CGTM NO. 86
JOHN R. EHRMAN
JANUARY, 1970

SMEDIT:

A SIMPLE-MINDED EDITOR

There have been a number of occasions when I have found it necessary to update files of card images which did not conform to the rather rigid restrictions on sequencing required by the IBM data set utilities IEBUPDAT and IEBUPDTE. In particular, some of the files consist of card images which contain data in all 80 columns, intermixed with cards containing sequence information that bears no necessary relationship to the position of the card in the file. I therefore wrote a short, simple-minded file-editing program that can be used to update and modify files of card images. It has the following advantages:

1. the updating of the items in the file depends only on their relative position in the file;

2. it is possible to locate cards for insertion by searching for a match for a particular string of characters in a given set of columns;

3. the updated file can be listed, sequenced, and identified; and

4. the program is not restricted to the manipulation of a single file per invocation, making it possible to update multiple-file tapes or several data sets at a time.

SMEDIT Control Cards

All control cards have the two characters "./" in columns 1 and 2, as is the case for the IBM utilities. In addition, the two characters in columns 3 and 4 specify the operation to be performed. The rest of the card may be occupied by optional numeric or literal string parameters, beginning in column 5 and appearing freely placed up to and including column 72. The general sequence of operations for updating a single file is as follows: the old master is read from Data Set Reference Number (DSRN) 8, and the new master is written on DSRN 9. All modifications come from the "update file" on DSRN 5, which is the usual input file for a Fortran program. The old master is then edited under the control of the update file, and the new master is written accordingly. The editing modifications must be in increasing order.

In the description of the control cards, the quantities "n1", "n2", and so forth represent numeric data; any non-numeric character is treated as a delimiter for the numeric value. The quantity "s1" represents a character string, which is punched as an apostrophe followed by the desired characters and terminated by an apostrophe. An imbedded apostrophe in a character string must be represented by a <u>pair</u> of apostrophes, as is the usual System/360 convention. Default values are assigned to all parameters (except for the ./IN and ./DE cards), so they may be omitted entirely if desired.

./NM n1 n2

The New Master (NM) card causes a listing of the new master to be made on unit n2 (if omitted, the default unit is 10). The parameter n1 is the number of lines per page (56 is the default) in addition to 2 lines of heading information. The length of the records written is 100 bytes, and the record form is FBA. The default action is that the new master is not listed.

./CH

This card causes a listing of all changes made to the file to be listed with the new master listing. The same LRECL and RECFM are used as for ./NM. Only INSERTs and DELETEs are listed, and only if the new master is listed. The default action is that no changes are listed.

./UP n1

This card causes a listing of the Update File (from unit 5) to be written on unit n1, with parameters LRECL=90, RECFM=FBA. The default action is that no update listing is made.

./ID n1 n2 s1 n3

This card specifies that sequencing and/or identifying information is to be placed in the cards of the new master. The meanings of the three numeric parameters are as follows: n1 is the column in which a numeric sequence number is to begin; it must be between 73 and 80 (the default value is 73). The quantity n2 is the increment to be applied to successive sequence numbers (its default value is 100). If the value of n2 is too small to be observable in the number of columns available for sequencing, it is ignored. The character string s1 is used for identifying the output, and the (optional) quantity n3 is the column in which the character string is to be placed on the output records. If n3 is omitted, it is assumed to be 73. However, if the length of the string is such that it will not fit in the number of columns allowed by the value of n3 (whether implied or explicit), then n3 is reset to right-justify the character string at the right end of the output record. Note that the operation of the sequencing and identifying operations is such that the sequence number is <u>first</u> placed on the output record, and then the character string is put into place. The default action is that no sequence or identification is placed on the output record.

./IN n1 s1 n2

The ./IN card controls the insertion of new text into the output stream. The quantity n1 specifies a value giving the <u>number</u> of the source record (from the old master) <u>after</u> which the following cards are to be inserted. The optional character string s1 allows the user to specify that the card after which the insertion is to take place is to be found by matching the character string with a string from the card. The column in which the match must be made may optionally be specified

by n2; if it is omitted it is assumed to be 73, unless the character string is longer than 8 characters in which case it is assumed that the string is to match the rightmost columns of the incoming card. Note that the occurrence of either insertion condition (the correct number of cards have been read from the old master, or the string match occurs) then the other insertion condition will be ignored. Note that either n1 or s1 (and n2 if then desired) must be present.

./DE n1 n2

This card allows for the deletion of a card or group of cards from the old master. If n2 is omitted, it is assumed to be equal to n1. The range of cards deleted is from n1 to n2 inclusive. Note that deletions may not be made under the control of a string match. Any text cards following the control card will replace the cards that have been deleted.

./EF

This card indicates the end of an update for a single file. All files being worked on are ENDFILEd, and a new edit is begun with the next files in sequence, unless the last ./EF card is followed by the /* delimiter card (the last ./EF card may be omitted). The values of all internal parameters and switches are reset to their default values. Note that the sequencing of the ddnames is determined by the usual FORTRAN conventions for more than one file associated with a given DSRN.

Text Cards

Any card that is not recognized as a control card is considered to be a text card, and it will be inserted into the output stream at the earliest opportunity that is consistent with any preceding control cards. Thus, to place some new cards at the front of the new master, simply precede the first ./DE or ./IN card with the new statements. Similarly, to place some statements after the last card from the old master, precede them with a ./IN card with a numeric operand such as 999999.

Error Messages

SMEDIT will attempt to find simple errors in the data from the update file. Excess literals, oversize numbers, or inconsistent data will be diagnosed under most circumstances. It is important to remember that the updates to be applied to the old master must appear in ascending order: that is, the updates are not sorted before the updating takes place.

Listing Features

The new master contains a title on each page, giving the file number and the new ordinal sequence number of the data on the new master file; that is, the new position of each statement, whether or not data has been placed in them under control of a ./ID card.

Example

Suppose we want to insert a card at the front of each of three datasets which identifies the dataset (an exceedingly simple-minded example). The update file might be constructed as follows:

```
./UP
./NM
THIS CARD GOES ON THE FRONT OF THE DATA SET SPECIFIED BY FT09F001
./EF
./UP
./NM
./DE 1
THIS CARD REPLACES THE FIRST CARD GOING ONTO FT09F002
./EF
./UP
THIS CARD GOES AT THE FRONT OF THE DATA SET GOING TO FT09F003
./IN 9999999
AND THIS ONE WILL GO AT THE BACK.
./EF
/*
```

Listing Of Program

```
C
      IMPLICIT INTEGER*4 (A-Z)
C
      COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT
C
      COMMON /COUNTS/ KNEW, KOLD, LINPAG, LISTLN, NUPDAT, NLOW, NHIGH
C
      COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
     C               SEQUEN, DATA
C
      COMMON /CQUENS/ COLUMN, INCR, SEQNO, IDCOL, IDLEN, IDLIT(68)
      LOGICAL*1 IDLIT
C
      COMMON /CARDS/ UPDATE, SOURCE
      REAL*8 UPDATE(10), SOURCE(10)
C
      COMMON /PARMS/ LITLEN, LITCOL, MARK, N(3), LITRAL(68)
      LOGICAL*1 LITRAL
C
C
C
      INTEGER*2 MSG1(11) /101,9/, M1A(9) /'PAST INSERT POINT'/
      INTEGER*2 MSG2(12) /113,10/, M2A(10)/'NO LITERALS ALLOWED'/
      INTEGER*2 MSG3(12) /112,10/, M3A(10)/'INVALID DELETE RANGE'/
      INTEGER*2 MSG4(11) /111,9/, M4A(9) /'PAST DELETE POINT'/
      INTEGER*2 MSG5(12) /121,10/, M5A(10)/'INVALID UNIT NUMBER'/
      INTEGER*2 MSG6(10) /141,8/, M6A(8) /'BAD ID PARAMETER'/
C
      EQUIVALENCE  (MSG1(3),M1A(1)),
     C             (MSG2(3),M2A(1)),
     C             (MSG3(3),M3A(1)),
     C             (MSG4(3),M4A(1)),
     C             (MSG5(3),M5A(1)),
     C             (MSG6(3),M6A(1))
C
      INTEGER*4 KEYWRD(7)
      DATA KEYWRD /'./IN./DE./EF./CH./UP./NM./ID'/
      DATA MAX /2147483647/, HITAPE /99/
      EQUIVALENCE (TYPE,UPDATE(1))
C
1     FORMAT(10A8)
2     FORMAT(10X,10A8)
C
C     *** START OF PROGRAM ***
C
      NUPDAT = 0
C
C     INITIALIZE VARIABLES AND FLAGS
C
100   CALL INIT
C
```

```
C     READ FROM UPDATE FILE
C
200   READ (5, 1, END=210) UPDATE
C
C     NOTE THAT DATA HAS BEEN READ FOR THIS FILE
C
      DATA = 1
C
C     LIST UPDATE INPUT IF REQUESTED
C
      IF (LISTUP .NE. 0) WRITE (LUPDAT, 2) UPDATE
C
C     SCAN FOR UPDATE CONTROL KEYWORD (NOTE DEPENDENCE ON SIZE OF
C     THE 'KEYWRD' ARRAY)
C
      DO 201 K = 1, 7
      IF (TYPE .EQ. KEYWRD(K)) GO TO 211
201   CONTINUE
C
C     NOT A CONTROL CARD
C
      CALL OUTNEW(UPDATE)
      IF (LISTCH .NE. 0) CALL OUTCH(UPDATE,0)
C
      GO TO 200
C
C     PROCESS CONTROL CARD
C
211   CALL SCAN(UPDATE, 5, 72)
C
      NLOW = N(1)
      NHIGH = N(2)
C
      GO TO (220, 230, 240, 250, 260, 270, 280), K
C              IN   DE   EF   CH   UP   NM   ID
C
210   EOFUPD = 1
      IF (DATA .EQ. 0) GO TO 700
C
C     ENDFILE ON UPDATE FILE
C
240   ENDJOB = 1
      NLOW = MAX
      NHIGH = MAX
C
C     INSERT
C
220   IF (MARK .EQ. 0) NLOW = MAX
      IF (LITCOL .EQ. 0) LITCOL = 73
      IF (NLOW .LT. KOLD) CALL ERROR(MSG1)
      GO TO 400
C
C     DELETE
```

```
C
230   IF (N(2) .EQ. 0) NHIGH = NLOW
      IF (MARK .NE. -1) CALL ERROR(MSG2)
      IF (NLOW .GT. NHIGH) CALL ERROR(MSG3)
      IF ((NLOW .LT. KOLD) .OR. (NHIGH .LT. KOLD)) CALL ERROR(MSG4)
      GO TO 400
C
C     LIST CHANGES
C
250   LISTCH = 1
      GO TO 200
C
C     LIST UPDATE FILE
C
260   LISTUP = 1
      IF (MARK .NE. -1) CALL ERROR(MSG2)
      IF (N(1) .EQ. 0) GO TO 262
      IF (N(1) .LE. HITAPE) GO TO 261
      CALL ERROR(MSG5)
      GO TO 262
261   LUPDAT = 1
262   WRITE (LUPDAT, 2) UPDATE
      GO TO 200
C
C     LIST THE NEW MASTER
C
270   LISTNM = 1
      IF (MARK .NE. -1) CALL ERROR(MSG2)
      IF (N(1) .NE. 0) LINPAG = N(1)
      IF (N(2) .EQ. 0) GO TO 200
      IF (N(2) .LE. HITAPE) GO TO 271
      CALL ERROR(MSG5)
      GO TO 200
271   NEWLIS = N(2)
      GO TO 200
C
C     SET UP SEQUENCING INFORMATION
C
280   SEQUEN = 1
      IDLEN = 0
      IF (MARK .EQ. -1) GO TO 200
      IF (MARK .EQ. 0) GO TO 285
      IF (N(1) .GT. 72  .AND.  N(1) .LE. 80) GO TO 281
      CALL ERROR(MSG6)
      GO TO 282
281   COLUMN = N(1)
282   IF (MARK .EQ. 1) GO TO 285
      IF (N(2) .EQ. 0) GO TO 285
      IF (10**(81-COLUMN) .LE. N(2)) GO TO 283
      INCR = N(2)
      GO TO 285
283   CALL ERROR(MSG6)
      INCR = 1
```

```
285   IF (LITLEN .EQ. 0) GO TO 200
      IDLEN = LITLEN
      IF (LITCOL .NE. 0) IDCOL = LITCOL
      DO 290 I = 1, IDLEN
290   IDLIT(I) = LITRAL(I)
      LITLEN = 0
      GO TO 200
C
C     BEGIN UPDATE OPERATION
C
400   IF (EOFOLD .NE. 0) GO TO 600
C
      ASSIGN 410 TO SWITCH
C
410   KOLD = KOLD + 1
      READ (OLDMAS, 1, END=420) SOURCE
C
C     CHECK FOR DOING A LITERAL MATCH
C
      IF (LITLEN .NE. 0) CALL MATCH (SOURCE, &415)
C
C     BRANCH TO 415 IF MATCH IS FOUND
C
C     CHECK FOR INSIDE RANGE OF DELETE OR INSERT
C
      IF (KOLD .LT. NLOW) GO TO 430
C
C     CHECK FOR DELETE OPERATION
C
415   LITLEN = 0
C
C     RESET LITERAL PRESENCE FLAG
C
      IF (K .EQ. 2) GO TO 440
C
C     INSERT OPERATION
C
      ASSIGN 200 TO SWITCH
C
C     WRITE SOURCE CARD ON NEW MASTER
C
430   CALL OUTNEW(SOURCE)
C
500   GO TO SWITCH, (200,410)
C
C     CHECK FOR END OF DELETE RANGE
C
440   IF (KOLD .LT. NHIGH) GO TO 450
C
      ASSIGN 200 TO SWITCH
C
450   IF (LISTCH .NE. 0) CALL OUTCH(SOURCE,KOLD)
C
```

```
      GO TO 500
C
C     END OF FILE ON OLD MASTER
C
420   EOFOLD = 1
C
600   IF (ENDJOB .EQ. 0) GO TO 200
C
C     ENDFILE LISTING FILES
C
C
610   IF (LISTUP .EQ. 0) GO TO 620
C
      LISTUP = -1
      ENDFILE LUPDAT
C
620   IF (LISTNM .EQ. 0) GO TO 630
C
      IF (NEWLIS .EQ. LUPDAT  .AND.  LISTUP .EQ. -1) GO TO 630
      ENDFILE NEWLIS
C
630   ENDFILE NEWMAS
C
      IF (EOFUPD .EQ. 0) GO TO 100
700   STOP
      END
      SUBROUTINE ERROR(MSG)
C
      IMPLICIT INTEGER*4 (A-Z)
C
      COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT
C
      INTEGER*2 MSG(20)
C
C
      N = MSG(2) + 2
      WRITE (LUPDAT,1) MSG(1), (MSG(K), K = 3, N)
1     FORMAT(' **ERROR',I4,2X,35A2)
      RETURN
      END
      SUBROUTINE OUTNEW(CARD)
C
      IMPLICIT INTEGER*4 (A-Z)
C
      COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT
C
      COMMON /COUNTS/ KNEW, KOLD, LINPAG, LISTLN, NUPDAT, NLOW, NHIGH
C
      COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
     C               SEQUEN, DATA
C
C
C
```

```
C
      REAL*8 CARD(10)
C
      KNEW = KNEW+1
      IF (SEQUEN .NE. 0) CALL SEQID(CARD)
      WRITE (NEWMAS, 100) CARD
100   FORMAT(10A8)
      IF (LISTNM .EQ. 0) RETURN
      IF (MOD(LISTLN,LINPAG) .NE. 0) GO TO 1
      LISTLN = 0
      WRITE (NEWLIS, 101) NUPDAT
101   FORMAT('1',9X,'NEW MASTER, UPDATE FILE NO.',I4,T94,'NEW NO.'/)
1     LISTLN = LISTLN + 1
      WRITE (NEWLIS, 102) CARD, KNEW
102   FORMAT(10X,10A8,I10)
      RETURN
      END
      SUBROUTINE INIT
C
      IMPLICIT INTEGER*4 (A-Z)
C
      COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT
C
      COMMON /COUNTS/ KNEW, KOLD, LINPAG, LISTLN, NUPDAT, NLOW, NHIGH
C
      COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
     C               SEQUEN, DATA
C
      COMMON /CQUENS/ COLUMN, INCR, SEQNO, IDCOL, IDLEN, IDLIT(68)
      LOGICAL*1 IDLIT
C
C
C
C     INCREMENT UPDATE COUNT
      NUPDAT = NUPDAT + 1
C     SET NEW MASTER TAPE NUMBER
      NEWMAS = 9
C     SET OLD MASTER TAPE NUMBER
      OLDMAS = 8
C     LISTING OF NEW MASTER GOES ON THIS
      NEWLIS = 10
C     LISTING OF UPDATE INPUT
      LUPDAT = 6
C     CARD COUNT ON NEW MASTER
      KNEW = 0
C     CARD COUNT FROM OLD MASTER
      KOLD = 0
C     LINES PER PAGE ON NEW MASTER LISTING
      LINPAG = 56
C     LINE COUNT FOR NEW MASTER LISTING
      LISTLN = 0
C     SET 'LIST CHANGES' FLAG OFF
      LISTCH = 0
```

```
C     SET 'LIST UPDATES' FLAG OFF
      LISTUP = 0
C     SET 'LIST NEW MASTER' FLAG OFF
      LISTNM = 0
C     SET OLD MASTER ENDFILE FLAG OFF
      EOFOLD = 0
C     SET UPDATE FILE ENDFILE FLAG OFF
      EOFUPD = 0
C     SET END-OF-A-JOB FLAG OFF
      ENDJOB = 0
C     SET RESEQUENCE FLAG OFF
      SEQUEN = 0
C     SET DEFAULT SEQUENCE COLUMN
      COLUMN = 73
C     SET DEFAULT SEQUENCE INCREMENT
      INCR = 100
C     SET INITIAL SEQUENCE NUMBER
      SEQNO = 0
C     NO ID CHARACTERS TO BE INSERTED
      IDLEN = 0
C     DEFAULT COLUMN FOR ID INSERTION
      IDCOL = 73
C     RESET DATA-CARD READ FLAG TO NO DATA
      DATA = 0
C
      RETURN
      END
      SUBROUTINE OUTCH(CARD,MODE)
C
      IMPLICIT INTEGER*4 (A-Z)
C
      COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT
C
      COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
     C               SEQUEN, DATA
C
C
C
      REAL*8 CARD(10)
C
      IF (LISTNM .EQ. 0) GO TO 3
      IF (MODE .NE. 0) GO TO 1
      WRITE (NEWLIS, 100) CARD
100   FORMAT(' INSERT   ',10A8)
      RETURN
1     WRITE (NEWLIS, 200) CARD
200   FORMAT(' DELETE   ',10A8)
3     RETURN
      END
      SUBROUTINE SEQID(CARD)
C
      IMPLICIT INTEGER*4 (A-Z)
C
```

```
      COMMON /CQUENS/ COLUMN, INCR, SEQNO, IDCOL, IDLEN, IDLIT(68)
      LOGICAL*1 IDLIT
C
C
C
C
      LOGICAL*1  DIGITS(10) /'0123456789'/, CARD(80)
C
      SEQNO = SEQNO + INCR
      K = 80
      N = SEQNO
C
100   MODN10 = MOD(N,10)
      CARD(K) = DIGITS(MODN10+1)
      N = N / 10
      K = K - 1
      IF (K .GE. COLUMN) GO TO 100
      IF (IDLEN .EQ. 0) GO TO 300
      N = IDCOL + IDLEN - 1
      DO 200 K = IDCOL, N
200   CARD(K) = IDLIT(K+1-IDCOL)
300   RETURN
      END
      SUBROUTINE MATCH(CARD,*)
C
      IMPLICIT INTEGER*4 (A-Z)
C
      COMMON /PARMS/ LITLEN, LITCOL, MARK, N(3), LITRAL(68)
      LOGICAL*1 LITRAL
C
      LOGICAL*1 CARD(80)
C
C
C
      DO 100 K = 1, LITLEN
      IF (CARD(K+LITCOL-1) .NE. LITRAL(K)) RETURN
100   CONTINUE
      RETURN 1
      END
      SUBROUTINE SCAN(STRING, KLEFT, KRIGHT)
C
      IMPLICIT INTEGER*4 (A-Z)
C
      COMMON /PARMS/ LITLEN, LITCOL, MARK, N(3), LITRAL(68)
      LOGICAL*1 LITRAL
C
      INTEGER*4 ZIP/0/, ZEROCH/240/, QUOTE/125/
C
      LOGICAL*1 G(4), STRING(80)
      EQUIVALENCE (ZIP,G(1))
C
      INTEGER*2 MSGA(9) /201,7/,MA(7) /'INVALID NUMBER'/
      INTEGER*2 MSGB(10) /202,8/, MB(8) /'EXCESS LITERALS'/
```

```
      INTEGER*2 MSGC(10) /203,8/,MC(8) /'TOO MANY NUMBERS'/
      INTEGER*2 MSGD(11) /204,9/,MD(9) /'LITERAL TRUNCATED'/
      INTEGER*2 MSGF(13) /206,11/,MF(11)/'INVALID LITERAL COLUMN'/
C
      EQUIVALENCE (MSGA(3),MA(1)),(MSGB(3),MB(1)),(MSGC(3),MC(1)),
     C            (MSGD(3),MD(1)),(MSGF(3),MF(1))
C
C
C
      MARK = -1
      BADNUM = 0
      QUOTSW = 0
      WORK = 0
      LITCOL = 0
      LITLEN = 0
      LITSW = 0
      N(1) = 0
      N(2) = 0
      N(3) = 0
      M = 1
      BRANCH = 1
      K = KLEFT
C
C      START OF SCAN LOOP
C
1     G(4) = STRING(K)
      IF (ZIP .NE. QUOTE) GO TO 50
C
C     HAVE A QUOTE
C
      IF (BRANCH .NE. 0) GO TO 10
      QUOTSW = 1
      IF (LITSW .NE. 0) CALL ERROR(MSGB)
      GO TO 80
10    IF (QUOTSW .NE. 0) GO TO 15
      IF (LITSW .EQ. 0) GO TO 11
      CALL ERROR(MSGB)
      LITLEN = 0
11    QUOTSW = 1
      GO TO 100
15    IF (K .EQ. KRIGHT) GO TO 100
C
C     CHECK FOR PAIRED QUOTES
C
      G(4) = STRING(K+1)
      IF (ZIP .EQ. QUOTE) GO TO 20
      QUOTSW = 0
      GO TO 100
20    K = K + 1
C     PAIRED QUOTE
      GO TO 60
50    IF (QUOTSW .NE. 0) GO TO 60
      IF (ZIP .LT. ZEROCH) GO TO 70
```

```
      IF (WORK .GE. 214748363) BADNUM = 1
      WORK = 10 * WORK + (ZIP-ZEROCH)
      BRANCH = 0
      GO TO 100
C
C     MOVE A CHARACTER INTO THE LITERAL STRING
C
60    LITLEN = LITLEN + 1
      LITRAL(LITLEN) = STRING(K)
      LITSW = 1
C
C     NOTE HOW MANY INTEGERS OCCURRED BEFORE LITERAL
C
      MARK = M - 1
      GO TO 100
C
C     HAVE A NON-NUMERIC, SEE IF IT ENDS A NUMBER
C
70    IF (BRANCH .NE. 0) GO TO 100
C     STORE THE NUMBER
80    IF (BADNUM .EQ. 0) GO TO 81
      CALL ERROR(MSGA)
      WORK = 0
      BADNUM = 0
81    IF (M .LE. 3) GO TO 82
      CALL ERROR(MSGC)
      GO TO 83
82    N(M) = WORK
83    WORK = 0
      BRANCH = 1
      M = M + 1
C
C     END OF SEARCH LOOP, INCREMENT INDEX
C
100   K = K + 1
      IF (K .LE. KRIGHT) GO TO 1
C
C     END OF SCAN
C
      IF (BRANCH .NE. 0) GO TO 200
      IF (M .GT. 3) GO TO 190
      N(M) = WORK
      GO TO 200
190   CALL ERROR(MSGC)
200   IF (MARK .EQ. -1) GO TO 300
      IF (MARK .LE. 2) GO TO 210
      CALL ERROR(MSGC)
      GO TO 300
210   LITCOL = N(MARK+1)
      IF (LITCOL .LT. 81) GO TO 220
      LITCOL = 73
      CALL ERROR(MSGF)
220   IF ((LITCOL + LITLEN) .LE. 81) GO TO 300
```

```
      LITLEN = 81 - LITCOL
      CALL ERROR(MSGD)
300   RETURN
      END
```