SMEDIT2:
A SIMPE-MINDED EDITOR
FOR CARD FILES

(SON OF SMEDIT)
There have been occasions when I have found it necessary to update files of card images which did not conform to the rather rigid restrictions on card format and 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 insertion points by searching for a match for a particular string of characters in a given set of columns. (This provides a facility similar to the IBM utilities.)
3. The updated file can be listed, sequenced, and identified, and the sequencing and identification may be changed within a file.
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.
5. The editing process may move groups of cards from their current positions to different positions within the file, as well as including groups of externally-defined cards into the file at arbitrary positions. This type of movement makes use of a user-specified temporary file called a "shelf" in this description.

SMEDIT2 Control Cards

All control cards have the two characters "/n" 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. Note that the parameters are order-dependent; they must be given in the correct sequence.

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 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 seven different types of control cards, the quantities "n1", "n2", and "n3" represent numeric data; any non-
numeric character is treated as a delimiter for the numeric value. The quantity "s1" represents a character string, which appears 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 pair of apostrophes, as is the usual System/360 convention. Except on the .IN and .DE cards, default values are assigned to all parameters, so they may be omitted from other control cards entirely if desired. On .IN cards, only one parameter is required; on .DE cards, one parameter is required, but two may be needed to specify a range.

Listing and Identification Control Cards

LIST THE NEW MASTER FILE

./NM n1 n2

The New Master (NM) card causes a numbered listing of the new master to be made on unit n2 (if omitted, the default unit is 10). The numbering information can then be used in subsequent edits. The parameter n1 is the number of lines per page (if n1 and n2 are omitted, 56 is the default) in addition to 2 lines of heading information. The default action is that the new master is not listed.

LIST CHANGES TO THE OLD MASTER

./CH n1

This card causes a listing of all changes made to the file to be listed on unit n1 (the default is 7). The same number of lines per page are printed as on the "NM" file. Only INSERTs and DELETEs are listed; the old sequence number is listed with each deletion, so that a check can be made for correct deletions. The default action is that no changes are listed.

LIST UPDATE FILE

./UP n1

This card causes a listing of the Update File (from unit 5) to be written on unit n1. The default action is that no update listing is made. An additional flag is placed on lines containing insertions from "shelf" files (see the "n2" parameter on the .IN card). The default unit is 6, the standard Fortran print file.
IDENTIFY AND SEQUENCE NEW MASTER

.\ID n1 n2 sl 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 sl 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 first 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.

Editing Control Cards

INSERT CARDS

.\IN n1 sl n2 n3

The .\IN card controls the insertion of new text into the output stream. The quantity n1 specifies a value giving the number of the source record (from the old master) after which the following cards are to be inserted. The optional character string sl 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 then 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 sl (and n2 if then desired) must be present.

The parameter n3 is the DSRN of a "shelf" file whose contents are to be copied into the new master at the insertion point. If sl is omitted, then only n1 and n3 are present; if sl is present, then n2 must be present also. After the cards from DSRN n3 have been copied, that unit is "rewound" (repositioned at the start of the file) so that the
Same data may be copied again if desired.

**DELETE (AND Optionally SAVE) CARDS**

`./DE n1 n2 n3`

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.

The parameter n3 specifies the DSRN of a "shelf" onto which the deleted statements are to be written, for possible later insertion into the new master, or for safekeeping. If it is desired to delete only a single card onto a "shelf" file, that card number must be given for both n1 and n2. That is, if n3 is present, n1 and n2 must also be.

**COMPLETE AN EDITED FILE**

`./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 dnames is determined by the usual FORTRAN conventions for more than one file associated with a given DSRN. Because all "shelf" files are automatically rewound (repositioned at the start) after they have been read, they are unaffected by the ./EF card. Thus, for example, the same data may be copied into each of several files, or data may be moved from one file to another.

**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

SMEDIT2 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 will not be sorted before the updating takes place.

Listing Features

The new master listing 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 it has been sequenced under control of a ./ID card.

Examples

Some sample JCL is given with some of the examples; the DD statements would of course be preceded by appropriate JOB, JOBLIB, and EXEC statements.

Example 1

A file of cards is to be sequenced by 10's, and the name SUBA should appear in columns 73-76. Replace cards 1 through 14 with a single Fortran comment statement, and then punch the resulting deck.

```
//PT09F001 DD SYSOUT=B
//PT08F001 DD *
----- old deck ----- */
//PT05F001 DD *
./ID 10 'SUBA'
=./DE 1 14
C THIS IS A FORTRAN COMMENT */
```

Example 2

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

```plaintext
//** SPECIFY 3 INPUT FILES FROM OLD MASTER
//PT08F001 DD UNIT=TAPE9,VOL=SER=MYTAPE,LABEL=(1,NL), ETC ETC
//PT08F002 DD UNIT=AFF=PT08F001,VOL=REF=*,PT08F001,LABEL=(2,NL), ETC
//PT08F003 DD UNIT=AFF=PT08F001,VOL=REF=*,PT08F001,LABEL=(3,NL), ETC
//** SPECIFY 3 OUTPUT FILES FOR NEW MASTER
//PT09F001 DD UNIT=TAPE9,VOL=SER=NEWMAS,LABEL=(1,NL), ETC ETC
//PT09F002 DD UNIT=AFF=PT09F001,VOL=REF=*,PT09F001,LABEL=(2,NL), ETC
//PT09F003 DD UNIT=AFF=PT09F001,VOL=REF=*,PT09F001,LABEL=(3,NL), ETC
//** DEFINE 3 FILES FOR LISTING OF NEW MASTER
//PT10F001 DD SYSOUT=A
//PT10F002 DD SYSOUT=A
//PT10F003 DD SYSOUT=A
//** DEFINE 3 FILES FOR UPDATE LISTING
//PT06F001 DD SYSOUT=A
//PT06F002 DD SYSOUT=A
//PT06F003 DD SYSOUT=A
//** UPDATE FILE
//PT05F001 DD *
-\UP
-\MN
THIS CARD GOES ON THE FRONT OF THE DATA SET SPECIFIED BY PT09F001
-\EF
-\UP
-\MN
-DE 1
THIS CARD REPLACES THE FIRST CARD GOING ONTO PT09F002
-\EF
-\UP
THIS CARD GOES AT THE FRONT OF THE DATA SET GOING TO PT09F003
-\IN 9999999
AND THIS ONE WILL GO AT THE BACK.
-\EF
/**

Example 3

Move cards 22 through 26 to appear after card 48.

//** DEFINE A SCRATCH FILE ON DSRN 99
//PT99F001 DD UNIT=SYSDA,SPACE=(CYL,1),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3520)
//PT05F001 DD *
-DE 22 26 99
-\IN 48 99
/**
Example 4

Obtain a numbered listing of two files from a tape, so that the relative positions of each card will be known for later editing.

//PT08F001 DD UNIT=TAPE9, ETC
//PT08F002 DD UNIT=APP=PT08F001, ETC
//* DEFINE DUMMY NEW MASTER FILES
//PT09F001 DD DUMMY,DCB=(RECFM=F,BLKSIZE=80)
//PT09F002 DD DUMMY,DCB=(RECFM=F,BLKSIZE=80)
//* NEW MASTER LISTING FILES
//PT10F001 DD SYSOUT=A
//PT10F002 DD SYSOUT=A
//PT05F001 DD *
-/*NM
-/*EP
-/*NM
/*

Example 5

Copy the cards in data set COMMON into a file named SOURCE after the 3rd', 200th', and 479th statements. Punch and list the result.

//PT08F001 DD DSNNAME=SOURCE,DISP=SHR
//PT09F001 DD SYSOUT=B
//PT25F001 DD DSNNAME=COMMON,DISP=SHR
//PT10F001 DD SYSOUT=A
//PT05F001 DD *
-/*NM
-/*IN 3 25
-/*IN 200 25
-/*IN 479 25
/*
Summary of File Specifications

<table>
<thead>
<tr>
<th>File</th>
<th>DSRN</th>
<th>Optional?</th>
<th>LRECL</th>
<th>RECFM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old Master</td>
<td>8</td>
<td>no</td>
<td>80</td>
<td>PB</td>
</tr>
<tr>
<td>New Master</td>
<td>9</td>
<td>no</td>
<td>80</td>
<td>PB</td>
</tr>
<tr>
<td>Updates</td>
<td>5</td>
<td>no</td>
<td>80</td>
<td>PB</td>
</tr>
<tr>
<td>Update Listing</td>
<td>(6)</td>
<td>yes</td>
<td>90</td>
<td>PBA</td>
</tr>
<tr>
<td>New Master Listing</td>
<td>(10)</td>
<td>yes</td>
<td>100</td>
<td>PBA</td>
</tr>
<tr>
<td>Changes Listing</td>
<td>(7)</td>
<td>yes</td>
<td>100</td>
<td>PBA</td>
</tr>
</tbody>
</table>

1 Default unit numbers are parenthesized
2 Other record formats are possible, of course.

Summary of Control Cards

<table>
<thead>
<tr>
<th>Card</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>.//UP</td>
<td>List control cards and updates on unit 6</td>
</tr>
<tr>
<td>.//UP n1</td>
<td>List control cards and updates on unit n1</td>
</tr>
<tr>
<td>.//CH</td>
<td>List changes on unit 7</td>
</tr>
<tr>
<td>.//CH n1</td>
<td>List changes on unit n1</td>
</tr>
<tr>
<td>.//NM</td>
<td>List New Master on unit 10, 58 lines/page</td>
</tr>
<tr>
<td>.//NM n1</td>
<td>List New Master on unit 10, n1 lines/page</td>
</tr>
<tr>
<td>.//NM n1 n2</td>
<td>List New Master on unit n2, n1 lines/page</td>
</tr>
<tr>
<td>.//ID</td>
<td>Sequence New Master by 100 in columns 73-80</td>
</tr>
<tr>
<td>.//ID n1</td>
<td>Sequence New Master by 100 in columns n1-80</td>
</tr>
<tr>
<td>.//ID n1 n2</td>
<td>Sequence New Master by n2 in columns n1-80</td>
</tr>
<tr>
<td>.//ID 's1'</td>
<td>Sequence New Master by 100 in columns 73-80 and place string s1 starting in column 73</td>
</tr>
<tr>
<td>.//ID 's1' n3</td>
<td>Sequence New Master by 100 in columns 73-80 and place string s1 starting in column n3</td>
</tr>
</tbody>
</table>
Sequence New Master by 100 in columns n1-80 and place string s1 starting in column 73

Sequence New Master by 100 in columns n1-80 and place string s1 starting in column n3

Sequence New Master by n2 in columns n1-80 and place string s1 starting in column 73

Sequence New Master by n2 in columns n1-80 and place string s1 starting in column n3

Insert the following cards after card n1 from the Old Master

Insert the following cards after the first card from the Old Master which matches string s1 starting in column 73

Insert the following cards after the first card from the Old Master which matches the string s1 starting in column n2

Insert the following cards after either card n1, or the first card to match string s1 (starting in column 73) from the Old Master, whichever occurs first

Insert after card n1 from the Old Master the cards to be found on DSRN n3

Insert the cards on DSRN n3 after card n1, or after the card which matches the string s1 (beginning in column n2), whichever occurs first

Delete card n1 from the Old Master

Delete cards n1 through n2 inclusive from the Old Master

Delete cards n1 through n2 from the Old Master, and save them on DSRN n3 as a group
Listing Of Program

IMPLICIT INTEGER*4 (A-Z)
COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPFILE, DELETE,
C CHANGE
COMMON /COUNTS/ KNEW, KOLD, LIPAG, LISTNM, NUPDAT, MLOW, NHIGH,
C NMAPAGE, CMAPAGE, CMGLIN
COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFPOL, EOPUPD, ENDJOB,
C SEQUEN, DATA
COMMON /CQUEENS/ COLUMN, INCR, SEQNO, IDCOL, IDELEN, IDLIT(68)
LOGICAL*1 IDLIT
COMMON /CARDS/ UPDATE, SOURCE
REAL*8 UPDATE (10), SOURCE (10)
COMMON /PARMS/ LITLEN, LITCOL, MARK, N(3), LITRAL(68)
LOGICAL*1 LITRAL
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"/
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))
INTEGER*4 KEYWRD(7)
DATA KEYWRD /.'./IN./DE./EP./CH./UP./MR./ID'./
DATA MAX /2147483647/, HITAPE /99/
EQUIVALENCE (TYPE,UPDATE(1))
1 FORMAT(10A8)
2 FORMAT(10X,10A8)
3 FORMAT(* FROM',I3,2X,10A8)
C *** START OF PROGRAM ***
NUPDAT = 0
C INITIALIZE VARIABLES AND FLAGS
100 CALL INIT
C READ FROM UPDATE FILE
200 READ (UPFILE, 1, END=210) UPDATE
C NOTE THAT DATA HAS BEEN READ FOR THIS FILE
DATA = 1
C LIST UPDATE INPUT IF REQUESTED
IF (LISTUP .EQ. 0) GO TO 202
IF (UPFILE .EQ. 5) WRITE (LUPDAT, 2) UPDATE
IF (UPFILE .EQ. 5) WRITE (LUPDAT, 3) UPFILE, UPDATE
C SCAN FOR UPDATE CONTROL KEYWORD (NOTE DEPENDENCE ON SIZE OF
THE 'KEYWRD' ARRAY)
202 DO 201 K = 1, 7
   IF (TYPE .EQ. KEYWRD(K)) GO TO 211
201 CONTINUE
\begin{verbatim}
NOT A CONTROL CARD
CALL OUTNEW(UPDATE)
IF (LISTCH .NE. 0) CALL OUTCH(UPDATE, 0)
GO TO 200

211 CALL SCAN(UPDATE, 5, 72)
NLOW = N(1)
NHIGH = N(2)
GO TO (220, 230, 240, 250, 260, 270, 280), K

210 IF (UPFILE .EQ. 5) GO TO 215
REWIND UPFILE
UPFILE = 5
GO TO 200

215 EOFUPD = 1
IF (DATA .EQ. 0) GO TO 700

240 EMDJOB = 1
NLOW = MAX
NHIGH = MAX

220 IF (MARK) 223, 224, 225
225 UK = N(3)
IF (LITCOL .EQ. 0) LITCOL = 73
GO TO 222
224 NLOW = MAX
223 UK = N(2)
222 IF (NLOW .LT. KOLD) CALL ERROR(MSG1)
IF (UK .EQ. 0) GO TO 226
IF (UK .LE. HITAPE) GO TO 221
CALL ERROR(MSG5)
GO TO 400

221 UPFILE = UK
C TEST FOR IMMEDIATE INSERTION FROM UPFILE
226 IF (NLOW .EQ. KOLD) GO TO 200
GO TO 400

C DELETE
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)
IF (N(3) .EQ. 0) GO TO 400
IF (N(3) .LE. HITAPE) GO TO 231
CALL ERROR(MSG5)
GO TO 400

231 DELETE = N(3)
GO TO 400

C LIST CHANGES
250 IF (MARK .NE. -1) CALL ERROR(MSG2)
IF (N(1) .EQ. 0) GO TO 252
IF (N(1) .LE. HITAPE) GO TO 251
CALL ERROR(MSG5)
GO TO 200
\end{verbatim}
CHANGE = N(1)
LISTCH = 1
GO TO 200

C LIST UPDATE FILE

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
LUPDAT = N(1)

IF (LISTUP .EQ. 0) WRITE (LUPDAT, 2) UPDATE
LISTUP = 1
GO TO 200

C LIST THE NEW MASTER

IF (MARK .NE. -1) CALL ERROR(MSG2)
IF (N(1) .EQ. 0) LINPAG = N(1)
IF (N(2) .EQ. 0) GO TO 272
IF (N(2) .LE. HITAPE) GO TO 271
CALL ERROR(MSG5)
GO TO 200
NEWLIS = N(2)

LISTMM = 1
GO TO 200

C SET UP SEQUENCING INFORMATION

SEQUEW = 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
COLUMN = N(1)

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
IMCR = N(2)
GO TO 285
CALL ERROR(MSG6)
IMCR = 1

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 BEGIN UPDATE OPERATION

IF (EOFOLD .NE. 0) GO TO 600
ASSIGN 410 TO SWITCH

410 KOLD = KOLD + 1
READ (OLDMAS, 1, END=420) SOURCE

C CHECK FOR DOING A LITERAL MATCH
C BRANCH TO 415 IF MATCH IS FOUND
IF (LITLEN .NE. 0) CALL MATCH (SOURCE, 6415)

C CHECK FOR INSIDE RANGE OF DELETE OR INSERT
IF (KOLD .LT. NLOW) GO TO 430
C CHECK FOR DELETE OPERATION
415 LITLEN = 0
C RESET LITERAL PRESENCE FLAG
IF (K .EQ. 2) GO TO 440
C INSERT OPERATION
ASSIGN 200 TO SWITCH
C WRITE SOURCE CARD ON NEW MASTER
430 CALL OUTNEW(SOURCE)
500 GO TO SWITCH, (200, 410)
440 IF (LISTCH .NE. 0) CALL OUTCH(SOURCE, KOLD)
IF (DELETE .NE. 0) WRITE (DELETE, 1) SOURCE
C CHECK FOR END OF DELETE RANGE
IF (KOLD .LT. NHIGH) GO TO 500
ASSIGN 200 TO SWITCH
IF (DELETE .EQ. 0) GO TO 500
ENDFILE DELETE
REWIND DELETE
DELETE = 0
GO TO 500
C END OF FILE ON OLD MASTER
420 EOFOLD = 1
600 IF (ENDJOB .EQ. 0) GO TO 200
C ENDFILE LISTING PILES
610 IF (LISTUP .EQ. 0) GO TO 620
LISTUP = -1
ENDFILE LUPDAT
620 IF (LISTNN .EQ. 0) GO TO 630
IF (NEWLIS .EQ. LUPDAT .AND. LISTUP .EQ. -1) GO TO 630
ENDFILE NEWLIS
LISTNN = -1
630 IF (LISTCH .EQ. 0) GO TO 640
IF (CHANGE .EQ. LUPDAT .AND. LUPDAT .EQ. -1) GO TO 640
IF (CHANGE .EQ. NEWLIS .AND. LISTNN .EQ. -1) GO TO 640
ENDFILE CHANGE
640 ENDFILE NEWMAS
IF (EOFUPD .EQ. 0) GO TO 100
700 STOP
END
C
C SUBROUTINE ERROR(MSG)
IMPLICIT INTEGER*4 (A-Z)
COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPFILE, DELETE,
C CHANGE
INTEGER*2 MSG(20)
N = MSG(2) + 2
WRITE (LUPDAT, 1) MSG(1), (MSG(K), K = 3, N)
1 FORMAT (' **ERROR*,I4,2X,35A2)
RETURN
END
SUBROUTINE OUTNER(CARD)
IMPLICIT INTEGER*4 (A-Z)
COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C CHANGE
COMMON /COUNTS/ KNEW, KOLD, LINPAG, LISTLN, NUPDAT, NLOW, NHIGH,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C
REAL*8 CARD(10)
KNEW = KNEW + 1
IF (SEQUEN .NE. 0) CALL SEQID(CARD)
WRITE (NEWMAS, 100) CARD
100 FORMAT(10A8)
IF (LISTNM .EQ. 0) RETURN
IF (MOD(LISTNM, LINEPAG) .NE. 0) GO TO 1
NMPAGE = NMPAGE + 1
LISTLN = 0
WRITE (NEWLIS, 101) NUPDAT, NMPAGE
101 FORMAT('NEW MASTER, UPDATE FILE NO.', I4, T50, 'PAGE', I5, T94,
C 'NEW NO.' /)
LISTLN = LISTLN + 1
WRITE (NEWLIS, 102) CARD, KNEW
102 FORMAT(10X, 10A8, I10)
RETURN
END

SUBROUTINE INIT
IMPLICIT INTEGER*4 (A-Z)
COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C CHANGE
COMMON /COUNTS/ KNEW, KOLD, LINPAG, LISTLN, NUPDAT, NLOW, NHIGH,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /COUNTS/ KNEW, KOLD, LINPAG, LISTLN, NUPDAT, NLOW, NHIGH,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPPFILE, DELETE,
C COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C DELETION SAVE FILE
DELETE = 0
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 PAGE NUMBER FOR LISTING OF NEW MASTER
MMPAGE = 0
C LINE COUNT FOR NEW MASTER LISTING
LISTLN = 0
C PAGE NUMBER FOR CHANGE LISTING
CHPAGE = 0
C LINE COUNT FOR CHANGE LISTING
CHGLIN = 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
RETURN
END
C
C SUBROUTINE OUTCH(CARD,MODE)
IMPLICIT INTEGER*4 (A-Z)
COMMON /TAPES/ NEWMAS, OLDMAS, NEWLIS, LUPDAT, UPFILE, DELETE,
C CHANGE
COMMON /FLAGS/ LISTCH, LISTUP, LISTNM, EOFOLD, EOFUPD, ENDJOB,
C SEQUEN, DATA
COMMON /COUNTS/ KNEW, KOLD, LINPAG, LISTLN, NUPDAT, NLOW, NHIGH.
C     REAL*8 CARD(10)
ELSE IF (LISTM.EQ.0) GO TO 3
ELSE IF (MOD(CHGLIN,LINPAG).NE.0) GO TO 4
CHGLIN = 0
CHPAGE = CHPAGE + 1
WRITE (CHANGE, 300) NUPDAT, CHPAGE
FORMAT('11',9X,'CHANGES TO OLD MASTER, FILE NO.','I4,T50,'PAGE',
C     I5, T94,'OLD NO.'/)
ELSE IF (BOD(CHGLIN,LINPAG).NE.0) GO TO 4
CHGLIN = CHGLIN + 1
IF (MODE.NE.0) GO TO 1
WRITE (CHANGE, 100) CARD
FORMAT(' INSERT ',10A8)
RETURN
1 WRITE (CHANGE, 200) CARD, MODE
FORMAT(' DELETE ',10A8,I10)
RETURN
END
SUBROUTINE SEQID(CARD)
IMPLICIT INTEGER*4 (A-Z)
COMMON /CQUEs/ COLUMN, INCR, SEQNO, IDCOL, IDLEN, IDLIT(68)
LOGICAL*1 IDLIT
LOGICAL*1 DIGITS(10) /'0123456789'/: CARD(80)
SEQNO = SEQNO + INCR
K = 80
N = SEQNO
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
CARD(K) = IDLIT(K+1-IDCOL)
RETURN
END
SUBROUTINE MATCH(CARD,*)
IMPLICIT INTEGER*4 (A-Z)
COMMON /PARMS/ LITLEN, LITCOL, MARK, N(3), LITRAL(68)
LOGICAL*1 LITRAL
LOGICAL*1 CARD(80)
INTEGER*4 DUM1 /0/, DUM2 /0/
LOGICAL*1 D1(4), D2(4)
EQUIVALENCE (D1(1),DUM1), (D2(1),DUM2)
DO 100 K = 1, LITLEN
D1(4) = CARD(K+LITCOL-1)
D2(4) = LITRAL(K)
IF (CARD(K+LITCOL-1).NE. LITRAL(K)) RETURN
IF (DUM1 .NE. DUM2) RETURN
100 CONTINUE
RETURN 1
END

C

SUBROUTINE SCAN(STRING, KLEFT, KRIGHT)
IMPLICIT INTEGER*4 (A-Z)
COMMON /PARMS/ LITLEN, LITCOL, MARK, N(3), LITRAL(68)
LOGICAL*1 LITRAL
INTEGER*4 ZIP/O/, ZEROCH/240/, QUOTE/125/
LOGICAL*1 G(4), STRING(80)
EQUIVALENCE (ZIP,G(1))
INTEGER*2 MSGA(9) /201,7/,NA(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'/
EQUIVALENCE (MSGA(3),MA(1)),(MSGB(3),MB(1)), (MSGC(3),MC(1)),
(CMSGD(3),MD(1)),(MSGF(3),MF(1))

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  START OF SCAN LOOP
1 G(4) = STRING(K)
C  HAVE A QUOTE
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  CHECK FOR PAIRED QUOTES
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 MOVE A CHARACTER INTO THE LITERAL STRING
60 LITLEN = LITLEN + 1
LITRAL(LITLEN) = STRING(K)
LITSW = 1
C NOTE HOW MANY INTEGERS OCCURRED BEFORE LITERAL
MARK = M - 1
GO TO 100
C HAVE A NON-NUMERIC, SEE IF IT ENDS A NUMBER
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 END OF SEARCH LOOP, INCREMENT INDEX
100 K = K + 1
C END OF SCAN
IF (BRANCH .NE. 0) GO TO 200
IF (N .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
FORMAT RELEASE 4 CONTROL CARDS

GROUP PAGE COLUMN LINE NO.

1 1 1 0
1 CONTROL CARD ENDS IN COLUMN 72
2 REPEAT TITLE ON EACH PAGE
3 CAPITALIZE AUTOMATICALLY
4 JUSTIFY AUTOMATICALLY
5 SENTENCES SEPARATED BY 2 BLANKS
6 LINES PER PAGE IS 58
7 TEXT STARTS ON LINE 6, PRINT POSITION 1
8 DARK PRINT 2 TIMES
9 BACKSPACE CHARACTER IS NUMBER 45
10 NULL CHARACTER SWITCH SET TO 2
11 INDENT (20,0)
12 NONTRIVIAL BLANK EQUIVALENT TO SPECIAL CHARACTER 46
13 CARD FIELD ENDS IN COLUMN 72
14 RIGHT PAGE NUMBER AT 1
15 TABS AT 50
16 WIDTH = 72
17 PAGE 1
18 GO

2 9 1 10 19
19 TABS AT 20, 28, 40, 50
20 GO

3 9 1 11 21
21 TABS AT 22, 31, 41, 51
22 GO

4 9 1 16 23
23 TABS AT 21 31 41 51
24 GO

00001800 00001900 00002000 00002100 00002200 00002300 00002400 00002500 00002600 00002700 00002800 00002900 00003000 00003100 00003200 00003300 00003400 000021400 000021500 000021700 000021800 000022020 000022040