U.S. GEOLOGICAL SURVEY

SAUDI ARABIAN MISSION

MISCELLANEOUS DOCUMENT 34

(INTERAGENCY REPORT 370)

REFBIB: A SYSTEM FOR THE STORAGE AND RETRIEVAL OF BIBLIOGRAPHIC DATA USED BY THE U. S. GEOLOGICAL SURVEY SAUDI ARABIAN MISSION, JIDDAH, SAUDI ARABIA.

by

G.I. Selner, M.E. Gettings, and B.M. North

U.S. Geological Survey
Open-File Report 81-826

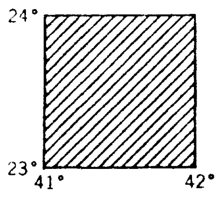
U.S. Geological Survey

Jiddah, Saudi Arabia

1981

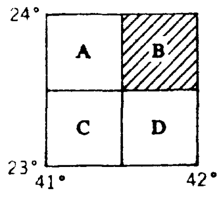
The work on which this report was based was performed in accordance with a cooperative agreement between the U.S. Geological Survey and the Ministry of Petroleum and Mineral Resources.

The quadrangle identification method used in U.S. Geological Survey Saudi Arabian Mission reports is shown below.



23/41

1-degree
quadrangle



23/41 B

30-minute
quadrangle

CONTENTS

Page

ILLUSTRATIONS

REFBIB:

A SYSTEM FOR THE STORAGE AND RETRIEVAL OF BIBLIOGRAPHIC DATA

USED BY THE U.S. GEOLOGICAL SURVEY SAUDI ARABIAN MISSION, JIDDAH, SAUDI ARABIA

By G.I. Selner, M.E. Gettings, and B.M. North

ABSTRACT

The U.S. Geological Survey Saudi Arabian Mission Reference/Bibliography System (REFBIB) consists of a series of computer programs for the storage and retrieval of bibliographic data. The system provides a mechanism for entering bibliographic data into a data set and allows for selective retrieval of all or part of this information. The bibliographic data includes author, title, reference, and keyword information for citations, all entered in a standard format. The system is capable of storing and updating citations in standard format and making retrievals that result in listings alphabetized by author. Retrievals may be done by logical expression(s) of author(s) or keyword(s).

INTRODUCTION

This report describes the current implementation of the U.S. Geological Survey Saudi Arabian Mission Reference/Bibliography System (REFBIB). The system is based on work done by M.E. Gettings and B.M. North in 1975. This version is based on a project defined by a committee of U.S. Geological Survey Saudi Arabian Mission (USGS) personnel in 1979. System design was completed by M.E. Gettings and G.I. Selner; programming and documentation were written by G.I. Selner.

The system design has been chosen so that bibliographic citations will appear in the format followed by the U.S. Geological Survey (Bishop and others, 1978).

The purpose of this report is to acquaint the reader with the REFBIB system, describe the interaction of the programs, and provide detailed instructions for the use of the various programs.

THE REFBIB DATA SET

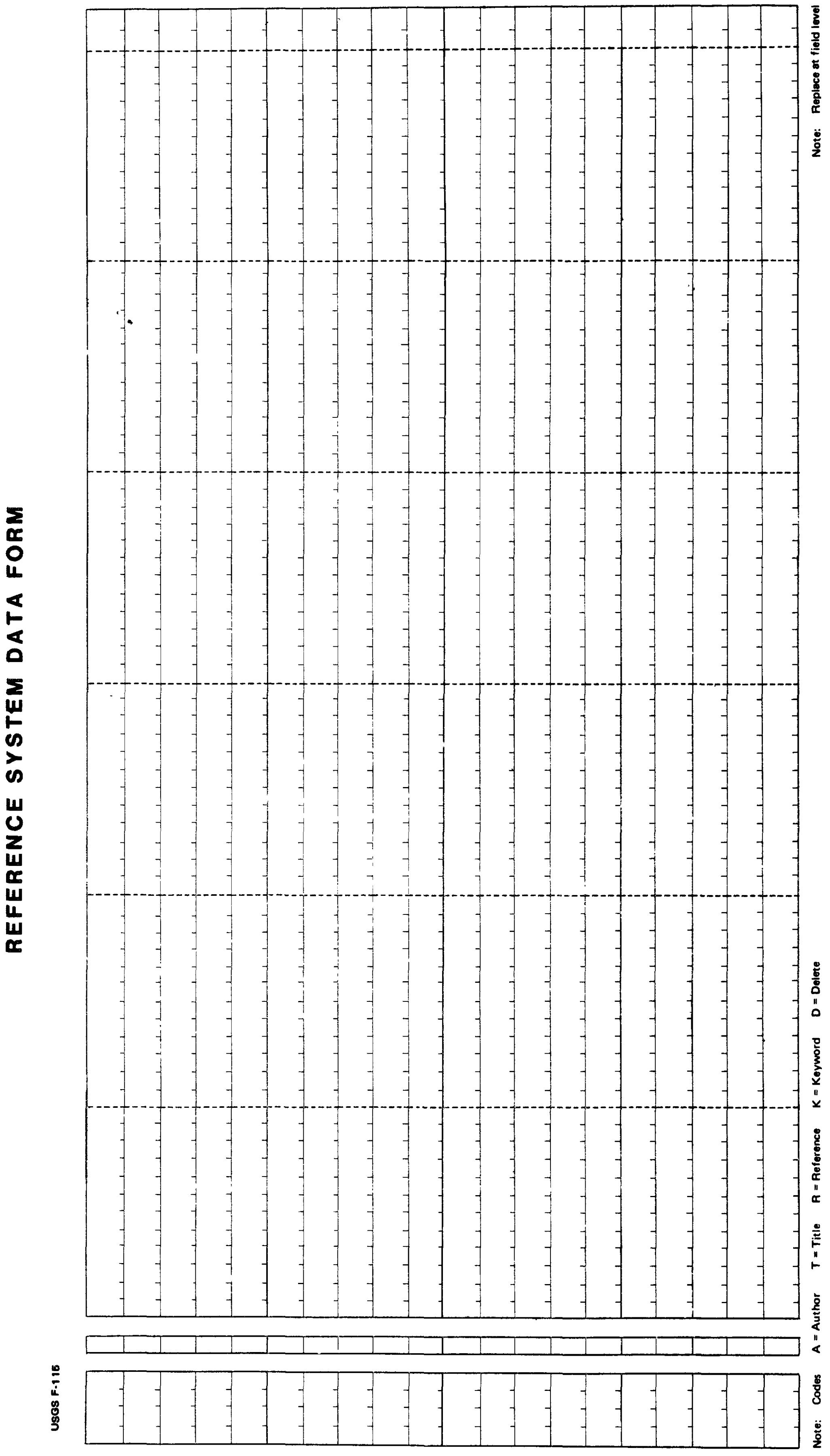
The REFBIB system is designed to store and retrieve bibliographic material. The data set, upon which storage and retrieval operations are performed, is a sequential master file of multiple entries, one entry for each bibliographic citation. Each entry consists of the following fields:

Reference number	- A four-digit integer assigned by the Data Set Administrator to uniquely identify one bibliographic citation.
Author	- The author(s) of the paper, abstract, bulletin, or entry cited.
Title	- The title of the paper, abstract, bulletin, or entry cited. The first five characters of the entry will be the year of publication and a comma. If the year of publication is unknown, the first four characters should be zeros, followed by a comma.
Reference	- Complete information needed to locate the citation, that is, publisher, place of publication, series, journal, volume, pagination, and so forth.
Keywords	- A group of alphanumeric terms that describe relevant subjects, geographic areas, and so on covered by the citation. The keywords used must be a part of the dictionary maintained by the Data Set Administrator. Keywords each contain a maximum of 12 characters; a maximum of 12 keywords is allowed per entry.

The data is entered onto a form designed for the system (fig. 1). The system utilizes this sample form for several purposes:

1. To enter a new citation into the data set by coding all the information for the citation onto the data form.

2. To change data for citations previously entered. Note that the entire field (author, title, reference, or keyword) to be changed must be entered.



REFERENCE SYSTEM DATA FORM

USGS F-115

Note: Codes A = Author T = Title R = Reference K = Keyword D = Delete

Note: Replace at field level

Figure 1.–Reference system data form.

3. To delete an existing citation from the data set by coding the reference number and a code "D".

The Data Set Administrator enters a citation onto the form according to the following rules:

1. The reference number must be repeated on all lines used to contain the citation. This number must be unique and cannot be used for any other citation.

2. Each field (author, title, reference, and keyword) must start on a new line, but can be continued onto other lines.

3. All lines must contain a code that specifies that the line is part of a field. The codes are "A" for author lines, "T" for title lines, "R" for reference lines, and "K" for keyword lines. Thus each line contains the reference number and a field code.

4. The complete citation must not exceed 35 lines.

It is extremely important that the data be coded in the sequence, author, title, reference, keyword, and that once keypunched, the data cards be maintained in this sequence.

After the data is coded and keypunched, the cards are stored until the next update of the data set is made. The cards do not have to be in sequence by reference number, but all cards for a given reference must be together and in the proper sequence.

The card format for keypunching is as follows:

Columns 1-4	Reference number, right-justified,
Column 6	Code A, T, R, K, or D,
Columns 7-80	Text of author, title, reference, or keyword field as appropriate. Keywords are 12 characters long and start in columns 7, 19, 31, 43, 55, and 67. A maximum of 12 keywords may be entered. The beginning of each keyword is indicated by a dashed vertical line on the form (fig. 1).

When the updating process is scheduled, the cards are entered onto a disk file, using the system utility program PIP to read the cards and place them on disk. The cards should be kept until the entire update processing is completed, at which time they may be discarded.

The data set is kept in two sequential data files. The first file is organized in ascending reference number sequence and is used for data set maintenance. The second file is organized in author sequence (alphabetically) and is used by the retrieval program.

Computer programs that provide the maintenance function (PDP530, PDP531, PDP532, and PDP533) and the retrieval function (PDP535) are described in separate sections of this report. Keyword indexes can be generated by programs PDP534 and PDP536. Utility programs for printing data sets (PDP537) and for renumbering data sets (PDP545) are also described.

RETRIEVAL CAPABILITIES

The REFBIB system provides the ability to retrieve listings of bibliographic citations from the data set by reference number, author, or keyword. In all cases the listing is produced in author sequence (alphabetically). The retrieval program permits the user to form logical expressions of authors or logical expressions of keywords. A logical expression consists of a prefix, a data value, and a connector.

Prefix	-	"FOR" specifies retrieval of citations that contain the data value as part of the author field or keyword field.
	-	"NOT" specifies retrieval of citations that do not contain the data value as part of the author field or keyword field.
Data value	-	A string of characters specifying an author name (maximum, 24 characters) or keyword (12 characters). Note that an author name can be of varying length while a keyword must be of fixed length.
Connectors	-	"AND" logically connects the prefix and data value on the current line with the prefix and data value on the following line. Thus both conditions must be true for selection.
	-	"OR" logically disconnects the prefix and data value on the current line with the prefix and data value on the following line.

The proper utilization of the prefix, data value, and connector fields permits the user to form more complex retrieval criteria. For example, a user wishes to select all citations that contain the keyword "SAUDI ARABIA" as a geographic term and that are concerned with geology in general. He also wishes to exclude any economic geology citations. The following set of logical expressions will produce the desired listing.

```
FOR        SAUDI ARABIA        AND
FOR        AREAL GEOL          AND
NOT        ECON GEOL
```

Note that the selection is based on keywords that were included as part of the original entry of the citation and is not based on the occurrence of a specified string of characters appearing in the citation as part of the title or reference lines. Examples of retrievals using three types of selection--author, keyword, and reference number--are given in the program documentation for PDP535.

The REFBIB system also allows the user to save selected citations on temporary data sets. Thus the user can make a retrieval by keyword and then further select from that data set either by author, keyword, or reference number, or the user can merge separate retrieval results into a single data set for the purposes of printing.

Complete details for using the retrieval program are given in the documentation of program PDP535.

PROGRAM DOCUMENTATION

The REFBIB system consists of nine programs written in FORTRAN IV. The relationships between programs and data files are illustrated in figure 2.

All programs are executed by a user responding to prompting questions while working at a terminal connected to the host computer system. The individual program documentation specifies the manner in which the program is executed, the questions that are asked, and the responses executed. Each program documentation provides examples of the terminal sessions that take place when the programs are used.

PDP530

This program edits any file that is in the REFBIB card format. The user may print the entire input file as well as any error messages, or print only the cards in error and the error messages.

Description

Multiple passes are made through the specified input file. If any errors occur on a pass through the data, then the subsequent passes are not executed. The purpose of each pass is as follows:

Pass One

1. Checks columns 1-4 to be right justified with no imbedded blanks.
2. Checks columns 1-4 for valid numeric character: b, 0, 1, 2, 3, 4, ..., 9.
3. Checks column 5 to be blank.
4. Checks column 6 for valid code: A, T, R, K, D.

Pass Two

Checks that cards for the same reference number are in correct sequence by codes A, T, R, K.

Pass Three

Checks for multiple entries with the same reference number.

Pass Four

Checks each keyword against the list of valid keywords in the file named MASKEY.WRD.

Input data preparation

The input data must be coded, keypunched, and entered onto disk as previously described. This program uses a set of valid keywords that must have been established in a data file named MASKEY.WRD. Normally this file is prepared by using the system utility editor program. The file consists of multiple records, each containing one 12-character keyword in columns 1 through 12.

Example

A user has coded a series of changes for references that exist in a REFBIB data set. The cards have been keypunched and loaded onto disk using the system utility program PIP. The user wishes to check the update cards for errors prior to updating the REFBIB data set. Execution of the program is as follows:

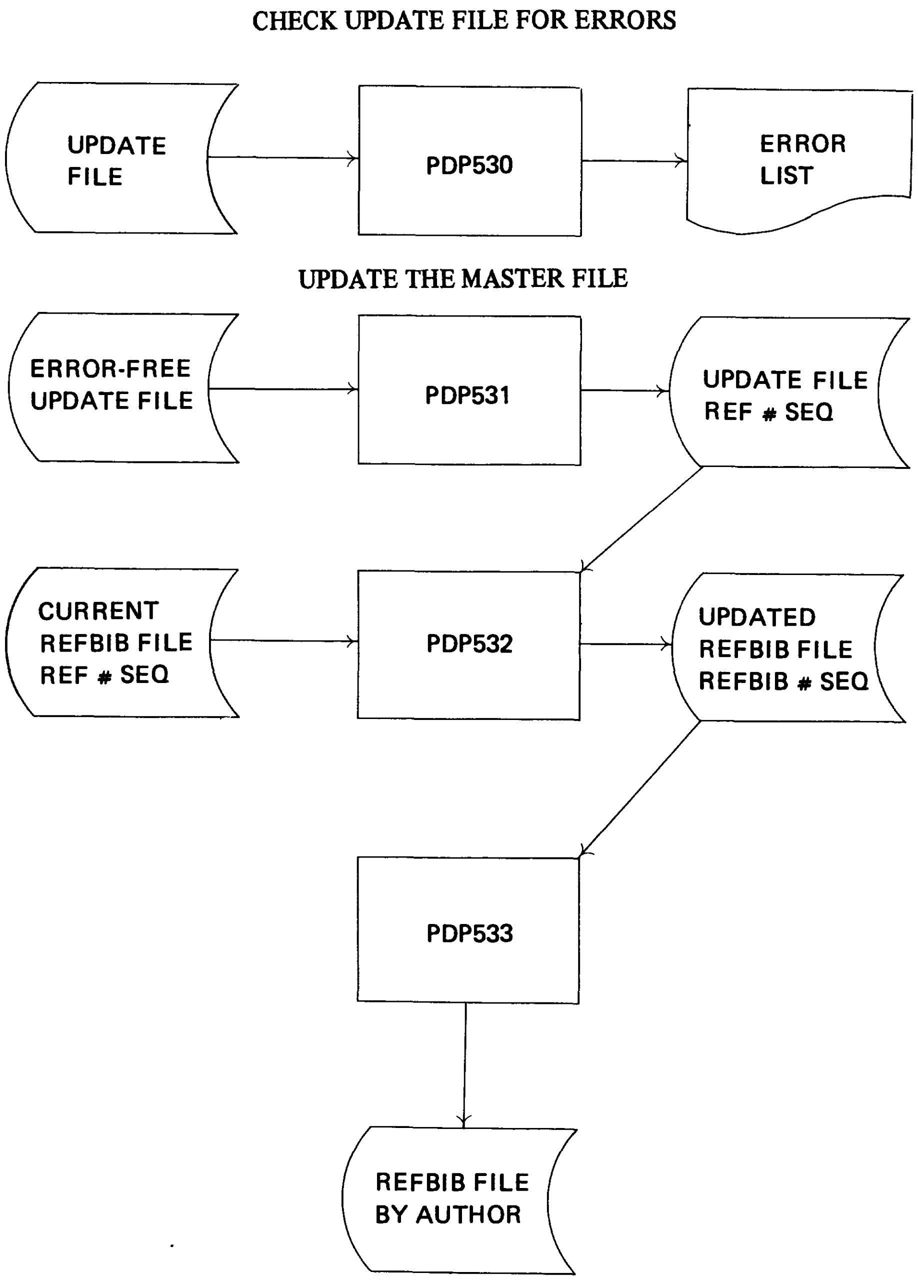
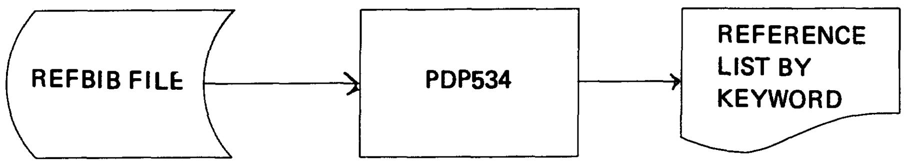
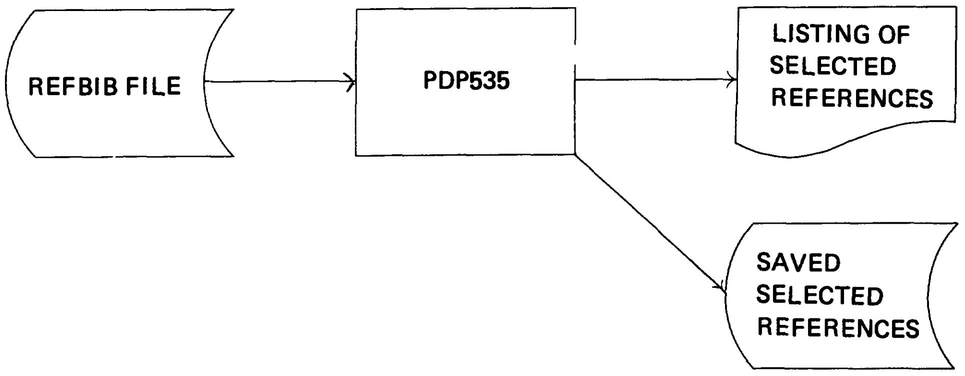


Figure 2.–Relationship of programs and data files.

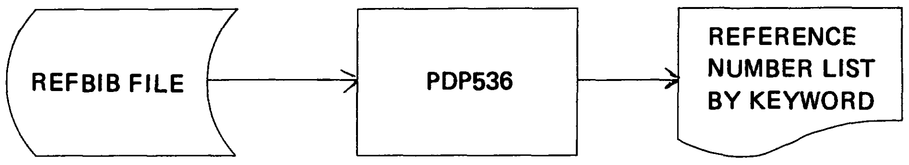
GENERATE REFERENCE LIST BY KEYWORD FOR AN ENTIRE FILE



SELECTION BY AUTHOR, KEYWORD, OR REFERENCE NUMBER



GENERATE REFERENCE LIST BY KEYWORD



RENUMBER AN ENTIRE FILE

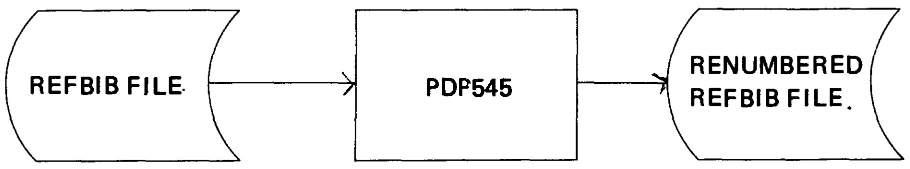


Figure 2.–Continued

```
MCR>RUN $PDP530@[1]

ENTER INPUT FILENAME:UPDATE.CRD

UPDATE.CRD

OK?Y

ENTER FILENAME FOR ERRORS:TT1:

TT1:

OK?Y

PRINT ERRORS ONLY (Y/N):Y

PDP530 -- STOP

↑C[2]

MCR>
```

The error messages (if any) have been directed to the terminal named TT1:. Only the cards in error and the appropriate error messages will be printed.

PDP531

This program reads as input all update data for a REFBIB data set and creates as output a file sorted by reference number to be used by the main update program PDP532.

Description

This program normally is used only when the REFBIB data set master file is to be updated. All information to be added to the master file, changed in the master file, or deleted from the master file is coded on the Reference System Data Form (fig. 1). The information on this form is then keypunched and the card data entered onto a disk file using the system utility program PIP.

[1] The character @ indicates the ALTMODE or ESCape key on the terminal. Underlining indicates user responses.
[2] The character ↑C indicates the control C key combination on the terminal.

The user is first prompted for the filename of the input data file and a filename for the output file, which will be sorted by reference number. This output file will be used as an input file by the main update program (PDP532). After the output file is sorted properly, PDP531--STOP is typed on the terminal and the program terminates. Normally the user would then continue the file maintenance process by executing program PDP532. The documentation of this program follows in the next section.

Input data preparation

The input for this program is the same data file that was created on disk using the system utility program PIP and checked by program PDP530.

Example

```
MCR>RUN $PDP531@

ENTER INPUT FILENAME:DATA.NEW

DATA.NEW

OK?Y

ENTER OUTPUT FILENAME:DATA.UPD

DATA.UPD

OK?Y

PDP531--STOP

↑C

MCR>
```

PDP532

This program performs the following maintenance functions in a REFBIB data set: addition of new citations to the data set, changing of a field(s) in existing citations, and deletion of existing citations.

Description

This program is used to update a REFBIB data set by the addition of new citations, the changing of existing citations at the field(s) level, or the deletion of existing citations. The user is first prompted for the filename of an update file, which has been sorted in reference number sequence.

This file was generated by program PDP531. The filename of the REFBIB data set is then requested. Note that this data set is sorted in reference number sequence. The user is then requested to supply a filename for the output file, a new updated REFBIB data set.

An entry is then read from the update file. The entry consists of all information specified about a citation with a given reference number. This information can consist of 1) a reference number and a delete code, 2) a reference number and the data for a specific field, or 3) a reference number and the data for all fields. The current REFBIB data set is then searched for the citation with the specified reference number. If such a citation is found, a check is made for a delete code in the update data. If such a code exists, the entire citation is deleted from the new REFBIB data set. If the deletion code is not specified, the appropriate fields are replaced with the data from the update file. However, if no citation with the specified reference number is found on the current REFBIB data set, then the data from the update file for the specified reference number is added to the new REFBIB data set. This cycle is repeated until all data from the update file and current REFBIB data set have been processed.

Input data preparation

The input to this program is a data file sorted by reference number and the current REFBIB data set sorted in ascending reference number sequence.

Example

```
MCR>RUN $PDP532@

ENTER FILENAME FOR UPDATE DATA:DATA.UPD

DATA.UPD

OK?Y

ENTER FILENAME FOR MASTER FILE:DATA.REF

DATA.REF

OK?Y

ENTER FILENAME FOR NEW MASTER:NEWEST.REF

NEWEST.REF

OK?Y
```

```
PDP532--STOP

↑C

MCR>
```

PDP533

This program reads in a REFBIB data set that is sorted in reference number sequence and creates a REFBIB data set that is sorted in author (alphabetical) sequence. It also produces a listing that contains all citations in author sequence.

Description

The purpose of this program is to create a REFBIB data file that is sorted by author and to produce a listing of all citations sorted by author. The first prompt by the program is for the filename of the REFBIB data set that is sorted in reference number sequence. Normally this data set has been created by running programs PDP531 and PDP532. The second prompt is for the filename of the author-sequenced output file.

The input data file is read in, one reference at a time, and a scratch file is created on disk; each record contains a 34-character key for sorting. This key is formed by taking one character at a time from the author field and discarding all special characters such as apostrophes, dashes, commas, blanks, and periods. After 30 characters have been accumulated, four characters are appended containing the year of publication.

The scratch file is then sorted using the key. The sorted scratch file is read, the keys are stripped off, and the REFBIB data file is created in author sequence. At the same time a listing file is created that can be directed to the line printer.

Input data preparation

Normally the input data file is created by executing programs PDP531 and PDP532.

Example

```
MCR>RUN $PDP533@

ENTER INPUT FILENAME:NEWEST.REF

NEWEST.REF
```

```
OK?Y

ENTER OUTPUT FILENAME:NEWEST.AUT

NEWEST.AUT
OK?Y
PDP533--STOP

↑C

MCR>
```

PDP534

This program creates a printer listing that contains all references for each keyword. The keyword lists are sorted in alphabetical sequence and each list of references is in author sequence (alphabetical).

Description

The user is first prompted for the filename of the current REFBIB data file. This file is in alphabetical sequence by author. The user is then requested to enter a filename for the printer output. A record is read from the input file and multiple records are written to a disk scratch file. A record (for each nonblank keyword) is created that contains a 12-character keyword, a four-digit sequential input record number, and the entire text for the reference. The four-digit sequential record number is necessary only to preserve the alphabetical sequence by author in the input file. After all input records are processed, the disk scratch file is sorted by the first 16 characters of the record. The sorted file is then processed and a printer listing is created. This listing contains a keyword followed by all references containing that keyword in alphabetical sequence by author. The listing for a keyword always starts on a new page and a citation is always fully contained on a page. When the sorted file is completely processed, the disk scratch file is deleted and PDP534--STOP is typed on the terminal. The user must request that the printer listing file be printed by issuing a QUE (system level) command.

Input data preparation

The input to this program is the REFBIB data set that contains all citations in alphabetical sequence by author.

Example

```
MCR>RUN $PDP534@

ENTER INPUT FILENAME:NEWEST.AUT

NEWEST.AUT

OK?Y

ENTER FILENAME (OR DEVICE) FOR PRINTED OUTPUT:

                                            KEYWD.PRT

KEYWD.PRT

OK?Y

PDP534--STOP

↑C

MCR>
```

PDP535

This program allows the user to select bibliographic citations from the REFBIB data set. All selection criteria result in listing of citations in alphabetical sequence by author. The selection criteria can be one of the following: logical expression of author name(s); logical expression of keyword(s); or selection by reference number(s). The program also provides the ability to save selections for further refinement and(or) combination with other selections.

Description

This program provides the user with the ability to retrieve bibliographic citations from the REFBIB data set. Each execution of the program can utilize only one of the three methods of retrieval. Therefore, a series of questions are first asked by the program regarding the method of selection. After an affirmative response (Y=yes), questions regarding the input file are asked and specific information about the selection criteria requested.

The program is designed to be tolerant to user typing mistakes. Each response to a prompt is retyped and the user has the option to confirm it as correct or incorrect, in which case the question is repeated. While the experienced user may find this confirmation annoying, he should be aware that incorrect answers can result in a useless retrieval.

The three selection methods are:

1. selection by logical expression of author,

2. selection by logical expression of keyword, or

3. selection by reference number.

Each of these will be discussed in detail. The two methods of retrieval by logical expression use a common methodology and will be discussed first.

The expression of logical values uses three fields: a prefix, a data value, and a connector. The prefix specifies the positive equal value by using the word FOR, the negative not equal value by using the word NOT. The data value is the alphabetic string to be tested. For author selection, the data value can be from 1 to 24 characters. For keyword selection, the data value has a fixed length of 12 characters. The connector is used to group logical expressions into clauses. The connector is either AND or OR. A clause is a group of logical expressions, all of which must be true for selection or rejection of a citation. The AND allows the user to specify multiple expressions to form a clause; the OR allows the user to specify the beginning of a new clause and the end of the previous clause. If any one of the specified clauses results in all logical expressions within that clause being true, the citation is selected. If any one of the logical expressions specified in a clause is false, the entire clause is false. If all clauses result in false, the citation is rejected. A detailed discussion of each type of selection method will provide examples that should help clear up any confusion.

Selection by author.-- Selection by author produces a list of all citations in which the author is included, either as sole author or one of the co-authors. At this point it is useful to review the content and format of the author field in the citations stored in the REFBIB data set. The last name of an author is followed by a comma, a space (b), and one or two initials. Each initial is followed by a period. If the citation was written by multiple authors, then the period after the last initial is followed by a comma and a space. If the next author is the last one, then the word AND and a space are followed by the last author's name. For purposes of illustration, several examples are given:

```
ABA-HUSAYN,bM.M.,bANDbSAYEGH,bA.H.
ABBAS,bH.L.
AKAAD,bM.K.,bEL-GABY,bS.,bANDbATTAS,bA.
```

Of course, the selection process is not limited to a single author. For instance, the user may wish to see all citations that two particular people wrote in collaboration or perhaps all citations by one particular author, rejecting all citations that were co-authored by a second particular author. This can best be explained by the following examples:

Example 1
A selection is made of all citations by D.G. HADLEY excluding those citations that were co-authored by D. SCHMIDT (fig. 3).

Example 2
A selection is made of all citations that were co-authored by M.E. GETTINGS and H.R. BLANK (fig. 4).

The manner in which logical expressions are entered is very important. The sequence is prefix followed by a tab character, then data value followed by a tab character, and then the connector. On the last line the connector can be omitted, but the second tab must be present. Each logical expression is entered on a separate line ending with a carriage-return character. After the last line is entered, a second carriage return indicates the end of the selection criteria. Note that the test conditions are retyped by the program, grouping logical clauses, and the user is requested to assert their correctness.

After the selection process is completed, the number of citations selected is typed by the program on the terminal and the user is asked whether he wishes to print them. If the response is affirmative, a standard listing is generated. The user is given the opportunity to direct the list to his terminal (TI:), the system line printer (LP:), or to a disk file for printing later.

The user is then asked if he wishes to save the selections. If he wishes to do further selections on this file, he should respond Y; if not, he should answer N. If the answer is N, the file is deleted.

Selection by keyword.-- The first point that must be made is that a keyword is a member of an arbitrarily chosen set of terms that define items of interest. The type of bibliographic citations stored in a REFBIB data set will somewhat define the set of terms or keywords. The selection of terms that comprise the dictionary of keywords relevant to a data set is the responsibility of the Data Set Administrator. Once keywords are chosen each citation that is encoded is entered with those keywords considered relevant to the article described by the citation. The relative merit

```
MCR>
MCR>HEL [72,340]
MCR>RUN $PDP535$

------------------------------------------------------------------------


 KEYWORD SELECTION(Y/N)? : N
N
OK ?
 REFERENCE # SELECTION(Y/N)? : N
N
OK ?
 SELECTION BY AUTHOR(Y/N)? : Y
Y
OK ?
 STANDARD MASTER FILE(Y OR N)? : Y
Y
OK ?
 ENTER FILENAME FOR SELECTED SUBSET : EXAMPLE1.AUT
EXAMPLE1.AUT
OK ?
PREFIX  VALUE                   CONNECTOR
FOR     HADLEY, D.G.    AND
NOT     SCHMIDT, D.

YOU HAVE SPECIFIED ON    2 LINES OF INPUT
THE FOLLOWING TESTS:
FOR       HADLEY, D.G.                 AND
NOT       SCHMIDT, D.

THIS CONSTITUTES    1 TEST CLAUSE(S)
OK?Y
PROCESSING IN PROGRESS; PLEASE WAIT FOR FURTHER PROMPTING
  26 REFERENCES SELECTED.
 DO YOU WISH TO PRINT THE SELECTIONS(Y/N)? : N
N
OK ?
 SAVE FILE OF SELECTED SUBSET(Y/N)? : Y
Y
OK ?
END OF SELECTION RUN


------------------------------------------------------------------------


PDP535  --  STOP

MCR>
```

Figure 3.–Selection by author, example 1.

```
    RUN $PDP535$

--------------------------------------------------------------------------------

 KEYWORD SELECTION(Y/N)? : N
N
OK ?
 REFERENCE # SELECTION(Y/N)? : N
N
OK ?
 SELECTION BY AUTHOR(Y/N)? : Y
Y
OK ?
 STANDARD MASTER FILE(Y OR N)? : Y
Y
OK ?
 ENTER FILENAME FOR SELECTED SUBSET : EXAMPLE2.AUT
EXAMPLE2.AUT
OK ?
PREFIX  VALUE                        CONNECTOR
FOR     GETTINGS, M.E.   AND
FOR     BLANK, H.R.

YOU HAVE SPECIFIED ON    2 LINES OF INPUT
THE FOLLOWING TESTS:
FOR        GETTINGS, M.E.                AND
FOR        BLANK, H.R.

THIS CONSTITUTES     1 TEST CLAUSE(S)
OK?Y
PROCESSING IN PROGRESS; PLEASE WAIT FOR FURTHER PROMPTING
   3 REFERENCES SELECTED.
 DO YOU WISH TO PRINT THE SELECTIONS(Y/N)? : N
N
OK ?
 SAVE FILE OF SELECTED SUBSET(Y/N)? : Y
Y
OK ?
END OF SELECTION RUN

    ----------------------------------------------------------------------------

PDP535  --  STOP

MCR>
```

Figure 4.–Selection by author, example 2.

of the dictionary and the choice of keywords for each citation are extremely important since they determine the usefulness of the data set to the researcher.

The U.S. Geological Survey Saudi Arabian Mission has established a dictionary of keywords that are relevant to a data set containing citations from the earth sciences (Appendix 1). The dictionary contains geographic and political terms as well as geoscience terms. In addition, the dictionary includes general terms that are further separated into specific terms. A user attempting to select citations based on keywords should become familiar with the dictionary.

The specification of selection criteria involves the use of logical expressions that may be grouped into clauses. A logical expression consists of a prefix (FOR or NOT), a data value (in this case a keyword), and a connector. Since these terms were explained in the section on retrieval capabilities, an attempt will now be made to illustrate their usage by citing two examples:

Example 1
A selection is made of all citations that contain the keywords EGYPT and COAL (fig. 5).

Example 2
A selection is made of all citations that contain the keywords SAUDI ARABIA and ECON GEOL, but excludes those that contain the keyword NONMETALS (fig. 6).

The manner in which the logical expressions are entered is the same as for selection by author. The sequence is prefix (FOR/NOT), tab character, keyword, tab character, connector (AND/OR), and carriage return. A second carriage return after the last tab character indicates the end of the selection criteria. Again, the test conditions are retyped by the program, grouping logical clauses, and the user is requested to assert their correctness.

After the selection process is completed, the procedure is identical to the selection by author process described above. The user is given the opportunity to print the selected citations and to save the selections on a disk file.

Selection by reference number.-- This method of selection is very different from selection by author and selection by keyword. Those processes assume that it is not known which references are to be selected. Selection by reference number is specific as to reference. The user is asked to enter a series of reference numbers that identify specific citations. Usually the purpose of such a retrieval is to create a list of citations for inclusion with a draft of a manuscript.

```
    RU^U
MCR>




    RUN $PDP535$

-------------------------------------------------------------------------------


 KEYWORD SELECTION(Y/N)? : Y
Y
OK ?
 STANDARD MASTER FILE(Y OR N)? : Y
Y
OK ?
 ENTER FILENAME FOR SELECTED SUBSET : EXAMPLE1.KEY
EXAMPLE1.KEY
OK ?
PREFIX  VALUE            CONNECTOR
FOR     EGYPT   AND
FOR     COL\L\AL

YOU HAVE SPECIFIED ON     2 LINES OF INPUT
THE FOLLOWING TESTS:
FOR       EGYPT               AND
FOR       COAL

THIS CONSTITUTES     1 TEST CLAUSE(S)
OK?Y
PROCESSING IN PROGRESS; PLEASE WAIT FOR FURTHER PROMPTING
   3 REFERENCES SELECTED.
 DO YOU WISH TO PRINT THE SELECTIONS(Y/N)? : N
N
OK ?
 SAVE FILE OF SELECTED SUBSET(Y/N)? : Y
Y
OK ?
END OF SELECTION RUN


-------------------------------------------------------------------------------


PDP535   --   STOP

MCR>
```

Figure 5.—Selection by keyword, example 1.

```
    RUN $PDP535$

------------------------------------------------------------------------

 KEYWORD SELECTION(Y/N)? : Y
Y
OK ?
 STANDARD MASTER FILE(Y OR N)? : Y
Y
OK ?
 ENTER FILENAME FOR SELECTED SUBSET : EXAMPLE2.KEY
EXAMPLE2.KEY
OK ?
PREFIX  VALUE            CONNECTOR
FRO\ORF\FOR     SAUDI ARABIA    AND
FOR     ECON GEOL        AND
NOT     NONMETALS

YOU HAVE SPECIFIED ON    3 LINES OF INPUT
THE FOLLOWING TESTS:
FOR        SAUDI ARABIA      AND
FOR        ECON GEOL         AND
NOT        NONMETALS

THIS CONSTITUTES    1 TEST CLAUSE(S)
OK?Y
PROCESSING IN PROGRESS; PLEASE WAIT FOR FURTHER PROMPTING
 852 REFERENCES SELECTED.
 DO YOU WISH TO PRINT THE SELECTIONS(Y/N)? : N
N
OK ?
 SAVE FILE OF SELECTED SUBSET(Y/N)? : Y
Y
OK ?
END OF SELECTION RUN

------------------------------------------------------------------------

PDP535  --  STOP

MCR>
```

Figure 6.—Selection by keyword, example 2.

Since the REFBIB data set is sorted in author sequence, the reference numbers to be selected will be entered in sequence as they occur in the data set. Up to 250 citations may be selected using this method.

Figure 7 illustrates the selection of three citations. Confirmation of each entry is requested as it is entered. Note that the final carriage return to terminate the entry list results in a zero reference number. The user should respond affirmatively when questioned as to its correctness. The zero reference number signifies the end of the specification list.

As in the other selection processes, the user is given the opportunity to print the references selected and to save the data set selected on disk.

PDP536

This program prepares a listing of all keywords referenced in a REFBIB data set. The program sorts the keywords into alphabetic sequence and prints out the reference number of all references that contain the keyword. The sequence of the input file may be either by author or by reference number. The output list of references for each keyword is in ascending reference number sequence.

Example

```
MCR>RUN $PDP536@

ENTER INPUT FILENAME: DB1:KEYWD.REF

ENTER FILENAME FOR PRINTER OUTPUT: KEYWD.IND

PDP536--STOP

↑C

MCR>
```

In this example the user has created a file that contains all the keywords referenced in the REFBIB data set named KEYWD.REF and located on disk DB1:. The output listing is stored on the system disk (SY:) under filename KEYWD.IND. To print this file, use one of the following commands:

```
MCR>PIP TT1:=KEYWD.IND  or

MCR>QUE KEYWD.IND.
MCR>



    RUN $PDP535$

--------------------------------------------------------------------------------


 KEYWORD SELECTION(Y/N)? : N
N
OK ?
 REFERENCE # SELECTION(Y/N)? : Y
Y
OK ?
 STANDARD MASTER FILE(Y OR N)? : Y
Y
OK ?
 ENTER FILENAME FOR SELECTED SUBSET : EXAMPLE1.REF
EXAMPLE1.REF
OK ?
PLEASE ENTER REF NOS ONE AT A TIME UPON PROMPT(:). END BY REF # 0000
  : 870
VALUE :                  870 , OK ?
  : 874
VALUE :                  874 , OK ?
  : 875
VALUE :                  875 , OK ?
  :
VALUE :                    0 , OK ?
PROCESSING IN PROGRESS: PLEASE WAIT FOR FURTHER PROMPTING
   3 REFERENCES SELECTED.
 DO YOU WISH TO PRINT THE SELECTIONS(Y/N)? : N
N
OK ?
 SAVE FILE OF SELECTED SUBSET(Y/N)? : Y
Y
OK ?
END OF SELECTION RUN


--------------------------------------------------------------------------------


PDP535  --  STOP
```

Figure 7.–Selection by reference number, example.

PDP537

This program creates an output file that lists the contents of a user-specified input file. The sequence of the input file is irrelevant, except that all records for a reference must be in the REFBIB format.

Example

```
MCR>RUN $PDP537@

ENTER INPUT FILENAME:DB1:KEYWD.REF

ENTER FILENAME OR DEV:  FOR PRINTED OUTPUT:  TI:

ENTER TITLE (80 characters maximum)

PDP537 EXAMPLE
```

Since the user has directed the output to his terminal, the listing will appear immediately on the terminal. The title appears on each page. When the listing is completed, the following line will appear on the terminal.

```
PDP537--STOP

↑C

MCR>
```

PDP545

This program renumbers citations in a REFBIB data set, beginning with 1 and incrementing by 1. The input file can be in any sequence, except that all records for a reference must be in the REFBIB format.

Example

```
MCR>RUN $PDP545@

ENTER FILENAME:DB1:KEYWD.REF

PDP545--STOP

↑C

MCR>
```

The file KEYWD.REF is now completely renumbered, starting with the number 1 for the first reference and incrementing each reference by 1.

REFERENCE CITED

Bishop, E. E., Eckel, E. B., and others, 1978, Suggestions to authors of the reports of the United States Geological Survey (6th ed.): Washington, D.C., U.S. Geological Survey, 273 p.

Appendix 1.—*Keyword dictionary*

KEYWORD LIST 1. GEOGRAPHIC TERMS

- AFRICA
    - NUBIAN SHIEL
    - RIFT VALLEY
- ARABIAN PENI
- ARABIAN GULF (PERSIAN GULF)
- ASIA
- BAHRAIN
- DEAD SEA
- DJIBOUTI (T.F.A.I.)
- EGYPT
    - EAST DESERT
    - LIBYAN DESER
    - NILE
    - SINAI
- ETHIOPIA (INCLUDES ERITREA)
    - AFAR
- INDIA
- INDIAN OCEAN
    - ARABIAN SEA
    - GULF OF ADEN
    - GULF OF OMAN
- IRAN
- IRAQ
- ISRAEL
- JORDAN
- KUWAIT
- KENYA
- LEBANON
- MEDITERRANEA (MEDITERRANEAN SEA)
- MIDDLE EAST
- OMAN
- PAKISTAN
- PALESTINE
- QATAR
- RED SEA
    - GULF AQABA
    - KARAMAN ISL
    - PERIM ISL
- SAUDI ARABIA
    - ARAB SHIEL
    - ASIR
    - EAST PROVINC
    - NAJD
    - HIJAZ
    - TIHAMA
    - RUB AL KHALI
    - I-SERIES MAP (I200-I220)
    - QUADRANGLE CODE (FOR EXAMPLE, 19/43)
    - OTHER GEOGRAPHIC NAMES (KEYWORD LIST 5)
- SOMALIA
- SYRIA
- SUDAN
- TANZANIA
- TURKEY
- UAE
- UGANDA
- YEMEN (INCLUDES NORTH AND SOUTH YEMEN)
    - PERIM ISL
    - KARAMAN ISL

KEYWORD LIST 2. SUBJECT CLASSIFICATIONS

- ARCHAEOLOGY
- AREAL GEOL
- BIBLIOGRAPHY
- BIOLOGY
- COMPUTING
  - ADM DP
  - SCIENTIF DP
- DRILLING
- ECON GEOL
  - METALS
    - ANY COMMODITY TERM FROM KEYWORD LIST 3
  - NONMETALS
    - ANY COMMODITY TERM FROM KEYWORD LIST 3
  - ENERGY RES
    - GEOTHERMAL
    - COAL
    - PETROLEUM
    - NUCLEAR
  - MINERALIZATI
  - ANCIENT MINE
  - MINING
- ENGIN GEOL
- ENVIRON GEOL
- GEOCHEM
  - GEOCHEM SUR
  - TRACE ELEM
  - ISOTOPES
- GEOCHRON (USED FOR RADIOMETRIC DATES ONLY)
- GEOGRAPHY
- GEOMORPH
  - SABKHAH/PLA
  - EOLIAN FEAT (INCLUDES DUNES)
  - TERRACES
  - GEOMORPH PROC
  - DRAINAGE SYS
- GEOPHYSICS
  - AREAL GEOPH
  - SEISMOLOGY
  - EXPLOR GEOPH
  - GRAVITY
  - ELECTROMAG
  - MAGNETICS
  - RADIOMETRICS
  - HEAT FLOW
  - PALEOMAG
  - PHY PROP RX
- HIST GEOL
- HYDROGEOLOGY (INCLUDES HYDROLOGY)
- OCEANOGRAPHY
  - MARINE GEOL
  - HOT BRINES
  - OCEAN FLOOR
  - CONT SHELF

```
     REEFS
     SEA H20 PROP (CHEMICAL AND PHYSICAL PROPERTIES, INCLUDING SEA LEVEL CHANGES)
     MARINE LITH
METEORITES
METEOROLOGY
MISCELLANEOU
     ADM REPORTS
MINERALOGY
OPHIOLITES
PALEONTOLOGY
     BOTANY
     INVERTEBRATE
     VERTEBRATE
PETROL NSED
     IGNEOUS RX
        FELSIC IG RX
        MAFIC IG RX
     METAMOR RX
     METASOMATISM
     METAMORPHISM
PETROL SED
     EVAPO/BRINES
     NONCARB RX (EXCEPT EVAPORITES)
     CARBONATE RX (CAN INCLUDE REEFS)
     SEDIMENTS
     LACUST DEP
REMOTE SENS (INCLUDES PHOTOGEOLOGY)
STRAT/SED (INCLUDES FOSSIL DATA REFERENCES)
     PALEOCLIMATE
     PALEOGEOGR
     ANY GEOLOGIC AGE TERM FROM KEYWORD LIST 4
STRUCT GEOL
     FAULTS
     FOLDS
     DEFORMATION
SURFIC GEOL (INCLUDES SOILS AND WEATHERING)
TECT/TECTOPH
     PLATE TECT
     OROGENESIS (INCLUDES UPLIFT)
     STS-STN REG (STRESS-STRAIN REGIME)
TOPO/CARTO (INCLUDES AIR PHOTOGRAPHS, INDEXES)
VOLCANOLOGY
```

KEYWORD LIST 3. COMMODITY TERMS

ALUMINUM
ANTIMONY
ARSENIC
ASBESTOS
BARITE
BAUXITE
BENTONITE
BERYLLIUM
BISMUTH
BITUMENS (INCLUDES ASPHALT, OIL SANDS)
CEMENT MAT
CERAMIC MAT
CHROMITE
CLAYS
COAL
COBALT

CONSTRUCT MAT (BUILDING STONE, AGGREGATE, SAND AND GRAVEL)
COPPER
DIAMONDS
DIATOMITE
ENERGY RES
EVAPO/BRINES
FELDSPAR
FLUORITE
GEMS
GEOTHERMAL
GLAUCONITE
GOLD
GRANITE
GRAPHITE
GYPSUM (INCLUBES ANHYDRITE)
HEAVY MIN
IRON
KAOLIN
LEAD
LIMESTONE
LITHIUM

MAGNESITE
MANGANESE
MARBLE
MERCURY
METALS
MICA
MOLYBDENUM
NICKEL
NIOBIUM
NONMETALS (INCLUDES INDUSTRIAL MINERALS)
NUCLEAR (RESOURCES)
PEAT
PEGMATITE
PERLITE
PETROLEUM
PHOSPHATE
PLATINUM
POLYMET ORES
POTASH
PUMICE
PYRITE
RARE EARTHS
SALT
SAND (SAND AND GRAVEL UNDER CONSTRUCT MAT)
SILVER
SLATE
SULFUR
TALC (SOAPSTONE)
TANTALUM
THORIUM
TIN
TITANIUM
TUNGSTEN
URANIUM
VANADIUM
VERMICULITE
WATER RES
ZEOLITES
ZINC

KEYWORD LIST 4. GEOLOGIC AGE TERMS

ARCHEAN
CAMBRIAN
CARBONIFER
CENOZOIC
CRETACEOUS
DEVONIAN
EOCENE
HOLOCENE
JURASSIC
MESOZOIC
MIOCENE
MISSISSIPPI
NEOGENE
OLIGOCENE
ORDOVICIAN
PALEOCENE
PALEOGENE
PALEOZOIC
PENNSYLVANI
PERMIAN
PHANEROZOIC
PLEISTOCENE
PLIOCENE
PRECAMBRIAN
PROTEROZOIC
QUATERNARY
SILURIAN
TERTIARY
TRIASSIC

KEYWORD LIST 5. SAUDI ARABIAN PLACE NAMES

ABA AL QAZAZ	26/36
ABHA	18/42
ABLAH	20/41
ABU BIER	19/41
ABU RAQAH	27/37
AD DAFINAH	23/41
AD DAMMAM	26/50
AD DARAH	23/43
AD DAWADIMI	24/44
AD DIBDIBAH	28/45
AD DUGHUM	24/47
AFAIYAH	25/41
AFIF	23/42
AL AHSA	25/49
AL AMAR	23,24/45
AL AQIQ	20/41
AL AYS	25/38
AL BAD	28/35
AL HADA	21/39
AL HADIDAH	21/50
AL HAJIRA	18/43
AL HANAKIYAH	24/40
AL HUFUF	25/49
AL JIZL	26/37
AL JUBAYL	27/49
AL JUNAYNAH	20/42
AL KHARJ	24/47
AL LIDAM	26/49
AL LITH	20/40
AL MADINAH	24/39
AL MUWAYH	22/41
AL MUWAYLIH	27/35
AL QASIM	26/43
AL QUNFUDHAH	19/41
AL QUWAYIYAH	24/44,45; 23/44,45
AL ULA	26/37
AL WAJH	26/36;26/37
AN NIMAHR	25/41
AN NIMAS	19/42
AR RAGHBAH	23/43
AR RIDANIYAH	24/44
AR RIYAD	24/47
ARJAH	24/44
AS SAFRA	24/41
AS SARAT	17/43; 18/43
AS SIHAM	23/42; 22/42
AS SULAYM	22/39
ASH SHA'IB	19/43
ASH SHARMAH	28/35
AT TAYBI	23/45
AYN QUNAY	23/45
AYNUNAH	28/35
BAHRAH	21/39
BAHRAN	22/39
BILJURSHI	19/41
BIR AL BAYDA	26/36
BIR AL KHAIS	24/45
BIR BADRIYAH	22/45
BIR FURAYSH	24/39
BIR GHAMRAH	22/45
BIR HUSANI	23/38
BIR JAYDAH	26/37
BIR JUQJUQ	21/43
BISHAH	20/42
BURAYDAH	26/43
BURAYKAH	22/39
DARB ZUBAYDA	I-202
DUBA	27/35
FARASAN ISL	16, 17/41
GHURAYYAH	27/35
GITH GATH	23/45

HAIL 27/41
HALABAN 23/44
HAMDAH 19/43
HAMMAT TEEN 23/45
HAQL 29/34
HARADAH 22/46
HARRAT HADAN 21/41
HARRAT RAHAT 22,23/40;23,24/39
J. AL BAYDA 24/39
J. AL BUWANA 24/37

J. AL HASIR 19/42,43; 19F
J. AL HAWSHA 22/44
J. AL LAWZ 28/35
J. AL MUSAYR 28/34
J. AL WASK 25/37,38
J. AR ROKHUM 23/41
J. ASH SHAMI 25/39
J. ASH SHIZM 26/37
J. ASH SHUMT 24/42
J. BUDIYAH 20/41
J. DARHAFAH 25/40
J. DHAYLAN 25/37
J. DHUHAYLAN 24/44
J. DHULAYAH 25/38
J. FARASAN 22/39
J. GHARABAH 25/37
J. HUMAYYAN 24/44
J. IBRAHIM 20/41
J. KIRSH 23/43
J. MAKHRUQAH 24/41
J. MURDAMAH 23/43
J. MURRYYI 20/41
J. NIYAQAT 23/38
J. SALAJAH 24/37
J. SHAYBAN 22/39
J. SHUMRAH 22/44
J. SHUWAYT 25/39

JABAL ABYAD 25/39
JABAL AFAF 20/40
JABAL AYA 18/42
JABAL BITRAN 23/45
JABAL BUWAD 24/38
JABAL DAHUL 22/43
JABAL DAMKH 23/44
JABAL ESS 26/37
JABAL GUYAN 18/43
JABAL IDSAS 23/45
JABAL IN 21/41
JABAL ISHMAS 20/43
JABAL JEDAIR 21/43
JABAL KHIDA 21/44
JABAL KHUFF 23,24/45
JABAL MAWAN 25/41
JABAL RADWA 24/38
JABAL RIK 25/41
JABAL RUGAAN 23/45
JABAL RUMUR 20/41
JABAL SA'BAN 18/41
JABAL SAHAH 22/44
JABAL SAMRAN 22/39
JABAL SARBAN 18/41
JABAL SAWDAH 18/42
JABAL SAYID 23/40
JABAL SHA'I 18/42
JABAL SHADA 19/41
JABAL SIHAM 23/42
JABAL TAWLAH 28/35
JADMAH 19/41
JAWF-SAKAKAH 29/38
JIDDAH 21/39
JIZAN 16/42
JURDHAWIYAH 25/42
KHADRA 19/42
KHAMIS MUSHA 18/42
KHASHIM RADI 24/47

(p. 37 follows)

```
DB2:PDP530.FTN                                           22-JAN-81 09:59:40
--------------------------------------------------------------------------------
C******************************************************************
C     PDP530
C
C     THIS PROGRAM EDITS ANY FILE WHICH IS IN THE REFBIB CARD
C     FORMAT-- EITHER NEW ADDITIONS,  UPDATES TO CURRENT FILE
C     OR EXISTING FILES.
C
C     U S G S  -  G A R Y   S E L N E R
C
C******************************************************************
      LOGICAL*1 INPUT(80),FILNAM(33),OFIL(33),IERR,PRT,
      1 TEXT(2625),IEOF,MASKEY(1000,12),KEYIN(6,12)
      INTEGER*2 IO1,IO2,IO3,IO4
      DATA IO1/1/,IO2/2/,IO3/3/,IO4/4/
      CALL TTINAA('ENTER INPUT FILENAME ',22,FILNAM,33,5)
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      CALL TTINAA('ENTER FILENAME FOR ERRORS ',27,OFIL,33,5)
      OPEN(UNIT=IO2,NAME=OFIL,TYPE='NEW')
      CALL TTINAA('PRINT ERRORS ONLY(Y/N)?',24,PRT,1,5)
      WRITE(IO2,1000) (FILNAM(I),I=1,32)
1000  FORMAT('1','PDP530 ERROR LIST FOR FILE: ',32A1)
C
C     PASS ONE
C
      WRITE(IO2,1050)
1050  FORMAT('0','PASS ONE ERRORS')
      NOERR = 0
100   READ(IO1,110,END=199) INPUT
110   FORMAT(80A1)
      IF (PRT.NE.'Y') WRITE(IO2,120) INPUT
120   FORMAT(' ',80A1)
      CALL CHECK1(IO2,INPUT,1,4,IERR,PRT)
      IF (IERR) NOERR=NOERR+1
      GO TO 100
199   CLOSE(UNIT=IO1,DISP='SAVE')
      IF (NOERR.NE.0) GO TO 9999
C
C     PASS TWO
C
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      WRITE(IO2,1040)
1040  FORMAT('0','PASS TWO ERRORS')
      IERR=.FALSE.
      IEOF=.TRUE.
      NOERR=0
200   CALL SPCREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,IEOF,IERR)
      IF (.NOT.IERR) GO TO 220
      WRITE(IO2,2000)MREF
2000  FORMAT('0','CARD CODE SEQUENCE ERROR - REFERENCE ',I4)
      NL = NLA+NLT+NLR+NLK
      DO 210 I = 1,NL
      K1=(I-1)*75+1
      K2=I*75
      WRITE(IO2,2010)MREF,(TEXT(J),J=K1,K2)
2010  FORMAT(' ',I4,1X,75A1)
210   CONTINUE
      NOERR=NOERR+1
```

```
220   IF (.NOT.IEOF) GO TO 200
      CLOSE(UNIT=IO1,DISP='SAVE')
      IF (NOERR.NE.0) GO TO 9999
C
C     PASS THREE
C
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      NOERR = 0
      WRITE(IO2,3000)
3000  FORMAT('0','PASS THREE ERRORS')
      IEOF = .TRUE.
      IERR = .FALSE.
      OPEN(UNIT=IO3,NAME='DB1:FOR003.DAT',TYPE='NEW')
300   CALL SPCREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,IEOF,IERR)
      WRITE(IO3,3010) MREF
3010  FORMAT(I4)
      IF (.NOT.IEOF) GO TO 300
      CLOSE(UNIT=IO1,DISP='SAVE')
      REWIND IO3
      OPEN(UNIT=IO4,NAME='DB1:FOR004.DAT',TYPE='NEW')
      CALL SORTR(IO3,IO4,5,TEXT,4,1,4)
      CLOSE(UNIT=IO3,DISP='DELETE')
      REWIND IO4
      IMAX = 0
320   READ(IO4,3010,END=350) MREF
      IF (IMAX.EQ.0) GO TO 330
      IF (MREF.NE.PMREF) GO TO 330
      NOERR = NOERR + 1
      WRITE(IO2,3020) MREF
3020  FORMAT(' ','MULTIPLE ENTRIES FOR REFERENCE ',I4)
330   PMREF=MREF
      IMAX = 99
      GO TO 320
350   CLOSE(UNIT=IO4,DISPOSE='DELETE')
      IF (NOERR.NE.0) GO TO 9999
C
C     PASS FOUR
C
      OPEN(UNIT=IO3,NAME='DB0:MASKEY.WRD',TYPE='OLD')
      TYPE 355
355   FORMAT('0','PASS FOUR ERRORS')
      NOERR = 0
      NKEY=1
400   READ(IO3,4000,END=410) (MASKEY(NKEY,J),J=1,12)
4000  FORMAT(12A1)
      NKEY=NKEY+1
      IF (NKEY.LE.1000) GO TO 400
      TYPE 4010
4010  FORMAT(' ','ERROR IN MASKEY.WRD  -- MORE THAN 1000 KEYWORDS')
410   NKEY=NKEY-1
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
450   READ(IO1,110,END=9999) INPUT
      IF (INPUT(6).NE.'K') GO TO 450
      DECODE(78,4030,INPUT) MREF,((KEYIN(I,J),J=1,12),I=1,6)
4030  FORMAT(I4,2X,6(12A1))
```

```
      DO 480 I = 1,6
      DO 460 J = 1,12
      IF (KEYIN(I,J).NE.' ') GO TO 465
460   CONTINUE
      GO TO 479
465   DO 470 J=1,NKEY
      DO 466 K=1,12
      IF (MASKEY(J,K).NE.KEYIN(I,K)) GO TO 469
466   CONTINUE
      GO TO 479
469   CONTINUE
470   CONTINUE
      WRITE(IO2,4040) MREF,(KEYIN(I,K),K=1,12)
4040  FORMAT(' ','INCORRECT KEYWORD REFERENCE ',I5,5X,'-',12A1,'-')
479   CONTINUE
480   CONTINUE
      GO TO 450
9999  CLOSE(UNIT=IO1,DISP='SAVE')
      WRITE(IO2,1010)
1010  FORMAT('1','END OF ERROR LISTING')
      CLOSE(UNIT=IO2,DISP='SAVE')
      STOP
      END
C*****************************************************************************
      SUBROUTINE SPCREF(IOU,MREF,NLA,NLT,NLR,NLK,TEXT,IEOF,IERR)
C     SPECIAL VERSION OF GETREF THAT CHECKS FOR SEQUENCE OF CARD
C     TYPES INTERNALLY AND RETURNS AN ERROR CODE, IERR IF AN
C     ERROR OCCURS.
      LOGICAL*1 TEXT(2625),IEOF,IERR,IPREVC
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,IOU
C     LOCAL VARIABLES
      LOGICAL*1 INPUT(75)
      INTEGER*2 PTR,IPREV,NIN
      IERR = .FALSE.
      PTR = 0
      NLA=0
      NLT=0
      NLR=0
      NLK=0
      DO 10 I = 1,2625
      TEXT(I)=' '
10    CONTINUE
      IF (.NOT.IEOF) GO TO 110
      READ(IOU,1000,ERR=980,END=998) NIN,INPUT
1000  FORMAT(I4,1X,75A1)
      IEOF=.FALSE.
      IPREV=NIN
      IPREVC = INPUT(1)
      GO TO 110
100   READ(IOU,1000,ERR=980,END=998) NIN,INPUT
110   IF (NIN.NE.IPREV) GO TO 999
      IF (INPUT(1).EQ.'A') GO TO 200
      IF (INPUT(1).EQ.'T') GO TO 300
      IF (INPUT(1).EQ.'R') GO TO 400
      IF (INPUT(1).EQ.'K') GO TO 500
      IF (INPUT(1).EQ.'D') GO TO 600
      TYPE 1010,NIN,INPUT
1010  FORMAT('  ','INVALID CARD IN MASTER:'/
     1      ,I4,1X,75A1)
      GO TO 100
200   NLA=NLA+1
      IF (IPREVC.NE.'A') IERR = .TRUE.
      IPREVC='A'
      GO TO 600
300   NLT=NLT+1
      IF (IPREVC.NE.'A' .AND.IPREVC.NE.'T') IERR=.TRUE.
      IPREVC='T'
      GO TO 600
400   NLR=NLR+1
      IF (IPREVC.NE.'T'.AND.IPREVC.NE.'R') IERR=.TRUE.
      IPREVC='R'
      GO TO 600
500   NLK=NLK+1
      IF (IPREVC.NE.'K'.AND.IPREVC.NE.'R') IERR=.TRUE.
600   DO 610 I = 1,75
      KI=PTR+I
      TEXT(KI)=INPUT(I)
610   CONTINUE
      PTR=PTR+75
      IF (PTR.LT.2625) GO TO 100
      TYPE 1020,NIN,INPUT
1020  FORMAT(' ','FOLLOWING RECORD WAS AT MAX RECORD SIZE:'/
     1   ' ',I4,1X,75A1)
      STOP
C     READ ERROR
980   TYPE 1030,IPREV
1030  FORMAT(' ','ERROR READING: PREV OR CURRENT REF= ',I4)
      GO TO 100
C     EOF READ
998   IEOF=.TRUE.
999   MREF=IPREV
      IPREV=NIN
      IPREVC=INPUT(1)
      RETURN
      END
C*********************************************************************
      SUBROUTINE CHECK1(IO2,INPUT,I1,I2,IERR,PRT)
      LOGICAL*1 INPUT(80),IERR,START,TDATA(10),PRT
      DATA TDATA/'0','1','2','3','4','5','6','7','8','9'/
      IERR=.FALSE.
      START=.FALSE.
      DO 100 I = I1,I2
      IF (.NOT.START.AND.INPUT(I).EQ.' ') GO TO 99
      START=.TRUE.
      IF (INPUT(I).NE.' ') GO TO 50
      IF (PRT.EQ.'Y') WRITE(IO2,900)INPUT
900   FORMAT('0',80A1)
      WRITE(IO2,1000) I1,I2
1000  FORMAT(' ','COL ',I3,' TO ',I3,' IS NOT RIGHT-',
     1 'JUSTIFIED OR CONTAINS IMBEDDED BLANKS')
```

```
      IERR=.TRUE.
      GO TO 110
50    DO 60 J=1,10
      IF (INPUT(I).EQ.TDATA(J)) GO TO 99
60    CONTINUE
      IF(PRT.EQ.'Y') WRITE(IO2,900) INPUT
      WRITE(IO2,1010) I1,I2
1010  FORMAT(' ','COL ',I3,' TO ',I3,' CONTAINS NON-',
      1  'NUMERIC CHARACTER')
      IERR=.TRUE.
      GO TO 110
99    CONTINUE
100   CONTINUE
      IF (START) GO TO 110
      IF (PRT.EQ.'Y') WRITE(IO2,900) INPUT
      WRITE(IO2,1020) I1,I2
1020  FORMAT(' ','COL ',I3,' TO ',I3,' IS ALL BLANK')
      IERR = .TRUE.
110   CONTINUE
      IF (INPUT(5).EQ.' ') GO TO 120
      IF ((.NOT.IERR).AND.(PRT.EQ.'Y')) WRITE(IO2,900) INPUT
      WRITE(IO2,1030)
1030  FORMAT(' ','COL 5 IS NOT BLANK')
      IERR=.TRUE.
120   CONTINUE
      IF (INPUT(6).EQ.'A') GO TO 130
      IF (INPUT(6).EQ.'T') GO TO 130
      IF (INPUT(6).EQ.'R') GO TO 130
      IF (INPUT(6).EQ.'K') GO TO 130
      IF (INPUT(6).EQ.'D') GO TO 130
      IF ((.NOT.IERR).AND.(PRT.EQ.'Y')) WRITE(IO2,900) INPUT
      WRITE(IO2,1040)
1040  FORMAT(' ','COL 6 CONTAINS CHARACTER NOT "A,T,R,K,D"')
      IERR=.TRUE.
130   CONTINUE
      RETURN
      END
```

```
C*********************************************************************
C
C     P D P 5 3 1
C
C     CREATES A 'BIG' REFERENCE FILE SORTED BY REFERENCE NUMBER
C     FOR THE MAIN UPDATE PROGRAM PDP532.
C
C     U S G S  -  G A R Y   S E L N E R
C
C*********************************************************************
C
      LOGICAL*1 FILNAM(33),DUM,TEXT(2625),EOF,SAVE(80)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,I01,I03,I04,ITT
      DATA I01/1/,I02/2/,I03/3/,I04/4/,ITT/5/,SAVE/80*' '/
      CALL TTINAA('ENTER INPUT FILENAME',20,FILNAM,33,ITT)
      OPEN(UNIT=I01,NAME=FILNAM,TYPE='OLD')
C     CREATE FILE FOR SORTING INTO REF NUMBER SEQUENCE
      OPEN(UNIT=I03,NAME='SCRATCH.DAT',TYPE='NEW')
      EOF = .TRUE.
      IMAX=0
100   CALL GETREF(I01,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      CALL LENGTH(TEXT,NCH,2625)
      IF (NCH.GT.IMAX) IMAX=NCH
      WRITE(I03,1999) MREF,(TEXT(I),I=1,NCH)
1999  FORMAT(I4,35(75A1))
      IF (.NOT.EOF) GO TO 100
      REWIND I03
      CLOSE(UNIT=I01,DISPOSE='SAVE')
      IMAX = IMAX + 4
      TYPE 3030
3030  FORMAT(' ','ENTER FILENAME FOR SORTED OUTPUT:'$)
      ACCEPT 3031,FILNAM
3031  FORMAT(33A1)
      FILNAM(33)=0
      OPEN(UNIT=I04,NAME=FILNAM,TYPE='NEW')
      CALL SORTR(I03,I04,ITT,TEXT,IMAX,1,4)
9999  CLOSE(UNIT=I04,DISPOSE='SAVE')
      CLOSE(UNIT=I03,DISPOSE='DELETE')
      STOP
      END
```

```
C*************************************************************
C
C      P D P 5 3 2
C      THIS PROGRAM DOES THE UPDATE OF
C      THE TOTAL REFERENCE FILE.
C      1.THE UPDATE FILE IS IN
C      REFERENCE # SEQUENCE AND IS IN THE
C      "EIC" REFERENCE FORMAT
C      2.THE CURRENT MASTER FILE IS IN
C      REFERENCE # SEQUENCE AND IS IN
C      THE 'CARD-IMAGE' FORMAT
C      3.THE NEW MASTER FILE IS WRITTEN
C      OUT IN REFERENCE # SEQUENCE AND IS
C      IN THE 'CARD-IMAGE' FORMAT.
C
C      U S G S  -  G A R Y  S E L N E R
C*************************************************************
      LOGICAL*1 TEXT1(2625),TEXT2(2625),TEXT3(2625),EOF1,EOF2,
      1  DELET,FILNAM(33),SAVE2(80)
      INTEGER*2  MREF1,NLA1,NLT1,NLR1,NLK1,
      1  MREF2,NLA2,NLT2,NLR2,NLK2,
      2  IO1,IO2,IO3,PTR,ITT,
      3  MREF3,NLA3,NLT3,NLR3,NLK3
      DATA SAVE2/80*' '/,ITT/5/
      IO1 = 1
      IO2 = 2
      IO3 = 3
C
      GET FILENAMES AND OPEN THEM
C
      CALL TTINAA('ENTER FILENAME FOR UPDATE INFO',30,FILNAM,33,ITT)
      OPEN (UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      CALL TTINAA('ENTER FILENAME FOR CURRENT MASTER',33,FILNAM,33,ITT)
      OPEN(UNIT=IO2,NAME=FILNAM,TYPE='OLD')
      CALL TTINAA('ENTER FILENAME FOR NEW MASTER',29,FILNAM,33,ITT)
      OPEN(UNIT=IO3,NAME=FILNAM,TYPE='NEW')
C
C      SETUP FOR MAIN LOOP
C
      EOF1=.FALSE.
      CALL SPCREF(IO1,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1,
      1  EOF1,DELET)
      EOF2=.TRUE.
      CALL GETREF(IO2,MREF2,NLA2,NLT2,NLR2,
      1  NLK2,TEXT2,EOF2,SAVE2)
C
C      MAIN LOOP
C
100   CONTINUE
      IF (MREF2.EQ.MREF1) GO TO 200
      IF (MREF2.GT.MREF1) GO TO 500
      IF (MREF2.LT.MREF1) GO TO 400
C
C      WE HAVE A MATCH.
C      TAKE CARE OF AUTHOR FIELD FIRST
200   CONTINUE
      IF (DELET) GO TO 290
      IF (NLA1.EQ.0) GO TO 210
      K1 = NLA1*75
      DO 205 I=1,K1
      TEXT3(I)=TEXT1(I)
205   CONTINUE
      NLA3 = NLA1
      PTR = K1
      GO TO 220
210   K1 = NLA2*75
      DO 215 I=1,K1
      TEXT3(I) = TEXT2(I)
215   CONTINUE
      NLA3 = NLA2
      PTR = K1
C     TAKE CARE OF TITLE FIELD
220   IF (NLT1.EQ.0) GO TO 230
      K1 = (NLA1*75) + 1
      K2 = (NLA1 + NLT1)*75
      DO 225 I=K1,K2
      PTR = PTR + 1
      TEXT3(PTR) = TEXT1(I)
225   CONTINUE
      NLT3 = NLT1
      GO TO 240
230   K1 = (NLA2*75) + 1
      K2 = (NLA2 + NLT2) * 75
      DO 235 I=K1,K2
      PTR = PTR + 1
      TEXT3(PTR) = TEXT2(I)
235   CONTINUE
      NLT3 = NLT2
C
C     TAKE CARE OF REFERENCE FIELD
C
240   IF (NLR1.EQ.0) GO TO 250
      K1 = ((NLA1+NLT1)*75) + 1
      K2 = (NLA1+NLT1+NLR1)*75
      DO 245 I=K1,K2
      PTR = PTR + 1
      TEXT3(PTR) = TEXT1(I)
245   CONTINUE
      NLR3 = NLR1
      GO TO 260
250   IF (NLR2.EQ.0) GO TO 260
      K1 = ((NLA2+NLT2)*75) + 1
      K2 = (NLA2+NLT2+NLR2)*75
      DO 255 I=K1,K2
      PTR = PTR + 1
      TEXT3(PTR) = TEXT2(I)
255   CONTINUE
      NLR3 = NLR2
C
C     TAKE CARE OF KEYWORDS FIELD
C
260   IF (NLK1.EQ.0) GO TO 270
```

```
      K1 = ((NLA1+NLT1+NLR1)*75) + 1
      K2 = (NLA1+NLT1+NLR1+NLK1)*75
      DO 265 I= K1,K2
      PTR = PTR + 1
      TEXT3(PTR) = TEXT1(I)
265   CONTINUE
      NLK3 = NLK1
      GO TO 280
270   IF (NLK2.EQ.0) GO TO 280
      K1 = ((NLA2+NLT2+NLR2)*75) + 1
      K2 = (NLA2+NLT2+NLR2+NLK2)*75
      DO 275 I = K1,K2
      PTR = PTR + 1
      TEXT3(PTR) = TEXT2(I)
275   CONTINUE
      NLK3 = NLK2
C
C     EVERYTHING IS IN PLACE
C     OUTPUT RECORD TO NEW
C     MASTER
280   MREF3 = MREF1
      CALL PUTREF(IO3,MREF3,NLA3,NLT3,NLR3,NLK3,TEXT3)
290   IF (EOF1.AND.EOF2) GO TO 900
      IF (EOF1) GO TO 300
      CALL SPCREF(IO1,MREF1,NLA1,NLT1,NLR1,
      1  NLK1,TEXT1,EOF1,DELET)
      IF (EOF1) MREF1 = 15000
300   IF (EOF2) GO TO 310
      CALL GETREF(IO2,MREF2,NLA2,NLT2,NLR2,
      1  NLK2,TEXT2,EOF2,SAVE2)
      GO TO 350
310   MREF2 = 15000
C
C     OK GO TO MAIN LOOP
C
350   GO TO 100
C
C     HAVEN'T REACHED IT YET
C
400   CONTINUE
      CALL PUTREF(IO3,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2)
      IF (EOF1.AND.EOF2) GO TO 900
      IF (EOF2) GO TO 410
      CALL GETREF(IO2,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2,EOF2,SAVE2)
      GO TO 450
410   MREF2 = 15000
C
C     OK GO TO MAIN LOOP
C
450   GO TO 100
C
C     MUST BE AN INSERT OR AN ERROR
C     FOR A DELETE. CHECK ERROR FIRST
C
500   CONTINUE
```

```
      IF (DELET) GO TO 290
      CALL PUTREF(IO3,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1)
      IF (EOF1.AND.EOF2) GO TO 900
510   IF (EOF1) GO TO 520
      CALL SPCREF(IO1,MREF1,NLA1,NLT1,NLR1,
      1  NLK1,TEXT1,EOF1,DELET)
520   IF (EOF1) MREF1 = 15000
C
C     OK GO TO MAIN LOOP
C
550   GO TO 100
C
C     OK WRAP IT UP
C
900   CLOSE (UNIT=IO1,DISPOSE='SAVE')
      CLOSE (UNIT=IO2,DISPOSE='SAVE')
      CLOSE (UNIT=IO3,DISPOSE='SAVE')
      STOP
      END
C************************************************************************
      SUBROUTINE SPCREF(IO2,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,DELET)
C
C     SPECIAL VERSION OF BIGREF----NO KEY IN FRONT OF RECORD SINCE
C     SORT WAS ON REFERENCE NUMBER WHICH IS FIRST FOUR CHARACTERS.
C
      LOGICAL*1 TEXT(2625),EOF,DELET
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,IO2,ISPEC
      DO 200 I = 1,2625
      TEXT(I)=' '
200   CONTINUE
      EOF = .FALSE.
      READ(IO2,1999,END=999) MREF,TEXT
1999  FORMAT(I4,35(75A1))
      NLA=0
      NLT=0
      NLR=0
      NLK=0
      DELET=.FALSE.
      IF (TEXT(1).EQ.'D') DELET=.TRUE.
      DO 100 I = 1,2551,75
      IF (TEXT(I).EQ.'A') NLA=NLA+1
      IF (TEXT(I).EQ.'T') NLT=NLT+1
      IF (TEXT(I).EQ.'R') NLR=NLR+1
      IF (TEXT(I).EQ.'K') NLK=NLK+1
      IF (TEXT(I).EQ.'A'.OR.TEXT(I).EQ.'T'.OR.TEXT(I).EQ.'R'
      1 .OR.TEXT(I).EQ.'K'.OR.TEXT(I).EQ.' '.OR.TEXT(I).EQ.'D')
      2 GO TO 99
      TYPE 1998,MREF
1998  FORMAT(' ','INVALID CARD CODE MREF:',I4)
99    CONTINUE
100   CONTINUE
222   RETURN
999   EOF = .TRUE.
      GO TO 222
      END
```

```
C***********************************************************************
C     P D P 5 3 3
C
C     GENERATES AUTHOR-SORTED REFERENCE FILE FROM REFERENCE-ORDERED
C     FILE.
C
C     U S G S - G A R Y  S E L N E R
C
C***********************************************************************
      LOGICAL*1 TEXT(2625),KEY(34),EOF,FILNAM(33),MASFIL(33),SAVE(80)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,I01,I02,I03,IMAX,KEYLEN,ITT
      DATA I01/1/,I02/2/,I03/3/,KEYLEN/34/,ITT/5/
      DATA SAVE/80*' '/
      CALL TTINAA('ENTER INPUT FILENAME',20,FILNAM,33,ITT)
      OPEN(UNIT=I01,NAME=FILNAM,TYPE='OLD')
      CALL TTINAA('ENTER FILENAME FOR NEW MASTER FILE',34,MASFIL,33,ITT)
      OPEN(UNIT=I02,NAME='DB1:FOR002.DAT',TYPE='NEW')
      EOF = .TRUE.
      IMAX=0
100   CALL GETREF(I01,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      CALL BLDKEY(KEY,MREF,TEXT,NLA)
      CALL LENGTH(TEXT,NCH,2625)
      IF (NCH.GT.IMAX) IMAX=NCH
      WRITE(I02,1020) KEY,MREF,(TEXT(I),I=1,NCH)
1020  FORMAT(34A1,I4,35(75A1))
      IF (.NOT.EOF) GO TO 100
      IMAX = IMAX + 38
      CLOSE(UNIT=I01,DISPOSE='SAVE')
C     SORT I02 FILE ON KEY
      REWIND I02
      OPEN(UNIT=I03,NAME='DB1:FOR003.DAT',TYPE='NEW')
      CALL SORTR(I02,I03,5,TEXT,IMAX,1,34)
      REWIND I03
      CLOSE(UNIT=I02,DISPOSE='DELETE')
      OPEN(UNIT=I01,NAME=MASFIL,TYPE='NEW')
      EOF = .FALSE.
10    CALL BIGREF(KEYLEN,KEY,I03,MREF,NLA,NLT,NLR,NLK,TEXT,EOF)
      IF (EOF) GO TO 999
      CALL PUTREF(I01,MREF,NLA,NLT,NLR,NLK,TEXT)
      GO TO 10
999   CLOSE(UNIT=I01,DISPOSE='SAVE')
      CLOSE(UNIT=I03,DISPOSE='DELETE')
      STOP
      END
C***********************************************************************
      SUBROUTINE BLDKEY(KEY,MREF,TEXT,NLA)
      LOGICAL*1 TEXT(2625),KEY(34)
      INTEGER O
      DO 5 I = 1,34
      KEY(I) = ' '
5     CONTINUE
      I=2
      O=1
10    IF (TEXT(I).EQ.'.') GO TO 20
      IF (TEXT(I).EQ.',') GO TO 20
      IF (TEXT(I).EQ.'-') GO TO 20
      IF (TEXT(I).EQ.' ') GO TO 20
```

```
C     GOOD CHARACTER FOR KEY
      KEY(J)=TEXT(I)
      J=J+1
20    I=I+1
      IF(J.GT.30) GO TO 100
      IF(I.GT.74) GO TO 100
      GO TO 10
C     ADD YEAR TO 30 CHARS OF AUTHOR NAME FIELDS
100   K1 = NLA*75 + 2
      K2 = K1+3
      K3 = 31
      DO 110 I = K1,K2
      KEY(K3)=TEXT(I)
      K3 = K3 + 1
110   CONTINUE
      RETURN
      END
```

```
C*********************************************************************
C
C     P D P 5 3 4
C
C     CREATES LISTING OF ALL REFERENCE FOR A KEYWORD
C
C     U S G S - G A R Y   S E L N E R
C
C*********************************************************************
      LOGICAL*1 TEXT(2625),EOF,FILNAM(33),SECOND,PREV(12),NEW,
      1  KEY(12),PRTFIL(33),SAVE(80),SPKEY(16)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,ISPEC,IO1,IO2,IO3,IMAX,NL,
      1 LINCT,KEYLEN
      REAL*4 DAT(3)
      DATA PREV/12*' '/,DAT/3*'    '/,IO1/1/,IO2/2/,IO3/3/
      DATA KEYLEN/16/
      DATA SAVE/80*' '/
      CALL DATE(DAT)
      TYPE 1000
1000  FORMAT(' ','ENTER INPUT FILENAME:'$)
      ACCEPT 1010,FILNAM
1010  FORMAT(33A1)
      FILNAM(33)=0
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      OPEN(UNIT=IO2,NAME='DB1:FOR002.DAT',TYPE='NEW')
      TYPE 1020
1020  FORMAT(' ENTER FILENAME FOR PRINTER OUTPUT:'$)
      ACCEPT 1010,PRTFIL
      PRTFIL(33)=0
      EOF = .TRUE.
      IMAX=0
      ISPEC = 1
100   CALL GETREF(1,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      CALL LENGTH(TEXT,NCH,2625)
      IF (NCH.GT.IMAX) IMAX=NCH
      IF (NLK.EQ.0) GO TO 510
      SECOND=.FALSE.
      K1 = (NLA+NLT+NLR)*75+2
300   DO 500 I = K1,K1+60,12
      DO 400 J=I,I+11
      K2 = J - I + 1
      IF (TEXT(J).NE.PREV(K2)) GO TO 410
400   CONTINUE
      GO TO 499
410   DO 420 J = I,I+11
      K2 = J-I+1
      KEY(K2)=TEXT(J)
420   CONTINUE
      WRITE(IO2,1999)KEY,ISPEC,MREF,(TEXT(J),J=1,NCH)
499   CONTINUE
500   CONTINUE
1999  FORMAT(12A1,I4,I4,35(75A1))
      IF (NLK.EQ.1) GO TO 510
      IF (SECOND) GO TO 510
      K1=(NLA+NLT+NLR+1)*75+2
      SECOND = .TRUE.
      GO TO 300
```

```
510   CONTINUE
      ISPEC = ISPEC + 1
      IF (.NOT.EOF) GO TO 100
      IMAX = IMAX + 20
      CLOSE(UNIT=IO1,DISPOSE='SAVE')
      REWIND IO2
      OPEN(UNIT=IO3,NAME='DB1:FOR003.DAT',TYPE='NEW')
      CALL SORTR(IO2,IO3,5,TEXT,IMAX,1,16)
      CLOSE(UNIT=IO2,DISPOSE='DELETE')
      REWIND IO3
      OPEN(UNIT=IO1,NAME=PRTFIL,TYPE='NEW')
      IMAX = 60
      LINCT=6
      EOF = .FALSE.
10    CALL BIGREF(KEYLEN,SPKEY,IO3,MREF,NLA,NLT,NLR,NLK,TEXT,EOF)
      IF (EOF) GO TO 999
      DO 13 I = 1,12
      KEY(I)=SPKEY(I)
13    CONTINUE
      NEW = .FALSE.
      DO 15 I = 1,12
      IF (PREV(I).EQ.KEY(I)) GO TO 14
      NEW=.TRUE.
14    CONTINUE
15    CONTINUE
      IF (.NOT.NEW) GO TO 16
      DO 18 I = 1,12
      PREV(I)=KEY(I)
18    CONTINUE
      WRITE(IO1,2030) KEY,DAT
      LINCT = 6
16    ILINE = 1
      ILINE = ILINE + NLA+NLT+NLR
      ILINE=ILINE + NLK
20    IF (LINCT+ILINE.GT.IMAX) CALL NEWPAG(IO1,LINCT,KEY)
      WRITE(IO1,2000)MREF,(TEXT(J),J=2,75)
      DO 30 I = 2,NLA+NLT+NLR
      K1 = (I-1)*75+2
      K2 = K1+73
      WRITE(IO1,2010) (TEXT(J),J=K1,K2)
30    CONTINUE
      IF (NLK.EQ.0) GO TO 40
      K1 = (NLA+NLT+NLR)*75 + 2
      K2 = K1 + 71
      WRITE(IO1,2040) (TEXT(J),J=K1,K2)
      IF (NLK.EQ.1) GO TO 40
      K1 = (NLA+NLT+NLR+1)*75+2
      K2 = K1 + 71
      WRITE(IO1,2040) (TEXT(J),J=K1,K2)
40    LINCT=LINCT+ILINE
      GO TO 10
999   WRITE(IO1,2050)
      CLOSE(UNIT=IO3,DISPOSE='DELETE')
      CLOSE(UNIT=IO1,DISPOSE='SAVE')
      STOP
```

```
2000  FORMAT('0',3X,I4,7X,74A1)
2010  FORMAT(' ',19X,74A1)
2030  FORMAT('1',//'0',19X,'KEYWORD:    ',12A1,40X,2A4,A1//)
2040  FORMAT(' ',18X,6('  ',12A1))
2050  FORMAT('1','END OF LISTING')
      END
C***********************************************************************
      SUBROUTINE NEWPAG(IOUT,LINCT,KEY)
      LOGICAL*1 KEY(12),CONT(9)
      INTEGER*2 LINCT,IOUT
      DATA CONT/' ','(','C','O','N','T','''','D',')'/
      WRITE(IOUT,1000) KEY,CONT
1000  FORMAT('1',///'0',19X,'KEYWORD:   ',12A1,9A1//)
      LINCT = 6
      RETURN
      END
```

```
C     REFERENCE PROGRAM
C
C     THIS PROGRAM IS USED TO PERFORM THE FOLLOWING:
C
C     1. TO RETRIEVE A SUBSET OF THE MAIN REFERENCE
C        FILE BY LOGICAL EXPRESSION OF AUTHOR(S)
C
C     2. TO RETRIEVE A SUBSET OF THE MAIN REFERENCE
C        FILE BY LOGICAL EXPRESSION OF KEYWORD(S)
C
C     3. TO RETRIEVE A SUBSET OF HE MAIN REFERENCE
C        FILE BY REFERENCE NUMBER(S)
C
C     4. TO COMBINE TWO SUBSETS OF THE MAIN REFERENCE
C        FILE INTO A SINGLE FILE.
C
      LOGICAL*1 Y1,N1,A1,FILNAM(33),TEXT(2625)
      INTEGER*2 REF(250),IO1,IO2,ITT,IO3
      REAL*4 DAT(4),TIM(2),BLNK
      DATA BLNK/'    '/,IO1/1/,IO2/2/,ITT/5/,Y1/'Y'/,IO3/3/
      WRITE(ITT,1020)
      DAT(3) = BLNK
      CALL DATE(DAT)
      CALL TIME(TIM)
      CALL TTINAA(' KEYWORD SELECTION(Y/N)?',24,A1,1,ITT)
      IF (A1.NE.Y1) GO TO 20
C
C     KEYWORD SELECTION
C
      CALL SELKE(ITT,IO1,IO2,NREF,TEXT,FILNAM)
      GO TO 50
20    CALL TTINAA(' REFERENCE # SELECTION(Y/N)?',28,A1,1,ITT)
      IF (A1.NE.Y1) GO TO 30
C
C     REFERENCE NUMBER SELECTION
C
      CALL SELRE(ITT,IO1,IO2,NREF,TEXT,FILNAM)
      GO TO 50
30    CALL TTINAA(' SELECTION BY AUTHOR(Y/N)?',26,A1,1,ITT)
      IF (A1.NE.Y1) GO TO 35
C
C     SELECTION BY AUTHOR
C
      CALL SELAU(ITT,IO1,IO2,NREF,TEXT,FILNAM)
      GO TO 50
35    CALL TTINAA(' MERGE TWO SUBSETS(Y/N)?',24,A1,1,ITT)
      IF (A1.NE.Y1) GO TO 40
C
C     MERGE TWO SUBSETS
C
      CALL MERGE(ITT,IO1,IO2,IO3,NREF,FILNAM,TEXT)
      GO TO 50
C
C     NO METHOD SELECTED CLOSE UP SHOP AND GO HOME
40    CLOSE(UNIT=IO1,DISPOSE='SAVE')
      CLOSE(UNIT=IO2,DISPOSE='DELETE')
      GO TO 70
```

```
50    CLOSE(UNIT=IO1,DISPOSE='SAVE')
      CLOSE(UNIT=IO2,DISPOSE='SAVE')
      WRITE(ITT,1000) NREF
1000  FORMAT(' ',I4,' REFERENCES SELECTED.')
      CALL TTINAA(' DO YOU WISH TO PRINT THE SELECTIONS(Y/N)?',
      1     42,A1,1,ITT)
      IF (A1.NE.Y1) GO TO 60
C
C     PRINT THEM
C
      OPEN(UNIT=IO2,NAME=FILNAM,TYPE='OLD')
      CALL OUTPT(ITT,IO2,IO3,DAT,TEXT)
      WRITE(ITT,1030)
1030  FORMAT('0',/////)
60    CALL TTINAA(' SAVE FILE OF SELECTED SUBSET(Y/N)?',35,A1,1,ITT)
      IF (A1.NE.Y1) CLOSE(UNIT=IO2,DISPOSE='DELETE')
      IF (A1.EQ.Y1) CLOSE(UNIT=IO2,DISPOSE='SAVE')
70    WRITE(ITT,1010)
1010  FORMAT(' END OF SELECTION RUN')
      WRITE(ITT,1020)
1020  FORMAT(//,1X,8('----------')//)
      STOP
      END
C**********************************************************************
      SUBROUTINE SETUP(ITT,IO1,IO2,OUTFIL)
      LOGICAL*1 FILNAM(33),A1,Y1,N1,OUTFIL(33)
      INTEGER*2 IO1,IO2,ITT
      DATA FILNAM/'D','B','1',':','[','7','2',',','3','4','0',']',
      1       'S','A','E','S','B','1','.','D','A','T',11*' '/
      DATA Y1/'Y'/
      FILNAM(33)=0
      CALL TTINAA(' STANDARD MASTER FILE(Y OR N)?',30,A1,1,ITT)
      IF (A1.EQ.Y1) GO TO 10
      CALL TTINAA(' ENTER MASTER FILENAME:',23,FILNAM,33,ITT)
10    OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      CALL TTINAA(' ENTER FILENAME FOR SELECTED SUBSET',35,OUTFIL,33,ITT)
20    OPEN(UNIT=IO2,NAME=OUTFIL,TYPE='NEW')
30    RETURN
      END
C**********************************************************************
      SUBROUTINE GETTST(ITT,PREFIX,TVALUE,CONJ,NLINE,NCH)
C     THIS SUBROUTINE DOES THE PROMPTING FOR THE LOGICAL TESTS
C     FOR KEYWORD SELECTION AND FOR AUTHOR SELECTION
C
C     WRITTEN BY GARY SELNER SEPT 1979
      INTEGER*4 PREFIX(45),CONJ(45),NLINE,ITEST,ITT,NCH
      LOGICAL*1 INPUT(40),ITAB,TVALUE(NCH,45)
      DATA ITAB/'011'/
      DO 4 I = 1,45
      PREFIX(I) = '    '
      CONJ(I) = '    '
      DO 3 J = 1,NCH
      TVALUE(J,I) = ' '
3     CONTINUE
4     CONTINUE
```

```
5     IF (NCH.EQ.12) WRITE(ITT,900)
900   FORMAT(' ','PREFIX  VALUE           CONNECTOR')
      IF (NCH.EQ.24) WRITE(ITT,901)
901   FORMAT(' ','PREFIX  VALUE                       CONNECTOR')
      I = 1
10    READ(ITT,910) INPUT
910   FORMAT(40A1)
      IF(INPUT(1).EQ.' ') GO TO 80
      IF (INPUT(1).EQ.'F'.AND.INPUT(2).EQ.'O'.AND.
      1 INPUT(3).EQ.'R') GO TO 20
      IF (INPUT(1).EQ.'N'.AND.INPUT(2).EQ.'O'.AND.
      1 INPUT(3).EQ.'T') GO TO 20
      WRITE (ITT,920) INPUT
920   FORMAT(' ','ERROR FIRST THREE CHARACTERS MUST BE FOR OR NOT'/
      1 ' ',40A1)
      GO TO 10
20    ENCODE(4,930,PREFIX(I)) (INPUT(K),K=1,3)
930   FORMAT(3A1,' ')
C     FIND FIRST TAB AND SAVE POSITION
      K=1
30    IF (INPUT(K).EQ.ITAB) GO TO 40
      K = K + 1
      GO TO 30
40    ITAB1 = K
      K = K + 1
50    IF (INPUT(K).EQ.ITAB) GO TO 60
      K = K + 1
      GO TO 50
60    ITAB2 = K
C     NOW PULL OUT TEST VALUE AND STORE IT
      K1 = ITAB1 + 1
      K2 = ITAB2 -1
      IF (K2.LE.K1) GO TO 70
      IF ((K2-K1+1).GT.NCH) GO TO 70
      ENCODE(NCH,940,TVALUE(1,I))(INPUT(K),K=K1,K2)
940   FORMAT(24A1)
C     NOW PULL OUT CONJUNCTION
      K1 = ITAB2 + 1
      ENCODE(3,950,CONJ(I))(INPUT(K),K=K1,K1+2)
950   FORMAT(3A1)
      I = I + 1
      IF (I.LE.50) GO TO 10
      WRITE(ITT,1030)
1030  FORMAT(' ','YOU HAVE EXCEEDED THE MAXIMUM NUMBER OF LINES(45)')
      STOP
70    WRITE(ITT,960) INPUT
960   FORMAT(' ','IMPROPER USE OF TABS OR NO TEST VALUE GIVEN'/
      1 ' ',40A1)
      GO TO 10
80    NLINE = I - 1
      WRITE(ITT,970)NLINE
970   FORMAT(' ','YOU HAVE SPECIFIED ON ',I4,' LINES OF INPUT'/
      1 ' ','THE FOLLOWING TESTS:')
      ITEST = 1
      DO 90 I = 1,NLINE
      IF (CONJ(I).EQ.'OR') GO TO 85
      IF (NCH.EQ.12)
     1 WRITE (ITT,971) PREFIX(I),(TVALUE(J,I),J=1,NCH),CONJ(I)
971   FORMAT(' ',A4,6X,12A1,6X,A4)
      IF (NCH.EQ.24)
     1 WRITE(ITT,972) PREFIX(I),(TVALUE(J,I),J=1,NCH),CONJ(I)
972   FORMAT(' ',A4,6X,24A1,6X,A4)
      GO TO 89
85    WRITE(ITT,980) PREFIX(I),(TVALUE(J,I),J=1,NCH)
980   FORMAT(' ',A4,6X,24A1)
      WRITE(ITT,990) CONJ(I)
990   FORMAT(' ','---------- ',A4,' ------------')
      ITEST=ITEST+1
89    CONTINUE
90    CONTINUE
      WRITE(ITT,1000) ITEST
1000  FORMAT('0','THIS CONSTITUTES ',I4,' TEST CLAUSE(S)'/
     1 ' ','OK?'$)
      READ(ITT,1010)IANS
1010  FORMAT(A1)
      IF (IANS.NE.'N') GO TO 100
      WRITE(ITT,1020)
1020  FORMAT(' ','TRY AGAIN')
      GO TO 5
100   WRITE(ITT,214)
214   FORMAT(' PROCESSING IN PROGRESS; PLEASE WAIT FOR',
     1' FURTHER PROMPTING')
      RETURN
      END
C************************************************************************
      SUBROUTINE SELKE(ITT,IO1,IO2,ICNT,TEXT,FILNAM)
C     SELECTION BY KEYWORD LOGICAL EXPRESSION
      LOGICAL*1 TEXT(2625),MKEY(12,12),EOF,RESULT,KEY(12,45),
     1 FILNAM(33),SAVE(80)
      INTEGER*2 ITT,IO1,IO2,MREF,NLA,NLT,NLR,NLK
      INTEGER*4 PREFIX(45),CONJ(45),ITEST
      DATA SAVE/80*' '/
      CALL SETUP(ITT,IO1,IO2,FILNAM)
      CALL GETTST(5,PREFIX,KEY,CONJ,NLINE,12)
      EOF = .TRUE.
      ICNT = 0
C     READ MASTER FILE RECORD AND CHECK KEYWORDS FOR MATCH
500   CALL GETREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      IF (NLK.EQ.0) GO TO 330
      DO 155 J=1,12
      DO 154 K=1,12
      MKEY(J,K)=' '
154   CONTINUE
155   CONTINUE
      K1 = (NLA+NLT+NLR)*75 +2
      DECODE(72,1000,TEXT(K1))((MKEY(I,J),I=1,12),J=1,6)
1000  FORMAT(6(12A1))
      IF (NLK.EQ.1) GO TO 501
      K1 = (NLA+NLT+NLR+1)*75 +2
      DECODE(72,1000,TEXT(K1))((MKEY(I,J),I=1,12),J=7,12)
```

```
501   CONTINUE
      RESULT=.TRUE.
      DO 320 I = 1,NLINE
      IF (PREFIX(I).EQ.'NOT') GO TO 200
C     TEST FOR KEY EQUAL TO TEST VALUE
      DO 156 J = 1,12
      DO 157 K = 1,12
      IF (KEY(K,I).NE.MKEY(K,J)) GO TO 156
157   CONTINUE
C     MATCH
      GO TO 300
156   CONTINUE
C     NO MATCH SET RESULT TO FALSE
      RESULT = .FALSE.
      GO TO 300
C     TEST IS NOT EQUAL TO TEST VALUE
200   DO 256 J = 1,12
      DO 257 K = 1,12
      IF (KEY(K,I).NE.MKEY(K,J)) GO TO 256
257   CONTINUE
C     MATCH SO SET RESULT TO FALSE
      RESULT=.FALSE.
      GO TO 300
C     NO MATCH SO LEAVE RESULT AS IS
256   CONTINUE
C     TEST TO SEE IF AT END OF CLAUSE
300   IF (CONJ(I).NE.'OR') GO TO 320
C     AT END OF CLAUSE IF RESULT IS TRUE
C     SEND RECORD TO OUTPT AND GO GET NEXT RECORD
C     IF RESULT IS FALSE, RESET RESULT AND TRY NEXT CLAUSE.
      IF (RESULT) GO TO 310
      RESULT=.TRUE.
320   CONTINUE
310   IF (RESULT) CALL PUTREF(IO2,MREF,NLA,NLT,NLR,NLK,TEXT)
      IF (RESULT) ICNT=ICNT+1
330   IF (EOF) GO TO 999
      GO TO 500
999   RETURN
      END
C*********************************************************************
      SUBROUTINE SELRE(ITT,IO1,IO2,ICNT,TEXT,FILNAM)
C     SELECTION BY REFERENCE # FOR <250 REFERENCES
      LOGICAL*1 MKEY(12,12),TEXT(2625),EOF,FILNAM(33),SAVE(80)
      INTEGER*2 REF(250),MREF,NLA,NLT,NLR,NLK,ITT,IO1,IO2,ICNT,NREF
      DATA SAVE/80*' '/
      CALL SETUP(ITT,IO1,IO2,FILNAM)
      EOF = .TRUE.
      ICNT = 0
      WRITE(ITT,208)
208   FORMAT(' PLEASE ENTER REF NOS ONE AT A TIME UPON PROMPT(:)',
     1 '. END BY REF # 0000')
      DO 55 I = 1,250
      CALL TTINSI(' ',1,REF(I),ITT)
      NREF=I
      IF (REF(I).EQ.0) GO TO 56
```

```
[illegible] CONTINUE
      GO TO 57
56    MREF=NREF-1
57    WRITE(ITT,214)
214   FORMAT('  PROCESSING IN PROGRESS: PLEASE WAIT FOR FURTHER PROMPTING')
C     READ MASTER FILE AND CHECK FOR MATCH WITH REF #
      N=1
500   CALL GETREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      IF(MREF.EQ.REF(N)) GO TO 520
      IF (EOF) GO TO 999
      GO  TO 500
520   N=N+1
      CALL PUTREF(IO2,MREF,NLA,NLT,NLR,NLK,TEXT)
      ICNT = ICNT + 1
      IF(N.GT.NREF) GO TO 999
      IF (EOF) GO TO 999
      GO TO 500
999   RETURN
      END
C***********************************************************************
      SUBROUTINE SELAU(ITT,IO1,IO2,ICNT,TEXT,FILNAM)
C     SELECTION BY AUTHOR LOGICAL EXPRESSION
      INTEGER*2 ITT,IO1,IO2,ICNT,MREF,NLA,NLT,NLK
      INTEGER*4 PREFIX(45),CONJ(45),NLINE
      DIMENSION LAU(45)
      LOGICAL*1 AUTH(24,45),MKEY(12,12),TEXT(2625),EOF,
     1 YES,RESULT,B1,TEMP(444),FILNAM(33),SAVE(80)
      DATA YES/'Y'/,B1/' '/,SAVE/80*' '/
      CALL SETUP(ITT,IO1,IO2,FILNAM)
      CALL GETTST(5,PREFIX,AUTH,CONJ,NLINE,24)
      DO 400 I = 1,NLINE
      DO 461 K = 1,23
      IF (AUTH(K,I).EQ.B1.AND.AUTH(K+1,I).EQ.B1) GO TO 462
      IF (K.EQ.23.AND.AUTH(24,I).EQ.B1) GO TO 463
461   CONTINUE
      LAU(I)=24
      GO TO 464
462   LAU(I)=K-1
      GO TO 464
463   LAU(I)=23
464   CONTINUE
400   CONTINUE
      EOF = .TRUE.
      ICNT = 0
C     READ A MASTER FILE RECORD
500   CALL GETREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      IF (NLA.GT.0) GO TO 505
      TYPE 1111,MREF
1111  FORMAT(' ','NO AUTHOR DATA ON REF:',I5)
      GO TO 311
505   IF (NLA.LE.6) GO TO 506
      TYPE 1112,MREF
1112  FORMAT(' ','MORE THAN 444 CHARACTER OF AUTHOR DATA FOR MREF:',I5)
      GO TO 311
506   NCH = 0
```

```
      DO 508 I = 1,NLA
      K1 = (I-1)*75 + 2
      K2 = K1 + 73
      DO 507 J = K1,K2
      NCH = NCH+1
      TEMP(NCH) = TEXT(J)
507   CONTINUE
508   CONTINUE
      RESULT=.TRUE.
      DO 320 I = 1,NLINE
      IF (PREFIX(I).EQ.'NOT') GO TO 200
C     TEST FOR AUTHOR EQUAL TO TEST VALUE
      NC = LAU(I)
      NCL=NCH-NC
      DO 156 J = 1,NCL
      DO 157 K = 1,NC
      IF (AUTH(K,I).NE.TEMP(J+K-1)) GO TO 156
157   CONTINUE
C     MATCH
      GO TO 300
156   CONTINUE
C     NO MATCH SET RESULT TO FALSE
      RESULT=.FALSE.
      GO TO 300
C     TEST IS NOT EQUAL TO TEST VALUE
200   NC = LAU(I)
      NCL=NCH-NC
      DO 256 J=1,NCL
      DO 257 K=1,NC
      IF (AUTH(K,I).NE.TEMP(J+K-1)) GO TO 256
257   CONTINUE
C     MATCH SO SET RESULT TO FALSE
      RESULT = .FALSE.
      GO TO 300
C     NO MATCH SO LEAVE RESULT AS IS
256   CONTINUE
C     TEST TO SEE IF AT END OF CLAUSE
300   IF (CONJ(I).NE.'OR') GO TO 320
C     AT END OF CLAUSE IF RESULT IS TRUE SEND RECORD TO
C     OUTPT AND GO GET NEXT RECORD. IF RESULT IS FALSE,
C     RESET RESULT AND TRY NEXT CLAUSE.
      IF (RESULT) GO TO 310
      RESULT = .TRUE.
320   CONTINUE
310   IF (RESULT) CALL PUTREF(IO2,MREF,NLA,NLT,NLR,NLK,TEXT)
      IF (RESULT) ICNT = ICNT + 1
311   IF (EOF) GO TO 999
      GO TO 500
999   RETURN
      END
C***********************************************************************
      SUBROUTINE MERGE(ITT,IO1,IO2,IO3,NREF,FILNAM,TEXT1)
C     MERGE SUBROUTINE
C
C     THIS SUBROUTINE MERGES TWO REFERENCE FILES
```

```
C     THAT ARE IN AUTHOR SEQUENCE.  THE
C     OUTPUT FILE IS A MASTER FILE IN
C     AUTHOR SEQUENCE.
C
C     UPDATE THE TOTAL MASTER FILE.
C
C     NOTE 2 - DUPLICATE RECORDS(FIRST THIRTY
C     CHARACTERS OF AUTHOR PLUS YEAR OF
C     PUBLICATION) WILL RESULT IN THE CONTENTS
C     OF THE RECORD FROM THE FIRST FILE
C     BEING WRITTEN TO THE OUTPUT FILE.
C
      LOGICAL*1 KEY1(34),KEY2(34),TEXT1(2625),TEXT2(2625),EOF1,EOF2,
      1  FILNAM(33),SAVE1(80),SAVE2(80)
      INTEGER*2 IO1,IO2,IO3,NLA1,NLA2,NLT1,NLT2,NLR1,NLR2,NLK1,NLK2,
      1    MREF1,MREF2,NREF
      DATA SAVE1/80*' '/,SAVE2/80*' '/
      NREF= 0
      TYPE 1000
1000  FORMAT(' ','ENTER FILENAME FOR 1ST DATA FILE:'$)
      ACCEPT 1010,FILNAM
1010  FORMAT(33A1)
      FILNAM(33) = 0
      OPEN (UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      TYPE 1020
1020  FORMAT(' ','ENTER FILENAME OF 2ND DATA FILE:'$)
      ACCEPT 1010,FILNAM
      FILNAM(33) = 0
      OPEN(UNIT=IO2,NAME=FILNAM,TYPE='OLD')
      TYPE 1030
1030  FORMAT(' ','ENTER FILENAME OF COMBINED DATA FILE:'$)
      ACCEPT 1010,FILNAM
      FILNAM(33) = 0
      OPEN(UNIT=IO3,NAME=FILNAM,TYPE='NEW')
C     SETUP FOR MAIN LOOP
      EOF1=.TRUE.
      CALL GETREF(IO1,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1,EOF1,SAVE1)
      CALL BLDKEY(KEY1,MREF1,TEXT1,NLA1)
      EOF2=.TRUE.
      CALL GETREF(IO2,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2,EOF2,SAVE2)
      CALL BLDKEY(KEY2,MREF2,TEXT2,NLA2)
C
C     MAIN LOOP
C
100   CONTINUE
      DO 110 I = 1,34
      IF (KEY1(I).EQ.KEY2(I)) GO TO 109
      IF (KEY1(I).LT.KEY2(I)) GO TO 200
      IF (KEY1(I).GT.KEY2(I)) GO TO 300
109   CONTINUE
110   CONTINUE
C
C     DUPLICATE RECORD---
C             CHECK CHARACTER FOR CHARACTER  TO SEE IF IDENTICAL
C             IF IDENTICAL, USE ONLY DATA FROM 1ST FILE.
```

```
C              IF NOT WRITE OUT BOTH...NOTE 2ND FILE THEN 1ST FILE.
C
      DO 115 I = 1,2625
      IF (TEXT1(I).NE.TEXT2(I)) GO TO 116
115   CONTINUE
      GO TO 120
116   CALL PUTREF(IO3,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2)
      NREF=NREF + 1
120   CALL PUTREF(IO3,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1)
      NREF = NREF + 1
      IF (EOF1) GO TO 400
      CALL GETREF(IO1,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1,EOF1,SAVE1)
      CALL BLDKEY(KEY1,MREF1,TEXT1,NLA1)
      IF (EOF2) GO TO 300
      CALL GETREF(IO2,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2,EOF2,SAVE2)
      CALL BLDKEY(KEY2,MREF2,TEXT2,NLA2)
      GO TO 100
C
C     WRITE OUT RECORD FROM FILE 1
C
200   CALL PUTREF(IO3,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1)
      NREF=NREF+1
      IF (EOF1) GO TO 400
      CALL GETREF(IO1,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1,EOF1,SAVE1)
      CALL BLDKEY(KEY1,MREF1,TEXT1,NLA1)
      GO TO 100
C
C     WRITE OUT RECORD FROM FILE 2
C
300   CALL PUTREF(IO3,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2)
      NREF=NREF+1
      IF (EOF2) GO TO 500
      CALL GETREF(IO2,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2,EOF2,SAVE2)
      CALL BLDKEY(KEY2,MREF2,TEXT2,NLA2)
      GO TO 100
C
C     EOF REACHED FILE 1
C     COPY REST OF DATA FROM FILE 2
C
400   CALL PUTREF(IO3,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2)
      NREF=NREF+1
      IF (EOF2) GO TO 999
      CALL GETREF(IO2,MREF2,NLA2,NLT2,NLR2,NLK2,TEXT2,EOF2,SAVE2)
      GO TO 400
C
C     EOF REACHED FILE 2
C     COPY REST OF DATA FROM FILE 1
C
500   CALL PUTREF(IO3,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1)
      NREF=NREF+1
      IF (EOF1) GO TO 999
      CALL GETREF(IO1,MREF1,NLA1,NLT1,NLR1,NLK1,TEXT1,EOF1,SAVE1)
      GO TO 500
C
C     WRAP IT UP
```

```
C
990   CLOSE(UNIT=IO1,DISPOSE='SAVE')
      CLOSE(UNIT=IO2,DISPOSE='SAVE')
      CLOSE(UNIT=IO3,DISPOSE='SAVE')
      RETURN
      END
C************************************************************************
      SUBROUTINE BLDKEY(KEY,MREF,TEXT,NLA)
      LOGICAL*1 TEXT(2625),KEY(34),TEST(5)
      INTEGER O
      DATA TEST/' ','A','N','D',' '/
      DO 5 I = 1,34
      KEY(I) = ' '
5     CONTINUE
      I=2
      O=1
10    IF (TEXT(I).EQ.'.') GO TO 20
      IF (TEXT(I).EQ.',') GO TO 20
      IF (TEXT(I).EQ.'-') GO TO 20
      IF (TEXT(I).EQ.' ') GO TO 20
C     GOOD CHARACTER FOR KEY
      KEY(O)=TEXT(I)
      O=O+1
20    I=I+1
      IF(O.GT.30) GO TO 100
      IF(I.GT.74) GO TO 100
      GO TO 10
C     ADD YEAR TO 30 CHARS OF AUTHOR NAME FIELDS
100   K1 = NLA*75 + 2
      K2 = K1+3
      K3 = 31
      DO 110 I = K1,K2
      KEY(K3)=TEXT(I)
      K3 = K3 + 1
110   CONTINUE
      RETURN
      END
```

```
C*****************************************************************
C
C     PDP536
C
C     GENERATES A LIST OF REFERENCE NUMBER FOR EACH KEYWORD
C
C     U S G S  -  G A R Y   S E L N E R
C******************************************************************
      LOGICAL*1 TEXT(2625),EOF,FILNAM(33),SECOND,PREV(12),NEW,
      1          KEY(12),PRTFIL(33),TITLE(80),SAVE(80)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,ISPEC,IO1,IO2,IO3,NL,
      1 LINCT,ICNT,IREF(10),PAGNO,IMAX,ITT
      REAL*4 DAT(3)
      DATA PREV/12*' '/,DAT/3*'    '/,IO1/1/,IO2/2/,IO3/3/,
      1    IMAX/60/,SAVE/80*' '/,ITT/5/
      CALL DATE(DAT)
      CALL TTINAA('ENTER INPUT FILENAME',20,FILNAM,33,ITT)
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      OPEN(UNIT=IO2,NAME='DB1:FOR002.DAT',TYPE='NEW')
      CALL TTINAA('ENTER FILENAME(OR DEVICE) FOR OUTPUT',36,PRTFIL,33,ITT)
      CALL TTINAA('ENTER TITLE(80 CHARS MAX)',25,TITLE,80,ITT)
      CALL TTINSI('ENTER STARTING PAGE NUMBER',26,PAGNO,ITT)
      EOF = .TRUE.
100   CALL GETREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      IF (NLK.EQ.0) GO TO 510
      SECOND=.FALSE.
      K1=(NLA+NLT+NLR)*75 + 2
300   DO 500 I = K1,K1+60,12
      DO 400 J = I,I+11
      K2 = J - I + 1
      IF (TEXT(J).NE.PREV(K2)) GO TO 410
400   CONTINUE
      GO TO 499
410   DO 420 J = I,I+11
      K2 = J-I+1
      KEY(K2)=TEXT(J)
420   CONTINUE
      WRITE(IO2,1999)  KEY,MREF
1999  FORMAT(12A1,I4)
499   CONTINUE
500   CONTINUE
      IF (NLK.EQ.1) GO TO 510
      IF (SECOND) GO TO 510
      K1 = (NLA+NLT+NLR+1)*75+2
      SECOND = .TRUE.
      GO TO 300
510   IF (.NOT.EOF) GO TO 100
      CLOSE(UNIT=IO1,DISPOSE='SAVE')
      REWIND IO2
      OPEN(UNIT=IO3,NAME='DB1:FOR003.DAT',TYPE='NEW')
      CALL SORTR(IO2,IO3,5,TEXT,16,1,16)
      CLOSE(UNIT=IO2,DISPOSE='DELETE')
      REWIND IO3
      OPEN(UNIT=IO1,NAME=PRTFIL,TYPE='NEW')
      WRITE(IO1,2000) TITLE,DAT
2000  FORMAT('1',//'0',16X,80A1,1X,2A4,A1//)
      LINCT = 6
      ICNT = 0
      SECOND=.TRUE.
10    READ(IO3,1999,END=999) KEY,MREF
      IF (.NOT.SECOND) GO TO 12
      SECOND=.FALSE.
      DO 11 I = 1,12
      PREV(I)=KEY(I)
11    CONTINUE
      IF (LINCT.GT.IMAX-3) CALL BREAK(IO1,LINCT,TITLE,PAGNO)
      WRITE(IO1,8010) KEY
      LINCT = LINCT + 2
12    NEW=.FALSE.
      DO 15 I = 1,12
      IF (PREV(I).EQ.KEY(I)) GO TO 14
      NEW=.TRUE.
14    CONTINUE
15    CONTINUE
      IF (NEW) GO TO 16
      ICNT = ICNT + 1
      IREF(ICNT) = MREF
      IF (ICNT.LT.10) GO TO 10
      IF (LINCT.GT.IMAX) CALL BREAK(IO1,LINCT,TITLE,PAGNO)
      WRITE(IO1,8000) IREF
8000  FORMAT(' ',34X,10I6)
      LINCT = LINCT + 1
      ICNT=0
      GO TO 10
16    IF (ICNT.EQ.0) GO TO 17
      IF (LINCT.GT.IMAX) CALL BREAK(IO1,LINCT,TITLE,PAGNO)
      WRITE(IO1,8000)(IREF(I),I=1,ICNT)
      LINCT=LINCT+1
17    ICNT=1
      IREF(ICNT)=MREF
      IF (LINCT.GT.IMAX-3) CALL BREAK(IO1,LINCT,TITLE,PAGNO)
      WRITE(IO1,8010) KEY
8010  FORMAT('0',20X,'-',12A1,'-',30(' -'))
      DO 18 I = 1,12
      PREV(I) = KEY(I)
18    CONTINUE
      LINCT = LINCT + 2
      GO TO 10
999   IF (LINCT.GT.IMAX) CALL BREAK(IO1,LINCT,TITLE,PAGNO)
      IF (ICNT.NE.0) WRITE(IO1,8000) (IREF(I),I=1,ICNT)
      CALL BREAK(IO1,LINCT,TITLE,PAGNO)
      CLOSE(UNIT=IO3,DISPOSE='DELETE')
      CLOSE(UNIT=IO1,DISPOSE='SAVE')
      STOP
      END
```

```
C************************************************************************
C
C     P D P 5 3 7
C
C     PRINTS A REFERENCE FILE IN STANDARD FORMAT
C
C     U S G S - G A R Y   S E L N E R
C
C***********************************************************************
      LOGICAL*1 TEXT(2625),KEY(34),EOF,FILNAM(33),TITLE(80),SAVE(80)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,IO1,IO2,IO3,IMAX,ISPC,
      1  LINCT,NL,MASFIL(33),PRTFIL(33),PAGNO,ITT
      REAL*4 DAT(3)
      DATA IO1/1/,IO2/2/,IO3/3/,DAT/3*'    '/,ISPC/1/,ITT/5/
      DATA SAVE/80*' '/
      CALL DATE(DAT)
      CALL TTINAA('ENTER INPUT FILENAME',20,FILNAM,33,ITT)
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      CALL OUTPT(ITT,IO1,IO2,DAT,TEXT)
999   CLOSE(UNIT=IO1,DISPOSE='SAVE')
      STOP
      END
```

```
C*******************************************************************
C
C     PDP545
C
C     RENUMBERS A REFERENCE FILE STARTING WITH ONE TO NUMBER CONTAINED
C     IN THE FILE.
C
C     U S G S  -  G A R Y  S E L N E R
C*******************************************************************
      LOGICAL*1 TEXT(2625),EOF,FILNAM(33),SAVE(80)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,IO1,IO2,NREF,ITT
      DATA SAVE/80*' '/,ITT/5/,IO1/1/,IO2/2/,NREF/1/
      CALL TTINAA('ENTER FILENAME',14,FILNAM,33,ITT)
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      OPEN(UNIT=IO2,NAME=FILNAM,TYPE='NEW')
      EOF = .TRUE.
100   CALL GETREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      CALL PUTREF(IO2,NREF,NLA,NLT,NLR,NLK,TEXT)
      NREF=NREF+1
      IF (.NOT.EOF) GO TO 100
      CLOSE(UNIT=IO1,DISPOSE='SAVE')
      CLOSE(UNIT=IO2,DISPOSE='SAVE')
      STOP
      END
      SUBROUTINE SORTR (IO1,IO2,ITT,RECADR,IRECSZ,KEYMIN,KEYMAX)
C     -----------------
C   CALLS 11/45 SORT ROUTINES TO SORT FILE IO1 (LUN = IO1).
C   RESULTS GO TO FILE IO2, ERROR MESSAGES GO TO FILE ITT.
C   KEY IS IN RIGHT ORDER ANYWHERE IN THE RECORD.
C          RECADR : ARRAY WHICH CONTAINS THE INPUT/OUTPUT RECORD
C          IRECSZ : BYTE COUNT OF RECORD SIZE
C          KEYMIN : BYTE NO OF BEGINNING OF KEY IN THE RECORD
C          KEYMAX : BYTE NO OF    END    OF KEY IN THE RECORD
C   NOTE : THE OUTPUT RECORDS HAVE A VARIABLE LENGTH; ALL THE BLANKS
C          AT THE END OF EACH RECORD ARE SUPPRESSED
C   WARNING : USE SPECIAL COMMAND FILE FOR TKB.  EXAMPLE :
C       PROG,PROG=PROG,[22,377]GENLBR/LB,[200,200]SORTS1,[200,200]SORTS2
C       /
C       ACTFIL=5
C       UNITS=10
C       MAXBUF=500
C       EXTSCT=$$FSR1:6200
C       ASG=SY0:1:2:3:4,TI:5
C       ASG=SY0:8:9:10
C       GBLDEF=FILES:3
C       GBLDEF=MULBUF:1
C       GBLDEF=RBTSZ:20
C       GBLDEF=FIRLUN:10
C       GBLDEF=INLUN:2
C       GBLDEF=OUTLUN:3
C       GBLDEF=RSTSSW:0
C       GBLDEF=XLUN:12
C       //
C
C   M.E.GETTINGS, MAR 77 / UPDATE M.DONZEAU, DEC 77, G.SELNER, FEB 80
C
      PARAMETER NW=5000
      PARAMETER NF=3
      LOGICAL*1 RECADR(IRECSZ),KEY(40)
      DOUBLE PRECISION SUBR(5)
      DIMENSION IWORK(NW),KEYS(20)
      EQUIVALENCE (KEY,KEYS)
      DATA SUBR/'RSORT','RELES','MERGE','RETRN','ENDS'/
100   FORMAT(<IRECSZ>A1)
101   FORMAT(<IR1>(250A1),<IR2>A1)
200   FORMAT(' ERROR IN ',A8,' , IERROR=',O3)
201   FORMAT(' RECORD NB : ',I5)
C-------------------- PRELIMINARY CALCULATIONS
      TEST(K)=FLOAT(K)/2.-K/2.
      KEYSIZ=KEYMAX-KEYMIN+1
      IF (TEST(KEYSIZ).NE.0.) KEYSIZ=KEYSIZ+1
      KEY(KEYSIZ)=1H
      KEYADR=KEYSIZ/2.
      MAXREC=IRECSZ
      IF (TEST(MAXREC).NE.0.) MAXREC=MAXREC+1
      IR1=IRECSZ/250
      IR2=MOD(IRECSZ,250)
      IF (IR1.LT.1) IR1=1
      IF (IR2.EQ.0) IR2=250
C-------------------- SORT PROCESSING
      CALL RSORT (IERROR,KEYSIZ,MAXREC,KEYS(KEYADR),IWORK(1),IWORK(NW),NF)
```

```
C       ----------
        IF (IERROR.NE.0) GOTO 901
        NREC=0
500     READ (IO1,101,END=550) RECADR
        NREC=NREC+1
        J=0
        DO 50 I=KEYMAX,KEYMIN,-1
        J=J+1
50      KEY(J)=RECADR(I)
        CALL RELES (IERROR,IRECSZ,RECADR)
C       ----------
        IF (IERROR) 902,500,902
550     CALL MERGE (IERROR)
C       ----------
        IF (IERROR.NE.0) GOTO 903
        NREC=0
560     CALL RETRN (IERROR,IRECSZ,RECADR)
C       ----------
        NREC=NREC+1
        IF (IERROR) 570,580,904
580     CALL LENGTH (RECADR,NCH,IRECSZ)
C       ------------
        WRITE(IO2,101) (RECADR(I),I=1,NCH)
        GOTO 560
570     CALL ENDS (IERROR)
C       ---------
        IF (IERROR.GT.0) GOTO 905
        RETURN
C-------------------- PRINT ERROR MESSAGES
901     ISUBR=1
        GOTO 910
902     ISUBR=2
        WRITE(ITT,201) NREC
        GOTO 910
903     ISUBR=3
        GOTO 910
904     ISUBR=4
        WRITE(ITT,201) NREC
        GOTO 910
905     ISUBR=5
910     WRITE(ITT,200) SUBR(ISUBR),IERROR
        STOP ' *** PROGRAM STOPS IN ROUTINE SORTR ***'
        END
```

```
      SUBROUTINE LENGTH(CHARAY,NCH,LEN)
      LOGICAL*1 CHARAY(LEN)
      NCH = 0
      K = LEN+1
      DO 10 I = 1,LEN
      J = K-I
      IF (CHARAY(J).EQ.' ') GO TO 9
      NCH = J
      GO TO 999
9     CONTINUE
10    CONTINUE
      NCH = 0
999   RETURN
      END
      SUBROUTINE GETREF(IOU,MREF,NLA,NLT,NLR,NLK,TEXT,IEOF,SAVE)
      LOGICAL*1 TEXT(2625),IEOF,SAVE(80)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,IOU
C     LOCAL VARIABLES
      LOGICAL*1 INPUT(75)
      INTEGER*2 PTR,IPREV,NIN
      PTR = 0
      NLA=0
      NLT=0
      NLR=0
      NLK=0
      DO 10 I = 1,2625
      TEXT(I)=' '
10    CONTINUE
      DECODE(80,1000,SAVE)NIN,INPUT
      IPREV=NIN
      IF (.NOT.IEOF) GO TO 110
      READ(IOU,1000,ERR=980,END=998) NIN,INPUT
1000  FORMAT(I4,1X,75A1)
      IEOF=.FALSE.
      IPREV=NIN
      GO TO 110
100   READ(IOU,1000,ERR=980,END=998) NIN,INPUT
110   IF (NIN.NE.IPREV) GO TO 999
      IF (INPUT(1).EQ.'A') GO TO 200
      IF (INPUT(1).EQ.'T') GO TO 300
      IF (INPUT(1).EQ.'R') GO TO 400
      IF (INPUT(1).EQ.'K') GO TO 500
      IF (INPUT(1).EQ.' ') GO TO 600
      TYPE 1010,NIN,INPUT
1010  FORMAT(' ','INVALID CARD IN MASTER:'/
     1 ' ',I4,1X,75A1)
      GO TO 100
200   NLA=NLA+1
      GO TO 600
300   NLT=NLT+1
      GO TO 600
400   NLR=NLR+1
      GO TO 600
500   NLK=NLK+1
600   DO 610 I = 1,75
      KI=PTR+I
      TEXT(KI)=INPUT(I)
610   CONTINUE
      PTR=PTR+75
      IF (PTR.LT.2625) GO TO 100
      TYPE 1020,NIN,INPUT
1020  FORMAT(' ','FOLLOWING RECORD WAS AT MAX RECORD SIZE:'/
     1 ' ',I4,1X,75A1)
      STOP
C     READ ERROR
980   TYPE 1030,IPREV
1030  FORMAT (' ','ERROR READING: PREV OR CURRENT REF= ',I4)
      GO TO 100
C     EOF READ
998   IEOF=.TRUE.
999   MREF=IPREV
```

```
ENCODE(80,1000,SAVE)NIN,INPUT
RETURN
END
```

```
      SUBROUTINE PUTREF(IOW,MREF,NLA,NLT,NLR,NLK,TEXT)
      LOGICAL*1 TEXT(2625),DUMMY,OUT(75)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK
C     LOCAL VARIABLES
      INTEGER*2 NL-K1,K2,NCH
      NL = NLA+NLT+NLR+NLK
      DO 200 1 = 1,NL
      K1=(I-1)*75 +1
      K2 = K1 + 74
      DO 100 J=K1,K2
      K = J-K1+1
      OUT(K) = TEXT(J)
100   CONTINUE
      CALL LENGTH(OUT,NCH,75)
      WRITE(IOW,1000) MREF,(OUT(J),J=1,NCH)
1000  FORMAT(I4,1X,75A1)
200   CONTINUE
      RETURN
      END
      SUBROUTINE OUTPT(ITT,IO2,IOUT,DAT,TEXT)
      LOGICAL*1 SUPPR,TITLE(80),TEXT(2625),Y1,EOF,N1,FILNAM(33),SAVE(80)
      INTEGER*2 IMAX,NL,IO2,LINCT,MREF,NLA,NLT,NLR,NLK,ITT,IOUT,PAGNO
      REAL*4 DAT(3)
      DATA Y1/'Y'/,IMAX/60/,N1/'N'/,SAVE/80*' '/
      CALL TTINAA('ENTER FILENAME OR DEV: FOR OUTPUT',33,FILNAM,33,ITT)
      OPEN(UNIT=IOUT,NAME=FILNAM,TYPE='NEW')
      CALL TTINAA('ENTER TITLE(80 CHAR MAX)',23,TITLE,80,ITT)
      CALL TTINAA('DO YOU WISH TO SUPPRESS REF NO AND KEYWORDS(Y/N)',48,
     1  SUPPR,1,ITT)
      CALL TTINSI('ENTER STARTING PAGE NUMBER',26,PAGNO,ITT)
      WRITE(IOUT,2030) TITLE,DAT
      LINCT=6
      EOF=.TRUE.
10    CALL GETREF(IO2,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      ILINE = 1
      ILINE = ILINE + NLA+NLT+NLR
      IF (SUPPR.EQ.Y1) GO TO 20
      ILINE=ILINE + NLK
20    IF (LINCT+ILINE.GT.IMAX) CALL BREAK(IOUT,LINCT,TITLE,PAGNO)
      IF (SUPPR.EQ.Y1) WRITE(IOUT,2020)(TEXT(J),J=2,75)
      IF (SUPPR.EQ.N1) WRITE(IOUT,2000)MREF,(TEXT(J),J=2,75)
      DO 30 I = 2,NLA+NLT+NLR
      K1 = (I-1)*75+2
      K2 = K1+73
      WRITE(IOUT,2010) (TEXT(J),J=K1,K2)
30    CONTINUE
      IF (SUPPR.EQ.Y1.OR.NLK.EQ.0) GO TO 40
      K1 = (NLA+NLT+NLR)*75 + 2
      K2 = K1 + 71
      WRITE(IOUT,2040) (TEXT(J),J=K1,K2)
      IF (NLK.EQ.1) GO TO 40
      K1 = (NLA+NLT+NLR+1)*75+2
      K2 = K1 + 71
      WRITE(IOUT,2040) (TEXT(J),J=K1,K2)
40    LINCT=LINCT+ILINE
      IF (EOF) GO TO 999
      GO TO 10
999   CALL BREAK(IOUT,LINCT,TITLE,PAGNO)
      WRITE(IOUT,2050)
      CLOSE(UNIT=IOUT,DISPOSE='SAVE')
      RETURN
2000  FORMAT('0',3X,I4,7X,74A1)
2010  FORMAT(' ',16X,74A1)
2020  FORMAT('0',14X,74A1)
2030  FORMAT('1',///'0',16X,80A1,1X,2A4,A1//)
2040  FORMAT(' ',15X,6(' ',12A1))
2050  FORMAT('0','END OF LISTING')
      END
      SUBROUTINE BREAK(IOUT,LINCT,TITLE,PAGNO)
      LOGICAL*1 TITLE(80),CONT(9)
      INTEGER*2 LINCT,IOUT,PAGNO
      DATA CONT/' ','(','C','O','N','T','''','D',')'/
      IF (LINCT.EQ.60) GO TO 500
      K = 60-LINCT
      DO 100 I = 1,K
      WRITE(IOUT,1010)
1010  FORMAT(' ',1X)
100   CONTINUE
500   WRITE(IOUT,1010)
      WRITE(IOUT,1020) PAGNO
1020  FORMAT(' ',51X,I4)
      PAGNO = PAGNO + 1
      WRITE(IOUT,1000) TITLE,CONT
1000  FORMAT('1',//'0',16X,80A1,1X,9A1//)
      LINCT = 6
      RETURN
      END
      SUBROUTINE BIGREF(KEYLEN,KEY,IO2,MREF,NLA,NLT,NLR,NLK,TEXT,EOF)
      LOGICAL*1 TEXT(2625),EOF,KEY(KEYLEN)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,IO2,KEYLEN
      DO 200 I = 1,2625
      TEXT(I)=' '
200   CONTINUE
      READ(IO2,1999,END=999) KEY,MREF,TEXT
1999  FORMAT(<KEYLEN>A1,I4,35(75A1))
      NLA=0
      NLT=0
      NLR=0
      NLK=0
      DO 100 I = 1,2551,75
      IF (TEXT(I).EQ.'A') NLA=NLA+1
      IF (TEXT(I).EQ.'T') NLT=NLT+1
      IF (TEXT(I).EQ.'R') NLR=NLR+1
      IF (TEXT(I).EQ.'K') NLK=NLK+1
      IF (TEXT(I).EQ.'A'.OR.TEXT(I).EQ.'T'.OR.TEXT(I).EQ.'R'
     1 .OR.TEXT(I).EQ.'K'.OR.TEXT(I).EQ.' ') GO TO 99
      TYPE 1998,MREF
1998  FORMAT(' ','INVALID CARD CODE MREF:',I4)
99    CONTINUE
100   CONTINUE
222   RETURN
999   EOF = .TRUE.
      GO TO 222
      END
```

```
C*******************************************************************
C
C     PDP545
C
C     RENUMBERS A REFERENCE FILE STARTING WITH ONE TO NUMBER CONTAINED
C     IN THE FILE.
C
C     U S G S  -  G A R Y   S E L N E R
C********************************************************************
      LOGICAL*1 TEXT(2625),EOF,FILNAM(33),SAVE(80)
      INTEGER*2 MREF,NLA,NLT,NLR,NLK,IO1,IO2,NREF
      DATA SAVE/80*' '/
      IO1 = 1
      IO2 = 2
      NREF = 1
      TYPE 1000
1000  FORMAT(' ','ENTER FILENAME:'$)
      ACCEPT 1010,FILNAM
1010  FORMAT(33A1)
      FILNAM(33)=0
      OPEN(UNIT=IO1,NAME=FILNAM,TYPE='OLD')
      OPEN(UNIT=IO2,NAME=FILNAM,TYPE='NEW')
      EOF = .TRUE.
100   CALL GETREF(IO1,MREF,NLA,NLT,NLR,NLK,TEXT,EOF,SAVE)
      CALL PUTREF(IO2,NREF,NLA,NLT,NLR,NLK,TEXT)
      NREF=NREF+1
      IF (.NOT.EOF) GO TO 100
      CLOSE(UNIT=IO1,DISPOSE='SAVE')
      CLOSE(UNIT=IO2,DISPOSE='SAVE')
      STOP
      END
      SUBROUTINE TTINAR (QUE,NQ,A,NA,ITT)
C     -----------------
C   GENERAL TTY ENTRY OF 1D REAL ARRAY DATA
C     QUE       ALPHANUMERIC TEXT CONTAINING THE QUESTION
C     NQ        NUMBER OF CHARACTERS OF TEXT 'QUE'
C     A         1D REAL ARRAY
C     NA        DIMENSION OF ARRAY A
C     ITT       L.U.N. OF TTY
C
C
C   ENTRIES :
C   --------
C
C     TTINAI (QUE,NQ,IA,NA,ITT)
C     ------
C   GENERAL TTY ENTRY OF 1D INTEGER ARRAY DATA
C     IA        1D INTEGER ARRAY
C
C     TTINAA (QUE,NQ,IB,NA,ITT)
C     ------
C   GENERAL TTY ENTRY OF 1D ALPHANUMERIC ARRAY DATA (LOGICAL*1)
C     IB        1D ALPHANUMERIC ARRAY (LOGICAL*1)
C
C     TTINSR (QUE,NQ,B,ITT)
C     ------
C   GENERAL TTY ENTRY OF REAL VARIABLE
C     B         REAL VARIABLE
C
C     TTINI2 (QUE,NQ,J,ITT)     OF    TTINSI (QUE,NQ,J,ITT)
C     ------                          ------
C   GENERAL TTY ENTRY OF INTEGER*2 VARIABLE
C     J         INTEGER*2 VARIABLE
C
C     TTINI4 (QUE,NQ,D,ITT)
C     ------
C   GENERAL TTY ENTRY OF INTEGER*4 VARIABLE
C     D         INTEGER*4 VARIABLE
C
C     TTINDR (QUE,NQ,C,ITT)
C     ------
C   GENERAL TTY ENTRY OF DOUBLE PRECISION VARIABLE
C     C         DOUBLE PRECISION VARIABLE
C
C
C   M.E.GETTINGS, MAY 77. / UPDATE M.D., DEC 77.
C
      LOGICAL*1 QUE(NQ),OR,IL1,IB(NA),NULL
      DIMENSION A(NA),IA(NA)
      REAL*8 C
      INTEGER*4 D
      DATA IL1/1H /,NULL/"000/
100   FORMAT(F10.0)
101   FORMAT(80A1)
102   FORMAT(I20)
103   FORMAT(Q,80A1)
104   FORMAT(1X,<NQ>A1,' : '$)
105   FORMAT(F17.0)
```

```
C
      WRITE(ITT,200) QUE
200   FORMAT(1X,79A1/' ENTER ARRAY ELEMENTS UPON PROMPT')
      DO 300 I=1,NA
301   WRITE(ITT,201) I
201   FORMAT(' A(',I3,' ) : ',$)
      READ(ITT,100,ERR=301) A(I)
      WRITE(ITT,202) I,A(I)
202   FORMAT(' A(',I3,' ) =',1PE14.7,' OK ? ',$)
      READ (ITT,101) QR
      IF (QR.EQ.'N') GO TO 301
300   CONTINUE
      RETURN
C
      ENTRY TTINAI (QUE,NQ,IA,NA,ITT)
C     ------------
C   GENERAL TTY ENTRY OF 1D INTEGER ARRAY DATA
C
      WRITE(ITT,200) QUE
      DO 310 I=1,NA
311   WRITE(ITT,201) I
      READ (ITT,102,ERR=311) IA(I)
      WRITE(ITT,210) I,IA(I)
210   FORMAT(' IA(',I3,' ) =',I10,' OK ? ',$)
      READ (ITT,101) QR
      IF(QR.EQ.'N') GO TO 311
310   CONTINUE
      RETURN
C
      ENTRY TTINAA (QUE,NQ,IB,NA,ITT)
C     ------------
C   GENERAL TTY ENTRY OF 1D ALPHANUMERIC ARRAY DATA (LOGICAL*1)
C
321   DO 320 I=1,NA
320   IB(I)=IL1
      WRITE(ITT,104) QUE
      READ (ITT,103) NO,IB
      IF (NO.LE.0) NO=1
      WRITE(ITT,220) (IB(I),I=1,NO)
220   FORMAT(1X,<NO>A1/' OK ? '$)
      READ (ITT,101) QR
      IF (QR.EQ.'N') GOTO 321
C   ADD NULL CHARACTER IN FIRST BLANK (FOR FILENAME INPUT)
      IF (NO.GE.NA) GO TO 900
      IB(NO+1)=NULL
900   RETURN
C
      ENTRY TTINSR (QUE,NQ,B,ITT)
C     ------------
C   GENERAL TTY ENTRY OF REAL VARIABLE
C
340   WRITE(ITT,104) QUE
      READ (ITT,100,ERR=340) B
      WRITE(ITT,240) B
240   FORMAT(' VALUE : ',1PE14.7,' , OK ? '$)
```

```
      READ (ITT,101) QR
      IF (QR.EQ.'N') GO TO 340
      RETURN
C
      ENTRY TTINI2 (QUE,NQ,J,ITT)
C     ------------
      ENTRY TTINSI (QUE,NQ,J,ITT)
C     ------------
C   GENERAL TTY ENTRY OF INTEGER*2 VARIABLE
C
350   WRITE(ITT,104) QUE
      READ (ITT,102,ERR=350) J
      WRITE(ITT,250) J
250   FORMAT(' VALUE : ',I20,' , OK ? '$)
      READ (ITT,101) QR
      IF (QR.EQ.'N') GO TO 350
      RETURN
C
      ENTRY TTINI4 (QUE,NQ,D,ITT)
C     ------------
C   GENERAL TTY ENTRY OF INTEGER*4 VARIABLE
C
360   WRITE(ITT,104) QUE
      READ (ITT,102,ERR=360) D
      WRITE(ITT,250) D
      READ (ITT,101) QR
      IF (QR.EQ.'N') GO TO 360
      RETURN
C
      ENTRY TTINDR (QUE,NQ,C,ITT)
C     ------------
C   GENERAL TTY ENTRY OF DOUBLE PRECISION VARIABLE
C
400   WRITE(ITT,104) QUE
      READ (ITT,105,ERR=400) C
      WRITE(ITT,260) C
260   FORMAT(' VALUE : ',D24.17,' , OK ? '$)
      READ (ITT,101) QR
      IF (QR.EQ.'N') GO TO 400
      RETURN
      END
```