0CATI0N IN IX.                                              PASS1    214
C     KK IS L0CATI0N IN JX.                                              PASS1    215
C     LL IS L0CATI0N IN JARG.                                            PASS1    216
C                                                                        PASS1    217
      IX(1) = 9                                                          PASS1    218
      ITYPE(1) = 2                                                       PASS1    219
      JJ = 2                                                             PASS1    220
      LL = 3                                                             PASS1    221
CIBMT0CDC                                                                PASS1    222
C                                                                        PASS1    223
C     REPLACED  KK = NW0RD*2 + 2  WITH F0LL0WING   (NW0RD = 1)           PASS1    224
C     THE USE 0F KK = 4 (0R 6 IF NW0RD=2) PICKS UP FIRST ITEM 0N RIGHT   PASS1    225
```

```
C         SIDE 0F = IN GENR STATEMENT
          KK = 4
CCDCT0IBM
C
          D0 220 I=4,N
          NTYPE = TYPE(I)
          IF(LAGGED)  G0 T0 530
CIBMT0CDC
C         TEST F0R NAME 0R NUMBER
CCDCT0IBM
          IF (NTYPE .NE. 2) G0 T0 205
CIBMT0CDC
C         TEST F0R LEFT PAREN
CCDCT0IBM
          IF(JX(KK) .EQ. 9)  G0 T0 500
C
C         0PERAT0R
C
  201 MMM = JX(KK)
CIBMT0CDC
C         TEST T0 SEE IF MINUS (C0DE 3) SH0ULD BE CHANGED T0 C0DE 8
C         (9 = LEFT PAREN, 2 = 0PERAT0R).  SWITCH MADE IF - IS PRECEDED
C         BY (   C0DE 3 MINUS IS INFIX, C0DE 8 MINUS IS PREFIX
CCDCT0IBM
          IF (MMM .EQ. 3 .AND. IX(JJ-1) .EQ. 9 .AND. ITYPE(JJ-1) .EQ. 2)
         I     MMM = 8
          IX(JJ) = MMM
          ITYPE(JJ) = 2
          JJ = JJ + 1
          KK = KK + 1
          G0 T0 220
CIBMT0CDC
C         HAVE LEFT PAREN.  SEE IF AN 0PERAT0R (C0DE 2) PRECEDES IT.
CCDCT0IBM
  500 IF (ITYPE(JJ-1) .EQ. 2) G0 T0 201
CIBMT0CDC
C         LEFT PAREN PRECEDED BY NAME, GET NAME AND SEE IF IT IS
C         TSP FUNCTI0N
CCDCT0IBM
          L00KUP = LL - 1
          CALL ARGGET(L00KUP,NAMED,JTYPE,LAG)
CIBMT0CDC
C
C         REPLACED  MATCH = IUC0MP(NAMED,NNAMES)  WITH THE F0LL0WING
C
C         DETERMINE IF NAMED(1) IS 0NE 0F TSP  FUNCTI0NS EXP,ABS,L0G,AL0G
          MATCH = 1
          IF( NAMED(1) .EQ. NNAMES(1) ) MATCH = 2
          IF( NAMED(1) .EQ. NNAMES(2) ) MATCH = 3
          IF( NAMED(1) .EQ. NNAMES(3) ) MATCH = 4
          IF( NAMED(1) .EQ. NNAMES(4) ) MATCH = 5
C         END 0F REPLACEMENT
CCDCT0IBM
          IF(MATCH .EQ. 1)  G0 T0 525
C
CIBMT0CDC
C         HAVE 0NE 0F THE FUNCTI0NS  ABS,EXP,AL0G,L0G
C         INSERT ITS C0DE AND SET TYPE T0 0PERAT0R (2)
CCDCT0IBM
C
  505 LL = L00KUP
          IX(JJ-1) = JC0DF(MATCH)
          ITYPE(JJ-1) = 2
          G0 T0 201
C
CIBMT0CDC
C         HAVE N0NE 0F THE FUNCTI0NS  ABS,EXP,AL0G,L0G
```

```
PASS1    226
PASS1    227
PASS1    228
PASS1    229
PASS1    230
PASS1    231
PASS1    232
PASS1    233
PASS1    234
PASS1    235
PASS1    236
PASS1    237
PASS1    238
PASS1    239
PASS1    240
PASS1    241
PASS1    242
PASS1    243
PASS1    244
PASS1    245
PASS1    246
PASS1    247
PASS1    248
PASS1    249
PASS1    250
PASS1    251
PASS1    252
PASS1    253
PASS1    254
PASS1    255
PASS1    256
PASS1    257
PASS1    258
PASS1    259
PASS1    260
PASS1    261
PASS1    262
PASS1    263
PASS1    264
PASS1    265
PASS1    266
PASS1    267
PASS1    268
PASS1    269
PASS1    270
PASS1    271
PASS1    272
PASS1    273
PASS1    274
PASS1    275
PASS1    276
PASS1    277
PASS1    278
PASS1    279
PASS1    280
PASS1    281
PASS1    282
PASS1    283
PASS1    284
PASS1    285
PASS1    286
PASS1    287
PASS1    288
PASS1    289
PASS1    290
PASS1    291
PASS1    292
```

```
      C
      C     LEFT PAREN PRECEDED BY N0N-TSP-FUNCTI0N NAME, HAVE LAGGED
      C     VARIABLE
      CCDCT0IBM
      C
        525 LAGGED = .TRUE.
            KK = KK + 1
            G0 T0 220
        530 IF(NTYPE .NE. 1)  G0 T0 540
      CIBMT0CDC
      C
      C     PR0CESSING SIGNED INTEGER ARGUMENT 0F LAGGED VARIABLE, ST0RED
      C     AS REAL TYPE, C0NVERT IT T0 INTEGER TYPE
      CCDCT0IBM
            MMM = X(KK)
            CALL ARGPUT(L00KUP,NAMED,JTYPE,MMM)
            IFRPAR = .TRUE.
            KK = KK +1
            G0 T0 220
      CIBMT0CDC
      C
      C     TURN 0F LAGGED-SWITCH WHEN GET T0 RIGHT PAREN
      CCDCT0IBM
       540 IF(.N0T. IFRPAR)   CALL ERG(69,JX(KK))
            IF(NTYPE .NE. 2 .0R. JX(KK) .NE. 1)   CALL ERG(69,JX(KK))
            IFRPAR = .FALSE.
            LAGGED = .FALSE.
            KK = KK + 1
            G0 T0 220
      CIBMT0CDC
      C
      C     PUT NAME 0R NUMBER INT0 JARG(.)
      CCDCT0IBM
        205 CALL ARGPUT(LL,JX(KK),NTYPE,0)
            IF(NTYPE.NE.1.0R.(ITYPE(JJ-1).NE.-1.AND.(ITYPE(JJ-1).NE.2.0R.IX(JJ
           1-1).NE.1))) G0 T0 210
            IX(JJ) = 2
            ITYPE(JJ) = 2
            JJ = JJ + 1
        210 IX(JJ) = LL - 2
            ITYPE(JJ) = -1
            JJ = JJ + 1
            KK = KK + 1
      CIBMT0CDC
      C
      C     DELETED THE F0LL0WING   (NW0RD = 1)
      C        IF(NTYPE .EQ. 3 .0R. NTYPE .EQ. 4) KK = KK + NW0RD - 1
      CCDCT0IBM
            LL = LL + 1
        220 C0NTINUE
      CIBMT0CDC
      C
      C     HAVE FINISHED ST0RING C0DED GENR STATEMENT, C0NVERT IT T0
      C     STRING WHICH EVAL CAN PR0CESS
      CCDCT0IBM
      C
            IX(JJ) = 1
            ITYPE(JJ) = 2
            JJ = JJ + 1
            IX(JJ) = 14
            ITYPE(JJ) = 2
            CALL C0MPLR(IX,ITYPE,C0DE,LENGTH,C0DE(501),C0DE(551))
      CIBMT0CDC
      C
      C     ST0RE STRING IN BLANK C0MM0N
      CCDCT0IBM
            CALL VPUT(NNAME,C0DE,LENGTH)
```

```
PASS1   293
PASS1   294
PASS1   295
PASS1   296
PASS1   297
PASS1   298
PASS1   299
PASS1   300
PASS1   301
PASS1   302
PASS1   303
PASS1   304
PASS1   305
PASS1   306
PASS1   307
PASS1   308
PASS1   309
PASS1   310
PASS1   311
PASS1   312
PASS1   313
PASS1   314
PASS1   315
PASS1   316
PASS1   317
PASS1   318
PASS1   319
PASS1   320
PASS1   321
PASS1   322
PASS1   323
PASS1   324
PASS1   325
PASS1   326
PASS1   327
PASS1   328
PASS1   329
PASS1   330
PASS1   331
PASS1   332
PASS1   333
PASS1   334
PASS1   335
PASS1   336
PASS1   337
PASS1   338
PASS1   339
PASS1   340
PASS1   341
PASS1   342
PASS1   343
PASS1   344
PASS1   345
PASS1   346
PASS1   347
PASS1   348
PASS1   349
PASS1   350
PASS1   351
PASS1   352
PASS1   353
PASS1   354
PASS1   355
PASS1   356
PASS1   357
PASS1   358
```

```
      NARG = LL - 1                                                    PASS1     360
CIBMT0CDC                                                              PASS1     361
  225 C0NTINUE                                                         PASS1     362
CCDCT0IBM                                                              PASS1     363
      G0 T0 50                                                         PASS1     364
C                                                                      PASS1     365
CIBMT0CDC                                                              PASS1     366
  300 C0NTINUE                                                         PASS1     367
C     HAVE TSP KEYW0RD  ******NAME******                               PASS1     368
C     ST0RE THE  CASE  PART 0F TSP STATEMENT $$NAME,CASE$              PASS1     369
C     IN USRNAM  F0R USE IN END-0F-EXECUTI0N 0UTPUT                    PASS1     370
CCDCT0IBM                                                              PASS1     371
C                                                                      PASS1     372
CIBMT0CDC                                                              PASS1     373
C     DELETED CALL NM0V(JARG(1),X)   (NW0RD+1)                         PASS1     374
C                                                                      PASS1     375
C     REPLACED CALL 0F 0NE LINE SUBR0UTINE                             PASS1     376
C     UNAME    CALL NM0V(...,JARG)    RETURN    END                   PASS1     377
C     WITH THE F0LL0WING   (NW0RD = 1)                                 PASS1     378
      USRNAM = JX(2)                                                   PASS1     379
CCDCT0IBM                                                              PASS1     380
      G0 T0 1                                                          PASS1     381
C                                                                      PASS1     382
C     SET UP CHARACTER C0DE                                            PASS1     383
CIBMT0CDC                                                              PASS1     384
C                                                                      PASS1     385
  400 C0NTINUE                                                         PASS1     386
C     F0LL0WING                                                        PASS1     387
C 400 CALL BCD                                                         PASS1     388
C     G0 T0 1                                                          PASS1     389
C 410 CALL EBCDIC                                                      PASS1     390
C     G0 T0 1                                                          PASS1     391
C     REPLACED WITH                                                    PASS1     392
C                                                                      PASS1     393
C     HAVE TSP KEYW0RD  *****BCD*****                                  PASS1     394
C                                                                      PASS1     395
      JUMP = 1                                                         PASS1     396
                                                  G0 T0 415            PASS1     397
C                                                                      PASS1     398
  410 C0NTINUE                                                         PASS1     399
C     HAVE TSP KEYW0RD  *****EBCDIC*****                               PASS1     400
C                                                                      PASS1     401
      JUMP = 2                                                         PASS1     402
  415 C0NTINUE                                                         PASS1     403
      WRITE(6,420)                                                     PASS1     404
  420 F0RMAT(44H0 TSP STATEMENTS  BCD  AND  EBCDIC  DISABLED , /       PASS1     405
     * 53H  CDC6500 READS BCD.   IF S0URCE TSP DECK IS IN EBCDIC,      PASS1     406
     * 47H IT SH0ULD BE PREPR0CESSED T0 C0NVERT IT T0 BCD     )         PASS1     407
                                                  G0 T0 (1,430), JUMP  PASS1     408
  430 C0NTINUE                                                         PASS1     409
      ST0P                                                             PASS1     410
CCDCT0IBM                                                              PASS1     411
      END                                                              PASS1     412
05.23.17.J0 16 EP30       8 FEET
```

```
CIBMT0CDC                                                                       L0AD        1
      SUBR0UTINE L0AD                                                           L0AD        2
CCDCT0IBM                                                                       L0AD        3
C                                                                               L0AD        4
C      THIS IS A STRICTLY TEMP0RARY VERSI0N BASED 0N THE 0LD 1620/7094          LL0AD       5
C                                                                               L0AD        6
CIBMT0CDC                                                                       L0AD        7
C                                                                               TSPC0M      1
      C0MM0N /TSPC0M/                                                           TSPC0M      2
     * MEMSIZ, N0B   , NSPARG, NW0RD , LENGTH,                                  TSPC0M      3
     * NTYPE , IFDBUG, IFTITL, NCHAR , NSUP  ,                                  TSPC0M      4
     * MEMST , N0REG , IFPL0T, IFFAST, NPAGE ,                                  TSPC0M      5
     * NUMLIN, IFREPL, PR0FF , SKIP(11), JPHAS ,                                TSPC0M      6
     * LIMARG, LINE  , NJARG , NARG  , NAME ,                                   TSPC0M      7
     *  JARG(4)                                                                 TSPC0M      8
C             LENGTH 0F JARG SET IN MAIN 0VERLAY                                TSPC0M      9
C                                                                               TSPC0M      10
C     DELETED  NAME2  BETWEEN  NAME  AND  JARG  IN  /TSPC0M/ (NW0RD = 1)        TSPC0M      11
C                                                                               TSPC0M      12
      L0GICAL IFDBUG, IFTITL, IFPL0T, IFFAST,                                   TSPC0M      13
     *        IFREPL, PR0FF                                                     TSPC0M      14
C                                                                               TSPC0M      15
C                                                                               MEMC0M      1
C     NEW C0MM0N BL0CK ADDED                                                    MEMC0M      2
      C0MM0N /MEMC0M/ IMNSZ(7), USRNAM, PSSVL(20)                               MEMC0M      3
      INTEGER USRNAM                                                            MEMC0M      4
      EQUIVALENCE                                                               MEMC0M      5
     * (IMNSZ(1),MMMSIZ), (IMNSZ(2),LLMARG), (IMNSZ(3),LLMD0T),                 MEMC0M      6
     * (IMNSZ(4),LLMBUF), (IMNSZ(5),LLMSYM), (IMNSZ(6),LLMSMP),                 MEMC0M      7
     * (IMNSZ(7),LLM0UT)                                                        MEMC0M      8
C                                                                               MEMC0M      9
C     IMMSZ USED T0 KEEP TRACK 0F MEM0RY USE                                    MEMC0M      10
      DIMENSI0N IPSSVL(20)                                                      MEMC0M      11
      EQUIVALENCE (IPSSVL(1),PSSVL(1))                                          MEMC0M      12
C     PSSVL AND IPSSVL USED T0 PASS VARIABLES BETWEEN SUBR0UTINES               MEMC0M      13
C     IN DIFFERENT 0VERLAYS                                                     MEMC0M      14
C     IPSSVL(1) IS USED T0 KEEP REC0RD 0F USE 0F C0MM0N IN L0AD                 MEMC0M      15
C        AND IN GENR                                                            MEMC0M      16
C     IPSSVL(3) PASSES C0MPUTED G0 T0 INDEX FR0M EXEC T0 MATRIX                 MEMC0M      17
C     IPSSVL(2) PASSES C0MPUTED G0 T0 INDEX FR0M EXEC T0 0VERLAYS               MEMC0M      18
C     IPSSVL(4) IS USED T0 KEEP REC0RD 0F USE 0F JARG-C0MM0N                    MEMC0M      19
C     IPSSVL(5) IS USED T0 KEEP REC0RD 0F USE 0F D0T-C0MM0N                     MEMC0M      20
C     IPSSVL(6) IS USED T0 KEEP REC0RD 0F USE 0F BUFFER-C0MM0N                  MEMC0M      21
C     IPSSVL(7) IS USED T0 KEEP REC0RD 0F USE 0F JSML-C0NM0N                    MEMC0M      22
C     IPSSVL(8) IS USED T0 KEEP REC0RD 0F USE 0F 0UTBUF-C0MMN0N                 MEMC0M      23
      EQUIVALENCE                                                               MEMC0M      24
     * (L0DUSE, IPSSVL(1)), (JT00VL, IPSSVL(2)), (JT0MAT, IPSSVL(3))           MEMC0M      25
     *,(JARUSE, IPSSVL(4)), (D0TUSE, IPSSVL(5)), (BUFUSE,IPSSVL(6)),           MEMC0M      26
     * (SMPUSE, IPSSVL(7)), (0BFUSE, IPSSVL(8))                                MEMC0M      27
      INTEGER D0TUSE, BUFUSE, SMPUSE, 0BFUSE                                   MEMC0M      28
C                                                                               MEMC0M      29
C                                                                               L0AD        10
C     REM0VED /SMPC0M/                                                          L0AD        11
C                                                                               L0AD        12
C     ADDED  JX(1)  EQUIVALENCED T0  X(1)                                       L0AD        13
      DIMENSION JX(1)                                                           L0AD        14
      EQUIVALENCE ( JX(1), X(1) )                                               L0AD        15
C                                                                               L0AD        16
CCDCT0IBM                                                                       L0AD        17
      INTEGER TYPE                                                              L0AD        18
CIBMT0CDC                                                                       L0AD        19
C                                                                               L0AD        20
      C0MM0N                                                                    L0AD        21
     * TYPE(100), MASK(200), X(1)                                              L0AD        22
C                                                                               L0AD        23
      INTEGER FRONT, RUNNER
```

```
      DATA    FRONT / 300 /, LIMTYP / 100 /                                  LOAD      25
C             FRONT IS LENGTH OF ARRAYS TYPE PLUS MASK                       LOAD      26
C                                                                            LOAD      27
C     DELETED LOGICAL IFCOMP                                                 LOAD      28
C                                                                            LOAD      29
C     CHANGE DIMENSION OF NAMES TO6 FROM 20                                  LOAD      30
      DIMENSION NAMES(6)                                                     LOAD      31
C                                                                            LOAD      32
      DATA NAMES /                                                           LOAD      33
     * 8RLOAD    , 8REND     , 8RMASK    , 8RSMPL    , 8RNOPRINT ,          LOAD      34
     * 8RFORMAT   /                                                         LOAD      35
C                                                                            LOAD      36
C / / / / / / / / / / / / / / / / / / / / / / / / BEGINNING OF LOAD / / / / LOAD      37
CCDCTOIBM                                                                    LOAD      38
      PROFF=.FALSE.                                                          LOAD      39
      CALL OUTPT                                                             LOAD      40
      NMASK = 2                                                              LOAD      41
      MASK(1) = 1                                                            LOAD      42
      MASK(2) = 80                                                           LOAD      43
C                                                                            LOAD      44
CIBMTOCDC                                                                    LOAD      45
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ LOAD      46
C     MAJOR ERROR IN IBM VERSION.  INPT CAN DESTROY DATA IN BLANK            LOAD      47
C     COMMON.                                                                LOAD      48
C     INSERTED TEST AT RETURN FROM INPT.                                     LOAD      49
C     REPLACED  1 CALL XNPT(X,N,TYPE)                                        LOAD      50
C                 IF(TYPE(1).EQ.1)  GO TO 1                                LOAD      51
C     WITH THE FOLLOWING                                                     LOAD      52
    1 CONTINUE                                                               LOAD      53
C                                                                            LOAD      54
C     SEARCH FOR TSP STATEMENT KEYWORD OR OPERATOR                           LOAD      55
      NMLEFT = MEMSIZ - FRONT                                                LOAD      56
    2 CONTINUE                                                               LOAD      57
      CALL INPT( X, N, TYPE )                                                LOAD      58
      IF( N .LT. NMLEFT )                                                    LOAD      59
C       THEN HAVE ENOUGH ROOM                                                LOAD      60
     *                                                                       LOAD      61
C       ELSE HAVE STORED OVER DATA IN BLANK COMMON          GO TO 10004      LOAD      62
        RUINED = N - NMLEFT                                                  LOAD      63
        WRITE( 6, 10003 ) RUINED                                             LOAD      64
10003   FORMAT( 50HO*** ERROR.  DURING LOAD SECTION, DESTROYED LAST ,        LOAD      65
     *    I5, 30H WORDS OF DATA IN BLANK COMMON  )                          LOAD      66
                                                                             LOAD      67
C                                                        CALL ABORT          LOAD      68
10004   CONTINUE                                                             LOAD      69
      IF( TYPE(1) .EQ. 1 )                                                   LOAD      70
C       THEN CONTINUE READING INPUT                                          LOAD      71
     *                                                                       LOAD      72
C       ELSE HAVE FOUND A NON-NUMERIC INPUT VALUE          GO TO 2           LOAD      73
C                                                                            LOAD      74
C     END OF CORRECTION                                                      LOAD      75
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ LOAD      76
C                                                                            LOAD      77
C                                                                            LOAD      78
C                                                                            LOAD      79
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ LOAD      80
C     INSERTED SLIGHTLY MORE EFFICIENT SEARCH                                LOAD      81
C     REPLACED 2 JJ = IUCOMP(X,NAMES)                                        LOAD      82
C                 GO TO (10,100,200,300,400,500,10),JJ                       LOAD      83
C     WITH THE FOLLOWING                                                     LOAD      84
C                                                                            LOAD      85
C     THE FOLLOWING USES THE COLLATING SEQUENCE OF THE CDC6500               LOAD      86
C     THE 8-BCD CHARACTER WORDS IN NAMES(I) SATISFY THE FOLLOWING            LOAD      87
C     CHAR. = 12345678    12345678    12345678    12345678    12345678       LOAD      88
C             END     .LT.LOAD    .LT.MASK    .LT.NOPRINT .LT.SMPL           LOAD      89
C     I     = 2                                                              LOAD      90
```

```
C
C     TEST FOR MASK                                                    LOAD       92
      IF( JX(1) - NAMES(3) )                    3,300, 4               LOAD       93
C        TEST FOR LOAD                                                 LOAD       94
     3    IF( JX(1) - NAMES(1) )                5,100,10               LOAD       95
C        TEST FOR NOPRINT                                              LOAD       96
     4    IF( JX(1) - NAMES(5) )               10,500, 6               LOAD       97
C           TEST FOR END                                               LOAD       98
     5       IF( JX(1) - NAMES(2) )            10,200,10               LOAD       99
C           TEST FOR SMPL                                              LOAD      100
     6       IF( JX(1) - NAMES(4) )            10,400,10               LOAD      101
C     END OF REPLACEMENT                                               LOAD      102
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ LOAD      103
C                                                                      LOAD      104
C     NONE OF THE KEYWORDS WERE FOUND                                  LOAD      105
CCDCT0IBM                                                              LOAD      106
   10 CALL ERG(11,X)                                                   LOAD      107
CIBMT0CDC                                                              LOAD      108
C     NO RETURN TO THIS POINT FROM ERG                                 LOAD      109
CCDCT0IBM                                                              LOAD      110
C                                                                      LOAD      111
CIBMT0CDC                                                              LOAD      112
C     HAVE ISP KEYWORD. ******LOAD******                               LOAD      113
C     HAVE JUST READ INPUT, CHECK TO SEE IF ALL ARE VARIABLE NAMES     LOAD      114
CCDCT0IBM                                                              LOAD      115
C                                                                      LOAD      116
  100 DO 110 I=2,N                                                     LOAD      117
  110 IF(TYPE(I) .NE. 3)  CALL ERG(12,X)                               LOAD      118
      NOVAR = N - 1                                                    LOAD      119
CIBMT0CDC                                                              LOAD      120
C     NOVAR IS THE NUMBER OF VARIABLE NAMES JUST READ                  LOAD      121
C                                                                      LOAD      122
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ LOAD      123
C     MAJOR ERROR IN IBM VERSION.  INPT CAN READ IN MORE VARIABLE NAMES LOAD      124
C     THAN ALLOWED FOR BY THE SIZE OF ARRAY TYPE.                      LOAD      125
      IF( NOVAR .LT. LIMTYP )                                          LOAD      126
C        THEN THE ARRAYS ARE LARGE ENOUGH                              LOAD      127
     *                                                                 LOAD      128
     *                                                  GO TO 116     LOAD      129
C        ELSE TOO MANY VARIABLE NAMES                                  LOAD      130
           WRITE( 6, 115 ) NOVAR, LIMTYP                               LOAD      131
  115      FORMAT( 20HO*** ERROR.  LOADED  , I5,                       LOAD      132
     *       29H VARIABLE NAMES.  MAXIMUM IS  ,  I5,                  LOAD      133
     *       17H PER CALL OF LOAD   )                                 LOAD      134
           CALL ERG( 20, NOVAR )                                       LOAD      135
C          NO RETURN TO THIS POINTFROM ERG                             LOAD      136
C                                                                      LOAD      137
 116       CONTINUE                                                    LOAD      138
C                                                                      LOAD      139
C     END OF CORRECTION                                                LOAD      140
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ LOAD      141
C                                                                      LOAD      142
C                                                                      LOAD      143
C     REPLACED  NLBUF=NWORD*N+1  WITH  (NWORD = 1)                     LOAD      144
      NLBUF = N + 1                                                    LOAD      145
CCDCT0IBM                                                              LOAD      146
      NLDAT = NLBUF + NOB                                              LOAD      147
CIBMT0CDC                                                              LOAD      148
C                                                                      LOAD      149
C     MOVED NEXT STATEMENT UP A FEW LINES TO CUT OUT SOME MULTIPLICATIONLOAD      150
      NDATA = NOB*NOVAR                                                LOAD      151
C                                                                      LOAD      152
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ LOAD      153
C     MAJOR ERROR IN IBM VERSION                                       LOAD      154
C     NTEST USED TO TEST THE SIZE OF AVAILABLE BLANK COMMON            LOAD      155
C     SPACE CONTAINS ERROR.                                            LOAD      156
C                                                                      LOAD      157
C     REPLACED  NTEST = NLDAT + NOB*NOVAR  WITH                        LOAD      158
```

```
      NTEST = FR0NT + N + 1 + N0B + 2*NDATA                                  L0AD      159
C       BLANK C0MM0N USE                                                     L0AD      160
C         ARRAYS TYPE, MASK    = FR0NT                                       L0AD      161
C         INPUT IN X           = N                                           L0AD      162
C         TEMP0RARY            = N0B                                         L0AD      163
C         WILL READ            = N0B*N0VAR + 1  (THE PLUS 1 IS F0R THE       L0AD      164
C                                                TYPE 0F LAST DATUM READ)    L0AD      165
C         WILL ST0RE           = N0B*N0VAR                                   L0AD      166
C       T0TAL IS ASSIGNED T0 NTEST AB0VE                                     L0AD      167
C                                                                            L0AD      168
      IF( NTEST .LT. MEMSIZ )                                                L0AD      169
C       THEN HAVE EN0UGH R00M                                                L0AD      170
     *                                                          G0 T0 119     L0AD      171
C       ELSE WILL DESTR0Y DATA IN BLANK C0MM0N                               L0AD      172
          INEED = NTEST - NDATA                                              L0AD      173
          WRITE(6,118) N0B,N0VAR,NDATA,NTEST,MEMSIZ,INEED,NTEST              L0AD      174
  118     F0RMAT( 47H0*** ERR0R.  TRYING T0 L0AD (N0. 0BSERVATI0NS)*  ,      L0AD      175
     *      18H(N0. VARIABLES) =  ,I6,1H*,I6,3H = ,I10,7H VALUES/            L0AD      176
     *      13X, 13HTHIS REQUIRES, I10, 23H W0RDS IN BLANK C0MM0N,  ,        L0AD      177
     *      10H 0NLY HAVE,I10, 11H AVAILABLE.  / 13X, 8HINCREASE  ,          L0AD      178
     *      53H BLANK C0MM0N SIZE 0R REWRITE SUBR0UTINE L0AD T0 L0AD ,       L0AD      179
     *      12H VALUES WITH, I10, 17H W0RDS INSTEAD 0F , I10 /               L0AD      180
     *      10H0AB0RT RUN  )                                                 L0AD      181
                                                          CALL AB0RT         L0AD      182
C         (N0TE THAT A CALL 0F ERG WHICH CALLS SUPER WHICH CALLS             L0AD      183
C          INPT IN IBM VERSI0N W0ULD DESTR0Y THE DATA)                       L0AD      184
C                                                                            L0AD      185
  119     C0NTINUE                                                           L0AD      186
C                                                                            L0AD      187
C     END 0F C0RRECTI0N                                                      L0AD      188
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ L0AD      189
C                                                                            L0AD      190
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ L0AD      191
C     N0TE 0N EFFICIENT USE 0F ST0RAGE.                                      L0AD      192
C     THE IBM VERSI0N AND THIS VERSI0N USE  NTEST  W0RDS IN BLANK            L0AD      193
C     C0MM0N T0 READ  N0B*N0VAR  DATA.  THIS AM0UNT CAN BE REDUCED           L0AD      194
C     BY  N0B*N0VAR  IF THE L0ADING IS D0NE IN A D0-L00P.                    L0AD      195
C     EACH ITERATI0N READS 0NLY  N0VAR  DATA, THE VALUES 0F THE              L0AD      196
C     NEXT 0BSERVATI0N F0R EACH VARIABLE.  THIS REQUIRES THE                 L0AD      197
C     C0NSTRUCTI0N 0F A NEW ENTRY T0  INPT  T0 READ A SPECIFIED              L0AD      198
C     NUMBER (N0VAR) 0F VALUES.  SUBR0UTINE  F0RMAT  HAS T0 BE               L0AD      199
C     CHANGED S0 THAT IT READS 0NLY  N0VAR  VALUES.                          L0AD      200
C                                                                            L0AD      201
C     END 0F N0TE                                                            L0AD      202
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ L0AD      203
C                                                                            L0AD      204
C     REC0RD THE USE 0F BLANK C0MM0N BY L0AD                                 L0AD      205
      L0DUSE = MAX0( L0DUSE, NTEST - N0B*N0VAR )                             L0AD      206
CCDCT0IBM                                                                    L0AD      207
      NDT = 0                                                                L0AD      208
      NP0INT = NLDAT                                                         L0AD      209
CIBMT0CDC                                                                    L0AD      210
C                                                                            L0AD      211
C     REPLACED  NW0RD+1, NW0RD+2  WITH  2,3   (NW0RDD = 1)                   L0AD      212
      JARG( 2 ) = 3                                                          L0AD      213
      JARG( 3 ) = 0                                                          L0AD      214
C                                                                            L0AD      215
  120 CALL XNPT( X(NP0INT), N, X(NP0INT+1) )                                 L0AD      216
C           XNPT IS AN ENTRY T0 NEW VERSI0N 0F INPT                          L0AD      217
C                                                                            L0AD      218
C     REPLACED                                                               L0AD      219
C     IF(IFC0MP(X(NP0INT),NAMES(13))) CALL F0RMAT(X(NP0INT),N,N0VAR)         L0AD      220
C                                                                            L0AD      221
C     TEST F0R KEYW0RD ******F0RMAT******                                    L0AD      222
      IF( JX(NP0INT) .EQ. NAMES(6) )                                         L0AD      223
     *      CALL F0RMAT( X(NP0INT), N, N0VAR )                               L0AD      224
CCDCT0IBM                                                                    L0AD      225
```

```
      IF (N0VAR .GT. 1) G0 T0 121                                       L0AD    226
      NLBUF = NLDAT                                                     L0AD    227
CIBMT0CDC                                                               L0AD    228
C                                                                       L0AD    229
C     REPLACED  NW0RD + 1  WITH  2   (NW0RD = 1)                        L0AD    230
      L = 2                                                             L0AD    231
CCDCT0IBM                                                               L0AD    232
      G0 T0 140                                                         L0AD    233
CIBMT0CDC                                                               L0AD    234
  121 C0NTINUE                                                          L0AD    235
C                                                                       L0AD    236
C     REM0VED F0LL0WING FR0M INSIDE D0-130-L00P                         L0AD    237
      K = NLBUF + N0B - 1                                               L0AD    238
C                                                                       L0AD    239
      D0 130 I = 1, N0VAR                                               L0AD    240
C                                                                       L0AD    241
C     REPLACED  I*NW0RD + 1  WITH  I + 1   (NW0RD = 1)                  L0RD    242
      L = I + 1                                                         L0AD    243
CCDCT0IBM                                                               L0AD    244
      NP0INT = NLDAT + I - 1                                            L0AD    245
      D0 125 J=NLBUF,K                                                  L0AD    246
      X(J) = X(NP0INT)                                                  L0AD    247
  125 NP0INT = NP0INT + N0VAR                                           L0AD    248
CIBMT0CDC                                                               L0AD    249
C                                                                       L0AD    250
C     REPLACED  CALL NM0V(JARG(1),X(L))  (NW0RD = 1)                    L0AD    251
      JARG(1) = JX(L)                                                   L0AD    252
CCDCT0IBM                                                               L0AD    253
  130 CALL TSPUT(JARG(1),X(NLBUF))                                      L0AD    254
  135 IF(N.NE.NDATA) CALL ERG(10,N)                                     L0AD    255
      G0 T0 1                                                           L0AD    256
CIBMT0CDC                                                               L0AD    257
C     REPLACED  CALL NM0V(JARG(1),X(L))  (NW0RD = 1)                    L0AD    258
  140 JARG(1) = JX(L)                                                   L0AD    259
CCDCT0IBM                                                               L0AD    260
      CALL TSPUT(JARG(1),X(NLBUF))                                      L0AD    261
      G0 T0 135                                                         L0AD    262
C                                                                       L0AD    263
CIBMT0CDC                                                               L0AD    264
C     HAVE TSP KEYW0RD  ******END******                                 L0AD    265
CCDCT0IBM                                                               L0AD    266
C                                                                       L0AD    267
  200 PR0FF=.FALSE.                                                     L0AD    268
CIBM!0CDC                                                               L0AD    269
C                                                                       L0AD    270
C     NEXT IS ENTRY T0 SUPER                                            L0AD    271
CCDCT0IBM                                                               L0AD    272
      CALL RETURN                                                       L0AD    273
CIBMT0CDC                                                               L0AD    274
C     N0 RETURN T0 THIS P0INT FR0M ENTRY RETURN 0F SUPER                L0AD    275
CCDCT0IBM                                                               L0AD    276
C                                                                       L0AD    277
C     SECTI0N F0R MASK                                                  L0AD    278
C                                                                       L0AD    279
  300 D0 310 I=1,N                                                      L0AD    280
  310 MASK(I) = X(I)                                                    L0AD    281
      G0 T0 1                                                           L0AD    282
C                                                                       L0AD    283
CIBMT0CDC                                                               L0AD    284
C     HAVE TSP KEYW0RD  ******SMPL******                               L0AD    285
CCDCT0IBM                                                               L0AD    286
C                                                                       L0AD    287
  400 NARG = N - 1                                                      L0AD    288
CIBMT0CDC                                                               L0AD    289
C                                                                       L0AD    290
C     REPLACE  NW0RD + 1  WITH  2                                       L0AD    291
      KK = 2                                                            L0AD    292
```

```
      CCDCT0IBM
            D0 520 I=2,N
            CALL ARGPUT(I-1,X(KK),TYPE(I),0)
        520 KK = KK + 1
            CALL SMPL
            G0 T0 1
      C
      CIBMT0CDC
      C     HAVE TSP KEYW0RD  ******N0PRINT******
      CCDCT0IBM
      C     TURN 0FF PRINTING 0F DATA AS L0ADED
      C
        500 PR0FF=.TRUE.
            G0 T0 1
            END
05.23.33.J0 16 EP30          6 FEET
```

```
LOAD      293
LOAD      294
LOAD      295
LOAD      296
LOAD      297
LOAD      298
LOAD      299
LOAD      300
LOAD      301
LOAD      302
LOAD      303
LOAD      304
LOAD      305
LOAD      306
LOAD      307
```

```
      SUBROUTINE GRAPH                                                  GRAPH      1
C                                                                       GRAPH      2
C          THIS IS AN ADAPTATION OF **PLOT-C**                          GRAPH      3
C                                                                       GRAPH      4
CIBMTOCDC                                                               GRAPH      5
C                                                                       TSPCOM     1
      COMMON /TSPCOM/                                                   TSPCOM     2
     * MEMSIZ, NOB   , NSPARG, NWORD , LENGTH,                          TSPCOM     3
     * NTYPE , IFDBUG, IFTITL, NCHAR , NSUP  ,                          TSPCOM     4
     * MEMST , NOREG , IFPLOT, IFFAST, NPAGE ,                          TSPCOM     5
     * NUMLIN, IFREPL, PROFF , SKIP(11), JPHAS ,                        TSPCOM     6
     * LIMARG, LINE  , NJARG , NARG  , NAME ,                           TSPCOM     7
     *  JARG(4)                                                         TSPCOM     8
C            LENGTH OF JARG SET IN MAIN OVERLAY                         TSPCOM     9
C                                                                       TSPCOM    10
C     DELETED  NAME2  BETWEEN  NAME  AND  JARG  IN  /TSPCOM/ (NWORD = 1)TSPCOM    11
C                                                                       TSPCOM    12
      LOGICAL IFDBUG, IFTITL, IFPLOT, IFFAST,                           TSPCOM    13
     *        IFREPL, PROFF                                             TSPCOM    14
C                                                                       TSPCOM    15
C                                                                       MEMCOM     1
C     NEW COMMON BLOCK ADDED                                            MEMCOM     2
      COMMON /MEMCOM/ IMNSZ(7), USRNAM, PSSVL(20)                       MEMCOM     3
      INTEGER USRNAM                                                    MEMCOM     4
      EQUIVALENCE                                                       MEMCOM     5
     * (IMNSZ(1),MMNSIZ), (IMNSZ(2),LLMARG), (IMNSZ(3),LLMDOT),         MEMCOM     6
     * (IMNSZ(4),LLMBUF), (IMNSZ(5),LLMSYM), (IMNSZ(6),LLMSMP),         MEMCOM     7
     * (IMNSZ(7),LLMOUT)                                                MEMCOM     8
C                                                                       MEMCOM     9
C     IMMSZ USED TO KEEP TRACK OF MEMORY USE                            MEMCOM    10
      DIMENSION IPSSVL(20)                                              MEMCOM    11
      EQUIVALENCE (IPSSVL(1),PSSVL(1))                                  MEMCOM    12
C     PSSVL AND IPSSVL USED TO PASS VARIABLES BETWEEN SUBROUTINES       MEMCOM    13
C     IN DIFFERENT OVERLAYS                                             MEMCOM    14
C     IPSSVL(1) IS USED TO KEEP RECORD OF USE OF COMMON IN LOAD         MEMCOM    15
C        AND IN GENR                                                    MEMCOM    16
C     IPSSVL(3) PASSES COMPUTED GO TO INDEX FROM EXEC TO MATRIX         MEMCOM    17
C     IPSSVL(2) PASSES COMPUTED GO TO INDEX FROM EXEC TO OVERLAYS       MEMCOM    18
C     IPSSVL(4) IS USED TO KEEP RECORD OF USE OF JARG-COMMON            MEMCOM    19
C     IPSSVL(5) IS USED TO KEEP RECORD OF USE OF DOT-COMMON             MEMCOM    20
C     IPSSVL(6) IS USED TO KEEP RECORD OF USE OF BUFFER-COMMON          MEMCOM    21
C     IPSSVL(7) IS USED TO KEEP RECORD OF USE OF JSML-COMMON            MEMCOM    22
C     IPSSVL(8) IS USED TO KEEP RECORD OF USE OF OUTBUF-COMMMON         MEMCOM    23
      EQUIVALENCE                                                       MEMCOM    24
     * (LODUSE, IPSSVL(1)), (JTOOVL, IPSSVL(2)), (JTOMAT, IPSSVL(3))   MEMCOM    25
     *,(JARUSE, IPSSVL(4)), (DOTUSE, IPSSVL(5)), (BUFUSE,IPSSVL(6)),   MEMCOM    26
     * (SMPUSE, IPSSVL(7)), (OBFUSE, IPSSVL(8))                        MEMCOM    27
      INTEGER DOTUSE, BUFUSE, SMPUSE, OBFUSE                           MEMCOM    28
C                                                                       MEMCOM    29
C                                                                       GRAPH      8
C     INSERTED LIMITS ON ARRAY LENGTHS IN GRAPHX (CALLED BELOW)         GRAPH      9
      DATA LIMGRX / 5000 /                                              GRAPH     10
C                                                                       GRAPH     11
CCDCTOIBM                                                               GRAPH     12
      COMMON SPACE(1)                                                   GRAPH     13
      IF(NARG.GT.2)CALL ERG(130,NARG)                                   GRAPH     14
      IF(NOB.GT.1500)CALL ERG(131,NOB)                                  GRAPH     15
      I2=NOB+1                                                          GRAPH     16
      I3=NSPARG+1                                                       GRAPH     17
      I4=I2+NOB                                                         GRAPH     18
      IF(I4.GT.MEMSIZ)CALL ERG(132,I4)                                  GRAPH     19
CIBMTOCDC                                                               GRAPH     20
C                                                                       GRAPH     21
C ***************************************************************     GRAPH     22
C     MAJOR ERROR IN IBM VERSION                                        GRAPH     23
C                                                                       GRAPH     24
```

```
C     CALL 0F GRAPHX CAN DESTR0Y DATA IN BLANK C0MM0N.                    GRAPH    25
C     THE C0RRECTI0N BEL0W IS C0NSERVATIVE--MAXIMUM P0SSIBLE USE          GRAPH    26
C     0F BLANK C0MM0N BY GRAPHX IS TESTED F0R                             GRAPH    27
C                                                                         GRAPH    28
      IF(LIMGRX .LT. MEMSIZ)                                              GRAPH    29
C       THEN HAVE EN0UGH R00M IN BLANK C0MM0N                             GRAPH    30
     *                                                 G0 T0 20           GRAPH    31
C       ELSE N0 EN0UGH R00M                                               GRAPH    32
          WRITE(6,10) LIMGRX, MEMSIZ                                      GRAPH    33
   10     F0RMAT(45H0*** ERR0R.  EXECUTI0N 0F GRAPH MIGHT DESTR0Y  ,      GRAPH    34
     *      28H DATA IN BLANK C0MM0N.  NEED, I10, 6H W0RDS, ,            GRAPH    35
     *      10H 0NLY HAVE, I10, 10H AVAILABLE )                          GRAPH    36
          CALL ERG(132, ITEST)                                            GRAPH    37
C         N0 RETURN T0 THIS P0INT FR0M ERG                                GRAPH    38
C                                                                         GRAPH    39
   20     C0NTINUE                                                        GRAPH    40
C                                                                         GRAPH    41
C     END 0F C0RRECTI0N                                                   GRAPH    42
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ GRAPH    43
C                                                                         GRAPH    44
C     ADDED A REC0RD 0F USE 0F BLANK C0MM0N USE                           GRAPH    45
      L0DUSE = MAX0(L0DUSE, ITEST)                                        GRAPH    46
CCDCT0IBM                                                                 GRAPH    47
      CALL TSMGET(NARG, JARG, SPACE)                                      GRAPH    48
      CALL GRAPHX(SPACE(1), SPACE(I2))                                    GRAPH    49
      RETURN                                                              GRAPH    50
      END                                                                 GRAPH    51
      SUBR0UTINE GRAPHX(YVAR, XVAR)                                       GRAPHX    1
C                                                                         GRAPHX    2
C       THIS IS A REVISI0N 0F **PL0T C**                                  GRAPHX    3
C       22 JUNE 1968                                                      GRAPHX    4
C                                                                         GRAPHX    5
C        N0TE THAT SPACE MUST BE DIMENSI0NED .GE. 5000 IN FRESH           GRAPHX    6
C        THIS R0UTINE IS GENERALLY D0RPPED IN THE 7094 VERSI0N            GRAPHX    7
CIBMT0CDC                                                                 GRAPHX    8
C                                                                         TSPC0M    1
      C0MM0N /TSPC0M/                                                     TSPC0M    2
     * MEMSIZ, N0B   , NSPARG, NW0RD , LENGTH,                           TSPC0M    3
     * NTYPE , IFDBUG, IFTITL, NCHAR , NSUP  ,                           TSPC0M    4
     * MEMST , N0REG , IFPL0T, IFFAST, NPAGE ,                           TSPC0M    5
     * NUMLIN, IFREPL, PR0FF , SKIP(11), JPHAS ,                         TSPC0M    6
     * LIMARG, LINE  , NJARG , NARG  , NAME ,                            TSPC0M    7
     *  JARG(4)                                                          TSPC0M    8
C             LENGTH 0F JARG SET IN MAIN 0VERLAY                          TSPC0M    9
C                                                                         TSPC0M   10
C     DELETED  NAME2  BETWEEN  NAME  AND  JARG  IN  /TSPC0M/ (NW0RD = 1)TSPC0M   11
C                                                                         TSPC0M   12
      L0GICAL IFDBUG, IFTITL, IFPL0T, IFFAST,                             TSPC0M   13
     *        IFREPL, PR0FF                                              TSPC0M   14
C                                                                         TSPC0M   15
CCDCT0IBM                                                                 GRAPHX   10
      C0MM0N SPACE(3000), IBEGIN(250), IEND(250), INDY(1500)              GRAPHX   11
      DIMENSI0N A(16), N0TIE(100), TAB(100), TA(100), TABLE(100), X(100), Y(10GRAPHX   12
     D4), XVAR(1), YVAR(1)                                                GRAPHX   13
      INTEGER TA                                                          GRAPHX   14
      L0GICAL JQ                                                          GRAPHX   15
      DATA A/1H ,1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1HX,1H ,1H+,1H-, GRAPHX   16
     C,1H*/                                                               GRAPHX   17
      D0 7 I=1,250                                                        GRAPHX   18
      IEND(I)=0                                                           GRAPHX   19
    7 IBEGIN(I)=0                                                         GRAPHX   20
C                                                                         GRAPHX   21
C     ARRANGE Y DATA BY DECREASING 0RDER                                  GRAPHX   22
C                                                                         GRAPHX   23
                                                                          GRAPHX   24
    5 D0 32 ID=1,N0B                                                      GRAPHX   25
      D0 32 JD=ID,N0B                                                     GRAPHX   26
      IF(YVAR(ID).GE.YVAR(JD))G0 T0 32
```

```
      DIAGS=YVAR(ID)                                                    GRAPHX     27
      YVAR(ID)=YVAR(JD)                                                 GRAPHX     28
      YVAR(JD)=DIAGS                                                    GRAPHX     29
      DIAGS=XVAR(ID)                                                    GRAPHX     30
      XVAR(ID)=XVAR(JD)                                                 GRAPHX     31
      XVAR(JD)=DIAGS                                                    GRAPHX     32
   32 CONTINUE                                                          GRAPHX     33
C                                                                       GRAPHX     34
C     FIND RANGES AND INCREMENTS                                        GRAPHX     35
C                                                                       GRAPHX     36
      XMA=XVAR(1)                                                       GRAPHX     37
      XMIN=XMA                                                          GRAPHX     38
      DO 18 JB=2,N0B                                                    GRAPHX     39
      IF(XMA.LT.XVAR(JB))XMA=XVAR(JB)                                   GRAPHX     40
      IF(XMIN.GT.XVAR(JB))XMIN=XVAR(JB)                                 GRAPHX     41
   18 CONTINUE                                                          GRAPHX     42
      IF(XMA.LE.0)GO TO 404                                             GRAPHX     43
      IF(XMIN.GE.0)GO TO 404                                            GRAPHX     44
      RANGEX=XMA+ABS(XMIN)                                              GRAPHX     45
      GO TO 405                                                         GRAPHX     46
  404 RANGEX=ABS(XMA-XMIN)                                              GRAPHX     47
  405 IF(YVAR(1).LE.0)GO TO 408                                         GRAPHX     48
      IF(YVAR(N0B).GE.0)GO TO 408                                       GRAPHX     49
      RANGEY=YVAR(1)+ABS(YVAR(N0B))                                     GRAPHX     50
      GO TO 409                                                         GRAPHX     51
  408 RANGEY=ABS(YVAR(1)-YVAR(N0B))                                     GRAPHX     52
  409 XINC=RANGEX/99.                                                   GRAPHX     53
      YINC=RANGEY/103.                                                  GRAPHX     54
      X(1)=XMIN                                                         GRAPHX     55
      Y(1)=YVAR(1)                                                      GRAPHX     56
      DO 27 IA=1,99                                                     GRAPHX     57
   27 X(IA+1)=X(IA)+XINC                                                GRAPHX     58
      DO 28 IB=1,103                                                    GRAPHX     59
   28 Y(IB+1)=Y(IB)-YINC                                                GRAPHX     60
      Y(104)=Y(104)-.00049                                              GRAPHX     61
      JI1=1                                                             GRAPHX     62
      DO 102 II=1,N0B                                                   GRAPHX     63
      DO 101 JI=JI1,104                                                 GRAPHX     64
      IF(YVAR(II).LT.Y(JI))GO TO 101                                    GRAPHX     65
      INDY(II)=JI                                                       GRAPHX     66
      JI1=JI                                                            GRAPHX     67
      GO TO 102                                                         GRAPHX     68
  101 CONTINUE                                                          GRAPHX     69
  102 CONTINUE                                                          GRAPHX     70
C                                                                       GRAPHX     71
C     CHECK FOR Y TIES, INITIAL + FINAL INDICES STORED IN IBEGIN + IEND GRAPHX     72
C          (IT=TIE NO.)                                                 GRAPHX     73
C                                                                       GRAPHX     74
   97 IT=0                                                              GRAPHX     75
      NA1=1                                                             GRAPHX     76
      NA2=N0B-1                                                         GRAPHX     77
   34 CONTINUE                                                          GRAPHX     78
      DO 38 NA=NA1,NA2                                                  GRAPHX     79
      NB=NA+1                                                           GRAPHX     80
      IF(INDY(NA).NE.INDY(NB))GO TO 38                                  GRAPHX     81
      IT=IT+1                                                           GRAPHX     82
      IF(IT.GT.250)CALL ERG(133,IT)                                     GRAPHX     83
      IBEGIN(IT) = NA                                                   GRAPHX     84
      DO 37 MB=NB,N0B                                                   GRAPHX     85
      IF(INDY(NA).EQ.INDY(MB))GO TO 37                                  GRAPHX     86
      NA1=MB                                                            GRAPHX     87
      IEND(IT) = MB - 1                                                 GRAPHX     88
      GO TO 34                                                          GRAPHX     89
   37 CONTINUE                                                          GRAPHX     90
      IEND(IT)=N0B                                                      GRAPHX     91
      GO TO 111                                                         GRAPHX     92
   38 CONTINUE                                                          GRAPHX     93
```

```
  111 CALL 0UTPT
      IF(NUMLIN.GT.0) CALL HEDING                                          GRAPHX    94
C                                                                          GRAPHX    95
C     0UTPUT L00P (THR0UGH 87), WRITES 0NE LINE 0F GRAPH EACH TIME THRU.   GRAPHX    96
C                                                                          GRAPHX    97
      IM1=0                                                                GRAPHX    98
      JE1 = 1                                                              GRAPHX    99
      LUCK = 0                                                             GRAPHX   100
      D0 86 IQ=1,104,2                                                     GRAPHX   101
      II=IQ+1                                                              GRAPHX   102
      D0 123 NC=1,100                                                      GRAPHX   103
      TA(NC)=0                                                             GRAPHX   104
  123 TAB(NC)=A(1)                                                         GRAPHX   105
   40 JE2 = JE1 + 1                                                        GRAPHX   106
      JE3 = JE1                                                            GRAPHX   107
      D0 70 NC = 1,100                                                     GRAPHX   108
   70 TABLE(NC) = A(1)                                                     GRAPHX   109
      JQ=.TRUE.                                                            GRAPHX   110
      IF(INDY(JE1).NE.IQ)G0 T0 71                                          GRAPHX   111
      JQ=.FALSE.                                                           GRAPHX   112
      G0 T0 72                                                             GRAPHX   113
   71 IF(INDY(JE1).NE.II)G0 T0 48                                          GRAPHX   114
   72 IF(INDY(JE1).NE.INDY(JE2))G0 T0 74                                   GRAPHX   115
      IM1=IM1+1                                                            GRAPHX   116
      JE1 = IBEGIN(IM1)                                                    GRAPHX   117
      JE3 = IEND(IM1)                                                      GRAPHX   118
   74 D0 77 JG = JE1,JE3                                                   GRAPHX   119
      D0 113 IJ=1,100                                                      GRAPHX   120
      IF(XVAR(JG).LE.X(IJ))G0 T0 75                                        GRAPHX   121
  113 C0NTINUE                                                             GRAPHX   122
      IJ=100                                                               GRAPHX   123
   75 IF(TABLE(IJ).NE.A(1))G0 T0 170                                       GRAPHX   124
      IF(TAB(IJ).EQ.A(1))G0 T0 69                                          GRAPHX   125
      IF(TAB(IJ).EQ.A(16))G0 T0 45                                         GRAPHX   126
      TA(IJ)=TA(IJ)+1                                                      GRAPHX   127
   51 TAB(IJ)=A(1)                                                         GRAPHX   128
      G0 T0 54                                                             GRAPHX   129
  170 TA(IJ)=TA(IJ)+1                                                      GRAPHX   130
   54 IF(TA(IJ)-10)62,64,63                                                GRAPHX   131
   62 LAZY=TA(IJ)+2                                                        GRAPHX   132
      TABLE(IJ)=A(LAZY)                                                    GRAPHX   133
      TAB(IJ)=A(1)                                                         GRAPHX   134
      G0 T0 77                                                             GRAPHX   135
   64 LUCK = LUCK+1                                                        GRAPHX   136
      N0TIE(LUCK)=10                                                       GRAPHX   137
      TABLE(IJ)=A(12)                                                      GRAPHX   138
      TAB(IJ)=A(1)                                                         GRAPHX   139
      G0 T0 77                                                             GRAPHX   140
   63 N0TIE(LUCK)=N0TIE(LUCK) + 1                                          GRAPHX   141
      G0 T0 77                                                             GRAPHX   142
   69 IF(JQ)G0 T0 45                                                       GRAPHX   143
      TABLE(IJ)=A(16)                                                      GRAPHX   144
      G0 T0 65                                                             GRAPHX   145
   45 TABLE(IJ)=A(15)                                                      GRAPHX   146
   65 TA(IJ)=TA(IJ)+1                                                      GRAPHX   147
   77 C0NTINUE                                                             GRAPHX   148
      JE1=JE3 + 1                                                          GRAPHX   149
      IF(JQ)G0 T0 48                                                       GRAPHX   150
      D0 47 J=1,100                                                        GRAPHX   151
   47 TAB(J)=TABLE(J)                                                      GRAPHX   152
      G0 T0 40                                                             GRAPHX   153
   48 IF(II.EQ.2)G0 T0 12                                                  GRAPHX   154
   13 WRITE(6,2017) (TAB(ME),ME=1,100)                                     GRAPHX   155
 2017 F0RMAT(15X100A1)                                                     GRAPHX   156
      MBB=M0D(II,3)-2                                                      GRAPHX   157
      IF(MBB)85,84,85                                                      GRAPHX   158
CIBMT0CDC                                                                  GRAPHX   159
```

```
C
C      DELETED IF(NW0RD.EQ.1) G0 T0 22     (NW0RD = 1)
C              WRITE(6,2016) JARG(1),JARG(2)
C        2016  F0RMAT(7X,2A4)
C              G0 T0 13
C
   12 C0NTINUE
CCDCT0IBM
   22 WRITE(6,3016) JARG(1)
CIBMT0CDC
C
C     REPLACED 9X,A6
 3016 F0RMAT( 9X,  R8 )
CCDCT0IBM
      G0 T0 13
   84 WRITE(6,2008) Y(II),(TABLE(ME),ME=1,100)
 2008 F0RMAT(1H+F12.3,2H *100A1)
      G0 T0 86
   85 WRITE(6,2009) (TABLE(ME),ME=1,100)
 2009 F0RMAT(1H+13X1H*100A1)
   86 C0NTINUE
      I3=1+NSPARG
CIBMT0CDC
C
C     REM0VED IF(NW0RD.EQ.1) G0 T0 83    (NW0RD = 1)
C     AND A PRINT SIMILAR T0 THE 0NE AT 83
CCDCT0IBM
   83 WRITE(6,3010)JARG(I3),X(1),X(20),X(40),X(60),X(80),X(100),X(10),
     WX(30),X(50),X(70),X(90)
CIBMT0CDC
C
C     CHANGED A6 T0 R8 (CDC6500)
 3010 F0RMAT( 15X,  100(1H*),  R8 / 7X,  2(8X,1H*),  9(9X,
CCDCT0IBM
     F1H*)/F19.3,5(F19.3,1X)/F28.3,4F20.3)
   89 CALL HEDING
      IF(LUCK.EQ.0)G0 T0 60
      WRITE(6,2004)
 2004 F0RMAT(105H LIST 0F TIED P0INT C0UNTS WHERE NUMBER 0F TIES IS GREA
     1TER THAN 9 (READING D0WN Y-AXIS AND ACR0SS X-AXIS))
      L0V1=0
      L0V2=0
  150 L0V2=L0V1+30
      L0V1=L0V2
      IF(L0V1.EQ.LUCK)G0 T0 152
      L0V2=L0V2-(L0V1-LUCK)
  152 L=L0V1-29
      WRITE(6,2015) (N0TIE(J),J=L,L0V2)
 2015 F0RMAT(1H,30I4)
      IF(L0V1.LT.LUCK)G0 T0 150
   60 RETURN
      END
```

```
GRAPHX     161
GRAPHX     162
GRAPHX     163
GRAPHX     164
GRAPHX     165
GRAPHX     166
GRAPHX     167
GRAPHX     168
GRAPHX     169
GRAPHX     170
GRAPHX     171
GRAPHX     172
GRAPHX     173
GRAPHX     174
GRAPHX     175
GRAPHX     176
GRAPHX     177
GRAPHX     178
GRAPHX     179
GRAPHX     180
GRAPHX     181
GRAPHX     182
GRAPHX     183
GRAPHX     184
GRAPHX     185
GRAPHX     186
GRAPHX     187
GRAPHX     188
GRAPHX     189
GRAPHX     190
GRAPHX     191
GRAPHX     192
GRAPHX     193
GRAPHX     194
GRAPHX     195
GRAPHX     196
GRAPHX     197
GRAPHX     198
GRAPHX     199
GRAPHX     200
GRAPHX     201
GRAPHX     202
GRAPHX     203
GRAPHX     204
GRAPHX     205
GRAPHX     206
GRAPHX     207
GRAPHX     208
GRAPHX     209
GRAPHX     210
GRAPHX     211
GRAPHX     212
```

05 23.49.J0 16 EP30 6 FEET

```
      SUBR0UTINE CAPITL                                                  CAPITL     1
CIBMT0CDC                                                                CAPITL     2
C                                                                        TSPC0M     1
      C0MM0N /TSPC0M/                                                    TSPC0M     2
     * MEMSIZ, N0B   , NSPARG, NW0RD , LENGTH,                          TSPC0M     3
     * NTYPE , IFDBUG, IFTITL, NCHAR , NSUP  ,                          TSPC0M     4
     * MEMST , N0REG , IFPL0T, IFFAST, NPAGE ,                          TSPC0M     5
     * NUMLIN, IFREPL, PR0FF , SKIP(11), JPHAS ,                        TSPC0M     6
     * LIMARG, LINE  , NJARG , NARG  , NAME ,                           TSPC0M     7
     *  JARG(4)                                                         TSPC0M     8
C             LENGTH 0F JARG SET IN MAIN 0VERLAY                         TSPC0M     9
C                                                                        TSPC0M    10
C     DELETED  NAME2  BETWEEN  NAME  AND  JARG  IN  /TSPC0M/ (NW0RD = 1)TSPC0M    11
C                                                                        TSPC0M    12
      L0GICAL IFDBUG, IFTITL, IFPL0T, IFFAST,                            TSPC0M    13
     *        IFREPL, PR0FF                                             TSPC0M    14
C                                                                        TSPC0M    15
CCDCT0IBM                                                                CAPITL     4
      C0MM0N DINV(600),CAP(600),DNINV(600),REPL(600),XX(2)               CAPITL     5
CIBMT0CDC                                                                CAPITL     6
C                                                                        CAPITL     7
C     NEXT IS BLANK C0MM0N SPACE NEEDED BY ARRAYS AB0VE                  CAPITL     8
      DATA LIMSIZ / 2402 /                                               CAPITL     9
C                                                                        CAPITL    10
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ CAPITL    11
C     MAJ0R ERR0R IN IBM VERSI0N, N0 TEST 0N AVAILABLE BLANK             CAPITL    12
C     C0MM0N SIZE                                                        CAPITL    13
C                                                                        CAPITL    14
C     ADDED TEST                                                         CAPITL    15
      IF( LIMSIZ .LT. MEMSIZ )                                           CAPITL    16
C       THEN HAVE EN0UGH R00M                                            CAPITL    17
     *                                                                   CAPITL    18
C       ELSE N0T EN0UGH R00M                                  G0 T0 30   CAPITL    19
          WRITE(6,20) LIMSIZ, MEMSIZ                                     CAPITL    20
   20     F0RMAT(25H0*** ERR0R.  CAPITL NEEDS , I10,                     CAPITL    21
     *      29H W0RDS IN BLANK C0MM0N.  0NLY  , I10,                     CAPITL    22
     *      42H AVAILABLE.  PR0CEDURE CAPITL N0T EXECUTED  // )          CAPITL    23
          RETURN                                                         CAPITL    24
C                                                                        CAPITL    25
   30     C0NTINUE                                                       CAPITL    26
C                                                                        CAPITL    27
C     END 0F C0RRECTI0N                                                  CAPITL    28
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ CAPITL    29
CCDCT0IBM                                                                CAPITL    30
      CALL TSGET(JARG,DINV)                                              CAPITL    31
      CALL ARGGET(5,XX,JJ,JJ)                                            CAPITL    32
      NBENCH = XX(1)                                                     CAPITL    33
      CALL ARGGET(6,XX,JJ,JJ)                                            CAPITL    34
      DELTA = XX(1)                                                      CAPITL    35
      CALL ARGGET(7,XX,JJ,JJ)                                            CAPITL    36
      BENCH = XX(1)                                                      CAPITL    37
      CAP(NBENCH) = BENCH                                                CAPITL    38
      XDELT = 1. - DELTA                                                 CAPITL    39
      NN = NBENCH + 1                                                    CAPITL    40
      NNN = NBENCH - 1                                                   CAPITL    41
      IF (NN .GT. N0B) G0 T0 101                                         CAPITL    42
      D0 100 J=NN,N0B                                                    CAPITL    43
  100 CAP(J) = XDELT*CAP(J-1) + DINV(J)                                  CAPITL    44
  101 IF(NNN .EQ. 0) G0 T0 106                                           CAPITL    45
      D0 105 J=1,NNN                                                     CAPITL    46
      K = NBENCH - J                                                     CAPITL    47
  105 CAP(K)=(CAP(K+1)-DINV(K+1))/XDELT                                  CAPITL    48
  106 D0 110 J=1,N0B                                                     CAPITL    49
      REPL(J) = DELTA*CAP(J)                                             CAPITL    50
  110 DNINV(J) = DINV(J) - REPL(J)                                       CAPITL    51
      CALL ARGGET(2,XX,ITYPE,II)                                         CAPITL    52
```

```
      IF(ITYPE .EQ. 3)  CALL TSPUT(JARG(NSPARG+1),CAP)                  CAPITL     53
      CALL ARGGET(3,XX,ITYPE,JJ)                                        CAPITL     54
      IF(ITYPE .EQ. 3)  CALL TSPUT(JARG(2*NSPARG+1),DNINV)              CAPITL     55
      CALL ARGGET(4,XX,ITYPE,JJ)                                        CAPITL     56
      IF(ITYPE .EQ. 3)  CALL TSPUT(JARG(3*NSPARG+1),REPL)               CAPITL     57
      RETURN                                                            CAPITL     58
      END                                                               CAPITL     59
05.24.02.J0 16 EP30         2 FEET
```

```
      SUBROUTINE INPROD(N,JSA,JSB,A,B,PROD)                              INPROD     1
CIBMTOCDC                                                                INPROD     2
C                                                                        INPROD     3
C     CALLED BY GGGMLT GTGMLT G2YMLT ORTHOS                              INPROD     4
C               TGGMLT T2YMLT TINV  UNTRAN VGVMLT                        INPROD     5
C               YFACT                                                    INPROD     6
CCDCTOIBM                                                                INPROD     7
C                                                                        INPROD     8
C     THIS SUBROUTINE CALCULATES THE INNER PRODUCT, AOB, OF THE VECTORS  INPROD     9
C     A AND B.                                                           INPROD    10
CIBMTOCDC                                                                INPROD    11
C     SUM I = 1,N   A(I,JSA)*A(I,JSB)                                    INPROD    12
C                                                                        INPROD    13
C     CORRECTED MARCH 75 WITH ADDITION OF TEMP                           INPROD    14
C                                                                        INPROD    15
CCDCTOIBM                                                                INPROD    16
C                                                                        INPROD    17
C     NOTE THAT THE INNER PRODUCT OF VECTORS OF ZERO LENGTH IS RETURNED   INPROD    18
C     AS ZERO                                                            INPROD    19
C                                                                        INPROD    20
      DIMENSION A(1),B(1)                                                INPROD    21
      DOUBLE PRECISION XPROD                                             INPROD    22
CIBMTOCDC                                                                INPROD    23
     * , TEMPA, TEMPB                                                    INPROD    24
CCDCTOIBM                                                                INPROD    25
      J = 1                                                              INPROD    26
      XPROD = 0.                                                         INPROD    27
      IF (N) 150,150,50                                                  INPROD    28
   50 NN = JSA*N                                                         INPROD    29
      DO 100 I=1,NN,JSA                                                  INPROD    30
CIBMTOCDC                                                                INPROD    31
      TEMPA = A(I)                                                       INPROD    32
      TEMPB = B(J)                                                       INPROD    33
      XPROD = XPROD + TEMPA*TEMPB                                        INPROD    34
CCDCTOIBM                                                                INPROD    35
  100 J = J + JSB                                                        INPROD    36
  150 PROD = XPROD                                                       INPROD    37
      RETURN                                                             INPROD    38
      END                                                                INPROD    39
05.24.15.J0 16 EP30         2 FEET
```

References

[1] Time Series Processor User's Manual, Princeton University Department of Economics, TSP (360-91), February 1971.