KARP

Kansas Automated Retrieval Project

by

Michael J. McCullagh
Kansas Geological Survey
and
Department of Geography
The University of Kansas
Lawrence, Kansas

1972

TABLE OF CONTENTS

KARP

Kansas Automated Retrieval Project

Section A - General

Introduction

Many systems are available today for automated handling of bibliographies compiled by different disciplines. The most famous ones are the Chemical Society's Chemical Abstracts and Chemical Titles, and the National Institute of Health's Medlars scheme. While these are of great benefit to researchers in these respective fields there are few available systems for small bibliographic collections such as those undertaken by an individual or a small department. These collections, varying in size between a few hundred and a few thousand references, suffer from the distinct disadvantage that there is no way of retrieving information from them in a fast manner without the imposition of some mechanical system: effectively, articles become irretrievable. It would appear that an automated retrieval system that could deal with these problems of size and display in some meaningful way would be a useful addition to the research capability of an academic.

The design criteria of such a system have to allow for the small scale of operation. Where the large nation-wide systems can be relatively slow in disseminating information, it would be expected that a small system would be able to provide output almost instantaneously and hence would have to operate either in time sharing on a remote computer terminal or in a simulation thereof. Secondly, some of the advantages of the large system approach would have to be sacrificed to enable this small system to be viable; for instance, free text searching would probably be incompatible with a small time sharing system because of time and input/output considerations.

The Kansas Automated Retrieval Project was set up with the idea of developing a small easily used interactive system for editing, storing, and retrieving information in the form of bibliographic references from a set of disc files. A keyword approach was used as it is less cumbersome to incorporate than free text searching procedures when the minimum of text can be stored in the system. The design limit of the system is intended to be about 3000 records, with no lower limit. This capacity should be adequate for most researchers' needs for private bibliographic collections. The system has been designed to be as general as possible, and was in fact tested initially using a 1300 photographic slide set as the basis for the data file. Other applications will be listed later.

Operation of the System

The KARP system is broken down into a sequence of four individual programs written in FORTRAN. The decision to break the system into four programs was made to insure the fastest possible interaction time and the lowest possible costs of operation commensurate with the completion of a given specific goal for each program. The four sections are EDITOR, UPDATE, SEARCH, and LIST. As the names suggest: EDITOR is responsible for the editing and correction of material entering the system, UPDATE for placing the edited file into permanent disc storage, and SEARCH for searching an index to find the answers to specific user questions. LIST operates in the batch mode as it produces a listing of records based on keyword composition. This would be an impossible task on a remote terminal and is hence sent for printing to the high-speed line-printer.

These four programs use as input files created by the previous program in the sequence. Thus the original data input is on a file called INPUT which could be either cards or a previously created disc file. When using the system in a full time sharing mode it would be usual to store this input file on disc. If the system is being run entirely in the batch mode in a time sharing simulation the data files could be on disc, drum, or tape, with the input most likely on cards. As most medium to large sized machines have some time sharing capability the system has specifically been designed for time sharing. Altogether six files are used: INPUT - which holds incoming data; INFILE which contains the material in the process of editing; OUTFILE as a scratch working file; KEYLIST - a temporary file holding the newly generated keyword set coupled between EDITOR and UPDATE; PERMFILE - a permanent file to store the updated bibliography; and KEYWORD, also permanent, to hold the cross reference listing on which the searches will be made and upon which the program LIST operates.

The development of these programs has followed the principle of using as little core storage as possible compatible with a reasonably efficient program. This is because the effectiveness of a user on a time sharing system is related to the amount of core store he is occupying. The more core affected the less frequently will the machine run part of his program. Where there is going to be a lot of interaction between the program and the user, it is advisable to keep the store requirements as small as possible to insure faster response. Thus the EDITOR program requires about 500 words storage whereas the UPDATE program can use 2500 words as the amount of interaction is small.

FORTRAN Dialect

The programs were originally written for the Honeywell 635 computer using a time sharing FORTRAN compiler that contained various departures from standard FORTRAN IV. The major differences are that:

1) PRINT is-used instead of WRITE in many situations, especially when writing interactively, in a FORTRAN II manner.

2) REWIND n is replaced by BEGINFILE n, and where the file is later to be written upon, by BEGINFILE n followed by ENDFILE n. In most other compilers the REWIND n statement should be sufficient.

3) Connected with the file manipulation is the use of FILENAME to allocate the name of a file. Where standard FORTRAN is used, the equivalent will usually be a logical unit number for the drum/disc/tape file.

4) The & sign is used as a continuation indicator rather than the more customary number in the 6th column of the card.

5) Formatted reads and writes to files are used where binary I/O would be faster.

Apart from these differences, there should be no trouble in converting to another version of FORTRAN.

All the programs use the Free Format Function (McCullagh, 1972) type of input system that allows the user to punch data into the machine without regard for format. The system is briefly explained in a later section.

Section B - EDITOR

Use of the EDITOR Program

The EDITOR program, as its name suggests, is a program that enables the incoming data in the form of cards or a disc file, to be edited and any necessary corrections made before the PERMFILE or KEYWORD cross reference index is updated. It is structured in an interactive mode so that corrections to the input file can be made, checked, and recorrected if necessary, all in the same computer run. The program runs using a series of commands that act on the incoming data. In order to understand these commands it is necessary first to outline the type of data format used. A bibliographic reference can be represented for purposes of retrieval by the following characteristics:

1) Identification number,

2) Author list,

3) Date,

4) Keyword list.

Keyword Choice

The first three items are self-explanatory but the fourth needs consideration. A keyword is a word that is a key to the understanding of the content of a passage. Thus, if a particular bibliographic reference has a title: "A bibliography of papers concerning the Gettysburg Address," it would be reasonable to assume that "bibliography" would be one of the keywords chosen to represent the content of that paper. Other keywords may well suggest themselves on reading the paper. Often the title as such will not be entirely relevant to the subject matter of the article and in such cases care is necessary to insure that no mistakes are made in the choice of keywords. Also the choice will depend on the type of computer system that is being developed. A bibliography concerned with history would put far more emphasis on the Gettysburg Address than one on computer mapping. In fact, the Gettysburg paper would not appear in such a listing!

Choosing the appropriate keyword set of a given bibliography is one of the most difficult and subjective aspects of setting up a storage and retrieval system. One method to circumvent this is to use a free text searching procedure operating on the actual title or abstract of a paper, but this leads to errors; especially with non-scientific title searches because titles do not well represent the contents, and abstracts only slightly better. A thorough keyword search by a researcher in the same field as the article has many advantages but necessarily involves a lot of voluntary labor. For this reason many organizations use the free text method in preference as the title and abstract can both be searched and the probability of relevance is still fairly high. If only a title is used for the search, however, the probability of success declines.

In the present case, it is assumed that the researcher is familiar with the field in which he is working and hence that he can develop a good set of keywords. Any article for which he retains an index card will have been read and hence with only a little more effort could have been keyworded at the same time. A common difficulty is that when one starts to compile a keyword index one's idea of the best set of keywords is completely different from the set chosen in the end. Therefore, the set of keywords has to be built up over a period of time, and no attempt should be made to define a keyword set until a representative selection of articles has been considered. Another problem is the choice of the generality of the keyword set. For instance in an automated bibliography on the availability of computing machines it would be pointless to have a keyword "computer." But would it be worthwhile to include "IBM"? This word may be present in over half the references but would be needed nevertheless to distinguish between different types of machines. At the other extreme, there would be little point in having a keyword "Programma 101"; a better choice would be "programmable calculating machine."

Data Input to EDITOR

The program accepts the four categories of data listed above as a card image sequence. In the program listing of EDITOR the format given is (3X,I5,5A4,I2,10A4). The I5 field reads the identification number of the paper: the 5A4, the author name (16 of the 20 characters) and the date (the remaining four characters of the field). I2 is the number of keywords chosen to represent that paper. This number can be in the range 1 to 15. The first five keywords appear in the 10A4 field of the first card: thus every keyword has up to 8 significant characters, two computer words to every keyword. If the number of keywords is greater than 5, some form of continuation card for the remaining keywords is needed. This is input in the format (3X,I5,20X, I2, 10A4). The identification number is used to check that the continuation card has the same number. The I2 field should be kept blank as an additional check that the right card is following the original.

An example of a complete card could be:

```
         1         2         3         4         5         6         7
1234567890123456789012345678901234567890123456789012345678901234567890

     345SCREWTAPE,D BEEL1972 8DEVILS SATANIC WORMWOODUNCLE

     345                      ADVICE TORMENT ETERNAL
```

These cards are stored in the file INPUT.

The free format routine is not used to read in the INPUT data as usually they will be punched up on cards and can easily be verified beforehand. The interactive commands are read in by free format, however, as otherwise many program failures would occur owing to incorrect formatting at the terminal.

No two references should have the same identification number. In the later retrieval phases this would lead to confusion! On the other hand, the numbers do not have to be sequential, but the alphabetic order should correspond to the numeric one. Thus the sequence of IDS many be 124, 153, 183, 195, 305, 267, etc. The reason for this is that when the new references are merged with the PERMFILE it is assumed that that file is in numeric order and the incoming cards will be merged into the right gaps in the sequence. It is therefore worthwhile to leave fairly large gaps in the series of ID numbers at the beginning of a bibliographic project so that infilling can occur later. This means that the actual articles can then be sorted by ID number or author alphabetic order and still be in sequence. This will not always be possible, especially when the gaps in available IDs are filled. It is then necessary to start placing the articles in a second series of alphabetic order thus giving a continuous numbering sequence but with two complete cycles through the alphabet. This process can be continued indefinitely. Thus if a sequence of IDs is started 5, 10, 15, 20, 25, ...

100, when one of the gaps is full, maybe between 15 and 20, a new alphabetic series could be started from 105 to 200.

Editing Procedure

Many time sharing systems have some form of inbuilt editing commands. It is awkward, however, to run a program to generate a set of keywords, summon up the text editor to make corrections, and then run the keyword question and listing program again. For this reason EDITOR contains its own set of editing commands so that the whole process can take place under one program's control. These commands and most of their associated data are read in by free format. A list of the free format commands and data is given below:

	Command	Subsidiary Command	Data field 1	Data field 2
1)	BEGINFILE	-	-	-
2)	COPY	-	-	-
3)	DELETE	CARD	N	-
		KEYWORD	N	M
4)	INSERT	CARD	N	-
		KEYWORD	N	M
5)	NEWKEY	-	-	-
6)	PRINT	CARD	N	-
		KEYLIST	-	-
		KEYWORD	-	-
7)	READCARDS	-	K	-
8)	REPLACE	AUTHOR	N	-
		CARD	N	-

		DATE	N	-
		KEYWORD	N	M
9)	SUBSTITUTE	-	-	-
10)	STOP	-	-	-

It can be seen that there are ten main commands and a selection of adjectives describing them. In the data fields K is the number of cards to be read, M is the position of the keyword, author, or date to be replaced, and N is the card number in question. In addition to this data set, some of the commands require alphanumeric input as a character string to identify, insert, or replace an author name, date, or keyword.

Only the first four letters of any keyword are significant; thus, instead of typing READCARDS, the system would accept just READ. Similarly, instead of SUBSTITUTE, one could use SUBS. Any mistakes in spelling occurring after the fourth letter will be ignored; any before are significant and the program will ask for a new command to be given. Note that any commands typed at the same time as the error will have to be re-entered. The delimiter between commands, and the data for commands, is one or more spaces. Thus a command sequence might appear as follows:

```
READCARDS  100 NEWKEY PRINT KEYLIST STOP
```

The program would read in 100 cards (or bibliographic references which might contain more than one physical card, but a total of 100 'logical' cards), compute a keyword occurrence listing, print the list of keywords found, and stop execution of the program. Commands can be mixed in any logical order and up to 72 columns of the card image can be filled with commands. In this way it is possible to execute more than one command sequence at a time before the program will ask for another set of commands.

The input of commands to the program is accomplished by typing data in response to the printing of either:

```
ENTER COMMAND    or
ENTER DATA
```

The first is expecting an alpha command and the latter a number defining some parameter of the command. Either piece of information can be entered at this time or a whole series of command sequences. It is important to realize that if a series is entered the program will follow the sequence rigidly and that any missing data may cause the program to fail, or make it difficult to retrieve an error. An example of this is given later. If a subsidiary command is given without a main command, for instance:

```
CARD 146 COPY   instead of
DELETE CARD 146 COPY
```

the program will print:

```
COMMAND RECOGNIZED BUT HIGHER ORDER COMMAND NEEDED.
YOUR COMMAND WAS
CARD 146 COPY
TRY AGAIN
```

If a command is totally unrecognizable, the program will print:

```
ILLEGAL COMMAND
```

and ask by means of ENTER COMMAND for the command name to be repeated.

Elucidation of Editing Commands

1) BEGINFILE

BEGINFILE has the sole function of setting all reading and writing pointers on files INFILE and OUTFILE back from whatever positions they occupied to the very beginning of the files. This is the same as rewinding all the files. If a mistake has been made during the editing procedure which must be removed, BEGINFILE will simply reset the file to its unchanged position. As no call of COPY has been made, there will be no copying of the amended version of INFILE, that is kept in OUTFILE, back into INFILE. On successful completion, the message

```
POINTERS AT BEGINNING OF FILES
```

will be printed.

2) COPY

This command saves any changes that have been made and written to OUTFILE back into INFILE. Assume that some data had been read into INFILE from INPUT and then had been edited by using commands like REPLACE, INSERT, or DELETE. It is possible that the last card altered was not in fact the last card in the sequence read by READCARDS. All alterations and editing act on INFILE, but write the corrected version out to OUTFILE. Thus the corrected version needs to be copied back into INFILE before further editing takes place as OUTFILE is only a temporary file, whereas INFILE is the file passed onto the program UPDATE. Thus COPY copies any unread remainder of INFILE to OUTFILE and then copies the whole of OUTFILE, containing the corrected version, back to INFILE. The message printed on successful completion of this task is:

```
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
```

3) DELETE

This is the first of the editing commands. Either a card or a keyword can be deleted. If a card, all trace of it will disappear from INFILE; if a keyword, the number of keywords will be automatically reset to one less. Both require that the ID number of the reference in question be given so that the correct card can be found in INFILE. If a card of that ID does not appear in INFILE the message:

```
NO CARD OF ID XXX FOUND IN FILE
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
```

will be printed. As can be seen an automatic COPY command is initiated. Note that all the editing commands act from where the previous command left off. In other words, if a previous card of ID 250 was amended or printed, and the next command looks for an ID of 170, no card will be found as the pointers are beyond that point in INFILE. Therefore, editing should be performed in a sequential manner from the beginning of the file to the end, with a COPY command at the end of any particular sequence of editing. This will be demonstrated in the example later on. A typical command to delete a card might be:

```
DELETE CARD 124   DELETE KEYWORD 167 2
```

The second series of commands for deleting the keyword needs some explaining. The deleted keyword is on card 167 and is the second keyword on the card. It may be difficult to know which position the keyword occupies. For this reason there is a command PRINT CARD which will print out the card before the DELETE command takes effect. The sequence could be as follows:

DELETE CARD 124 PRINT CARD 167 DELETE KEYWORD 167

Once the card 167 has b-en printed the aberrant keyword can be readily identified as the second. As no more commands occur in the previous command list, the program will ask:

```
ENTER DATA
```

to which the response is:

```
2
```

and the keyword will then be deleted. Note that it is possible to perform more than one operation on a card before passing to the next card. Successful completion of the DELETE command is indicated by:

```
CARD XXX DELETED
```

for DELETE CARD, and:

```
DELETED KEYWORD IS XXXXXXXX
```

for the DELETE KEYWORD command.

4) INSERT

INSERT uses two subsidiary commands, either CARD or KEYWORD. If a card is to be inserted, the card ID given in data field one is the card immediately preceding the point at which insertion is desired. Thus, if card 197 is to be inserted after card 195 but before 200, the INSERT command would be:

```
INSERT CARD 195
```

On successful completion of card insertion the program will print out:

```
INSERTED CARD READS AS FOLLOWS
```

followed by a listing of the card. The card that is inserted is the next one read from the file INPUT. Thus if 100 cards were read originally by READCARDS, the inserted card will be the 101st.

Inserting a keyword follows much the same process as deleting one except that the keyword itself has to be typed in the form of a character string. Thus if the sequence is:

```
INSERT CARD 195 INSERT KEYWORD 234
```

the program will perform the insertion after card 195 then look for card 234, and having found it print the message:

```
ENTER CHARACTER STRING
```

to which the response is the desired keyword. The keyword must be typed in column one onwards as it is read in 2A4. The answer should be:

```
INSERTED KEYWORD IS XXXXXXXX
```

5) <u>NEWKEY</u>

The purpose of the editing program is to eliminate errors from INFILE so that when UPDATE calculates the cross reference index all the keywords included will be valid ones. It is therefore necessary to be able to check the keywords present in INFILE to determine whether further editing is necessary. NEWKEY creates the entire keyword list for INFILE and sends a copy of it, together with the frequency of occurrence of every keyword to the temporary file KEYLIST. No printing of the keyword listing is given, only the message:

```
KEYWORD LIST GENERATED
```

to indicate success. Typing PRINT KEYLIST will print a list of the keywords and their frequencies.

6) <u>PRINT</u>

The subsidiary commands associated with PRINT are CARD, KEYLIST, and KEYWORD. PRINT CARD does as could be guessed, print out a given card! The command sequence for printing a card with ID 345 would be:

```
PRINT CARD 345
```

which would give the following output for the example of an input card given in the section on data input to EDITOR:

```
ID IS 345  AUTHOR IS  SCREWTAPE,D BEEL  AND DATE 1972
THERE ARE  8 KEYWORDS, WHICH ARE
DEVILS  SATANIC  LETTERS  WORMWOOD  UNCLE
ADVICE  TORMENT  ETERNAL
```

If the card desired does not exist, the message:

```
NO CARD OF ID XXX FOUND IN FILE
```

is printed and the file saved by an automatic COPY with whatever corrections had been made up to that point.

PRINT KEYLIST prints out the keyword listing in its entirety together with the frequency of occurrence of every keyword. See the example given later for an indication of the printing format. The only failures are catastrophic and would be related to the fact that the file KEYLIST had not previously been written. A call on NEWKEY must precede any attempt to use PRINT KEYLIST.

PRINT KEYWORD can be used to find those cards on which a certain keyword exists. For instance, an error may have been made in the punching of the input reference cards so that two versions of one keyword exist, such as SURFACE and SURFACES, with frequencies of, respectively, 1 and maybe 10. So that the file can be edited in a sequential manner, it is vital to know which card contains the keyword SURFACE so that it can later be altered to SURFACES. PRINT KEYWORD has no data requirement but does need the keyword input as a character string. The program will print:

```
ENTER CHARACTER STRING
```

to which the response would be:

```
SURFACE
```

The output from the search could be:

```
KEYWORD OCCURS ON FOLLOWING CARDS
267
POINTERS AT BEGINNING OF FILES
```

Thus the SURFACE keyword is on card 267 only. If the keyword had not been found on any cards the output from the program would only have said:

```
POINTERS AT BEGINNING OF FILES
```

Note that as this command does not alter the file in any way. No COPY command and only an automatic BEGINFILE are necessary.

7) READCARDS

The function of READCARDS has been previously mentioned. Its purpose is to read a given number of cards into INFILE so that they can be edited. To read 100 cards would require:

```
READCARDS 100
```

and the only printed output is:

```
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILE
```

unless fewer than the required number of cards can be read. If 70 cards are in INPUT the program will fail catastrophically. No attempt is made to test for end of file. It is possible that some cards will not be complete or that the ID on them does not match the ID of any subsequent cards referring to the same article. In this case the entire logical card will be omitted from INFILE and reading will recommence at the start of the next complete card. The warning printed on such an occasion is:

```
CARD XXX HAS BEEN DELETED FROM THE INPUT FILE
```

8) REPLACE

Replacement of any part of a logical card may be necessary when wrong information has been entered or mistyped onto a card. Therefore, options are supplied to replace either author, the entire card, date, or a keyword. The

```
REPLACE CARD 260
```

command sequence would replace card 260 with the next one in INPUT in the same manner as the INSERT CARD, but card 260 would be entirely replaced, including the ID, with the new card read in. Care must be taken that the sequence order is maintained.

All the other options require as data the card number on which something is going to be replaced, and for KEYWORD the position of the replaced keyword in the keyword string. This latter case may mean that a PRINT CARD would have to have been used first to insure that the right keyword is replaced. The various forms of the command are:

```
REPLACE AUTHOR 278
REPLACE DATE  278
REPLACE KEYWORD 278 4
```

All three will ask for:

```
ENTER CHARACTER STRING
```

at which point the relevant new author/date/keyword is typed starting in column one. The output for successful completion of the command should read:

```
            REPLACED ITEM IS XXXXXXXXXXXXXXXXX

            REPLACING ITEM IS XXXXXXXXXXXXXXXXX
```

9) SUBSTITUTE

There are many occasions where a keyword is misspelt on more than one occasion in the INPUT file. If the word SURFACES used earlier as an example had been typed as SURFACE in perhaps 12 places throughout the file it would have been easier to correct the error by means of the SUBSTITUTE command. The command is:

```
            SUBSTITUTE
```

There are no card numbers indicated because the program will check every card for the presence of the irregular keyword. Thus two character strings are needed, the incorrect one and the corrected version. The interactive sequence would look like this:

```
            ENTER CHARACTER STRING

            SURFACE

            ENTER CHARACTER STRING

            SURFACES

            SURFACE REPLACED THOURGHOUT BY SURFACES

              12 CHANGES MADE

            DATA SAVED INTO INFILE

            POINTERS AT BEGINNING OF FILES
```

The corrected file is saved automatically into INFILE.

10) STOP

This command stops execution of the program and prints no message other than those supplied by whatever system is being used to run the program.

Program Details

The listing of the program has comment cards throughout to indicate the operation of different sections. As an overall guide, this section will outline the function of some of the arrays and subroutines.

Arrays

KOM Holds the command names. A data statement feeds the names in A1 into KARD from which they are copied into KOM.

KARD Used mostly to hold card images being read or edited.

KPEY General storage, used to develop keyword list, and holds character strings for editing purposes.

KNUM Used to generate the frequency distribution of the keywords.

NUMS } Hold data read into the program as commands: associated with
KYMB } the free format function.

Subroutines

MAINLINE Summons commands, calls other routines, and does the necessary switching to enter the right command.

ORDER Controls routine SORT which puts the keyword listings in alphabetical order. As the keywords are stored as 2A4, ordering proceeds on the first half of the keyword, and then sorts any remaining keywords that have the same first half and different second halves. Thus if two keywords are OPTICS and OPTIMIZATION, the first sort will place them together on the basis of their common OPTI, and the second sort on the last half of the keyword will place OPTICS before OPTIMIZA. At the same time, as keywords are ordered, the array KNUM has to be put into the same order to maintain a one to one correspondence between a keyword and its frequency.

READIT Reads in a card from INPUT in whatever format is supplied on lines 4920 and 4930. If KA is 1, the routine assumes it is looking for the beginning of a logical card; if 2, the end.

COM This routine is called from the main program to read in cards and data. If KC is 1, an alphanumeric command is to be written; if 2, its associated data. Both are read by means of the function IREAD. If a command name is read the value of IREAD should be negative and if data, positive. If the wrong sign is associated with the particular type of read being undertaken, the program prints:

WRONG TYPE OF DATA

and terminates execution.

RITE This subroutine writes a logical card to file OUT which is a file name, and may contain either the code for INFILE or OUTFILE depending on the call from the mainline.

FIND FIND is used to look through IN either for a given card ID (NUMB = desired ID number), or every card in the file (NUMB = 0), or a read from IN and a write to OUT (NUMB equals -1). NUMB is set to -1 for copying large sections of files in the COPY operation. IN and OUT contain the file codes for INFILE and OUTFILE.

COPY COPY copies the contents of any array KARD from elements KA to KB into an array KPEY from element KC+1.

START Rewinds INFILE and OUTFILE and writes the number of records to be written on file OUT as the first record on OUT. IN and OUT contain the file codes for INFILE and OUTFILE.

SORT This routine is a modification of the CACM routine published in CACM March 1969 by Singleton. It will sort the JAth column of array KA into order from low to high. Note that with alpha type information being sorted there may be a problem in that the sorted order will not always be alphabetical, owing to the bit pattern of some letters. This does not matter with regard to the program as long as the same order is maintained throughout. If alphabetic order is desired, a few small modifications could be made to input characters. JB indicates the other column of KA that is not being sorted, but together with the array NUM is simply following the ordering process based on the JAth column of KA.

NEWCRD Initiates the entry of a new card; part of the Free Format Function.

IREAD Reads in the commands and the associated data. The function can differentiate between numeric input and alpha input by assuming that anything non-numeric must be alphabetic. For details of operation see McCullagh (1972).

Error Messages Generated by the Program Editor

These messages are printed in the form:

```
PROGRAM STOPPED.  ERROR  X
```

where X is a value taken by the variable KERR. The various values of KERR have the following meanings:

Value of KERR	Interpretation
1	ID on continuation card and heading card not the same. Card deleted from input file.
2	Illegal command discovered on input sequence. New input sequence read without regard of any immediately following commands.
3*	Wrong type of data encountered.
4*	More than 100 individual keywords exist. Remedy is to read fewer cards.
5	No card of a particular ID found in the file.
6*	List of keywords is trying to grow larger than 100.
7	File fully read already, needs to be saved before more editing can take place.

*indicates execution stops if this state encountered.

Section C - UPDATE

Use of the UPDATE Program

The purpose of the UPDATE program is to take the edited file contained in INFILE, and the keyword list in KEYLIST, and to merge these with any previous bibliography held in PERMFILE. The program will then generate a cross reference index for the cards in INFILE and finally merge the old and the new cross reference indices to give an updated form in KEYWORD. As this process is largely automatic, there is very little in the way of data input to the program. What there is consists of two questions:

1) TYPE 1 IF THIS IS THE FIRST UPDATE

to which the reply is 0 if it is not. If successful, the program will print the message:

```
TOTAL NUMBER OF RECORDS IS NOW XXXX
UPDATE COMPLETE
THERE ARE NOW XXX KEYWORDS
```

followed by the second question:

2) TYPE 0 TO SUPPRESS TOTAL KEYWORD LISTING

Any other value than zero will cause the entire keyword list and frequency to be printed from KEYWORD. The final message before the programs ends is:

```
TRANSFER TO KEYWORD COMPLETED
```

The UPDATE program uses the same subroutines as EDITOR, needing RITE, FIND, and the Free Format Function. In UPDATE the FFF is in the standard form given in McCullagh (1972). Subroutine SORT is borrowed from program SEARCH to sort the ID lists for each keyword into order.

Section D - SEARCH

Use of Program SEARCH

The SEARCH program is designed to retrieve references on the basis of a question formulated by the user into a set of keywords. It operates on KEYWORD in the main, but may use PERMFILE if anything other than an identification number. is required as an answer. The program operates by searching through KEYWORD and forming a table of hits where a given card has a keyword on it corresponding to a keyword given in the input profile or question. When a certain number of hits has accrued to a particular ID, the reference referred to by that ID is considered to be retrieved; in other words, the question has been successfully answered. Obviously, more than one reference is a possible success; provision has to be made for testing all those likely to lead to success and discarding failures at the earliest possible moment. Because the number of IDs that have to be checked is a far larger group than those successful, about 2500 words of core are needed despite the fact that the program is interactive in the early stages of its execution. This can cause slow response, but is essential to a reasonably efficient running of the program when a large file is used.

Formation of a Profile or Question

The formation of a question that has meaning is one of the most difficult and most subjective parts of any retrieval system. In a human operated system implied parts of a question can be understood automatically and missing information easily found. But in a computer system a lot of care has to be given to the problem of making a question neither too general nor too specific. If too general, the proportion of correctly retrieved references will be very small compared with the total that the computer thinks relevant. Thus, if a bibliography on computer manufacturing had been developed and the keyword computer were used, it is quite likely that nearly the whole bibliography would be retrieved! The opposite is also true.

In a human system it is quite usual to ask questions like: "give me every paper to do with urban areas in England that does not include neighborhood communities, or villages, but maybe including any references to a CBD or sewage treatment in urban areas." This question can be broken into various parts and be rephrased into an AND, OR, and NOT formulation. AND, OR, and NOT are logical operators in a computer that can be used to answer questions and to provide a structure for them. The request just given could be shortened and formalized in the following manner:

		AND		NOT	
	URBAN		ENGLAND		NEIGHBORHOOD
OR					
	CBD		SEWAGE		VILLAGE

This can be translated to:(((URBAN.OR.CBD).AND.(ENGLAND.OR.SEWAGE)).NOT. (NEIGHBORHOOD.OR.VILLAGE)). This is not exactly what the questioner first meant; this question will give no information on sewage related to neighborhoods or villages, whereas there may have been an implied relationship that anything on sewage was worth keeping. This demonstrates the type of confusion that can easily occur. Other types of logic could be introduced at various levels of subtlety, but SEARCH has the capability of dealing only with AND, OR, or NOT. A very helpful way of developing a profile is to choose keywords to make up the question structure in relation to the frequency of occurrence listings given by EDITOR and UPDATE, which will show whether a given keyword will lead to too general or too specific an answer.

Data Input for SEARCH

It is advisable to have the question already partitioned into this type of framework so that the minimum time is spent on input. The data entry process will be demonstrated by using the question just considered. The program asks:

```
NUMBER OF ANDS
```

In this case, there are two. There is a maximum of five, usually more than sufficient. The second question is:

```
ENTER THE NUMBER OF ORS FOR EACH AND
```

This refers to the numbers in each column in the example. In each case the answer is two. Last, the program will ask:

```
ENTER NUMBER OF NOTS
```

which will once again be two. As with the editing program it is possible to type these numbers individually in response to the question or in one string after the first question. Then although the program will print out the remaining captions, it will not wait for the input of any data as it already has some. Thus, to answer all the interrogations at once:

```
2  2  2  2
```

could be typed, indicating two ANDS, two ORS for each AND, and two NOTS. The program will then start the input sequence for the keyword character strings. These are read in 2A4 and every keyword must start on a new line. Any characters after the eighth column will be ignored. The ORs for the first **AND** are listed, followed by the ones for AND two, and so on if necessary. Note that the OR list must be in alphabetical order, as must any NOT list entered. Therefore, in our example, the response to:

```
ENTER OR LIST FOR AND NUMBER 1

ENTER "OR" 1

CBD

ENTER "OR" 2
```

```
URBAN

ENTER OR LIST FOR AND NUMBER 2

ENTER "OR" 1

ENGLAND

ENTER "OR" 2

SEWAGE

ENTER 2 NOTS, ONE KEYWORD PER LINE

NEIGHBORHOOD

VILLAGE
```

would be as indicated in the sequence just given. The underlined keywords are the responses typed in answer to the questions. It is quite possible that a mistake may have been made in typing so the structure of your question is printed for checking:

```
YOUR STRUCTURE IS AS FOLLOWS

AND  1  IS CBD      URBAN

AND  2  IS ENGLAND  SEWAGE

NOTS ARE   NEIGHBOU VILLAGE
```

followed by:

```
ENTER 0 IF THERE IS NO MISTAKE IN THE LISTING
```

This entry of 1 or 0 enables the program to return to the start of the question answer sequence if a mistake has been found. Once the program has been informed that the input is correct the search will begin. At the end the output will be:

```
NUMBER OF ITEMS IS    5

THESE ARE AS FOLLOWS

  102  259  267  296  333

ENTER 0 IF NO CARD PRINT OUT REQUIRED
```

Usually it will not be necessary to print out the full reference card kept in PERMFILE as the ID alone is adequate to find an article.

Some modifications of this procedure can occur with big files, those of 1000 references or over. The array sizes are kept deliberately small to economize on the cost of running the program. This means that during a search it may not be possible to check all possible hits at once. The array KSTORE can hold up to 200 IDs for checking as successful hits at one time, but if any more need to be entered, the only alternative is to store them temporarily somewhere else and to move them to KSTORE when some of the possible hits have been deleted as failures. A message is printed when this happens as it necessitates researching the file KEYWORD for the temporarily stored remaining likely hits. The message is:

NO SPACE LEFT IN PRIMARY STORAGE FILE

This will not lead to a catastrophic failure as the temporary space used is NUMS which can hold up to 500 extra IDs. Every time that the program finishes a cycle through KEYWORD without exhausting all possibilities, the message will be printed. If it is printed more than once, it indicates that an increase in the size of KSTORE may be called for, regardless of possible storage economies. By enlarging KSTORE, the number of cycles through KEYWORD can be greatly reduced and the size of NUMS decreased. The present array limits are set to cope with a file of approximately 500 records. If only a small bibliography is being searched it would be very worthwhile to reduce the size of these arrays. The array NUMS is the one that holds the card image containing the free format input data relating to the numbers of ANDs, ORs, and NOTs. Once the question answer sequence has been concluded successfully it has no further function and hence can be used as backup store. It should not be shortened to less than 70 words. For reference files of less than 500 items, KSTORE could probably be reduced to dimensions of 100 by 6, and NUMS to 200. If the worst should happen and both KSTORE and NUMS are filled but more space is needed, the message:

NUMBER STORAGE EXCEEDED

will be printed and execution will stop.

It is also possible that the array KINAL that holds the successful hits will become full if a very broad question is asked. No message will be printed out in this instance but on most systems there will be a catastrophic failure caused by trying to access an array outside its limits.

Array and Subroutine Use in SEARCH

Array Use

KSUM	This array holds the count total for the number of hits scored for a given reference. There must be at least one hit for every AND for success. Thus the second hit on a particular AND will have no effect on the count.
KAND	Holds all the keywords in the AND/OR structure with a total of 50.
KPAND	Marks the beginning of every AND in KPAND.
NOT	Holds the NOT keywords to a limit of 10.
KINAL	Stores the definitely successful hit references. Can hold up to 200.
KSTORE	As a given ID becomes a possible hit, it is moved into KSTORE (column 6), and the other columns set to zero to

mark future hits for a given AND.

KARD This array is used to hold the cross reference index referring to a given keyword in the file KEYWORD.

Subroutines

KNOT Removes items from KSTORE and NUMS that are no longer candidates for a successful hit because they possess one of the stipulated NOTs as one of their keywords.

ZERO This subroutine removes unsuccessful hits from the arrays NUMS, KSTORE, and KINAL.

All other routines have been described previously. Note that the free format routines have been used here as subroutines rather than functions.

Section E - EXAMPLE

A Practical Example of Using the KARP Interactive System

The best way of demonstrating the use of KARP is to give an example from a real situation. The one given here is the basis for the bibliography on computer mapping given in another section of this issue. A bibliography has been built up over a few years based on incorporating all the papers, articles, and books related to the technical aspects of computer mapping and specifically to the problems associated with gridding, contouring, and block diagram construction. Although there have been many articles written where the use of computer mapping techniques has been important, there are relatively few dealing with the techniques themselves. At a time when more emphasis is being put on the use of graphis systems in many disciplines, it seems appropriate to include an example based on a computer literature search of computer graphics literature!

The purpose of this example is to show how the original bibliography was converted to a computer file that could be searched, how this could be updated, and how new additions in the form of extra titles could be added. The first problem encountered was that of the choice of the keyword set. The set is dynamic but has to have some basis. As a computer bibliography had already been created for references dealing with the use of statistics in geology, many of the keywords from that study were directly transferrable. Owing to the more specific problem of computer mapping, the keywords in the previous bibliography were very general and many new ones had to be created to fill the gaps. Some of the development of the keyword set will be seen as the example progresses.

The list below shows the bibliography as it was punched on cards at the beginning of the project. Mostly the ID numbers are spaced every five units, but here and there some infilling has occurred because of the growth of the bibliography during the card punching phase. Plenty of room is, however, left for expansion. This listing contains a number of mistakes most of which cannot easily be detected visually. The XXX and --- refer respectively to those mistakes detected while editing and to those discovered manually.

```
      00005ACI             197103CONTOURSTRIANGULMAPPING
      00010ACM             196904INTERPOLALGORITHIRREGULADATA
---   00015ALHBERG,JH      197103SPLINES INTERPOLENGINEER
      00020ANDERSON,WL     197103CUBIC   SPLINES REGULAR
      00025AUMEN,WC        196602NUMERICAMAPPING
xxx   00030BASINGER,D GVILD197106BLOCK DITHREE SIGRAPHICSSURFACESCOMPUTER
      00030                1971  PROGRAM
      00035BATCHA,JP REESE,196404GRIDDINGCONTOURSMINERALSEXPLORAT
      00040BENGTSSON,BE NOR196405CONTOURSGRIDDINGTRIANGULMAPPING COMPUTER
      00045BERTIN,J        196705CARTOGRAQUANTITACOMPUTERMAPPING FRENCH
---   00050BERTRAN,S       196502MAPPING ENGINEER
      00055BICKMORE,DP     196705MAP ANALCOMPUTERGENERAL CARTOGRAENGLAND
      00060BLUM,M          196903OPTIMISASMOOTHINMAPPING
      00065BOEHM,BW        196704MULTIVARMAPPING TERRAIN MODELS
```

```
        00070BOND,DH MCGLASHA196703COMPUTERGEOGRAPHMAPPING
xxx     00075BONHAM-CARTER,GF196605COMPUTERPROGRAMSBIBLIOGRCORRELATMULTIVAR
        00080BOUKNIGHT,WJ    196904HALF TONGRAPHICSCOMPUTERALGORITH
        00085BROWN,AE        196305MATRIX ATERRAIN PROFILESSIMULATIMAPPING
        00090BUREAU OF THE CE196904MAPPING COMPUTERMAP ANALCARTOGRA
        00095CAIN,JC NEILON,J196303COMPUTERMAPPING MAGNETIC
        00100CALCOMP         196805GENERAL CONTOURSPROGRAM MAPPING COMPUTER
        00105CALCOMP         196902PLOTTINGPROGRAM
        00110CENTRAL INTELLIG197004CARTOGRACOMPUTERMAPPING PROGRAM
        00115CERNY,JW        197103PROGRAM MAPPING TRANSFOR
        00120CHANG,K ADAMS,JM197106CARTOGRAPRINTER GRIDDINGPLANNINGCOMPUTER
        00120                1971  GEOGRAPH
        00125CHOHA,DD STIRLIN19  04GRAPHICSCOMPUTERMAPPING OCEANOGR
        00130CHRISTENSEN,P   196903QUANTITAMAPPING BIBLIOGR
xxx     00135CHUN,D SCHLATER,196905SURFACE FIT     CONTOURSPROGRAM COMPUTER
        00140CITIES SERVICE O197002CONTOURSPROGRAM
        00145CLENSHAW,CW HAYE196507CURVE FISURFACESMATHEMATALGORITHPOLYNOMI
        00145                1965  FOURIER SMOOTHIN
--      00150COLE,AJ JORDAN,C196708PROGRAM ITERATIVQUADRATILINEAR  FIT
        00150                1967  CONTOURSQUADRATIMAPPING
        00155COLE,AJ         196806COMPUTERMAP ANALCARTOGRAIRREGULAALGORITH
        00155                1968  CONTOURS
        00160COLE,AJ         196906PROGRAM ITERATIVFIT     TSA     CONTOURS
        00160                1969  MAPPING
        00165COLNER,BJ       196702MAPPING SIMULATI
        00170CONNELLY,DS     197104CONTOURSMAPPING PLOTTINGGENERAL
        00175COPPOCK,JT      197203GRAPHICSPRINTER GEOGRAPH
        00180CORNWELL,B ROBIN196604FILMS   GRAPHICSCARTOGRAANIMATION
        00185COTTAFAVA,G LE M196903ALGORITHCONTOURSMAPPING
xxx     00190COULTHARD,WJ LEI196904MAP ANALEXPLORATTHREE-D SPATIAL
        00195CRAIN,IK        197006COMPUTERINTERPOLBIBLIOGRCONTOURSMAP ANAL
        00195                1970  EXPLORAT
        00200CRAIN,IK BHATTAC196703IRREGULADATA    COMPUTER
        00205CUDE,WC         196204COMPUTERCARTOGRAMAP ANALMAPPING
        00210DAVIES,DK BEVAN,196705PROGRAM COMPUTERGRAPHICSGEOLOGY SEDIMENT
        00215DAYHOFF,MO      196306CONTOURSCOMPUTERPROGRAM MAPPING X-RAY
        00215                1963  CRYSTALL
        00220DEFENSE DOCUMENT196804BIBLIOGROPTICS  GRAPHICSINFORMAT
        00225DEGANI,A        197005CONTOURSMAPPING COMPUTERPLOTTINGSCANNING
        00230DEMETER,ER      196304CARTOGRACOMPUTERGENERAL MAPPING
        00235DEMETER,ER      196403CONTOURSCOMPUTERMAPPING
        00240DIELLO,J KIRK,K 196903CARTOGRACOMPUTERGENERAL
xxx     00245DODD,JR CAIN,JA 196505PATTERNSCONTOURSMAPPING RANDOM  DATA
        00250DODGE,HF        196402MAPPING ENGINEER
        00255DUNN,DE         197005PROBABILERRORS  CONTOURSMAPPING HYDROLOG
        00260ELLIOTT,D       196504QUANTITAORIENTATMAPPING STRUCTUR
        00265EWEN-SMITH,BM   197105ALGORITHCARTOGRACONTOURSMAPPING LINEAR
        00270EXPERIMENTAL CAR197104MAPPING CARTOGRAPLANNINGGENERAL
        00275FLESHEL,B STEWAR197003PROGRAM MAPPING CONTOURS
        00280FORGOTSON,JM    196303COMPUTEREXPLORATOIL
        00285FREEMAN,H MORSE,196704CONTOURSMAPPING TERRAIN BLOCK DI
---     00290FIRTSCH,JM      196906GRIDDINGINTERPOLCUBIC   MAPPING SPLINES
        00290                1969  METHODS
```

```
     00295GAITS,GM        196903PRINTER MAPPING COMPUTER
     00300GALIMBERTI,R MON196903ALGORITHHIDDEN LPOLYHEDR
     00305GARFINKEL,D     196203PRINTER GRAPHICSPROGRAM
xxx  00307GEOCOM BULLETIN 196907BIBLIOGRCOMPUTERMAPPING GRAPHICSCURVE-FI
     00307                1969  INTERPOLCONTOURS
     00310GILBERT,PR      196804COMPUTERNUMERICAMAPPING MAP ANAL
     00315GRAHAM,NY       197205PERSPECTSURFACESHIDDEN LBLOCK DIALGORITH
     00320HARVARD UNIVERSI197104MAPPING PRINTER GRAPHICSCONTOURS
     00325HEAP,BR PINK,MG 196902CONTOURSALGORITH
     00330HOWARTH,RJ      197103PROGRAM MAPPING SPATIAL
     00335HSU,ML ROBINSON,197004ERRORS  CONTOURSMAPPING CARTOGRA
---  00340HUIJBREGTS,C MAT197006KRIGING METHODS CONTOURSTSA     MAP ANAL
---  00340                1970  SPATIAL
xxx  00345IBM             19  04CONTOURSTIRANGULALGORITHMAPPING
     00350IBM             196304GRIDDINGINTERPOLALGORITHMAPPING
xxx  00355IBM             196508BIBLIOGRNUMERICASURFACE METHODS CONTOURS
     00355                1965  PLOTTINGMAPPING GENERAL
xxx  00360IBM             196905NUMERICASURFACE METHODS CONTOURSPLOTTING
     00365ICA             197102COMPUTERCARTOGRA
     00370JENKS,GF STEINKE197106NOTATIONCONTOURSSPACING SPATIAL THREE DI
     00370                1971  CARTOGRA
     00375JESPERSON,H     197003PROGRAM BLOCK DITHREE DI
     00380JOHNSON,SS HUXSA197107COMPUTERPROGRAM GRIDDINGMAPPING CONTOURS
     00380                1971  RANDOM  DATA
     00385JOHNSON,TE      196304PROGRAM DIAGRAMSCOMPUTERTHREE DI
     00390JOHNSON,WL SANDE196604SURFACESDESIGN  GRAPHICSPOLYHEDR
     00395JONES,RL        197104MAPPING GENERAL CONTOURSPROGRAM
     00400KATSANIS,T      196805COMPUTERPROGRAM INTERPOLSPLINES CURVE FI
     00405KERN,R RUSHTON,G196905COMPUTERPROGRAM CARTOGRAMAPPING NOTATION
xxx  00410KOEMAN,C VAN DER197004GRAPHIC COMPUTERPLOTTINGCARTOGRA
     00415KOLBERG,DW      197004DISTRIBUPOPULATICOMPUTERMAPPING
     00420KUBERT,B SZABO,J196805BLOCK DIPERSPECTALGORITHPLOTTINGCARTOGRA
     00425LANG,T          197103COMPUTERPROGRAM MAPPING
     00430LAWRENCE,CH     196702PLOTTINGSTEREOGR
     00435LAWRENCE,CJ     196804BLOCK DIGRAPHICSTERRAIN CLASSIFI
     00440LIGHT,DL        196604MAPPING GRIDDINGINTERPOLCONTOURS
     00445LINTNER,MA      196906ALGORITHCOMPUTERPROGRAM HIDDEN LSURFACES
     00445                1969  SINGLE V
     00450LOUTREL,PP      197004GRAPHICSALGORITHHIDDEN LCOMPUTER
     00455MACKAY,JR       195104CONTOURSMETHODS MAPPING GEOGRAPH
     00460MACKAY,JR       195302CONTOURSINTERPOL
     00465MASSEY,JS       197004BIBLIOGRCARTOGRACOMPUTERMAPPING
     00470MATHERON,G      196905THEORY  MAPPING ALGORITHMATHEMATKRIGING
     00475MCADAMS,HT      196903COMPUTERMAPPING DATA
xxx  00480MCCUE,GA DWORETS196505STEREOGRBLOCK DICONTOURSPROGRAMPPLOTTING
     00485MCCUE,GA GREEN,J196505GRAPHICSCOMPUTERSPATIAL TERRAIN CONTOURS
     00490MCINTYRE,DB POLL196803COMPUTERPROGRAM CONTOURS
     00495MCLAURIN,JD     196903GRAPHICSCOMPUTERPERCEPTI
     00500MCLELLAN,CD     197103COMPUTERALGORITHGEOGRAPH
     00505MCMAINS,F       196704CONTOURSPROGRAM MAPPING MATRIX
     00510MCMILLAN,RE JOHN196904COMPUTERMAP ANALBINARY  SCANNING
     00515MERRIAM,DF SNEAT196604QUANTITACONTOURSGEOPHYSIMAPPING
xxx  00520MILLY,SA        196906PROGRAM COMPUTERSPATIAL CONTOURSSURFACE
     00520                1969  AREA
```

```
    00525MISKELL,RV     197104GRIDDINGINTERPOLNUMERICACUBIC
xxx 00530MISULIA,MG SPOON195705PROFILE SCANNINGMODEL   GRAPHICSSTEREOGR
    00535MIT            196803CONTOURSPLOTTINGPROGRAM
    00540MONMONIER,MS   196504PRINTER SHADING MAPPING COMPUTER
xxx 00545MONMONIER,MS PFA196604SURFACE AREA    CONTOURSMAPPING
    00550MONMONIER,MS   196803COMPUTERMAPPING PLOTTING
    00555MONMONIER,MS   197104CORRELATMAPPING SLOPES  QUANTITA
    00560MONMONIER,MS   197104PLOTTINGMAPPING COMPUTERPROGRAM
    00565MOORE,RT STARK,M196403SCANNINGOPTICS  MAPPING
    00570MORRISON,JL    197106ERRORS  NUMERICAMETHODS MAPPING CONTOURS
    00570               1971  CARTOGRA
xxx 00575MORSE,SP       196803CONTOURSMODEL   THEORY
    00580MORSE,SP       196802CONTOURSCOMPUTER
    00585MORSE,SP       196902CONTOURSMAPPING
    00590MURRAY,FW      196803CONTOURSALGORITHMETHODS
    00595MURT,GE        197103CARTOGRAOCEANOGRCOMPUTER
    00600NAKATA,I       196704ALGORITHCOMPUTERSYSTEMS TREES
    00605NELSON,WB HENDRI196904PROGRAM COMPUTERPROBABILPLOTTING
    00610NOBLE,DC EBERLY,196402ALGORITHGRAPHICS
    00615NOLL,AM        196704HIDDEN LPOLYHEDRROTATIONALGORITH
    00620NOMA,AA MISULIA,195904COMPUTERPROGRAM TERRAIN BLOCK DI
xxx 00625NORDBECK,S RYSTE196705CARTOGRAPROGRAMSBIBLIOGRCOMPUTERPOLYGON
    00630NORDBECK,S RYSTE196902CARTOGRAMAPPING
    00635NOTT,CW        196803GRAPHICSMETHODS COMPUTER
    00640OJAKANGAS,DR BAS196404ALGORITHCOMPUTERCONTOURSEXPLORAT
    00645PALMER,JAB     196903COMPUTERMAPPING CARTOGRA
    00650PALMER,JAB     197004COMPUTERPROGRAM CONTOURSPLOTTING
    00655PALMER,JAB     197003PLOTTINGCONTOURSTHEORY
    00660PALMER,JAB     197002PLOTTINGCOMPUTER
    00665PARK,CM LEE,YH S197003SLOPES  CONTOURSMAPPING
    00670PFALTZ,JL ROSENF196703MAPPING PATTERN ALGORITH
    00675PLOC,RA BARNETT,197103STEREOGRCOMPUTERMAPPING
    00680RALL,LL        196602MAPPING ENGINEER
    00685RENS,FJ        196503PROGRAM COMPUTERMAPPING
    00690RENS,F         196703PLOTTINGGRAPHICSBLOCK DI
    00695RHIND,DW BARRETT197104CARTOGRAGENERAL COMPUTERCONTOURS
    00700RIEBER,JE LAMB,V197001GRAPHICS
    00705ROBERTSON,JC   196704CARTOGRAPRINTER COMPUTERMAPPING
--- 00710ROHLF,RJ       196904GRAPHICSPROGRAM COMPUTEROUTPUT
    00715ROSENFELD,A    196503REMOTE SPATTERN MAPPING
    00720ROSENFELD,A STRO196902MATHEMATMAPPING
    00725SCHMID,CF MACCAN195506THEORY  METHODS CARTOGRASTATISTICONTOURS
    00725               1955  MAPPING
    00730SCHOLER,H      196702STEREOGRMAPPING
    00735SCIENTIFIC COMPU197102MAPPING CONTOURS
--- 00740SCUDDEN,JD     197104SPLINES GRIDDINGQUADRATIINTERPOL
--- 00745SHAPR,JV CHRISTE196502NUMERICAMAPPING
    00750SHEPARD,D      196807SPATIAL INTERPOLGRIDDINGCOMPUTERIRREGULA
    00750               1968  MAPPING SPACING
    00755SLACK,HA BRUNTON196306OIL     INDUSTRYCOMPUTERMAPPING CARTOGRA
    00755               1963  GENERAL
xxx 00760SMITH,FG       196803CONTOURSMAPPING PROGRAMS
xxx 00765SMITH,PJ       197109QUADRATICUBIC   GRIDDINGPROGRAM FITTING
    00765               1971  INTERPOLPOLYNOMISPLINES LEAST SQ
    00770SPEIGHT,JG       196905CARTOGRAGRIDDINGCOMPUTERMAPPING LANDFORM
    00775SPRUNT,B         197205CONTOURSBLOCK DIPERSPECTISOMETRIALGORITH
    00780STEARNS,F        196805METHODS CONTOURSERRORS  QUANTITAMAPPING
xxx 00785SUNDBERG,WD      197005BLOCK DICOMPUTERPROGRAMSHIDDEN LISOMETRI
    00790SUTHERLAND,IE    197003COMPUTERGENERAL GRAPHICS
    00795SWANN,DH ET AL   197003COMPUTERGENERAL MAPPING
    00800SWINDLE,G VAN AN196903COMPUTERCONTOURSOCEANOGR
    00805SWITZER,P MOHR,C196405STATISTITERRAIN CONTOURSPLOTTINGOCEANOGR
    00810SWITZER,P        196904MAPPING GEOGRAPHCORRELATENVIRONM
    00815TAYLOR,DRF       197205BIBLIOGRCOMPUTERMAPPING CARTOGRAGEOGRAPH
    00820THOMAS,AL        196903GRAPHICSBLOCK DIGENERAL
    00825THOMAS,AL        197204COMPUTERCARTOGRACONTOURSBLOCK DI
    00830TOBLER,WR        196503COMPUTERCARTOGRAMAPPING
    00835TROEH,FR         19  04LANDFORMFIT     CONTOURSMAPPING
xxx 00840TURNER,AK        196805PROGRAMSCOMPUTERCONTOURSMAPPING THREE DI
    00845U.S. GEOLOGICAL 196903CARTOGRAMAPPING SCANNING
    00850WALTERS,RF       196904CONTOURSCOMPUTERALGORITHGENERAL
--- 00855WAHSINGTON,WM OL196803GRAPHICSCONTOURSMETEOROL
    00860WILLIAMSON,H     197203HIDDEN LPLOTTINGPROGRAM
    00865WOLFENDALE,PCF   196703COMPUTERERRORS  CARTOGRA
    00870WOODFORD,CH      196904INTERPOLALGORITHCURVE FISMOOTHIN
    00875WOON,P           197005BLOCK DIPLOTTINGCOMPUTERHIDDEN LPOLYHEDR
    00880WRAY,WB          197004BLOCK DIISOMETRIPROGRAM COMPUTER
```

The cards are punched ready for reading into the file INPUT for processing by EDITOR. The simplest eay to check for errors in the keyword list is to use the EDITOR program. The next output listing shows the sequence of operations. First, 100 of the cards are read, a totally arbitrary number that is somewhere near the amount that usually produces just under 100 keywords. A keyword list is then generated and printed.

```
ENTER COMMAND
= READCARDS 100 NEWKEY PRINT KEYLIST
ENTER DATA
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
KEYWORD LIST GENERATED
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
KEYWORD AND FREQUENCY LIST
 ALGORITH   ANIMATIO   BIBLIOGR   BLOCK DI   CARTOGRA   CLASSIFI   COMPUTER
       15          1          7          7         19          1         43
 CONTOURS   CORRELAT   CRYSTALL   CUBIC      CURVE FI   CURVE-FI   DATA
       37          1          1          2          2          1          5
```

```
DESIGN    DIAGRAMS  DISTRIBU  ENGINEER  ENGLAND   ERRORS    EXPLORAT
       1         1         1         3         1         2         4
FILMS     FIT       FOURIER   FRENCH    GENERAL   GEOGRAPH  GEOLOGY
       1         3         1         1         8         4         1
GRAPHIC   GRAPHICS  GRIDDING  HALF TON  HIDDEN L  HYDROLOG  INFORMAT
       1        15         7         1         4         1         1
INTERPOL  IRREGULA  ITERATIV  KRIGING   LINEAR    MAGNETIC  MAP ANAL
       9         3         2         2         2         1         8
MAPPING   MATHEMAT  MATRIX A  METHODS   MINERALS  MODELS    MULTIVAR
      54         2         1         5         1         1         2
NOTATION  NUMERICA  OCEANOGR  OIL       OPTICS    OPTIMISA  ORIENTAT
       2         4         1         1         1         1         1
PATTERNS  PERCEPTI  PERSPECT  PLANNING  PLOTTING  POLYHEDR  POLYNOMI
       1         1         2         2         9         2         1
POPULATI  PRINTER   PROBABIL  PROFILES  PROGRAM   PROGRAMP  PROGRAMS
       1         5         1         1        23         1         1
QUADRATI  QUANTITA  RANDOM    REGULAR   SCANNING  SEDIMENT  SIMULATI
       2         3         2         1         1         1         2
SINGLE V  SMOOTHIN  SPACING   SPATIAL   SPLINES   STEREOGR  STRUCTUR
       1         2         1         5         4         2         1
SURFACE   SURFACES  TERRAIN   THEORY    THREE DI  THREE SI  THREE-D
       3         5         5         1         3         1         1
TIRANGUL  TRANSFOR  TRIANGUL  TSA       X-RAY
       1         1         2         2         1
```

The keyword listing shows that a number of errors exist. Both CURVE FI(TTING) and CURVE-FI exist, and similarly with GRAPHIC/GRAPHICS, PATTERNS instead of PATTERN (RECOGNITION), PROGRAM/PROGRAMP/PROGRAMS, SURFACE/SURFACES, THREE DI/THREE SI/THREE-D, and with TIRANGUL/TRIANGUL. There remains the problem of which to choose as the 'correct' form: in some cases this is not an easy choice, such as SURFACE or SURFACES. The first occurs three times according to the listing and the latter five times, so the SURFACES have it!

In order not to read through INFILE (which contains these card images) too many times it is best to determine the location of the errors so that they can be dealt with sequentially. The command PRINT KEYWORD can be used for each of the errors to determine their position in the file.

```
ENTER COMMAND
= PRINT KEYWORD
ENTER COMMAND
ENTER CHARACTER STRING
= CURVE-FI
KEYWORD OCCURS ON FOLLOWING CARDS
   307
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
= PRIN KEYW PRIN KEY
ENTER COMMAND
ENTER CHARACTER STRING
= GRAPHIC
KEYWORD OCCURS ON FOLLOWING CARDS
   410
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ILLEGAL COMMAND
PRIN KEYW PRIN KEY

ENTER COMMAND
= KEYWORD PRIN KEYW PRIN KEYW PRIN KEYW PRIN KEYW
ENTER CHARACTER STRING
= PROGRAMP
KEYWORD OCCURS ON FOLLOWING CARDS
   480
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER CHARACTER STRING
= PROGRAMS
KEYWORD OCCURS ON FOLLOWING CARDS
    75
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER CHARACTER STRING
= SURFACE
KEYWORD OCCURS ON FOLLOWING CARDS
   135   355   360
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER CHARACTER STRING
= THREE SI
KEYWORD OCCURS ON FOLLOWING CARDS
    30
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER CHARACTER STRING
= THREE-D
```

```
KEYWORD OCCURS ON FOLLOWING CARDS
   190
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
= PRIN KEYW PRIN KEYW
ENTER COMMAND
ENTER CHARACTER STRING
= TIRANGUL
KEYWORD OCCURS ON FOLLOWING CARDS
   345
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER CHARACTER STRING
= BIBLIOGRAPHY
KEYWORD OCCURS ON FOLLOWING CARDS
    75    130    195    220    307    355    465
POINTERS AT BEGINNING OF FILES
```

For program checking purposes, a PRINT KEYWORD command has been used at the end of the sequence to find all occurrences of the word "bibliography". These are found to be on references 75, 130, 195, 220, 307, 355, and 465. Once the location of the errors is known it is then possible to produce a sequential list of the order in which the errors should be corrected:

ERROR	CORRECT FORM	OCCURRENCE
CURVE-FI	CURVE FI	307
GRAPHIC	GRAPHICS	410
PROGRAMP	PROGRAM	480
PROGRAMS	PROGRAM	75
SURFACE	SURFACES	135,355,360
THREE SI	THREE DI	30
THREE-D	THREE DI	190
TIRANGUL	TRIANGUL	345

The sequence for correction is then 30, 75, 135, 190, 307, 345, 355, 360, 410, and 480. To make the alterations each card has to be printed to determine the position of the keyword that has to be changed:

```
ENTER COMMAND
= PRINT CARD 30 REPLACE KEYWORD 30
ENTER COMMAND
ENTER DATA
ID IS   30   AUTHOR IS  BASINGER,D GVILD  AND DATE   1971
THERE ARE   6  KEYWORDS, WHICH ARE
 BLOCK DI  THREE SI  GRAPHICS  SURFACES  COMPUTER
 PROGRAM
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= THREE DI
ENTER DATA
= 2
KEYWORD  THREE SI  REPLACED BY  THREE DI
ENTER COMMAND
= PRINT CARD 75 REPL KEYW 75
ENTER COMMAND
ENTER DATA
ID IS   75   AUTHOR IS  BONHAM-CARTER,GF  AND DATE   1966
THERE ARE   5  KEYWORDS, WHICH ARE
 COMPUTER  PROGRAMS  BIBLIOGR  CORRELAT  MULTIVAR

ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= PROGRAMS
ENTER DATA
= 2 PRINT CARD 135 REPL KEYWORD 135
KEYWORD  PROGRAMS  REPLACED BY  PROGRAMS
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ID IS  135   AUTHOR IS  CHUN,D SCHLATER,  AND DATE   1969
THERE ARE   5  KEYWORDS, WHICH ARE
 SURFACE   FIT       CONTOURS  PROGRAM   COMPUTER

ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= SURFACES
ENTER DATA
= 1 PRINT CARD 190 REPLLCE CARD
KEYWORD  SURFACE   REPLACED BY  SURFACES
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ID IS  190   AUTHOR IS  COULTHARD,WJ LEI  AND DATE   1969
```

```
THERE ARE   4  KEYWORDS, WHICH ARE
 MAP ANAL  EXPLORAT  THREE-D   SPATIAL
ENTER COMMAND
ENTER COMMAND
ENTER DATA
= 590
NO CARD OF ID  590  FOUND IN FILE
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
```

Notice that at the end of the list an error was made in calling REPLACE CARD rather than REPLACE KEYWORD. The solution was to give the program a card ID that did not exist thus causing it to search through the rest of the file unsuccessfully, and then save all the previously made corrections as an automatic response. The correct command REPLACE KEYWORD 190 could then be made on the second pass through the file. The listing below shows the rest of the editing process.

```
ENTER COMMAND
= PRINT CARD 135
ENTER COMMAND
ENTER DATA
ID IS   135   AUTHOR IS  CHUN,D SCHLATER,   AND DATE  1969
THERE ARE   5  KEYWORDS, WHICH ARE
 SURFACES  FIT       CONTOURS  PROGRAM   COMPUTER

ENTER COMMAND
= REPL KEYW 190 3 PRINT CARD 307 REPL KEYW 307
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= THREE DI
ENTER DATA
KEYWORD  THREE-D   REPLACED BY  THREE DI
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ID IS   307   AUTHOR IS  GEOCOM BULLETIN    AND DATE  1969
THERE ARE   7  KEYWORDS, WHICH ARE
 BIBLIOGR  COMPUTER  MAPPING   GRAPHICS  CURVE-FI
 INTERPOL  CONTOURS
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= CURVE FI
```

```
ENTER DATA
= 5
KEYWORD  CURVE-FI  REPLACED BY  CURVE FI
ENTER COMMAND
= PRINT CARD 345 REPLACE KEYWORD 345
ENTER COMMAND
ENTER DATA
ID IS  345   AUTHOR IS  IBM                  AND DATE  19
THERE ARE   4  KEYWORDS, WHICH ARE
 CONTOURS  TIRANGUL  ALGORITH  MAPPING
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= TRIANGULAR
ENTER DATA
= 2
KEYWORD  TIRANGUL  REPLACED BY  TRIANGUL
ENTER COMMAND
= PRINT CARD 355 REPL KEYW 355
ENTER COMMAND
ENTER DATA
ID IS  355   AUTHOR IS  IBM                  AND DATE  1965
THERE ARE   8  KEYWORDS, WHICH ARE
 BIBLIOGR  NUMERICA  SURFACE   METHODS   CONTOURS
 PLOTTING  MAPPING   GENERAL
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= SURFACES
ENTER DATA
= 3
KEYWORD  SURFACE   REPLACED BY  SURFACES
ENTER COMMAND
= PRINT CARD 360 REPL KEYW 360
ENTER COMMAND
ENTER DATA
ID IS  360   AUTHOR IS  IBM                  AND DATE  1969
THERE ARE   5  KEYWORDS, WHICH ARE
 NUMERICA  SURFACE   METHODS   CONTOURS  PLOTTING

ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= SURFACES
ENTER DATA
= 2
KEYWORD  SURFACE   REPLACED BY  SURFACES
ENTER COMMAND
= PRINT CARD 410 REPL KEYW 410
ENTER COMMAND
ENTER DATA
ID IS  410   AUTHOR IS  KOEMAN,C VAN DER  AND DATE  1970
THERE ARE   4  KEYWORDS, WHICH ARE
 GRAPHIC   COMPUTER  PLOTTING  CARTOGRA
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= GRAPHICS
ENTER DATA
= 1
KEYWORD  GRAPHIC   REPLACED BY  GRAPHICS
ENTER COMMAND
= PRINT CARD 480 REPL KEYW 480
ENTER COMMAND
ENTER DATA
ID IS  480   AUTHOR IS  MCCUE,GA DWORETS  AND DATE  1965
THERE ARE   5  KEYWORDS, WHICH ARE
 STEREOGR  BLOCK DI  CONTOURS  PROGRAMP  PLOTTING

ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= PROGRAM
ENTER DATA
= 4
KEYWORD  PROGRAMP  REPLACED BY  PROGRAM
ENTER COMMAND
= COPY NEWKEY PRINT KEYLIST
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
KEYWORD LIST GENERATED
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
KEYWORD AND FREQUENCY LIST
 ALGORITH  ANIMATIO  BIBLIOGR  BLOCK DI  CARTOGRA  CLASSIFI  COMPUTER
       15         1         7         7        19         1        43
 CONTOURS  CORRELAT  CRYSTALL  CUBIC     CURVE FI  DATA      DESIGN
       37         1         1         2         3         5         1
 DIAGRAMS  DISTRIBU  ENGINEER  ENGLAND   ERRORS    EXPLORAT  FILMS
        1         1         3         1         2         4         1
 FIT       FOURIER   FRENCH    GENERAL   GEOGRAPH  GEOLOGY   GRAPHICS
        3         1         1         8         4         1        16
 GRIDDING  HALF TON  HIDDEN L  HYDROLOG  INFORMAT  INTERPOL  IRREGULA
        7         1         4         1         1         9         3
 ITERATIV  KRIGING   LINEAR    MAGNETIC  MAP ANAL  MAPPING   MATHEMAT
        2         2         2         1         8        54         2
 MATRIX A  METHODS   MINERALS  MODELS    MULTIVAR  NOTATION  NUMERICA
        1         5         1         1         2         2         4
 OCEANOGR  OIL       OPTICS    OPTIMISA  ORIENTAT  PATTERNS  PERCEPTI
        1         1         1         1         1         1         1
 PERSPECT   PLANNING   PLOTTING   POLYHEDR   POLYNOMI   POPULATI   PRINTER
        2          2          9          2          1          1          5
 PROBABIL   PROFILES   PROGRAM    PROGRAMS   QUADRATI   QUANTITA   RANDOM
        1          1         24          1          2          3          2
 REGULAR    SCANNING   SEDIMENT   SIMULATI   SINGLE V   SMOOTHIN   SPACING
        1          1          1          2          1          2          1
 SPATIAL    SPLINES    STEREOGR   STRUCTUR   SURFACES   TERRAIN    THEORY
        5          4          2          1          8          5          1
 THREE DI   TRANSFOR   TRIANGUL   TSA        X-RAY
        5          1          3          2          1
ENTER COMMAND
```

The keyword listing generated at the end of the sequences shows that some errors were not corrected and some further editing is necessary. Also the errors that were detected visually need to be changed; these errors are mostly concerned with the author and date fields of the reference cards:

```
ENTER COMMAND
= PRINT CARD 490 PRINT CARD 500
ENTER COMMAND
ENTER DATA
ID IS  490   AUTHOR IS  MCINTYRE,DB POLL  AND DATE  1968
THERE ARE   3  KEYWORDS, WHICH ARE
 COMPUTER  PROGRAM   CONTOURS
ENTER COMMAND
ENTER COMMAND
ENTER DATA
NO CARD OF ID  500  FOUND IN FILE
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
= SUBSTITUTE
ENTER CHARACTER STRING
= PROGRAMS
ENTER CHARACTER STRING
= PROGRAM
PROGRAMS  REPLACED THROUGHOUT BY  PROGRAM
    1  CHANGES MADE
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
= REPLACE AUTHOR 15 REPLACE AUTHOR 50 REPL AUTH 290
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= AHLBERG,JH
REPLACED ITEM IS  ALHBERG,JH
REPLACING ITEM IS  AHLBERG,JH
```

```
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= BERTRAM,S
REPLACED ITEM IS  BERTRAN,S
REPLACING ITEM IS  BERTRAM,S
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= FRITSCH,JM
REPLACED ITEM IS  FIRTSCH,JM
REPLACING ITEM IS  FRITSCH,JM
ENTER COMMAND
= REPLACE DATE 240@@340
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1972
REPLACED ITEM IS  1970
REPLACING ITEM IS  1972
ENTER COMMAND
= PRINT CARD 340
ENTER COMMAND
ENTER DATA
ID IS  340   AUTHOR IS  HUIJBREGTS,C MAT  AND DATE  1972
THERE ARE   6  KEYWORDS, WHICH ARE
 KRIGING   METHODS   CONTOURS  TSA       MAP ANAL
 SPATIAL
 ENTER COMMAND
 = COPY PRINT KEYWORD
 DATA SAVED INTO INFILE
 POINTERS AT BEGINNING OF FILES
 ENTER COMMAND
 ENTER COMMAND
 ENTER CHARACTER STRING
 = PATTERNS
 KEYWORD OCCURS ON FOLLOWING CARDS
    245
 POINTERS AT BEGINNING OF FILES
 ENTER COMMAND
 = PRINT CARD 245
 ENTER COMMAND
 ENTER DATA
 ID IS  245   AUTHOR IS  DODD,JR CAIN,JA   AND DATE  1965
 THERE ARE   5  KEYWORDS, WHICH ARE
  PATTERNS  CONTOURS  MAPPING   RANDOM    DATA

 ENTER COMMAND
 = REPLACE KEYWORD 245 1 COPY NEWKEY PRINT KEYLIST
 ENTER COMMAND
 ENTER DATA
 ENTER CHARACTER STRING
 = PATTERN
```

```
ENTER DATA
 EYWORD  PATTERNS  REPLACED BY  PATTERN
ENTER COMMAND
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
KEYWORD LIST GENERATED
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
KEYWORD AND FREQUENCY LIST
 ALGORITH  ANIMATIO  BIBLIOGR  BLOCK DI  CARTOGRA  CLASSIFI  COMPUTER
       15         1         7         7        19         1        43
 CONTOURS  CORRELAT  CRYSTALL  CUBIC     CURVE FI  DATA      DESIGN
       37         1         1         2         3         5         1
 DIAGRAMS  DISTRIBU  ENGINEER  ENGLAND   ERRORS    EXPLORAT  FILMS
        1         1         3         1         2         4         1
 FIT       FOURIER   FRENCH    GENERAL   GEOGRAPH  GEOLOGY   GRAPHICS
        3         1         1         8         4         1        16
 GRIDDING  HALF TON  HIDDEN L  HYDROLOG  INFORMAT  INTERPOL  IRREGULA
        7         1         4         1         1         9         3
 ITERATIV  KRIGING   LINEAR    MAGNETIC  MAP ANAL  MAPPING   MATHEMAT
        2         2         2         1         8        54         2
 MATRIX A  METHODS   MINERALS  MODELS    MULTIVAR  NOTATION  NUMERICA
        1         5         1         1         2         2         4
 OCEANOGR  OIL       OPTICS    OPTIMISA  ORIENTAT  PATTERN   PERCEPTI
        1         1         1         1         1         1         1
 PERSPECT  PLANNING  PLOTTING  POLYHEDR  POLYNOMI  POPULATI  PRINTER
        2         2         9         2         1         1         5
 PROBABIL  PROFILES  PROGRAM   QUADRATI  QUANTITA  RANDOM    REGULAR
        1         1        25         2         3         2         1
 SCANNING  SEDIMENT  SIMULATI  SINGLE V  SMOOTHIN  SPACING   SPATIAL
        1         1         2         1         2         1         5
 SPLINES   STEREOGR  STRUCTUR  SURFACES  TERRAIN   THEORY    THREE DI
        4         2         1         8         5         1         5
 TRANSFOR  TRIANGUL  TSA       X-RAY
        1         3         2         1
ENTER COMMAND
= STOP

PROGRAM STOP AT 4390
```

The final keyword listing is correct and has all the keywords in their standard forms. The command STOP stops execution of the program and leaves INFILE containing the edited cards, and KEYLIST with the corrected keyword list.

Program UPDATE can now be used to generate the cross reference index, KEYWORD, and to write the edited reference list to PERMFILE. As it is the first update the response to the first question asked is 1, and 0 to the second. A copy of the keyword listing has already been printed.

```
TYPE 1 IF THIS IS THE FIRST UPDATE
= 1
TOTAL NUMBER OF RECORDS IS NOW  100
UPDATE COMPLETE
THERE ARE NOW   88  KEYWORDS
TYPE 0 TO SUPRESS TOTAL KEYWORD LISTING
= 0
TRANSFER TO KEYWORD COMPLETED

PROGRAM STOP AT 2680
```

There is very little program output! The listing below gives an idea of the structure of the files KEYWORD and PERMFILE:

KEYWORD

```
 88
   2   15                                          ALGORITH
  10   80  145  155  185  265  300  315  325  345
 350  420  445  450  470    0    0    0   -2  180
   1    1                                          ANIMATIO
 180    0    0    0   -3   75  130  195  220  307
   1    7                                          BIBLIOGR
  75  130  195  220  307  355  465    0    0    0
   1    7                                          BLOCK DI
  30  285  315  375  420  435  480    0    0    0
   2   19                                          CARTOGRA
  45   55   90  110  120  155  180  205  230  240
```

PERMFILE

```
100
 5   ACI              1971 3CONTOURSTRIANGULMAPPING
10   ACM              1969 4INTERPOLALGORITHIRREGULADATA
15   AHLBERG,JH       1971 3SPLINES INTERPOLENGINEER
20   ANDERSON,WL      1971 3CUBIC   SPLINES REGULAR
25   AUMEN,WC         1966 2NUMERICAMAPPING
30   BASINGER,D GVILD1971 6BLOCK DITHREE DIGRAPHICSSURFACESCOMPUTER
  PROGRAM
35   BATCHA,JP REESE,1964 4GRIDDINGCONTOURSMINERALSEXPLORAT
40   BENGTSSON,BE NOR1964 5CONTOURSGRIDDINGTRIANGULMAPPING COMPUTER
```

```
45   BERTIN,J          1967 5CARTOGRAQUANTITACOMPUTERMAPPING FRENCH
50   BERTRAM,S         1965 2MAPPING ENGINEER
55   BICKMORE,DP       1967 5MAP ANALCOMPUTERGENERAL CARTOGRAENGLAND
60   BLUM,M            1969 3OPTIMISASMOOTHINMAPPING
65   BOEHM,BW          1967 4MULTIVARMAPPING TERRAIN MODELS
70   BOND,DH MCGLASHA1967 3COMPUTERGEOGRAPHMAPPING
75   BONHAM-CARTER,GF1966 5COMPUTERPROGRAM BIBLIOGRCORRELATMULTIVAR
```

So far only 100 of the cards have been read. The remaining 77 have yet to be edited and merged with PERMFILE and KEYWORD. The next sequence of operations reads and edits the 77 remaining cards to produce a keyword listing that is compatible with the one given earlier. The previous set of keywords now forms a lexicon for the spelling and nomenclature of any later keywords. The list can be added to, but no further alterations can be made to the word list itself without causing near duplicates to exist. Thus extreme care is needed in checking the listing given for the remaining 77 cards to insure that is agrees in all respects with the earlier one.

```
ENTER COMMAND
= READCARDS 77 NEWKEY PRINT KEYLIST
ENTER DATA
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
KEYWORD LIST GENERATED
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
KEYWORD AND FREQUENCY LIST
 ALGORITH  AREA      BIBLIOGR  BINARY    BLOCK DI  CARTOGRA  COMPUTER
        10         2         2         1         8        15        37
 CONTOURS  CORRELAT  CUBIC     CURVE FI  ENGINEER  ENVIRONM  ERRORS
        27         2         2         1         1         1         3
 EXPLORAT  FIT       FITTING   GENERAL   GEOGRAPH  GEOPHYSI  GRAPHICS
         1         1         1         6         3         1         9
 GRIDDING  HIDDEN L  INDUSTRY  INTERPOL  IRREGULA  ISOMETRI  LANDFORM
         5         4         1         5         1         3         2
 LEAST SQ  MAP ANAL  MAPPING   MATHEMAT  MATRIX    METEOROL  METHODS
         1         1        36         1         1         1         5
 MODEL     NUMERICA  OCEANOGR  OIL       OPTICS    OUTPUT    PATTERN
         2         3         3         1         1         1         2
 PERSPECT  PLOTTING  POLYGON   POLYHEDR  POLYNOMI  PRINTER   PROBABIL
         1        11         1         2         1         2         1
```

```
  PROFILE   PROGRAM   PROGRAMS  QUADRATI  QUANTITA  REMOTE S  ROTATION
        1        12         4         2         3         1         1
  SCANNING  SHADING   SLOPES    SMOOTHIN  SPACING   SPATIAL   SPLINES
        4         1         2         1         1         2         2
  STATISTI  STEREOGR  SURFACE   SYSTEMS   TERRAIN   THEORY    THREE DI
        2         3         2         1         2         3         1
  TREES
        1
 ENTER COMMAND
 = REPLACE AUTHOR 740 REPL AUTH 745 REPL AUTH 855
 ENTER COMMAND
 ENTER DATA
 ENTER CHARACTER STRING
 = SCUDDER,JD
 REPLACED ITEM IS  SCUDDEN,JD
 REPLACING ITEM IS  SCUDDER,JD
 ENTER COMMAND
 ENTER COMMAND
 ENTER DATA
 ENTER CHARACTER STRING
 = SHARP,JV CHRISTE
 REPLACED ITEM IS  SHAPR,JV CHRISTE
 REPLACING ITEM IS  SHARP,JV CHRISTE
 ENTER COMMAND
 ENTER COMMAND
 ENTER DATA
 ENTER CHARACTER STRING
 = WASHINGTON,WM OL
 REPLACED ITEM IS  WAHSINGTON,WM OL
 REPLACING ITEM IS  WASHINGTON,WM OL
 ENTER COMMAND
 = COPY PRINT KEYWORD PRINT KEYWORD PRIN KEYW PRIN KEYW PRIN KEYW
 DATA SAVED INTO INFILE
 POINTERS AT BEGINNING OF FILES
 ENTER COMMAND
 ENTER COMMAND
 ENTER CHARACTER STRING
 = FITTING
 KEYWORD OCCURS ON FOLLOWING CARDS
    765
 POINTERS AT BEGINNING OF FILES
 ENTER COMMAND
 ENTER COMMAND
 ENTER CHARACTER STRING
 = MODEL
 KEYWORD OCCURS ON FOLLOWING CARDS
    530   575
 POINTERS AT BEGINNING OF FILES
 ENTER COMMAND
 ENTER COMMAND
 ENTER CHARACTER STRING
 = PROFILE
 KEYWORD OCCURS ON FOLLOWING CARDS
    530
```

```
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER CHARACTER STRING
= PROGRAMS
KEYWORD OCCURS ON FOLLOWING CARDS
   625   760   785   840
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER CHARACTER STRING
= SURFACE
KEYWORD OCCURS ON FOLLOWING CARDS
   520   545
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
= SUBSTTITUE PRINT CARD 530
ENTER CHARACTER STRING
= PROGRAMS
ENTER CHARACTER STRING
= PROGRAM
PROGRAMS  REPLACED THROUGHOUT BY  PROGRAM
    4  CHANGES MADE
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ID IS  530   AUTHOR IS  MISULIA,MG SPOON  AND DATE  1957
THERE ARE   5  KEYWORDS, WHICH ARE
 PROFILE   SCANNING  MODEL     GRAPHICS  STEREOGR

ENTER COMMAND
= REPL KEYW 530 1 REPL KEYW 530 3 PRINT CARD 530 PRINT CARD 575
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= PROFILES
ENTER DATA
KEYWORD  PROFILE   REPLACED BY  PROFILES
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= MODELS
ENTER DATA
KEYWORD  MODEL     REPLACED BY  MODELS
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ID IS  530   AUTHOR IS  MISULIA,MG SPOON  AND DATE  1957
THERE ARE   5  KEYWORDS, WHICH ARE
 PROFILES  SCANNING  MODELS    GRAPHICS  STEREOGR
```

```
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ID IS  575   AUTHOR IS  MORSE,SP          AND DATE  1968
THERE ARE   3  KEYWORDS, WHICH ARE
 CONTOURS  MODEL     THEORY
ENTER COMMAND
= REPL KEYW 575 2 PRINT CARD 765
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= MODELS
ENTER DATA
KEYWORD  MODEL     REPLACED BY  MODELS
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ID IS  765   AUTHOR IS  SMITH,PJ          AND DATE  1971
THERE ARE   9  KEYWORDS, WHICH ARE
 QUADRATI  CUBIC     GRIDDING  PROGRAM   FITTING
 INTERPOL  POLYNOMI  SPLINES   LEAST SQ
ENTER COMMAND
= REPLACE KEYWORD 765 5 COPY SUBS NEWKEY PRINT KEYL
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= FIT
ENTER DATA
KEYWORD  FITTING   REPLACED BY  FIT
ENTER COMMAND
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER CHARACTER STRING
= SURFACE
ENTER CHARACTER STRING
= SURFACES
SURFACE   REPLACED THROUGHOUT BY  SURFACES
    2  CHANGES MADE
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
KEYWORD LIST GENERATED
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
```

```
KEYWORD AND FREQUENCY LIST
 ALGORITH  AREA      BIBLIOGR  BINARY    BLOCK DI  CARTOGRA  COMPUTER
       10         2         2         1         8        15        37
 CONTOURS  CORRELAT  CUBIC     CURVE FI  ENGINEER  ENVIRONM  ERRORS
       27         2         2         1         1         1         3
 EXPLORAT  FIT       GENERAL   GEOGRAPH  GEOPHYSI  GRAPHICS  GRIDDING
        1         2         6         3         1         9         5
 HIDDEN L  INDUSTRY  INTERPOL  IRREGULA  ISOMETRI  LANDFORM  LEAST SQ
        4         1         5         1         3         2         1
 MAP ANAL  MAPPING   MATHEMAT  MATRIX    METEOROL  METHODS   MODELS
        1        36         1         1         1         5         2
 NUMERICA  OCEANOGR  OIL       OPTICS    OUTPUT    PATTERN   PERSPECT
        3         3         1         1         1         2         1
 PLOTTING  POLYGON   POLYHEDR  POLYNOMI  PRINTER   PROBABIL  PROFILES
       11         1         2         1         2         1         1
 PROGRAM   QUADRATI  QUANTITA  REMOTE S  ROTATION  SCANNING  SHADING
       16         2         3         1         1         4         1
 SLOPES    SMOOTHIN  SPACING   SPATIAL   SPLINES   STATISTI  STEREOGR
        2         1         1         2         2         2         3
 SURFACES  SYSTEMS   TERRAIN   THEORY    THREE DI  TREES
        2         1         2         3         1         1
ENTER COMMAND
= STOP

PROGRAM STOP AT 4390
```

Note that the SUBSTITUTE command has been used to change the occurrence of PROGRAMS to PROGRAM, with a total of four changes in the list. Running the UPDATE program again gives the merged PERMFILE and KEYWORD, with a total of 177 records and 107 keywords. The latter is an increase of 19 keywords over the previous set. This would be expected as the list is not very large. An example printing of the merged versions of PERMFILE and KEYWORD is given below.

```
TYPE 1 IF THIS IS THE FIRST UPDATE
= 0
TOTAL NUMBER OF RECORDS IS NOW   177
UPDATE COMPLETE
THERE ARE NOW   107   KEYWORDS
TYPE 0 TO SUPRESS TOTAL KEYWORD LISTING
= 0
TRANSFER TO KEYWORD COMPLETED

PROGRAM STOP AT 2680
 OLD KEYWORD
READY
*LIST
```

```
 107
   3   25                                         ALGORITH
  10   80  145  155  185  265  300  315  325  345
 350  420  445  450  470  500  590  600  610  615
 640  670  775  850  870    0    0    0    0    0
   1    1                                         ANIMATIO
 180    0    0    0   -3   75  130  195  220  307
   1    2                                         AREA
 520  545    0    0   -3  625  815    0    0   -4
   1    9                                         BIBLIOGR
  75  130  195  220  307  355  465  625  815    0
   1    1                                         BINARY
 510    0    0    0   -5  620  690  775  785  820
   2   15                                         BLOCK DI
  30  285  315  375  420  435  480  620  690  775
 785  820  825  875  880  500  590  600  610  615
   4   34                                         CARTOGRA
  45   55   90  110  120  155  180  205  230  240
 265  270  335  365  370  405  410  420  465  570
 595  625  630  645  695  705  725  755  770  815
 825  830  845  865    0    0    0    0    0    0

*OLD PERMFILE
READY
*LIST

 177
   5  ACI                1971 3CONTOURSTRIANGULMAPPING
  10  ACM                1969 4INTERPOLALGORITHIRREGULADATA
  15  AHLBERG,JH         1971 3SPLINES INTERPOLENGINEER
  20  ANDERSON,WL        1971 3CUBIC   SPLINES REGULAR

*LIST 470

 470  MATHERON,G         1969 5THEORY  MAPPING ALGORITHMATHEMATKRIGING
 475  MCADAMS,HT         1969 3COMPUTERMAPPING DATA
 480  MCCUE,GA DWORETS1965 5STEREOGRBLOCK DICONTOURSPROGRAM PLOTTING
 485  MCCUE,GA GREEN,J1965 5GRAPHICSCOMPUTERSPATIAL TERRAIN CONTOURS
 490  MCINTYRE,DB POLL1968 3COMPUTERPROGRAM CONTOURS
 495  MCLAURIN,JD        1969 3GRAPHICSCOMPUTERPERCEPTI
 500  MCLELLAN,CD        1971 3COMPUTERALGORITHGEOGRAPH
 505  MCMAINS,F          1967 4CONTOURSPROGRAM MAPPING MATRIX
 510  MCMILLAN,RE JOHN1969 4COMPUTERMAP ANALBINARY  SCANNING
 515  MERRIAM,DF SNEAT1966 4QUANTITACONTOURSGEOPHYSIMAPPING
 520  MILLY,SA           1969 6PROGRAM COMPUTERSPATIAL CONTOURSSURFACES
   AREA    DATA    GENERAL
 525  MISKELL,RV         1971 4GRIDDINGINTERPOLNUMERICACUBIC

*LIST 870

 870  WOODFORD,CH        1969 4INTERPOLALGORITHCURVE FISMOOTHIN
 875  WOON,P             1970 5BLOCK DIPLOTTINGCOMPUTERHIDDEN LPOLYHEDR
 880  WRAY,WB            1970 4BLOCK DIISOMETRIPROGRAM COMPUTER

READY

*
```

Now all the original bibliography has been entered into the system. Meanwhile, some more references have been found and need to be added. A new file is created and labeled INPUT containing these new cards.

```
        00037BEDIENT,H NEILON196203CONTOURSMETEOROLMAPPING
        00053BHATTACHARYYA,BK197207MAPPING SPATIAL CONTOURSGRIDDINGIRREGULA
        00053                1972  SPLINES CUBIC
xxx     00077BOREN,HE        WN6802CURVE FIPROGRAM
xxx     00107CANRIGHT, RB SWI 66803THREE DIPROGRAM GRAPHICS
xxx     00253DOUGHERTY,E     8-6603FILTERINMAPPING LINEAR
        00317GRIM,PJ         197005PROGRAM PLOTTINGOCEANOGRMAGNETICPROFILES
xxx     00327HERSHEY,AV       36903NOTATIONCARTOGRAPROGRAM
xxx     00417KUBERT,BR       536803SURFACESCOMPUTERPERSPECT
xxx     00422KUBERT,BR        66903PROGRAM PLOTTINGCONTOURS
xxx     00767SNOWDEN,JM      9-6905PROGRAM GRAPHICSGRAVITY CONTOURSGRIDDING
xxx     00832TOBLER,WR ED.   SP7004PROGRAM GENERAL GEOGRAPHBIBLIOGR
xxx     00847VERZUH           65803CONTOURSPLOTTINGPROGRAM
```

Once again they are edited, and the old files updated to include them.

```
ENTER COMMAND
= READCARDS 12 NEWKEY PRINT KEYLIST
ENTER DATA
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
KEYWORD LIST GENERATED
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
ENTER COMMAND
KEYWORD AND FREQUENCY LIST
 BIBLIOGR  CARTOGRA  COMPUTER  CONTOURS  CUBIC     CURVE FI  FILTERIN
       .1         1         1         5         1         1         1
 GENERAL   GEOGRAPH  GRAPHICS  GRAVITY   GRIDDING  IRREGULA  LINEAR
        1         1         2         1         2         1         1
 MAGNETIC  MAPPING   METEOROL  NOTATION  OCEANOGR  PLOTTING  PERSPECT
        1         3         1         1         1         3         1
 PROFILES  PROGRAM   SPATIAL   SPLINES   SURFACES  THREE DI
        1         8         1         1         1         1
ENTER COMMAND
= PRINT CARD 847
ENTER COMMAND
ENTER DATA
ID IS  847   AUTHOR IS  VERZUH              AND DATE   658
THERE ARE   3  KEYWORDS, WHICH ARE
 CONTOURS  PLOTTING  PROGRAM
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
= REPLACE DATE 77
```

```
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
=-1968
REPLACED ITEM IS  WN68
REPLACING ITEM IS  1968
ENTER COMMAND
= REPL DATE 107 REPL DATE 253 REPL DATE 327
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1968
REPLACED ITEM IS   668
REPLACING ITEM IS  1968
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1966
REPLACED ITEM IS  8-66
REPLACING ITEM IS  1966
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1969
REPLACED ITEM IS   369
REPLACING ITEM IS  1969
ENTER COMMAND
= REPL DATE 417 REPL DATE 422  REPL DATE 767 REPL DATE 832 REPL DATE
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1968
REPLACED ITEM IS  5368
REPLACING ITEM IS  1968
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1969
REPLACED ITEM IS   669
REPLACING ITEM IS  1969
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1969
REPLACED ITEM IS  9-69
REPLACING ITEM IS  1969
ENTER COMMAND
ENTER COMMAND
ENTER DATA
ENTER CHARACTER STRING
= 1970
```

```
REPLACED ITEM IS  SP70
REPLACING ITEM IS  1970
ENTER COMMAND
ENTER COMMAND
ENTER DATA
= 847
ENTER CHARACTER STRING
= 1958
REPLACED ITEM IS   658
REPLACING ITEM IS  1958
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND
= COPY STOP
DATA SAVED INTO INFILE
POINTERS AT BEGINNING OF FILES
ENTER COMMAND

PROGRAM STOP AT 4390
*
```

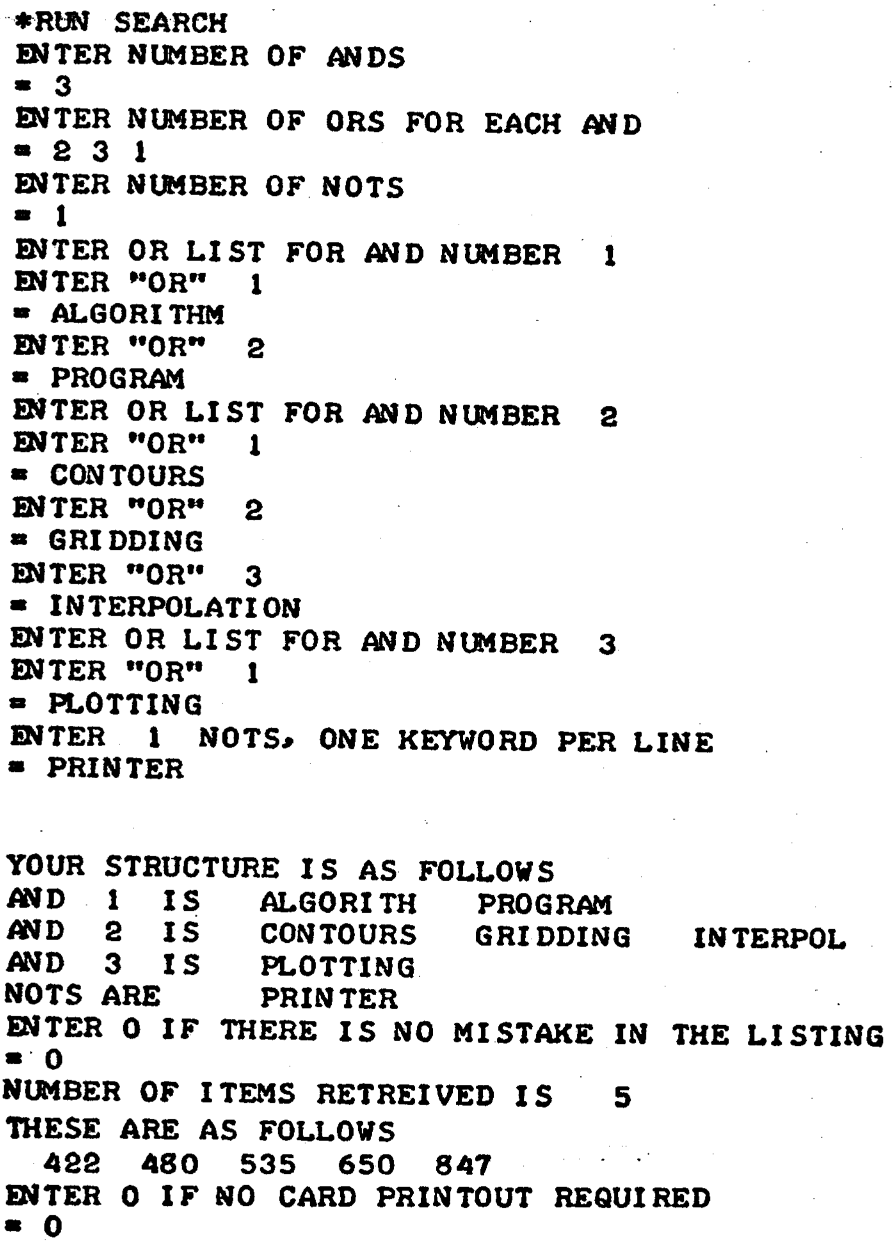
As these are the last cards entering the system for a while, a listing of the total keyword index is requested when using program UPDATE so that a record can be kept of the keywords and frequencies of occurrence in the updated bibliography.

```
*RUN UPDATE

TYPE 1 IF THIS IS THE FIRST UPDATE
= 0
TOTAL NUMBER OF RECORDS IS NOW   189
UPDATE COMPLETE
THERE ARE NOW  110  KEYWORDS
TYPE 0 TO SUPRESS TOTAL KEYWORD LISTING
= 1

ALGORITH      25                  DESIGN         1
ANIMATIO       1                  DIAGRAMS       1
AREA           2                  DISTRIBU       1
BIBLIOGR      10                  ENGINEER       4
BINARY         1                  ENGLAND        1
BLOCK DI      15                  ENVIRONM       1
CARTOGRA      35                  ERRORS         5
CLASSIFI       1                  EXPLORAT       5
COMPUTER      81                  FILMS          1
CONTOURS      69                  FILTERIN       1
CORRELAT       3                  FIT            5
CRYSTALL       1                  FOURIER        1
CUBIC          5                  FRENCH         1
CURVE FI       5                  GENERAL       15
DATA           5                  GEOGRAPH       8
```

```
GEOLOGY      1                 PLANNING     2
GEOPHYSI     1                 PLOTTING    23
GRAPHICS    27                 POLYGON      1
GRAVITY      1                 POLYHEDR     4
GRIDDING    14                 POLYNOMI     2
HALF TON     1                 POPULATI     1
HIDDEN L     8                 PRINTER      7
HYDROLOG     1                 PROBABIL     2
INDUSTRY     1                 PROFILES     3
INFORMAT     1                 PROGRAM     49
INTERPOL    14                 QUADRATI     4
IRREGULA     5                 QUANTITA     6
ISOMETRI     3                 RANDOM       2
ITERATIV     2                 REGULAR      1
KRIGING      2                 REMOTE S     1
LANDFORM     2                 ROTATION     1
LEAST SQ     1                 SCANNING     5
LINEAR       3                 SEDIMENT     1
MAGNETIC     2                 SHADING      1
MAP ANAL     9                 SIMULATI     2
MAPPING     93                 SINGLE V     1
MATHEMAT     3                 SLOPES       2
MATRIX       1                 SMOOTHIN     3
MATRIX A     1                 SPACING      2
METEOROL     2                 SPATIAL      8
METHODS     10                 SPLINES      7
MINERALS     1                 STATISTI     2
MODELS       3                 STEREOGR     5
MULTIVAR     2                 STRUCTUR     1
NOTATION     3                 SURFACES    11
NUMERICA     7                 SYSTEMS      1
OCEANOGR     5                 TERRAIN      7
OIL          2                 THEORY       4
OPTICS       2                 THREE DI     7
OPTIMISA     1                 TRANSFOR     1
ORIENTAT     1                 TREES        1
OUTPUT       1                 TRIANGUL     3
PATTERN      3                 TSA          2
PERCEPTI     1                 X-RAY        1
PERSPECT     4


TRANSFER TO KEYWORD COMPLETED

PROGRAM STOP AT 2680
```

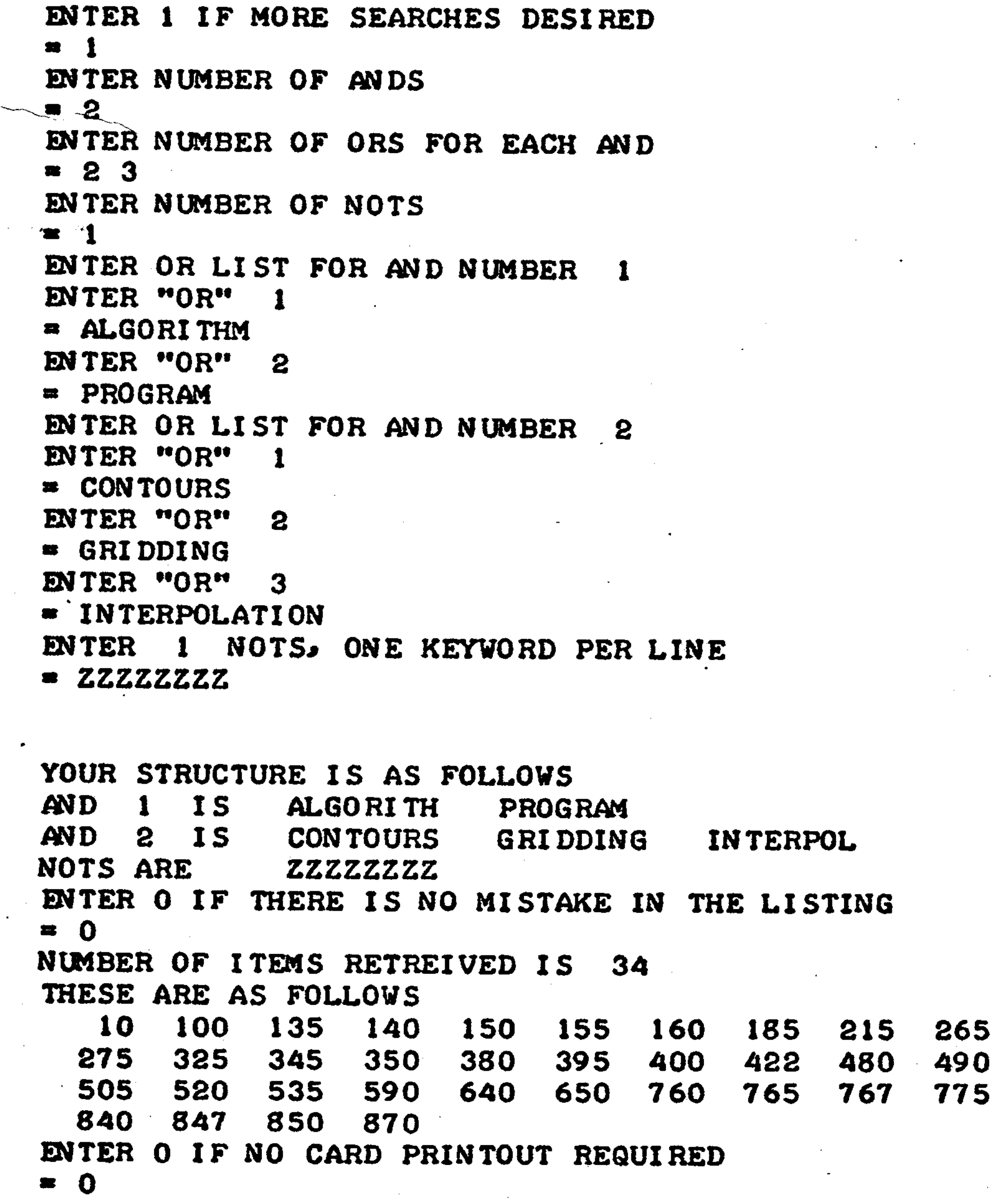
```
     PERMFILE

*LIST

 189
   5 ACI               1971 3CONTOURSTRIANGULMAPPING
  10 ACM               1969 4INTERPOLALGORITHIRREGULADATA
  15 AHLBERG,JH        1971 3SPLINES INTERPOLENGINEER
  20 ANDERSON,WL       1971 3CUBIC   SPLINES REGULAR
  25 AUMEN,WC          1966 2NUMERICAMAPPING
  30 BASINGER,D GVILD1971 6BLOCK DITHREE DIGRAPHICSSURFACESCOMPUTER
   PROGRAM POLYNOMISPLINES LEAST SQ
  35 BATCHA,JP REESE,1964 4GRIDDINGCONTOURSMINERALSEXPLORAT
  37 BEDIENT,H NEILON1962 3CONTOURSMETEOROLMAPPING
  40 BENGTSSON,BE NOR1964 5CONTOURSGRIDDINGTRIANGULMAPPING COMPUTER
  45 BERTIN,J          1967 5CARTOGRAQUANTITACOMPUTERMAPPING FRENCH
  50 BERTRAM,S         1965 2MAPPING ENGINEER
  53 BHATTACHARYYA,BK1972 7MAPPING SPATIAL CONTOURSGRIDDINGIRREGULA
   SPLINES CUBIC   SPLINES LEAST SQ
  55 BICKMORE,DP       1967 5MAP ANALCOMPUTERGENERAL CARTOGRAENGLAND
  60 BLUM,M            1969 3OPTIMISASMOOTHINMAPPING
  65 BOEHM,BW          1967 4MULTIVARMAPPING TERRAIN MODELS
  70 BOND,DH MCGLASHA1967 3COMPUTERGEOGRAPHMAPPING
  75 BONHAM-CARTER,GF1966 5COMPUTERPROGRAM BIBLIOGRCORRELATMULTIVAR
  77 BOREN,HE          1968 2CURVE FIPROGRAM
  80 BOUKNIGHT,WJ      1969 4HALF TONGRAPHICSCOMPUTERALGORITH
  85 BROWN,AE          1963 5MATRIX ATERRAIN PROFILESSIMULATIMAPPING


          KEYWORD

*LIST

 110
    3   25                                       ALGORITH
   10   80  145  155  185  265  300  315  325  345
  350  420  445  450  470  500  590  600  610  615
  640  670  775  850  870    0    0    0    0    0
    1    1                                       ANIMATIO
  180    0    0    0   -3   75  130  195  220  307
    1    2                                       AREA
  520  545    0    0   -3  625  815    0    0   -4
    1   10                                       BIBLIOGR
   75  130  195  220  307  355  465  625  815  832
    1    1                                       BINARY
  510    0    0    0   -5  620  690  775  785  820
    2   15                                       BLOCK DI
   30  285  315  375  420  435  480  620  690  775
  785  820  825  875  880  500  590  600  610  615
    4   35                                       CARTOGRA
   45   55   90  110  120  155  180  205  230  240
  265  270  335  365  370  405  410  420  465  570
  595  625  630  645  695  705  725  755  770  815
  825  830  845  865  327    0    0    0    0    0
```

As the bibliography is now complete, the point has been reached where it can be asked useful questions, and expected to give reasonable answers. Various examples are given below. The first is concerned with the discovery of published programs or algorithms concerned with the contouring problem.



```
*RUN SEARCH
ENTER NUMBER OF ANDS
= 3
ENTER NUMBER OF ORS FOR EACH AND
= 2 3 1
ENTER NUMBER OF NOTS
= 1
ENTER OR LIST FOR AND NUMBER  1
ENTER "OR"  1
= ALGORITHM
ENTER "OR"  2
= PROGRAM
ENTER OR LIST FOR AND NUMBER  2
ENTER "OR"  1
= CONTOURS
ENTER "OR"  2
= GRIDDING
ENTER "OR"  3
= INTERPOLATION
ENTER OR LIST FOR AND NUMBER  3
ENTER "OR"  1
= PLOTTING
ENTER  1  NOTS, ONE KEYWORD PER LINE
= PRINTER


YOUR STRUCTURE IS AS FOLLOWS
AND  1  IS   ALGORITH   PROGRAM
AND  2  IS   CONTOURS   GRIDDING   INTERPOL
AND  3  IS   PLOTTING
NOTS ARE     PRINTER
ENTER 0 IF THERE IS NO MISTAKE IN THE LISTING
= 0
NUMBER OF ITEMS RETREIVED IS   5
THESE ARE AS FOLLOWS
  422  480  535  650  847
ENTER 0 IF NO CARD PRINTOUT REQUIRED
= 0
```

The second problem shows what can happen when a question is made too general, and also how a dummy keyword can be used to supply a NOT.



```
ENTER 1 IF MORE SEARCHES DESIRED
= 1
ENTER NUMBER OF ANDS
= 2
ENTER NUMBER OF ORS FOR EACH AND
= 2 3
ENTER NUMBER OF NOTS
= 1
ENTER OR LIST FOR AND NUMBER  1
ENTER "OR"  1
= ALGORITHM
ENTER "OR"  2
= PROGRAM
ENTER OR LIST FOR AND NUMBER  2
ENTER "OR"  1
= CONTOURS
ENTER "OR"  2
= GRIDDING
ENTER "OR"  3
= INTERPOLATION
ENTER  1  NOTS, ONE KEYWORD PER LINE
= ZZZZZZZZ


YOUR STRUCTURE IS AS FOLLOWS
AND  1  IS   ALGORITH   PROGRAM
AND  2  IS   CONTOURS   GRIDDING   INTERPOL
NOTS ARE     ZZZZZZZZ
ENTER 0 IF THERE IS NO MISTAKE IN THE LISTING
= 0
NUMBER OF ITEMS RETREIVED IS  34
THESE ARE AS FOLLOWS
   10  100  135  140  150  155  160  185  215  265
  275  325  345  350  380  395  400  422  480  490
  505  520  535  590  640  650  760  765  767  775
  840  847  850  870
ENTER 0 IF NO CARD PRINTOUT REQUIRED
= 0
```

In the third question attention has turned to published programs concerned with block diagram construction with the option of printing out not only the ID but a brief reference to the article contents (contained in PERMFILE).

```
ENTER 1 IF MORE SEARCHES DESIRED
= 1
ENTER NUMBER OF ANDS
= 2 3 2 1
ENTER NUMBER OF ORS FOR EACH AND
ENTER NUMBER OF NOTS
ENTER OR LIST FOR AND NUMBER  1
ENTER "OR"  1
= BLOCK DI
ENTER "OR"  2
= ISOMETRIC
ENTER "OR"  3
= PERSPECTIVE
ENTER OR LIST FOR AND NUMBER  2
ENTER "OR"  1
= ALGORITHM
ENTER "OR"  2
= PROGRAM
ENTER  1  NOTS, ONE KEYWORD PER LINE
= PRINTER


YOUR STRUCTURE IS AS FOLLOWS
AND  1  IS   BLOCK DI   ISOMETRI   PERSPECT
AND  2  IS   ALGORITH   PROGRAM
NOTS ARE     PRINTER
ENTER 0 IF THERE IS NO MISTAKE IN THE LISTING
= 0
NUMBER OF ITEMS RETREIVED IS   9
THESE ARE AS FOLLOWS
   30  315  375  420  480  620  775  785  880
ENTER 0 IF NO CARD PRINTOUT REQUIRED
= 1

ID =   30  AUTHOR IS  BASINGER,D GVILD  DATE =  1971
THERE ARE  6  KEYWORDS, WHICH ARE
 BLOCK DI  THREE DI  GRAPHICS  SURFACES  COMPUTER  PROGRAM

ID =  315  AUTHOR IS  GRAHAM,NY         DATE =  1972
THERE ARE  5  KEYWORDS, WHICH ARE
 PERSPECT  SURFACES  HIDDEN L  BLOCK DI  ALGORITH

ID =  375  AUTHOR IS  JESPERSON,H       DATE =  1970
THERE ARE  3  KEYWORDS, WHICH ARE
 PROGRAM   BLOCK DI  THREE DI

ID =  420  AUTHOR IS  KUBERT,B SZABO,J  DATE =  1968
THERE ARE  5  KEYWORDS, WHICH ARE
 BLOCK DI  PERSPECT  ALGORITH  PLOTTING  CARTOGRA

ID =  480  AUTHOR IS  MCCUE,GA DWORETS  DATE =  1965
THERE ARE  5  KEYWORDS, WHICH ARE
 STEREOGR  BLOCK DI  CONTOURS  PROGRAM   PLOTTING

ID =  620  AUTHOR IS  NOMA,AA MISULIA,  DATE =  1959
THERE ARE  4  KEYWORDS, WHICH ARE
 COMPUTER  PROGRAM   TERRAIN   BLOCK DI
```

```
ID =  775  AUTHOR IS  SPRUNT,B          DATE =  1972
THERE ARE  5  KEYWORDS, WHICH ARE
 CONTOURS  BLOCK DI  PERSPECT  ISOMETRI  ALGORITH

ID =  785  AUTHOR IS  SUNDBERG,WD       DATE =  1970
THERE ARE  5  KEYWORDS, WHICH ARE
 BLOCK DI  COMPUTER  PROGRAM  HIDDEN L  ISOMETRI

ID =  880  AUTHOR IS  WRAY,WB           DATE =  1970
THERE ARE  4  KEYWORDS, WHICH ARE
 BLOCK DI  ISOMETRI  PROGRAM  COMPUTER
```

The last question is related to printer graphics only, but of all types.

```
ENTER 1 IF MORE SEARCHES DESIRED
= 1
ENTER NUMBER OF ANDS
= 302
ENTER NUMBER OF ORS FOR EACH AND
= 1 3
ENTER NUMBER OF NOTS
= 1
ENTER OR LIST FOR AND NUMBER  1
ENTER "OR"  1
= PRINTER
ENTER OR LIST FOR AND NUMBER  2
ENTER "OR"  1
= CONTOURS
ENTER "OR"  2
= GRAPHICS
ENTER "OR"  3
= OUTPUT
ENTER  1  NOTS, ONE KEYWORD PER LINE
= PLOTTING

YOUR STRUCTURE IS AS FOLLOWS
AND  1  IS   PRINTER
AND  2  IS   CONTOURS  GRAPHICS  OUTPUT
NOTS ARE     PLOTTING
ENTER 0 IF THERE IS NO MISTAKE IN THE LISTING
= 0
NUMBER OF ITEMS RETREIVED IS   3
THESE ARE AS FOLLOWS
  175  305  320
ENTER 0 IF NO CARD PRINTOUT REQUIRED
= 1
```

```
ID =  175  AUTHOR IS  COPPOCK,JT        DATE =  1972
THERE ARE  3  KEYWORDS, WHICH ARE
 GRAPHICS  PRINTER   GEOGRAPH

ID =  305  AUTHOR IS  GARFINKEL,D       DATE =  1962
THERE ARE  3  KEYWORDS, WHICH ARE
 PRINTER   GRAPHICS  PROGRAM

ID =  320  AUTHOR IS  HARVARD UNIVERSI  DATE =  1971
THERE ARE  4  KEYWORDS, WHICH ARE
 MAPPING   PRINTER   GRAPHICS  CONTOURS
ENTER 1 IF MORE SEARCHES DESIRED
= 0


END OF PROGRAM RETREIVER

PROGRAM STOP AT 3710
*
```

Section F - LIST

Use of Program LIST

LIST is different from the other programs in the KARP package in that it does not assume a time sharing environment. It is written as a batch program using both random and serial files to store data. The purpose of the program is to print an expanded version of the cross reference index storéd in KEYWORD on the lineprinter giving not only the author, date, and keywords contained in PERMFILE but also a full bibliographic reference.

Operation of LIST

Three files are read by LIST. File code 01 refers to a random file that is created by the program to hold the entire bibliographic record in compressed form. File 02 is a serial file holding the bibliographic information in uncompressed form. The cross reference index is held in file 03. The format of file 03 is the same as that for KEYWORD, and will probably be that file. File 02 needs explanation. It consists of a series of cards or card images holding every reference in the system. In a large system this may be a tape rather than a disc file. The first card or cards for a given ID will be in the same format as the card images stored in PERMFILE. These will be followed by the bibliographic reference typed in columns nine to 72 of the following cards. The only conventions to be followed on these N cards are:

1) Words can be carried across cards providing column 72 of the first of the cards and column 9 of the second contain non blank characters, and conversely,

2) a blank character in either column 72 of the first, or column 9 of the second is considered to be a spacer between two words.

As many gaps between words as desired may be left with as little or as much information per card. The format of a typical card for the Screwtape example used earlier might be:

```
         1         2         3         4         5         6         7
1234567890123456789012345678901234567890123456789012345678901234567890

 234  SCREWTAPE,D BEEL1972 8DEVILS  SATANIC LETTERS WORMWOODUNCLE

    ADVICE  TORMENT ETERNAL

        EXTRACT OF LETTERS FROM SCREWTAPE TO HIS NEPHEW,

*       NOW DECEASED.
```

Note the * in column one of the last card of the Screwtape reference. This indicates to the program that the last card has been read and that a new ID will follow. The program will read in these references, and then compress the free text reference to eliminate unnecessary spaces. This is performed by reading the text into the machine in A1 format and removing any extra blank characters. Then the number of computer words used is reduced by converting the text string from a maximum of 384A1 to 64A6 by means of the ENCODE function supplied on most modern compilers. This function could probably be replaced by a very simple machine code routine to achieve the same end. The doubly compressed record is then written out to the random file as record X. X is not the ID number as there may well be gaps in the ID sequence, but its position in the file. In other words, if it occurs as the tenth record although its ID may be 84, its reference in random file 01 will be 10. Once the entire list has been processed the reading of end of file on file 02 will initiate the start of the printing sequence.

The use of a random file is not obligatory as the files 02 and 03 are both in numeric order. It would be possible to use a serial storage system and to read right through file 01 for every keyword to find the appropriate IDs for that keyword. A random file is, however, more efficient. The output of the program is displayed in the bibliography printed in this issue.

Data Input to LIST

This consists of one number read from file 05, usually the card reader. This number is either 1 or 0 depending on whether a deck of the compressed reference list is required. One indicates a deck is to be punched, 0 not. As there is only one number to be input to the program, the free format function is not used, and the number is read in I5. When the cards are punched, columns 9 to 11 inclusive are punched with '=' signs. This is to pack out the line so that on rereading of the data although only 10A6 characters are being written from this punched deck, the program will not consider a gap to exist between records unless a true blank character exists at one end of the 10A6. The symbol '=' must not be used in the text of the reference as it is ignored by the compression algorithm.

Error Messages in LIST

There are two error messages printed out by LIST. One refers to a missing ID. It is fairly common that when typing out the reference cards the * will be omitted from the last card. This means that until another * is encountered the program will consider all following, even different references, to be part of the previous reference. If this happens, or if a reference of a given ID has been omitted completely from file 02, the message:

NO ID XXX FOUND IN FILE

will be printed wherever appropriate.

The second is related to the amount of input for a given record. This can be on as many cards as is desired, but must not consist of more than 379 significant characters including a space between every word. If more than this limit is present, the message:

RECORD XXXX TOO BIG FOR STORAGE

will be printed. The program will retain all information read up to that point, but will then skip forward in the file until an * is encountered, presumably at the start of a new record. When the overlarge record is printed only that portion consisting of less than 380 characters will appear.

Section G - PROGRAM

```
                         KARP/EDITOR      1
1000        DIMENSION KOM(15,4),KARD(60),KPEY(200),KNUM(100)
1010        COMMON NUMS(72),KYMB(6),KIN
1020        FILENAME IN,OUT
1030        DATA KPEY(1)/1H-/,KPEY(2)/1H+/,KPEY(3)/1H./
1040        DATA KPEY(4)/1H /,KPEY(5)/1H /,KPEY(6)/1H0/,I/1H1/
1050        DATA KARD /1HA,1HU,1HT,1HH,1HB,1HE,1HG,
1060&1HI,1HC,1HA,1HR,1HD,1HC,1HO,1HP,1HY,1HD,1HA,1HT,1HE,1HD,1HE,
1070&1HL,1HE,1HI,1HN,1HS,1HE,1HK,1HE,1HY,1HL,1HK,1HE,1HY,1HW,1HN,
1080&1HE,1HW,1HK,1HP,1HR,1HI,1HN,1HR,1HE,1HA,1HD,1HR,1HE,1HP,1HL,
1090&1HS,1HU,1HB,1HS,1HS,1HT,1HO,1HP/
1100C......SET UP COMMAND NAMES AND FREE FORMAT SYMBOLS
1110        L=0
1120        DO 147 J=1,15
1130        DO 147 K=1,4
1140        L=L+1
1150    147 KOM(J,K)=KARD(L)
1160        DO 146 J=1,6
1170    146 KYMB(J)=KPEY(J)
1180        KYMB(5)=I-KYMB(6)
1190        KARD(35)=KYMB(4)
1200        IN="INFILE"
1210        OUT="OUTFILE"
1220        CALL NEWCRD
1230        KTEST=0
1240        KNU=-11
1250        KX=0
1260        IP=1
1270C......IF INFILE HAS BEEN READ TO THE EOF GOTO 103
1280    100 IF((IP.LE.KX).OR.(KX.EQ.0))GOTO 103
1290        GOTO 116
1300C......THE MAJOR COMMAND NAME READ IN THE PROGRAM
1310    103 CALL COM(KR,KOM,KERR,1)
1320        IF(KERR.NE.0)GOTO 139
1330C......JUMP TO THE 15 POSSIBLE COMMANDS
1340        GOTO(101,141,101,116,101,102,102,101,101,104,102,106,102,
1350&108,140),KR
1360C......JUMPS HERE IF COMMAND JUST READ NOT FIRST LEVEL
1370    101 PRINT 201,(NUMS(I),I=1,72)
1380        GOTO 100
1390C......106 IS THE START OF THE READCARDS SEQUENCE
1400    106 CALL COM(KX,KOM,KERR,2)
1410        WRITE("INFILE",202)KX
1420        KAX=KX
1430    105 I=0
1440    110 I=I+1
1450C......READIT READS A CARD FROM THE INPUT FILE
1460        CALL READIT(1,0,KARD,NX,KERR)
```

```
1470  142 KQ=KARD(7)
1480      KD=KQ
1490      KC=0
1500C......COPY IS USED TO TRANSFER DATA FROM ONE ARRAY TO ANOTHER
1510      CALL COPY(KARD,1,17,KPEY,KC)
1520C......LOOP 111 READS IN REST OF RECORD FROM DATA FILE
1530      DO 111 J=1,2
1540      KD=KD-5
1550      IF(KD.LE.0)GOTO 155
1560      CALL READIT(2,0,KARD,NX,KERR)
1570      IF(KERR.NE.0)GOTO 113
1580      KB=KD*2
1590      IF(KB.GT.10)KB=10
1600      KB=7+KB
1610  111 CALL COPY(KARD,8,KB,KPEY,KC)
1620  155 CALL COPY(KPEY,1,KC,KARD,0)
1630C......RITE WRITES OUT TO A FILE DATA READ FROM THE INPUT FILE
1640  112 CALL RITE(IN,KARD,KC,KQ)
1650      IF((KR.EQ.6).OR.(KR.EQ.7))GOTO 125
1660      IF(KR.EQ.13)GOTO 129
1670      GOTO 144
1680C......CARD DELETED FROM INPUT FILE BECAUSE OF ERROR IN INPUT
1690  113 PRINT 203,NX
1700      KAX=KAX-1
1710  143 CALL READIT(1,0,KARD,NX,KERR)
1720      IF(KARD(7))142,143,142
1730  144 IF(I.LT.KAX)GOTO 110
1740      IF(KX.NE.KAX)KX=KX-1
1750C......START ZEROES THE FILE MARKERS FOR READING OR WRITING
1760      CALL START(KX,IN,OUT,IP)
1770      GOTO 116
1780C......SECOND LEVEL COMMAND READ BY COMM
1790  102 CALL COM(KA,KOM,KERR,1)
1800      IF(KA.EQ.8)GOTO 126
1810      IF((KA.EQ.9).AND.(KR.EQ.11))GOTO 108
1820      IF(KERR.EQ.0)GOTO 114
1830      PRINT 207
1840      GOTO 102
1850C......SEARCH FOR CARD KB
1860  114 CALL COM(KB,KOM,KERR,2)
1870      IF(KB.EQ.KNU)GOTO 117
1880C......IF CARD KB ALREADY FOUND GOTO 117, OTHERWISE CALL FIND
1890C......IF KTEST IS ONE A CARD NEEDS TO BE WRITTEN TO A FILE
1900C......BEFORE ANOTHER CARD IS READ.
1910      IF(KTEST.EQ.1)CALL RITE(OUT,KARD,KL,KQ)
1920      CALL FIND(IN,OUT,KARD,IP,KX,KB,KQ,KERR)
1930      KTEST=0
```

```
1940        KNU=KB
1950        IF(KERR)116,117,116
1960   117 IF(KR-7)107,118,119
1970   107 IF(KA.EQ.3)GOTO 120
1980C......DETERMINE POSITION OF KEYWORD TO BE DELETED
1990        CALL COM(KB,KOM,KERR,2)
2000        KL=KQ*2+6
2010        KB=KB*2+6
2020        PRINT 208,KARD(KB),KARD(KB+1)
2030        KQ=KQ-1
2040        DO 121 I=KB,KL,2
2050        KARD(I)=KARD(I+2)
2060   121 KARD(I+1)=KARD(I+3)
2070C......FILL EMPTY SPACES WITH A BLANK CHARACTER
2080        KARD(KL)=KARD(35)
2090        KARD(KL+1)=KARD(35)
2100        KARD(7)=KQ
2110        KL=KL-1
2120C......SET MARKER TO SHOW RECORD NOT TRANSFERRED TO OUTPUT FILE
2130   145 KTEST=1
2140        GOTO 100
2150C......DELETE RECORD KB
2160   120 PRINT 209,KB
2170        IP=IP-1
2180        KX=KX-1
2190   154 KTEST=0
2200        KNU=-11
2210        GOTO 100
2220C......WRITE RECORD TO OUTPUT FILE AND THEN PROCEED TO NEXT CASE
2230   122 CALL RITE(OUT,KARD,KL,KQ)
2240        IF((KR.EQ.7).AND.(KA.EQ.3))GOTO 105
2250        GOTO 100
2260C......INSERT A CARD OR KEYWORD
2270   118 IF(KA.NE.3)GOTO 123
2280C......BEGINNING OF READ OF NEW RECORD
2290        IN="OUTFILE"
2300        KAX=1
2310        GOTO 122
2320C......PRINT OF RECORD INSERTION
2330   125 PRINT 210
2340        KX=KX+1
2350        IP=IP+1
2360C......PRINT OF NEW REPLACEMENT RECORD
2370   129 PRINT 204,(KPEY(I),I=1,KC)
2380        IN="INFILE"
2390        GOTO 154
2400   123 PRINT 223
```

```
2410C......INSERTION OF KEYWORD INTO RECORD
2420          READ 222,(KPEY(I),I=1,2)
2430          PRINT 211,KPEY(1),KPEY(2)
2440          KL=KQ*2+9
2450          KQ=KQ+1
2460          KARD(7)=KQ
2470          KARD(KL-1)=KPEY(1)
2480          KARD(KL)=KPEY(2)
2490          GOTO 145
2500     119  KL=KQ*2+7
2510          IF(KR.NE.11)GOTO 124
2520C......PRINTS RECORD
2530          PRINT 204,(KARD(I),I=1,KL)
2540          GOTO 145
2550C......PRINT LIST OF KEYWORDS
2560     126  BEGIN FILE "KEYLIST"
2570          READ("KEYLIST",202)KC
2580          I=KC/2
2590          READ("KEYLIST",212)(KPEY(J),J=1,200)
2600          READ("KEYLIST",224)(KNUM(J),J=1,100)
2610          K=(I-1)/7+1
2620          PRINT 226
2630          DO 148 J=1,K
2640          KL=J*7
2650          KM=KL-6
2660          IF(KL.GT.I)KL=I
2670          KN=KL*2
2680          KO=KM*2-1
2690          PRINT 212,(KPEY(L),L=KO,KN)
2700     148  PRINT 224,(KNUM(L),L=KM,KL)
2710          GOTO 100
2720C......REPLACE ENTIRE RECORD
2730     124  IF(KA.NE.3)GOTO 128
2740          IN="OUTFILE"
2750          KAX=1
2760          PRINT 213
2770          PRINT 204,(KARD(I),I=1,KL)
2780          PRINT 214
2790          GOTO 105
2800C......REPLACE AUTHOR OR DATE OR KEYWORD
2810     128  K=0
2820          L=0
2830          LK=0
2840          IF(KA-5)130,131,132
2850C......REPLACE AUTHOR
2860     130  K=K+2
2870          L=L+1
```

```
2880       LK=-4
2890C......REPLACE DATE
2900  132 K=K+1
2910       L=L+1
2920C......REPLACE KEYWORD
2930  131 K=K+1
2940       L=L+1
2950       LK=LK+6
2960       PRINT 223
2970C......READS IN CHARACTER STRING FOR SUBSTITUTION
2980       READ 222,(KPEY(I),I=1,K)
2990       IF(KA.EQ.9)GOTO 133
3000       KC=LK+K-1
3010       PRINT 215,(KARD(I),I=LK,KC)
3020       PRINT 216,(KPEY(I),I=1,K)
3030       K=0
3040       DO 134 I=LK,KC
3050       K=K+1
3060  134 KARD(I)=KPEY(K)
3070       GOTO 145
3080C......KEYWORD REPLACEMENT SECTION
3090  133 CALL COM(KB,KOM,KERR,2)
3100       KC=KB*2+6
3110       PRINT 217,KARD(KC),KARD(KC+1),KPEY(1),KPEY(2)
3120       KARD(KC)=KPEY(1)
3130       KARD(KC+1)=KPEY(2)
3140       GOTO 145
3150C......INITIATES COPY CYCLE TO SAVE EDITED FILE BACK INTO INFILE
3160  116 IF(KTEST.EQ.1)CALL RITE(OUT,KARD,KL,KQ)
3170       CALL FIND("INFILE","OUTFILE",KARD,IP,KX,-1,KQ,KERR)
3180       CALL START(KX,OUT,IN,IP)
3190       CALL FIND(OUT,IN,KARD,IP,KX,-1,KQ,KERR)
3200       KTEST=0
3210       KNU=-11
3220       PRINT 219
3230C......BEGINFILE COMMAND
3240  141 CALL START(KX,IN,OUT,IP)
3250       PRINT 218
3260       GOTO 100
3270C......JUMPS HERE FOR SUBSTITUTION OF A KEYWORD THROUGHOUT FILE
3280  108 PRINT 223
3290C......READS IN KEYWORD TO BE REPLACED
3300       READ 222,(KPEY(I),I=1,2)
3310C......JUMP TO 151 IF KEYWORD OCCURENCE LIST IS TO BE MADE
3320       IF(KR.EQ.11)GOTO 151
3330       PRINT 223
3340C......READS IN REPLACING KEYWORD
```

```
3350        READ 222,KB,KC
3360        PRINT 220,KPEY(1),KPEY(2),KB,KC
3370C......CALCULATION OF KEYWORD OCCURENCE LISTING STARTS HERE. IT
3380C......PROCEEDS ALONGSIDE THE COMMAND SUBSTITUTE.
3390   151 KM=0
3400C......GO TO START OF FILE
3410        CALL START(KX,IN,OUT,IP)
3420        K=0
3430        DO 135 I=1,KX
3440C......FIND LOOKS FOR A PARTICULAR CARD IN INFILE
3450        CALL FIND(IN,OUT,KARD,IP,KX,0,KQ,KERR)
3460        LK=KQ*2+6
3470C......CHECK TO SEE IF KEYWORD EXISTS ON THIS RECORD
3480        DO 109 J=8,LK,2
3490        IF(KARD(J).NE.KPEY(1))GOTO 109
3500        IF(KARD(J+1).NE.KPEY(2))GOTO 109
3510        IF(KR.NE.11)GOTO 152
3520        KM=KM+1
3530        KNUM(KM)=KARD(1)
3540        GOTO 109
3550C......REPLACE KEYWORD IF FOUND
3560   152 KARD(J)=KB
3570        KARD(J+1)=KC
3580        K=K+1
3590   109 CONTINUE
3600        KL=LK+1
3610        IF(KR.EQ.11)GOTO 135
3620C......WRITE RECORD TO FILE AND PRINT NUMBER OF SUBSTITUTIONS
3630        CALL RITE(OUT,KARD,KL,KQ)
3640   135 CONTINUE
3650        IF(KR.NE.11)GOTO 153
3660        IF(KM.EQ.0)GOTO 141
3670        PRINT 225
3680        PRINT 227,(KNUM(J),J=1,KM)
3690        GOTO 141
3700   153 PRINT 221,K
3710        GOTO 100
3720C......KEYWORD LIST GENERATION SEQUENCE
3730   104 BEGIN FILE "KEYLIST"
3740        CALL START(KX,IN,OUT,IP)
3750        LIST=2
3760        DO 150 I=1,100
3770   150 KNUM(I)=1
3780        DO 136 I=1,KX
3790        CALL FIND(IN,OUT,KARD,IP,KX,0,KQ,KERR)
3800        LK=KQ*2+6
3810        DO 137 J=8,LK,2
```

```
3820        DO 138 K=2,LIST,2
3830        IF(KARD(J).LT.KPEY(K-1))GOTO 149
3840        IF(KARD(J).NE.KPEY(K-1))GOTO 138
3850        IF(KARD(J+1).NE.KPEY(K))GOTO 138
3860        KC=K/2
3870        KNUM(KC)=KNUM(KC)+1
3880        GOTO 137
3890   138 CONTINUE
3900   149 KPEY(LIST-1)=KARD(J)
3910        KPEY(LIST)=KARD(J+1)
3920C.....ORDER PUTS KEYWORD LIST INTO ALPHA ORDER
3930        CALL ORDER(KPEY,KERR,LIST/2,KARD,KNUM)
3940        IF(KERR.GT.0)GOTO 139
3950        LIST=LIST+2
3960        IF(LIST.LT.200)GOTO 137
3970        KERR=6
3980        GOTO 139
3990   137 CONTINUE
4000   136 CONTINUE
4010        LIST=LIST-2
4020        PRINT 205
4030        KC=LIST/2
4040        WRITE("KEYLIST",202)LIST
4050        WRITE("KEYLIST",212)(KPEY(I),I=1,200)
4060        WRITE("KEYLIST",224)(KNUM(I),I=1,100)
4070        GOTO 141
4080   139 PRINT 206,KERR
4090   200 FORMAT(1H ,66A1,A4)
4100   201 FORMAT(67H COMMAND RECOGNISED BUT HIGH ORDER COMMAND NEE
4110&DED. YOUR COMMAND WAS,/,1X,72A1,/,10H TRY AGAIN)
4120   202 FORMAT(I5)
4130   203 FORMAT(5H CARD,I6,2X,36HHAS BEEN DELETED FROM THE INPUT FILE)

4140   204 FORMAT(6H ID IS,I5,3X,9HAUTHOR IS,2X,4A4,2X,8HAND DATE,2X,A4,

4150&/,10H THERE ARE,I4,2X,19HKEYWORDS, WHICH ARE,/,3(5(2X,2A4)/))
4160   205 FORMAT(23H KEYWORD LIST GENERATED)
4170   206 FORMAT(23H PROGRAM STOPPED. ERROR,I5)
4180   207 FORMAT(38H THE COMMAND SHOULD BE CARD OR KEYWORD)
4190   208 FORMAT(19H DELETED KEYWORD IS,2X,2A4)
4200   209 FORMAT(5H CARD,I5,2X,7HDELETED)
4210   210 FORMAT(31H INSERTED CARD READS AS FOLLOWS)
4220   211 FORMAT(20H INSERTED KEYWORD IS,2X,2A4)
4230   212 FORMAT(7(2X,2A4))
4240   213 FORMAT(5H CARD,I5,2X,12HLISTED BELOW)
4250   214 FORMAT(26H HAS BEEN REPLACED BY CARD,I5)
4260   215 FORMAT(17H REPLACED ITEM IS,2X,4A4)
4270   216 FORMAT(18H REPLACING ITEM IS,2X,4A4)
4280   217 FORMAT(8H KEYWORD,2X,2A4,2X,11HREPLACED BY,2X,2A4)
```

```
  218 FORMAT(31H POINTERS AT BEGINNING OF FILES)
  219 FORMAT(23H DATA SAVED INTO INFILE)
  220 FORMAT(1H ,2A4,2X,22HREPLACED THROUGHOUT BY,2X,2A4)
  221 FORMAT(1H ,I5,2X,12HCHANGES MADE)
  222 FORMAT(4A4)
  223 FORMAT(23H ENTER CHARACTER STRING)
  224 FORMAT(7(2X,I8))
  225 FORMAT(34H KEYWORD OCCURS ON FOLLOWING CARDS/)
  226 FORMAT(27H KEYWORD AND FREQUENCY LIST/)
  227 FORMAT(1H ,10I6)
  140 STOP
      END
      SUBROUTINE ORDER(KEY,KERR,N,KSAME,KNUM)
      DIMENSION KEY(2,100),KSAME(2,15),KNUM(100),NUM(15)
C......ORDER PUTS THE KEYWORD LIST INTO ORDER FROM A TO Z
C......SORT ON FIRST HALF OF KEYWORD
      CALL SORT(KEY,N,1,2,KNUM)
      M=1
      NN=N-1
  103 L=M
      DO 100 I=L,NN
      K=1
      IA=I+1
      DO 101 J=IA,N
      IF(K.EQ.1)GOTO 105
      IF(KEY(1,J).NE.KEY(1,I))GOTO 106
  105 IF(KEY(1,J).NE.KEY(1,I))GOTO 100
      K=K+1
      IF(K.GT.15)GOTO 102
      NUM(K)=KNUM(J)
  101 KSAME(2,K)=KEY(2,J)
  106 KSAME(2,1)=KEY(2,I)
C......SORT ON SECOND HALF OF KEYWORD
      NUM(1)=KNUM(I)
      CALL SORT(KSAME,K,2,1,NUM)
      M=I+K-1
      K=0
      DO 104 J=I,M
      K=K+1
      KNUM(J)=NUM(K)
  104 KEY(2,J)=KSAME(2,K)
      IF(M.EQ.N)RETURN
      GOTO 103
  100 CONTINUE
      IF(K.LT.15)RETURN
  102 PRINT 200,K
      KERR=4
```

```
4760   200 FORMAT(15H K TOO LARGE AT,I3)
4770       RETURN
4780       END
4790       SUBROUTINE READIT(KA,KB,KEY,NUM,KERR)
4800       DIMENSION KEY(35)
4810       KERR=0
4820       GOTO(101,102),KA
4830C......READ AUTHOR CARD
4840   101 READ("INPUT",200)(KEY(I),I=1,17)
4850       NUM=KEY(1)
4860       RETURN
4870C......READ CONTINUATION CARD OF SAME AUTHOR
4880   102 READ("INPUT",201)K,(KEY(I),I=7,17)
4890       IF(NUM.EQ.K)RETURN
4900       PRINT 202,K,NUM
4910       KERR=1
4920   200 FORMAT(3X,I5,5A4,I2,10A4)
4930   201 FORMAT(3X,I5,20X,I2,10A4)
4940   202 FORMAT(18H CONTINUATION CARD,2X,I5,32H  HAS DIFFERENT ID FROM O
4950&INAL,2X,I5)
4960       RETURN
4970       END
4980       SUBROUTINE COM(KA,KOM,KERR,KC)
4990       COMMON NUMS(72),KYMB(6),IN
5000       DIMENSION KOM(15,4)
5010C......COMM IS DESIGNED TO READ IN IN FREE FORMAT ALPHA COMMANDS
5020   106 KERR=0
5030       GOTO(101,102),KC
5040   101 PRINT 202
5050       KB=-IREAD(0)
5060       IF(KB.LT.0)GOTO 105
5070       DO 103 I=1,15
5080       KD=KB
5090       IF(NUMS(KB).NE.KOM(I,1))GOTO 103
5100       DO 104 J=2,4
5110       KD=KD+1
5120   104 IF(NUMS(KD).NE.KOM(I,J))GOTO 103
5130       KA=I
5140       DO 100 J=KD,72
5150       IN=J
5160   100 IF(NUMS(J).EQ.KYMB(4))RETURN
5170   103 CONTINUE
5180       PRINT 200,(NUMS(I),I=1,72)
5190       KERR=2
5200       CALL NEWCRD
5210       GOTO 106
5220   102 PRINT 203
```

```
5230       KA=IREAD(0)
5240       IF(KA.GE.0)RETURN
5250   105 KERR=3
5260       PRINT 201
5270   200 FORMAT(16H ILLEGAL COMMAND,/, 1H 72A1)
5280   201 FORMAT(27H WRONG TYPE OF DATA ON CARD)
5290   202 FORMAT(14H ENTER COMMAND)
5300   203 FORMAT(11H ENTER DATA)
5310       RETURN
5320       END
5330       SUBROUTINE RITE(OUT,KARD,KL,KQ)
5340       DIMENSION KARD(60)
5350       FILENAME OUT
5360       WRITE(OUT,200)(KARD(I),I=1,17)
5370       IF(KQ.LT.6)RETURN
5380       WRITE(OUT,201)(KARD(I),I=18,33)
5390   200 FORMAT(I4,2X,5A4,I2,10A4)
5400   201 FORMAT(4X,16A4)
5410       RETURN
5420       END
5430       SUBROUTINE FIND(IN,OUT,KARD,IP,N,NUMB,KQ,KERR)
5440       DIMENSION KARD(60)
5450       FILENAME IN,OUT
5460C......FIND LOOKS FOR A PARTICULAR CARD NUMBER, OR
5470C......EVERY CARD, OR JUST COPIES WHAT IT FINDS
5480   204 FORMAT(2I5)
5490       KERR=0
5500       IF(IP.GT.N)GOTO 103
5510       DO 100 I=IP,N
5520       L=17
5530       READ(IN,200)(KARD(J),J=1,17)
5540       KQ=KARD(7)
5550       IF(KQ.LT.6)GOTO 101
5560       L=7+KQ*2
5570       READ(IN,201)(KARD(J),J=18,L)
5580   101 IF((NUMB.EQ.0).OR.(KARD(1).EQ.NUMB))GOTO 102
5590   100 CALL RITE(OUT,KARD,L,KQ)
5600       IP=N+1
5610       IF(NUMB.EQ.(-1))RETURN
5620       PRINT 202,NUMB
5630       KERR=5
5640       RETURN
5650   102 IP=I+1
5660       RETURN
5670   103 KERR=7
5680   200 FORMAT(I4,2X,5A4,I2,10A4)
5690   201 FORMAT(4X,16A4)
```

```
  202 FORMAT(14H NO CARD OF ID,I5,2X,13HFOUND IN FILE)
      RETURN
      END
      SUBROUTINE COPY(KARD,KA,KB,KPEY,KC)
      DIMENSION KARD(60),KPEY(200)
      DO 100 I=KA,KB
      KC=KC+1
  100 KPEY(KC)=KARD(I)
      RETURN
      END
      SUBROUTINE START(KX,IN,OUT,IP)
      FILENAME IN,OUT
      IP=1
      BEGIN FILE "INFILE"
      BEGIN FILE "OUTFILE"
      READ(IN,200)K
      WRITE(OUT,200)KX
  200 FORMAT(I5)
      RETURN
      END
      SUBROUTINE SORT(KA,JJ,JA,JB,NUM)
      DIMENSION KA(2,100),IU(10),IL(10),NUM(100)
C......THIS ROUTINE SORT IS A MODIFICATION OF CACM QUICKERSORT2
      M=1
      I=1
      J=JJ
  555 IF(I.GE.J)GOTO 700
  100 K=I
      IJ=(J+I)/2
      KT=KA(JA,IJ)
      KTA=KA(JB,IJ)
      KTB=NUM(IJ)
      IF(KA(JA,I).LE.KT)GOTO 200
      KA(JA,IJ)=KA(JA,I)
      KA(JB,IJ)=KA(JB,I)
      NUM(IJ)=NUM(I)
      KA(JA,I)=KT
      KA(JB,I)=KTA
      NUM(I)=KTB
      KT=KA(JA,IJ)
      KTA=KA(JB,IJ)
      KTB=NUM(IJ)
  200 L=J
      IF(KA(JA,J).GE.KT)GOTO 400
      KA(JA,IJ)=KA(JA,J)
      KA(JB,IJ)=KA(JB,J)
      NUM(IJ)=NUM(J)
```

```
6170       KA(JA,J)=KT
6180       KA(JB,J)=KTA
6190       NUM(J)=KTB
6200       KT=KA(JA,IJ)
6210       KTA=KA(JB,IJ)
6220       KTB=NUM(IJ)
6230       IF(KA(JA,I).LE.KT)GOTO 400
6240       KA(JA,IJ)=KA(JA,I)
6250       KA(JB,IJ)=KA(JB,I)
6260       NUM(IJ)=NUM(I)
6270       KA(JA,I)=KT
6280       KA(JB,I)=KTA
6290       NUM(I)=KTB
6300       KT=KA(JA,IJ)
6310       KTA=KA(JB,IJ)
6320       KTB=NUM(IJ)
6330       GOTO 400
6340   300 KA(JA,L)=KA(JA,K)
6350       KA(JB,L)=KA(JB,K)
6360       NUM(L)=NUM(K)
6370       KA(JA,K)=KS
6380       KA(JB,K)=KSA
6390       NUM(K)=KSB
6400   400 L=L-1
6410       IF(KA(JA,L).GT.KT)GOTO 400
6420       KS=KA(JA,L)
6430       KSA=KA(JB,L)
6440       KSB=NUM(L)
6450   500 K=K+1
6460       IF(KA(JA,K).LT.KT)GOTO 500
6470       IF(K.LE.L)GOTO 300
6480       IF((L-I).LE.(J-K))GOTO 600
6490       IL(M)=I
6500       IU(M)=L
6510       I=K
6520       M=M+1
6530       GOTO 800
6540   600 IL(M)=K
6550       IU(M)=J
6560       J=L
6570       M=M+1
6580       GOTO 800
6590   700 M=M-1
6600       IF(M.EQ.0)GOTO 120
6610       I=IL(M)
6620       J=IU(M)
6630   800 IF((J-I).GE.1)GOTO 100
```

```
6640        IF(I.EQ.1)GOTO 555
6650        I=I-1
6660   900 I=I+1
6670        IF(I.EQ.J)GOTO 700
6680        KT=KA(JA,I+1)
6690        KTA=KA(JB,I+1)
6700        KTB=NUM(I+1)
6710        IF(KA(JA,I).LE.KT)GOTO 900
6720        K=I
6730   110 KA(JA,K+1)=KA(JA,K)
6740        KA(JB,K+1)=KA(JB,K)
6750        NUM(K+1)=NUM(K)
6760        K=K-1
6770        IF(KT.LT.KA(JA,K))GOTO 110
6780        KA(JA,K+1)=KT
6790        KA(JB,K+1)=KTA
6800        NUM(K+1)=KTB
6810        GOTO 900
6820   120 RETURN
6830        END
6840        SUBROUTINE NEWCRD
6850        COMMON NUMS(72),KYMB(6),IN
6860C......SIMULATES A NEW CARD INPUT COMMAND FOR FREE FORMAT ROUTINE
6870        IN=72
6880        RETURN
6890        END
6900        FUNCTION IREAD(KZ)
6910        COMMON NUMS(72),KYMB(6),IN
6920C......IREAD READS IN A CARD IMAGE, AND ON COMMAND READS EITHER
6930C......AN ALPHA COMMAND, OR A NU BER. COMMANDS AND NUMBERS CAN BE
6940C       MIXED, AND CAN BE TYPED IN ANY OF THE 72 COLUMNS OF THE
6950C       CARD IMAGE
6960        IF(IN.GE.72)GOTO115
6970   117 NUM=0
6980        SIG=1.0
6990        LB=0
7000        IF((IN.NE.1).OR.(KZ.NE.1))GOTO116
7010        PRINT 202,(NUMS(I),I=1,72)
7020C......MAIN SEARCH LOOP FOR FREE FORMAT ROUTINE
7030   116 DO 100 I=IN,72
7040        L=NUMS(I)
7050        IF(L.EQ.KYMB(4))GOTO100
7060        DO 101 J=1,3
7070        IF(L.EQ.KYMB(J))GOTO102
7080   101 CONTINUE
7090        K=I
7100C......LOOP104 READS IN REQUIRED NUMBER
```

KARP/EDITOR 14

```
7110   103 DO 104 J=K,72
7120       JA=J
7130       IF(NUMS(J).EQ.KYMB(4))GOTO106
7140       LA=(NUMS(J)-KYMB(6))/KYMB(5)
7150       IF((LA.GT.9).OR.(LA.LT.0))GOTO105
7160       NUM=(NUM+LA)*10
7170       GOTO 104
7180   105 IF((NUMS(J).NE.KYMB(3)).OR.(LB.NE.0))GOTO 107
7190       LB=J
7200   104 CONTINUE
7210       GOTO 113
7220   106 JA=JA-1
7230   113 LB=LB+1
7240       IF(LB.EQ.1)GOTO108
7250       LB=JA-LB+2
7260C......FUNCTION RETURNS WITH THE NEW VALUE
7270   108 IREAD=IFIX(FLOAT(NUM)*(0.1**FLOAT(LB))*SIG+0.001)
7280       GOTO 109
7290   102 GOTO(110,114,112),J
7300   110 SIG=-1.0
7310       GOTO 114
7320   112 LB=I
7330   114 K=I+1
7340       GOTO 103
7350C......107 REACHED ONLY FOR TERMINATION BECAUSE OF ILLEGAL CHARACTER
                                                  S
7360   107 IREAD=-J
7370       RETURN
7380   100 CONTINUE
7390C......PRINTS OUT NEWLY ENTERED CARD IMAGE IF KZ=1
7400   115 READ 200,(NUMS(J),J=1,72)
7410       IN=1
7420       GOTO 117
7430   109 IN=JA+1
7440       RETURN
7450   200 FORMAT(72A1)
7460   202 FORMAT(1H ,72A1)
7470       END
```

KARP/UPDATE 1

```
1000       DIMENSION NEW(2,100),KEY(700),KTRAK(100),KARD(35),KUP(1310)
1010       DIMENSION NUMS(72),KYMB(6),LTRAK(100)
1020C......DATA STATEMENT SETS UP FREE FORMAT ROUTINE
1030       DATA KYMB/"-","+",".", " "," ","0"/,I/"1"/
1040       DATA NZZ/"ZZZZ"/
1050       KYMB(5)=I-KYMB(6)
1060       CALL NEWCRD(IN)
1070       IP=1
1080       KP=1
1090       BEGINFILE "OUTFILE"
1100       ENDFILE "OUTFILE"
1110       PRINT 204
1120C......READS IN 1 OR 0 TO INDICATE FIRST UPDATE
1130       LXK=IREAD(0,NUMS,KYMB,IN)
1140       IF(LXK-1)128,127,128
1150C......SET INDICATOR TO ZERO TO INDICATE NO PERMANENT FILE
1160C......HAS YET BEEN CREATED
1170   127 KX=0
1180       KARD(1)=999999
1190       GOTO 132
1200C......READ NUMBER OF CARDS IN PERMANENT FILE AND IN THE
1210C......INCOMING FILE OF NEW CARDS
1220   128 BEGINFILE "PERMFILE"
1230       READ("PERMFILE",200)KX
1240       CALL FIND("PERMFILE","OUTFILE",KARD,KP,KX,0,KQ,KERR)
1250   132 BEGINFILE "INFILE"
1260       READ("INFILE",200)IX
1270C......READ IN A CARD FROM EACH FILE TO DETERMINE WHICH HAS THE
1280C......LOWEST ID, CHECKING FOR THE LAST CARD FIRST.
1290   133 IF(IP.GT.IX)GOTO 138
1300       CALL FIND("INFILE","OUTFILE",LTRAK,IP,IX,0,IQ,KERR)
1310       GOTO 137
1320   135 IF(KP.GT.KX)GOTO 136
1330       CALL FIND("PERMFILE","OUTFILE",KARD,KP,KX,0,KQ,KERR)
1340       GOTO 137
1350C......ASSIGN IMPOSSIBLE ID TO COMPLETELY MERGED FILE TO INDICATE
1360C......THAT NO FURTHER CARDS CAN BE READ FROM IT.
1370   136 KARD(1)=999999
1380       GOTO 137
1390   138 LTRAK(1)=999999
1400C......COMPARE ID FROM BOTH FILES TO DETERMINE WHICH IS SMALLER
1410   137 IF(KARD(1)-LTRAK(1))134,139,140
1420C......WRITE OUT ONTO OUTFILE THE CARD WITH THE SMALLER ID
1430   134 CALL RITE("OUTFILE",KARD,KQ*2+7,KQ)
1440       GOTO 135
1450   140 CALL RITE("OUTFILE",LTRAK,IQ*2+7,IQ)
1460       GOTO 133
```

KARP/UPDATE 2

```
1470C......139 IS ONLY REACHED WHEN BOTH IDS ARE 999999, IN OTHER
1480C......WORDS WHEN BOTH FILES ARE EXHAUSTED.
1490  139 KX=KX+IX
1500      PRINT 201,KX
1510      BEGINFILE "OUTFILE"
1520      BEGINFILE "PERMFILE"
1530      ENDFILE "PERMFILE"
1540      IP=1
1550      WRITE("PERMFILE",200)KX
1560C......WRITES OUT MERGED FILES TO PERMFILE
1570      CALL FIND("OUTFILE","PERMFILE",KARD,IP,KX,-1,KQ,KERR)
1580      BEGINFILE "KEYLIST"
1590      READ("KEYLIST",200)KL
1600      N=KL/2
1610C......READS IN NEW SET OF KEYWORDS
1620      READ("KEYLIST",202)((NEW(I,J),I=1,2),J=1,N)
1630      DO 100 I=1,1010
1640  100 KUP(I)=0
1650      K=1
1660C......STARTS CALCULATION OF NEW CROSS REFERENCE LIST FOR
1670C......NEWLY INPU CARDS
1680      DO 101 I=1,N
1690      KUP(K)=-I
1700      KTRAK(I)=K+1
1710      LTRAK(I)=KTRAK(I)
1720  101 K=K+5
1730      KEND=K
1740      KUP(KEND)=-1000
1750      BEGINFILE "INFILE"
1760      READ("INFILE",200)I
1770      BEGINFILE "OUTFILE"
1780      ENDFILE "OUTFILE"
1790      IP=1
1800C......LOOP 102 ASSEMBLES REFERENCE LIST FROM NEW INPUT CARDS
1810      DO 102 I=1,IX
1820      CALL FIND("INFILE","OUTFILE",KARD,IP,IX,0,KQ,KERR)
1830      KA=6+KQ*2
1840      DO 103 J=8,KA,2
1850      KB=KARD(J)
1860      DO 104 K=1,N
1870      IF(KB.NE.NEW(1,K))GOTO 104
1880      IF(KARD(J+1).NE.NEW(2,K))GOTO 104
1890      KB=LTRAK(K)
1900C......CREATION OF REFERENCE LIST
1910      DO 105 L=KB,KEND
1920      IF(KUP(L))106,107,105
1930  107 KUP(L)=KARD(1)
```

```
1940      LTRAK(K)=L
1950      GOTO 103
1960C......STOPS IF SIZE (1295) OF REFERENCE LIST EXCEEDED
1970  108 PRINT 203
1980      STOP
1990  106 IF(KEND.GT.1295)GOTO 108
2000C......EXPANDS MARKERS FOR POSITIONS OF KEYWORDS IN LIST
2010      DO 109 M=L,KEND
2020      LA=KEND-M+L
2030  109 KUP(LA+7)=KUP(LA)
2040      LA=L+6
2050      DO 129 M=L,LA
2060  129 KUP(M)=0
2070      KEND=KEND+7
2080      DO 117 M=K,N
2090      IF(LTRAK(M).GE.L)LTRAK(M)=LTRAK(M)+7
2100  117 IF(KTRAK(M).GE.L)KTRAK(M)=KTRAK(M)+7
2110      GOTO 107
2120  105 CONTINUE
2130  104 CONTINUE
2140  103 CONTINUE
2150  102 CONTINUE
2160C......BEGINNING OF MERGE SEQUENCE OF OLD AND NEW CROSS REFERENCE
2170C......INDEXES
2180      KA=1
2190      KP=0
2200      BEGINFILE "OUTFILE"
2210      ENDFILE "OUTFILE"
2220      BEGINFILE "KEYWORD"
2230      LA=KTRAK(1)
2240      LB=LTRAK(1)
2250      LX=0
2260      NEW(1,N+1)=NZZ
2270      IF(LXK.EQ.1)GOTO 110
2280      READ("KEYWORD",200)NKEY
2290  126 IA=0
2300  111 IA=IA+1
2310      IF(IA.GT.NKEY)GOTO 119
2320      READ("KEYWORD",206)NCAR,KEYN,KB,KC
2330      NA=NCAR*10
2340      READ("KEYWORD",205)(KEY(I),I=1,NA)
2350      IF(LX.EQ.1)GOTO 118
2360      GOTO 123
2370  110 NKEY=0
2380      GOTO 126
2390  118 KA=KA+1
2400      LA=KTRAK(KA)
```

```
2410        LB=LTRAK(KA)
2420   123 LX=0
2430C......TEST FOR ENTRY OF OLD OR NEW KEYWORD INDEX
2440        IF(KB-NEW(1,KA))112,113,114
2450   114 LC=LB-LA+1
2460        LD=(LC-1)/10+1
2470        WRITE("OUTFILE",206)LD,LC,NEW(1,KA),NEW(2,KA)
2480        LD=LA-1+LD*10
2490        WRITE("OUTFILE",205)(KUP(I),I=LA,LD)
2500        KP=KP+1
2510        GOTO 118
2520   112 WRITE("OUTFILE",206)NCAR,KEYN,KB,KC
2530        DO 130 I=1,NCAR
2540        J=(I-1)*10+1
2550        K=J+9
2560   130 WRITE("OUTFILE",205)(KEY(L),L=J,K)
2570        GOTO 111
2580   113 IF(KB.EQ.NZZ)GOTO 124
2590        IF(KC-NEW(2,KA))112,122,114
2600C......MERGE OLD AND NEW LISTS FOR GIVEN KEYWORD
2610   122 DO 120 I=LA,LB
2620        KEYN=KEYN+1
2630   120 KEY(KEYN)=KUP(I)
2635        CALL SORT(KEY,KEYN)
2640        NCAR=(KEYN-1)/10+1
2650        NA=NCAR*10
2660        LX=1
2670        GOTO 112
2680   119 KB=NZZ
2690        IF(LX-1)123,118,123
2700   124 NKEY=NKEY+KP
2710C......FINAL WRITING OF NEW AND OLD MERGED INDEXES TO KEYWORD
2720        PRINT 207,NKEY
2730        BEGIN FILE "KEYWORD"
2740        ENDFILE "KEYWORD"
2750        BEGIN FILE "OUTFILE"
2760        WRITE("KEYWORD",200)NKEY
2770        PRINT 210
2780        J=IREAD(0,NUMS,KYMB,IN)
2790        DO 125 IA=1,NKEY
2800        READ("OUTFILE",206)NCAR,KEYN,KB,KC
2810        WRITE("KEYWORD",206)NCAR,KEYN,KB,KC
2820        IF(J.EQ.0)GOTO 131
2830        PRINT 209,KB,KC,KEYN
2840   131 NA=NCAR*10
2850        READ("OUTFILE",205)(KEY(I),I=1,NA)
2860   125 WRITE("KEYWORD",205)(KEY(I),I=1,NA)
2870        PRINT 208
```

```
200 FORMAT(I5)
201 FORMAT(31H TOTAL NUMBER OF RECORDS IS NOW,I5)
202 FORMAT(7(2X,2A4))
203 FORMAT(45H ARRAY TOO SMALL TO HOLD CROSS REFERENCE LIST)
204 FORMAT(35H TYPE 1 IF THIS IS THE FIRST UPDATE)
205 FORMAT(1X,10I5)
206 FORMAT(1X,2I5,32X,2A4)
207 FORMAT(16H UPDATE COMPLETE,/,14H THERE ARE NOW,I5,
&2X,8HKEYWORDS)
208 FORMAT(30H TRANSFER TO KEYWORD COMPLETED)
209 FORMAT(1H ,2A4,I7)
210 FORMAT(40H TYPE 0 TO SUPRESS TOTAL KEYWORD LISTING)
    STOP
    END
    SUBROUTINE RITE(OUT,KARD,KL,KQ)
    DIMENSION KARD(35)
    FILENAME OUT
    WRITE(OUT,200)(KARD(I),I=1,17)
    IF(KQ.LT.6)RETURN
    WRITE(OUT,201)(KARD(I),I=18,33)
200 FORMAT(I4,2X,5A4,I2,10A4)
201 FORMAT(4X,16A4)
    RETURN
    END
    SUBROUTINE FIND(IN,OUT,KARD,IP,N,NUMB,KQ,KERR)
    DIMENSION KARD(35)
    FILENAME IN,OUT
204 FORMAT(2I5)
    KERR=0
    IF(IP.GT.N)GOTO 103
    DO 100 I=IP,N
    L=17
    READ(IN,200)(KARD(J),J=1,17)
    KQ=KARD(7)
    IF(KQ.LT.6)GOTO 101
    L=7+KQ*2
    READ(IN,201)(KARD(J),J=18,L)
101 IF((NUMB.EQ.0).OR.(KARD(1).EQ.NUMB))GOTO 102
100 CALL RITE(OUT,KARD,L,KQ)
    IP=N+1
    IF(NUMB.EQ.(-1))RETURN
    PRINT 202,NUMB
    KERR=5
    RETURN
102 IP=I+1
    RETURN
103 KERR=7
```

```
  200 FORMAT(I4,2X,5A4,I2,10A4)
  201 FORMAT(4X,16A4)
  202 FORMAT(14H NO CARD OF ID,I5,2X,13HFOUND IN FILE)
      RETURN
      END
      SUBROUTINE NEWCRD(IN)
      IN=72
      RETURN
      END
      FUNCTION IREAD(KZ,NUMS,KYMB,IN)
      DIMENSION NUMS(72),KYMB(6)
      IF(IN.GE.72)GOTO115
  117 NUM=0
      SIG=1.0
      LB=0
      IF((IN.NE.1).OR.(KZ.NE.1))GOTO116
      PRINT 202,(NUMS(I),I=1,72)
C......MAIN SEARCH LOOP FOR FREE FORMAT ROUTINE
  116 DO 100 I=IN,72
      L=NUMS(I)
      IF(L.EQ.KYMB(4))GOTO100
      DO 101 J=1,3
      IF(L.EQ.KYMB(J))GOTO102
  101 CONTINUE
      K=I
C......LOOP104 READS IN REQUIRED NUMBER
  103 DO 104 J=K,72
      JA=J
      IF(NUMS(J).EQ.KYMB(4))GOTO106
      LA=(NUMS(J)-KYMB(6))/KYMB(5)
      IF((LA.GT.9).OR.(LA.LT.0))GOTO105
      NUM=(NUM+LA)*10
      GOTO 104
  105 IF((NUMS(J).NE.KYMB(3)).OR.(LB.NE.0))GOTO107
      LB=J
  104 CONTINUE
      GOTO 113
  106 JA=JA-1
  113 LB=LB+1
      IF(LB.EQ.1)GOTO108
      LB=JA-LB+2
C......FUNCTION RETURNS WITH THE NEW VALUE
  108 IREAD=IFIX(FLOAT(NUM)*(0.1**FLOAT(LB))*SIG+0.001)
      GOTO 109
  102 GOTO(110,114,112),J
  110 SIG=-1.0
      GOTO 114
```

```
3820  112 LB=I
3830  114 K=I+1
3840      GOTO 103
3850C......107 REACHED ONLY FOR TERMINATION BECAUSE OF ILLEGAL CHARACTER
     ......

3860  107 PRINT 201
3870      PRINT 202,(NUMS(J),J=1,72)
3880      STOP
3890  100 CONTINUE
3900C......PRINTS OUT NEWLY ENTERED CARD IMAGE IF KZ=1
3910  115 READ 200,(NUMS(J),J=1,72)
3920      IN=1
3930      GOTO 117
3940  109 IN=JA+1
3950      RETURN
3960  200 FORMAT(72A1)
3970  201 FORMAT(26H ILLEGAL CHARACTER ON CARD)
3980  202 FORMAT(1H ,72A1)
3990      END
4000      SUBROUTINE SORT(KA,JJ)
4010      DIMENSION KA(700),IU(10),IL(10)
4020      M=1
4030      I=1
4040      J=JJ
4050  555 IF(I.GE.J)GOTO 700
4060  100 K=I
4070      IJ=(J+I)/2
4080      KT=KA(IJ)
4090      IF(KA(I).LE.KT)GOTO 200
4100      KA(IJ)=KA(I)
4110      KA(I)=KT
4120      KT=KA(IJ)
4130  200 L=J
4140      IF(KA(J).GE.KT)GOTO 400
4150      KA(IJ)=KA(J)
4160      KA(J)=KT
4170      KT=KA(IJ)
4180      IF(KA(I).LE.KT)GOTO 400
4190      KA(IJ)=KA(I)
4200      KA(I)=KT
4210      KT=KA(IJ)
4220      GOTO 400
4230  300 KA(L)=KA(K)
4240      KA(K)=KS
4250  400 L=L-1
4260      IF(KA(L).GT.KT)GOTO 400
4270      KS=KA(L)
4280  500 K=K+1
```

```
    IF(KA(K).LT.KT)GOTO 500
    IF(K.LE.L)GOTO 300
    IF((L-I).LE.(J-K))GOTO 600
    IL(M)=I
    IU(M)=L
    I=K
    M=M+1
    GOTO 800
600 IL(M)=K
    IU(M)=J
    J=L
    M=M+1
    GOTO 800
700 M=M-1
    IF(M.EQ.0)GOTO 120
    I=IL(M)
    J=IU(M)
800 IF((J-I).GE.1)GOTO 100
    IF(I.EQ.1)GOTO 555
    I=I-1
900 I=I+1
    IF(I.EQ.J)GOTO 700
    KT=KA(I+1)
    IF(KA(I).LE.KT)GOTO 900
    K=I
110 KA(K+1)=KA(K)
    K=K-1
    IF(KT.LT.KA(K))GOTO 110
    KA(K+1)=KT
    GOTO 900
120 RETURN
    END
```

```
                  KARP/SEARCH  1
1000      DIMENSION KSUM(200),KAND(2,50),KPAND(6),NOT(2,10)
1010      DIMENSION KINAL(200),KSTORE(200,6),KARD(600)
1020      DIMENSION NUMS(500),KYMB(6)
1030C......DATA STATEMENT SETS UP FREE FORMAT ROUTINE, AND SETS
1040C......KEX TO ALPHA 9999 - THE CHARACTER STRING NEEDED TO
1050C......TERMINATE THE INPUT KEYWORD SEQUENCE
1060      DATA KYMB/"-","+","."," "," ","0"/,I/"1"/,KEX/"9999"/
1070      KYMB(5)=I-KYMB(6)
1080      CALL NEWCRD(IN)
1090      KPAND(1)=1
1100C......KLB IS SET TO THE SIZE OF ARRAY KSTORE. IT IS USED TO
1110C......TEST WHETHER THE ARRAY IS FULL AND BACKUP
1120C......STORAGE IS NEEDED FROM NUMS.
1130      KLB=200
1140  100 PRINT 200
1150C......READS IN THE NUMBER OF ANDS IN FREE FORMAT
1160      CALL IREAD(0,NUMS,KYMB,IN,NAND)
1170      PRINT 201
1180      DO 101 I=1,NAND
1190C......READS IN NUMBER OF ORS FOR EACH AND
1200      CALL IREAD(0,NUMS,KYMB,IN,KOR)
1210  101 KPAND(I+1)=KPAND(I)+KOR
1220      PRINT 203
1230C......READS IN NUMBER OF NOTS - MUST BE AT LEAST ONE
1240      CALL IREAD(0,NUMS,KYMB,IN,KGNOT)
1250      KSTORE(1,6)=0
1260      KINAL(1)=0
1270      KWAT=0
1280      KFIN=1
1290C......LOOP 103 READS THE ORS FOR EACH AND IN TURN
1300      DO 103 I=1,NAND
1310      PRINT 204,I
1320      KA=KPAND(I)
1330      KB=KPAND(I+1)-1
1340      DO 103 J=KA,KB
1350      K=J-KA+1
1360      PRINT 205,K
1370C......THE ORS ARE STORED IN KAND, WITH EACH AND FOLLOWING
1380C......THE ONE BEFORE IN A LINEAR ARRAY
1390      READ 206,(KAND(K,J),K=1,2)
1400C......KEX IS TESTING TO SEE WHETHER THE INPUT KEYWORD IS 9999
1410C......IF IT IS THE PROGRAM WILL GOTO TO 100 TO START THE
1420C......KEYWORD SEQUENCE AFRESH
1430  103 IF(KAND(1,J).EQ.KEX)GOTO 100
1440      PRINT 208,KGNOT
1450C......LOOP 106 READS IN THE NOT KEYWORDS, ONE TO A CARD IMAGE
1460      DO 106 I=1,KGNOT
```

```
1470        READ 206,(NOT(J,I),J=1,2)
1480   106 IF(NOT(1,I).EQ.KEX)GOTO 100
1490        PRINT 209
1500        DO 107 I=1,NAND
1510        KA=KPAND(I)
1520        KB=KPAND(I+1)-1
1530C......LOOP 107 PRINT OUT THE KEYWORD STRUCTURE
1540   107 PRINT 210,I,((KAND(J,K),J=1,2),K=KA,KB)
1550        PRINT 212,((NOT(I,J),I=1,2),J=1,KGNOT)
1560        PRINT 213
1570        NEW=0
1580C......A NUMBER, K, IS READ TO SEE WHETHER STRUCTURE IS ACCURATE
1590        CALL IREAD(0,NUMS,KYMB,IN,K)
1600        KNUM=1
1610        KST=1
1620        NUMS(1)=0
1630C......IF K IS 0, CONTROL MOVES TO THE NEXT PART OF THE PROGRAM
1640C......ELSE TO 100 TO START A NEW KEYWORD SEQUENCE
1650        IF(K)100,108,100
1660   108 BEGINFILE "KEYWORD"
1670        READ("KEYWORD",214)NKEY
1680C......LOOP 110 STARTS THE SEARCH PROCEDURE, READING FROM
1690C......THE FILE KEYWORD THE CROSS REFERENCE INDEX, A KEYWORD LIST
1700C......AT A TIME. THE NUMBER OF KEYWORDS IS NKEY
1710        DO 110 IA=1,NKEY
1720        READ("KEYWORD",215)NA,NB,KC,KD
1730        K=1
1740C......LOOP 111 READS IN A PARTICULAR KEYWORDS CROSS REFERENCE
1750C......LIST INTO KARD. KARD CONTAINS ALL ITEMS THAT USE THAT
1760C......KEYWORD, STORED IN THE SAME ORDER AS PERMFILE
1770        DO 111 I=1,NA
1780        L=K+9
1790        READ("KEYWORD",216)(KARD(J),J=K,L)
1800   111 K=K+10
1810C......SEARCH THROUGH THE ANDS IS PERFORMED BY LOOP 115
1820        DO 115 I=1,NAND
1830        KA=KPAND(I)
1840        KB=KPAND(I+1)-1
1850C......LOOP 116 SEARCHES THROUGH THE ORS FOR THE PARTICULAR
1860C......AND OF LOOP 115
1870        DO 116 J=KA,KB
1880C......THE IFS TEST FOR AN OR BEING THE KEYWORD INDICATED BY LOOP
1890C......110. KC AND KD ARE THIS KEYWORD (STORED IN 2A4).
1900        IF(KC.NE.KAND(1,J))GOTO 116
1910        IF(KD.NE.KAND(2,J))GOTO 116
1920C......LOOP 117 SEARCHES THROUGH THE LISTING FOR KEYWORD KC,KD
1930C......TO SEE IF THAT ITEM HAS ALREADY BEEN STORED AS A POSSIBLE
```

```
                    KARP/SEARCH   3
1940C......HIT. THIS LOOP ONLY REACHED IF KC,KD WERE ONE OF THE ORS.
1950          DO 117 K=1,NB
1960          KX=KARD(K)
1970C......LOOP 118 CHECKS TO SEE IF ITEM ALREADY PRESENT AS A HIT
1980          DO 118 L=1,KFIN
1990     118  IF(KX.EQ.KINAL(L))GOTO 117
2000C......LOOP 119 CHECKS FOR PRESENCE IN THE PROBABLE HIT TABLE,
2010C......KSTORE, AND INSERTS IT IF NOT THERE.
2020          DO 119 L=1,KST
2030          IF(KX.NE.KSTORE(L,6))GOTO 119
2040          IF(KSTORE(L,I).EQ.1)GOTO 117
2050C......KSTORE UPDATED TO INDICATE THE PRESENCE OF A HIT FOR
2060C......ONE OF THE ANDS
2070          KSTORE(L,I)=1
2080          KSUM(L)=KSUM(L)+1
2090C......IF KSUM(L)= THE NUMBER OF ANDS A DEFINITE HIT IS OBTAINED
2100          IF(KSUM(L).NE.NAND)GOTO 117
2110C......LOOP 136 FINDS A SPACE IN THE DEFINITE HIT ARRAY, KINAL,
2120          DO 136 M=1,KFIN
2130          IF(KINAL(M).EQ.0)GOTO 132
2140     136  CONTINUE
2150          M=KFIN
2160C......THE DEFINITE HIT IN KSTORE(L,6) IS PUT INTO KINAL
2170     132  KINAL(M)=KSTORE(L,6)
2180          KSTORE(L,6)=0
2190          IF(M.NE.KFIN)GOTO 117
2200          KFIN=KFIN+1
2210          KINAL(KFIN)=0
2220          GOTO 117
2230     119  CONTINUE
2240C......NEW EQUALS 1 IF THIS IS THE SECOND OR MORE TIME THROUGH
2250C......THE FILE KEYWORD AFTER THE PRIMARY STORAGE ARRAY KSTORE
2260C......HAS BEEN FILLED
2270          IF(NEW.EQ.1)GOTO 117
2280C......LOOP 121 SEARCHES FOR A SPACE IN KSTORE TO PUT A POSSIBLE
2290C......HIT
2300          DO 121 L=1,KST
2310          IF(KSTORE(L,6).EQ.0)GOTO 122
2320     121  CONTINUE
2330          IF(KWAT.EQ.1)GOTO 137
2340     122  DO 120 M=1,5
2350     120  KSTORE(L,M)=0
2360C......THE PROBABLE HIT, KX, IS PUT IN KSTORE
2370          KSUM(L)=1
2380          KSTORE(L,6)=KX
2390          KSTORE(L,I)=1
2400          IF(L.NE.KST)GOTO 117
```

```
2410C••••••TRANSFER TO 134 MEANS THAT KSTORE IS FULL AND NUMS
2420C••••••MUST NOW BE USED FOR STORAGE OF POSSIBLE HITS
2430       IF(KST•GT•(KLB-1))GOTO 134
2440       KST=KST+1
2450       KSTORE(KST,6)=0
2460       GOTO 117
2470C••••••FROM HERE TO 134 IS CONCERNED WITH USING NUMS AS
2480C••••••BACKUP STORE TO KSTORE (WHEN FULL)• LOOP 142 LOOKS FOR
2490C••••••A SPACE IN NUMS
2500  137 DO 142 L=1,KNUM
2510  142 IF(NUMS(L)•EQ•KX)GOTO 117
2520      DO 139 L=1,KNUM
2530  139 IF(NUMS(L)•EQ•0)GOTO 140
2540C••••••KX IS PLACED IN NUMS
2550  140 NUMS(L)=KX
2560      KNUM=KNUM+1
2570      NUMS(KNUM)=0
2580      IF(KNUM•LE•500)GOTO 117
2590C••••••138 IS REACHED ONLY IF NUMS CAPACITY AS BACKUP STORE IS
2600C••••••EXCEEDED, REMEDY IS TI INCREASE SIZE OF NUMS AND KSTORE
2610  138 PRINT 222
2620      GOTO 130
2630  134 KWAT=1
2640  117 CONTINUE
2650  116 CONTINUE
2660C••••••LOOP 123 ELIMINATES FROM KSTORE POSSIBLE HITS THAT HAVE
2670C••••••NOW BECOME DEFINITE FAILURES AS KC,KD IS FURTHER ON
2680C••••••ALPHABETICALLY THAN THE LARGEST KEYWORD OF THE PRESENT AND
2690      IF(KAND(1,KB)•GT•KC)GOTO 115
2700      DO 123 J=1,KST
2710      IF((KSTORE(J,6)•EQ•0)•OR•(KSTORE(J,I)•NE•0))GOTO 123
2720      KSTORE(J,6)=0
2730  123 CONTINUE
2740  115 CONTINUE
2750      IF(NEW•EQ•1)GOTO 110
2760C••••••KNOT DELETES FROM KSTORE,KINAL,AND NUMS THOSE POSSIBLE HITS
2770C••••••THAT ALSO HAVE A NOT IN THEIR LISTING
2780      CALL KNOT(KGNOT,NOT,KC,KD,KARD,KST,KSTORE,NB,5*KLB)
2790      CALL KNOT(KGNOT,NOT,KC,KD,KARD,KNUM,NUMS,NB,0)
2800      CALL KNOT(KGNOT,NOT,KC,KD,KARD,KFIN,KINAL,NB,0)
2810  110 CONTINUE
2820      IF(NEW•EQ•1)GOTO 143
2830C••••••AFTER THE FIRST SEARCH, THE FILE KEYWORD IS SEARCHED ONCE
2840C••••••AGAIN TO ERADICATE ANY ITEMS THAT CONTAIN A NOT
2850      BEGINFILE "KEYWORD"
2860      READ("KEYWORD",214)I
2870      DO 104 IA=1,NKEY
```

```
                        KARP/SEARCH  5
2880       READ("KEYWORD",215)NA,NB,KC,KD
2890       K=1
2900       DO 112 I=1,NA
2910       L=K+9
2920       READ("KEYWORD",216)(KARD(J),J=K,L)
2930   112 K=K+10
2940       CALL KNOT(KGNOT,NOT,KC,KD,KARD,KFIN,KINAL,NB,0)
2950   104 CALL KNOT(KGNOT,NOT,KC,KD,KARD,KNUM,NUMS,NB,0)
2960C......ZERO REMOVES THE 0 CREATED IN NUMS, AND KINAL BY NOT
2970   143 CALL ZERO(NUMS,KNUM)
2980C......THE IF TESTS WHETHER NUMS HAS BEEN USED FOR BACKUP STORE
2990       IF((KNUM.EQ.1).AND.(NUMS(1).EQ.0))GOTO 135
3000       PRINT 211
3010       NEW=1
3020C......HERE TO 135 SETS UP ANOTHER LOOP ROUND THE SEARCH PROCEDURE
3030C......TO TEST THE POSSIBLE HITS TEMPORARILY STORED IN NUMS
3040       DO 126 I=1,6
3050       DO 126 J=1,KLB
3060       KSUM(J)=0
3070   126 KSTORE(J,I)=0
3080C......UPTO KLB POSSIBLE HITS ARE TRANSFERRED FROM NUMS TO KSTORE
3090       KLA=KNUM
3100       IF(KLA.LE.KLB)GOTO 127
3110       KLA=KLB
3120   127 DO 141 I=1,KLA
3130       KSTORE(I,6)=NUMS(I)
3140   141 NUMS(I)=0
3150       KST=KLA
3160C......ANY ZEROS ARE REMOVED FROM THE LISTS IN NUMS AND KINAL
3170       CALL ZERO(NUMS,KNUM)
3180       CALL ZERO(KINAL,KFIN)
3190       GOTO 108
3200C......FINAL LIST OF HITS HAS ANY ZEROS IN IT REMOVED
3210   135 CALL ZERO(KINAL,KFIN)
3220       KFIN=KFIN-1
3230       PRINT 217,KFIN
3240       CALL NEWCRD(IN)
3250       IF(KFIN.EQ.0)GOTO 125
3260C......ITEMS NUMBERS OF DEFINITE HITS ARE PRINTED
3270       PRINT 202,(KINAL(I),I=1,KFIN)
3280       PRINT 220
3290C......THIS READ DETERMINES WHETHER A COMPLETE LISTING IS PRINTED
3300       CALL IREAD(0,NUMS,KYMB,IN,KOR)
3310       IF(KOR)129,125,129
3320C......FROM 129 TO 128 PRINTS OUT THE COMPLETE LISTING OF
3330C......EVERY ITEM THAT WAS A DEFINITE HIT
3340   129 BEGINFILE "PERMFILE"
```

```
3350       READ("PERMFILE",214)KX
3360       IP=1
3370       DO 128 I=1,KFIN
3380       CALL FIND("PERMFILE","OUTFILE",KARD,IP,KX,KINAL(I),KQ,KERR)
3390       KA=KQ*2+7
3400   128 PRINT 218,(KARD(J),J=1,KA)
3410   125 PRINT 221
3420C......READ NUMBER DETERMINES WHETHER MORE SEARCHES ARE REQUIRED
3430       CALL IREAD(0,NUMS,KYMB,IN,KOR)
3440       IF(KOR-1)130,100,130
3450   130 PRINT 219
3460   200 FORMAT(21H ENTER NUMBER OF ANDS)
3470   201 FORMAT(33H ENTER NUMBER OF ORS FOR EACH AND)
3480   202 FORMAT(1X,10I5)
3490   203 FORMAT(21H ENTER NUMBER OF NOTS)
3500   204 FORMAT(29H ENTER OR LIST FOR AND NUMBER,I3)
3510   205 FORMAT(11H ENTER "OR",I3)
3520   206 FORMAT(2A4)
3530   207 FORMAT(12H ENTER "NOT",I3)
3540   208 FORMAT(6H ENTER,I3,2X,26HNOTS, ONE KEYWORD PER LINE/)
3550   209 FORMAT(//29H YOUR STRUCTURE IS AS FOLLOWS)
3560   210 FORMAT(4H AND,I3,2X,2HIS,2(5(3X,2A4)/))
3570   211 FORMAT(38H NO SPACE LEFT IN PRIMARY STORAGE FILE)
3580   212 FORMAT(9H NOTS ARE,2X,5(3X,2A4))
3590   213 FORMAT(46H ENTER 0 IF THERE IS NO MISTAKE IN THE LISTING)
3600   214 FORMAT(I5)
3610   215 FORMAT(1X,2I5,32X,2A4)
3620   216 FORMAT(1X,10I5)
3630   217 FORMAT(29H NUMBER OF ITEMS RETREIVED IS,I4,/,
3640&21H THESE ARE AS FOLLOWS/)
3650   218 FORMAT(/5H ID =,I5,2X,9HAUTHOR IS,2X,4A4,2X,6HDATE =,2X,
3660&A4,/,10H THERE ARE,I3,2X,19HKEYWORDS, WHICH ARE,/,2(7(2X,2A4)/))
3670   219 FORMAT(///25H END OF PROGRAM RETREIVER)
3680   220 FORMAT(37H ENTER 0 IF NO CARD PRINTOUT REQUIRED)
3690   221 FORMAT(33H ENTER 1 IF MORE SEARCHES DESIRED)
3700   222 FORMAT(24H NUMBER STORAGE EXCEEDED)
3710       STOP
3720       END
3730       SUBROUTINE KNOT(KGNOT,NOT,KC,KD,KARD,KFIN,KINAL,NB,N)
3740       DIMENSION NOT(2,10),KARD(600),KINAL(1200)
3750C......SUBROUTINE KNOT REMOVES ANY ITEMS THAT HAVE A NOT AMONGST
3760C......THEIR KEYWORDS FROM THE LIST OF POSSIBLE HITS. KINAL
3770C......MAY BE NUMS, KSTORE, OR KINAL. N IS ZERO FOR NUMS AND
3780C......KINAL, BUT 5*KLB FOR KSTORE AS THE ARRAY KSTORE IS USUALLY
3790C......A 2D ARRAY AND IS BEING TREATED LINEARLY.
3800       DO 100 I=1,KGNOT
3810       IF(KC.NE.NOT(1,I))GOTO 100
```

```
3820        IF(KD.NE.NOT(2,I))GOTO 100
3830        DO 101 J=1,NB
3840        KA=KARD(J)
3850        DO 102 K=1,KFIN
3860        IF(KA.NE.KINAL(K+N))GOTO 102
3870        KINAL(K+N)=0
3880        GOTO 101
3890   102 CONTINUE
3900   101 CONTINUE
3910   100 CONTINUE
3920        RETURN
3930        END
3940        SUBROUTINE ZERO(KINAL,KFIN)
3950        DIMENSION KINAL(500)
3960C......SUBROUTINE ZERO FIRST SORTS TYHE INPUT ARRAY FROM LOW
3970C......TO HIGH AND THEN REMOVES ANY ZEROS THAT MAY BE PRESENT.
3980        CALL SORT(KINAL,KFIN)
3990        DO 100 IA=1,KFIN
4000   100 IF(KINAL(IA).NE.0)GOTO 101
4010        KFIN=1
4020        GOTO 103
4030   101 IA=IA-1
4040        KFIN=KFIN-IA
4050        DO 102 I=1,KFIN
4060   102 KINAL(I)=KINAL(I+IA)
4070        KFIN=KFIN+1
4080   103 KINAL(KFIN)=0
4090        RETURN
4100        END
4110        SUBROUTINE RITE(OUT,KARD,KL,KQ)
4120        DIMENSION KARD(35)
4130        FILENAME OUT
4140        WRITE(OUT,200)(KARD(I),I=1,17)
4150        IF(KQ.LT.6)RETURN
4160        WRITE(OUT,201)(KARD(I),I=18,33)
4170   200 FORMAT(I4,2X,5A4,I2,10A4)
4180   201 FORMAT(4X,16A4)
4190        RETURN
4200        END
4210        SUBROUTINE FIND(IN,OUT,KARD,IP,N,NUMB,KQ,KERR)
4220        DIMENSION KARD(35)
4230        FILENAME IN,OUT
4240   204 FORMAT(2I5)
4250        KERR=0
4260        IF(IP.GT.N)GOTO 103
4270        DO 100 I=IP,N
4280        L=17
```

```
4290        READ(IN,200)(KARD(J),J=1,17)
4300        KQ=KARD(7)
4310        IF(KQ.LT.6)GOTO 101
4320        L=7+KQ*2
4330        READ(IN,201)(KARD(J),J=18,L)
4340   101 IF((NUMB.EQ.0).OR.(KARD(1).EQ.NUMB))GOTO 102
4350   100 CALL RITE(OUT,KARD,L,KQ)
4360        IP=N+1
4370        IF(NUMB.EQ.(-1))RETURN
4380        PRINT 202,NUMB
4390        KERR=5
4400        RETURN
4410   102 IP=I+1
4420        RETURN
4430   103 KERR=7
4440   200 FORMAT(I4,2X,5A4,I2,10A4)
4450   201 FORMAT(4X,16A4)
4460   202 FORMAT(14H NO CARD OF ID,I5,2X,13HFOUND IN FILE)
4470        RETURN
4480        END
4490        SUBROUTINE NEWCRD(IN)
4500        IN=72
4510        RETURN
4520        END
4530        SUBROUTINE IREAD(KZ,NUMS,KYMB,IN,IRE)
4540        DIMENSION NUMS(72),KYMB(6)
4550        IF(IN.GE.72)GOTO115
4560   117 NUM=0
4570        SIG=1.0
4580        LB=0
4590        IF((IN.NE.1).OR.(KZ.NE.1))GOTO116
4600        PRINT 202,(NUMS(I),I=1,72)
4610C.....MAIN SEARCH LOOP FOR FREE FORMAT ROUTINE
4620   116 DO 100 I=IN,72
4630        L=NUMS(I)
4640        IF(L.EQ.KYMB(4))GOTO100
4650        DO 101 J=1,3
4660        IF(L.EQ.KYMB(J))GOTO102
4670   101 CONTINUE
4680        K=I
4690C.....LOOP104 READS IN REQUIRED NUMBER
4700   103 DO 104 J=K,72
4710        JA=J
4720        IF(NUMS(J).EQ.KYMB(4))GOTO106
4730        LA=(NUMS(J)-KYMB(6))/KYMB(5)
4740        IF((LA.GT.9).OR.(LA.LT.0))GOTO105
4750        NUM=(NUM+LA)*10
```

```
4760        GOTO 104
4770   105 IF((NUMS(J).NE.KYMB(3)).OR.(LB.NE.0))GOTO107
4780        LB=J
4790   104 CONTINUE
4800        GOTO 113
4810   106 JA=JA-1
4820   113 LB=LB+1
4830        IF(LB.EQ.1)GOTO108
4840        ~~LB=JA-LB+2~~
4850C......FUNCTION RETURNS WITH THE NEW VALUE
4860   108 IRE=IFIX(FLOAT(NUM)*(0.1**FLOAT(LB))*SIG+0.001)
4870        GOTO 109
4880   102 GOTO(110,114,112),J
4890   110 SIG=-1.0
4900        GOTO 114
4910   112 LB=I
4920   114 K=I+1
4930        GOTO 103
4940C......107 REACHED ONLY FOR TERMINATION BECAUSE OF ILLEGAL CHARACTER
4950   107 PRINT 201
4960        PRINT 202,(NUMS(J),J=1,72)
4970        STOP
4980   100 CONTINUE
4990C......PRINTS OUT NEWLY ENTERED CARD IMAGE IF KZ=1
5000   115 READ 200,(NUMS(J),J=1,72)
5010        IN=1
5020        GOTO 117
5030   109 IN=JA+1
5040        RETURN
5050   200 FORMAT(72A1)
5060   201 FORMAT(26H ILLEGAL CHARACTER ON CARD)
5070   202 FORMAT(1H ,72A1)
5080        END
5090        SUBROUTINE SORT(KA,JJ)
5100        DIMENSION KA(500),IU(8),IL(8)
5110        M=1
5120        I=1
5130        J=JJ
5140   555 IF(I.GE.J)GOTO 700
5150   100 K=I
5160        IJ=(J+I)/2
5170        KT=KA(IJ)
5180        IF(KA(I).LE.KT)GOTO 200
5190        KA(IJ)=KA(I)
5200        KA(I)=KT
5210        KT=KA(IJ)
5220   200 L=J
```

```
    IF(KA(J).GE.KT)GOTO 400
    KA(IJ)=KA(J)
    KA(J)=KT
    KT=KA(IJ)
    IF(KA(I).LE.KT)GOTO 400
    KA(IJ)=KA(I)
    KA(I)=KT
    KT=KA(IJ)
    GOTO 400
300 KA(L)=KA(K)
    KA(K)=KS
400 L=L-1
    IF(KA(L).GT.KT)GOTO 400
    KS=KA(L)
500 K=K+1
    IF(KA(K).LT.KT)GOTO 500
    IF(K.LE.L)GOTO 300
    IF((L-I).LE.(J-K))GOTO 600
    IL(M)=I
    IU(M)=L
    I=K
    M=M+1
    GOTO 800
600 IL(M)=K
    IU(M)=J
    J=L
    M=M+1
    GOTO 800
700 M=M-1
    IF(M.EQ.0)GOTO 120
    I=IL(M)
    J=IU(M)
800 IF((J-I).GE.1)GOTO 100
    IF(I.EQ.1)GOTO 555
    I=I-1
900 I=I+1
    IF(I.EQ.J)GOTO 700
    KT=KA(I+1)
    IF(KA(I).LE.KT)GOTO 900
    K=I
110 KA(K+1)=KA(K)
    K=K-1
    IF(KT.LT.KA(K))GOTO 110
    KA(K+1)=KT
    GOTO 900
120 RETURN
    END
```

KARP/LISTER 1

```
1040       DIMENSION KARD(33),KEY(500),KOUT(64),KPOS(500),KEN(384)
1050       DATA KSTAR/1H*/,IBLANK/1H /,KIX/6H      /,LASH/1H=/
1060C......RANSIZ IS A SYSTEM COMMAND TO INDICATE THE RANDOM FILE IS
1070C......FILE 01 AND IS ALWAYS 98 WORDS LONG FOR EVERY RECORD
1080       CALL RANSIZ(1,98)
1090       READ(5,205)IJK
1100       K=379
1110C......LOOP 100 READS THROUGH PERMFILE AND CONTRACTS THE RECORD
1120C......SO THAT ONLY ONE SPACE IS LEFT BETWEEN WORDS ON THE CARD
1130       DO 100 I=1,500
1140       K=K+5
1150       DO 104 J=1,K
1160   104 KEN(J)=IBLANK
1170       READ(2,200,END=102)(KARD(J),J=1,17)
1180C......READS IN KEYWORD CARD PUNCHED OUT FROM KEYWORD
1190       LB=7+KARD(7)*2
1200       IF(KARD(7).LT.6)GOTO 101
1210       READ(2,201)(KARD(J),J=18,LB)
1220   101 KPOS(I)=KARD(1)
1230C......BEGINS TO READ IN BIBLIOGRAPHIC INFORMATION REFERRING
1240C......TO THE KEYWORD CARD ALREADY READ INTO THE MACHINE
1250C......A STAR IN CLUMN ONE OF THESE CARDS INDICATES THE LAST
1260C......BIBLIOGRAPHIC CARD RELATING TO A GIVEN KEYWORD CARD
1270       K=0
1280       KEMP=0
1290   105 READ(2,202)(KEY(J),J=1,65)
1300C......LOOP 103 COMPRESSES THE BIBLIOGRAPHIC INFORMATION SO THAT
1310C......THERE IS ONLY ONE SPCAE BETWEEN TWO BIBLIOGRAPHIC WORDS
1320       DO 103 J=2,65
1330       IF(KEY(J).EQ.LASH)GOTO 103
1340       IF(KEY(J).NE.IBLANK)GOTO 106
1350       KEMP=1
1360       GOTO 103
1370   106 K=K+1
1380       IF(KEMP.EQ.1)K=K+1
1390       IF(K.LT.380)GOTO 115
1400C......IF ALL THE STORAGE SPACE HAS BEEN USED, THE PROGRAM WILL
1410C......READ ON THROUGH FILE 02 UNTIL IT FINDS THE END OF THE CASE
1420C......IT IS WORKING ON. THEN CONTROL WILL PASS TO COMPRESSING
1430C......THE PART OF THE RECORD ALREADY STORED.
1440       WRITE(6,212)KARD(1)
1450       DO 117 LC=1,100
```

KARP/LISTER 2

```
1460        IF(KEY(1).EQ.KSTAR)GOTO 116
1470   117 READ(2,202)(KEY(MA),MA=1,65)
1480        K=K-1
1490   115 KEN(K)=KEY(J)
1500        KEMP=0
1510   103 CONTINUE
1520C......CHECK TO SEE WHETHER THE LAST BIBLIOGRAPHIC CARD READ
1530C......CONTAINS A STAR IN COLUMN ONE
1540        IF(KEY(1).NE.KSTAR)GOTO 105
1550C......ENCODE READS IN IN 384A1 AND CONMVERTS TO 64A6
1560   116 M=0
1570        L=(K-1)/6+1
1580        KOUT(L)=KIX
1590        DO 112 IJ=1,61,20
1600        M=M+120
1610        MA=M-119
1620        IF(MA.GT.K)GOTO 113
1630        IF(M.GT.K)M=K
1640   112 ENCODE(KOUT(IJ),203)(KEN(J),J=MA,M)
1650   113 IF(IJK.EQ.0)GOTO 111
1660        LC=17
1670        IF(LC.GT.LB)LC=LB
1680        WRITE(43,200)(KARD(J),J=1,LC)
1690        IF(LB.LT.18)GOTO 118
1700        WRITE(43,201)(KARD(J),J=18,LB)
1710C......PUNCH OUT THE BIBLIOGRAPHIC REFERENCE WITH A * ON LAST CARD
1720   118 LA=IBLANK
1730        DO 114 IJ=1,51,10
1740        IF(IJ.GT.L)GOTO 111
1750        MA=IJ+9
1760        IF(MA.GT.L)MA=L
1770        IF(MA.EQ.L)LA=KSTAR
1780   114 WRITE(43,213)LA,(KOUT(J),J=IJ,MA)
1790C......WRITESD A COMPRESSED RECORD OF 98 WORDS TO RANDOM FILE 01
1800   111 WRITE(1'I)(KARD(J),J=1,33),L,(KOUT(J),J=1,64)
1810   100 MZ=I
1820   102 WRITE(6,204)
1830C......LOOP 107 SEARCHES THROUGH KEYWORD TO FIND ALL THE CARDS
1840C......RELATING TO A GIVEN KEYWORD. THE ID NUMBER IS LOOKED UP
1850C......IN ARRAY KPOS TO DETERMINE ITS POSITION IN THE RANDOM FILE.
1860C......THE RECORD IS THEN PICKED UP AND PRINTED OUT.
1870        READ(3,205)N
1880        DO 107 I=1,N
1890C......READS IN NAME OF KEYWORD (NC,ND) AND ITS FREQUENCY
1900C......(NB)
1910        READ(3,206)NA,NB,NC,ND
```

KARP/LISTER 3

```
      READ(3,207)(KEY(J),J=1,NB)
      WRITE(6,208)NC,ND,NB
      K=1
C......LOOKS FOR EACH ID NUMBER IN TURN STORED IN KEYWORD
      DO 108 J=1,NB
      DO 109 LB=K,MZ
      IF(KPOS(LB).EQ.KEY(J))GOTO 110
  109 CONTINUE
C......WRITES A FAILURE MESSAGE IF ID NOT PRESENT IN KPOS
      WRITE(6,209)KEY(J)
      GOTO 108
  110 K=LB
C......READS IN BIBLIOGRAPHIC INFORMATION FOR RECORD K OF ID
C......KEY(J) AND PRINTS IT OUT
      READ(1'K)(KARD(L),L=1,33),LA,(KOUT(M),M=1,64)
      L=7+KARD(7)*2
      WRITE(6,210)(KARD(M),M=2,6),KARD(1),(KARD(M),M=8,L)
      WRITE(6,211)(KOUT(M),M=1,LA)
  108 CONTINUE
  107 CONTINUE
  200 FORMAT(I4,2X,5A4,I2,10A4)
  201 FORMAT(4X,16A4)
  202 FORMAT(A1,7X,64A1)
  203 FORMAT(120A1)
  204 FORMAT(/34H KARP INFORMATION RETRIEVAL SYSTEM,//,
     126H KEYWORD OCCURENCE LISTING/)
  205 FORMAT(I5)
  206 FORMAT(1X,2I5,32X,2A4)
  207 FORMAT(1X,10I5)
  208 FORMAT(///11H KEYWORD IS,3X,2A4,3X,12HCONTAINED IN,I5,2X,
     17HRECORDS)
  209 FORMAT(6H NO ID,I7,2X,13HFOUND IN FILE)
  210 FORMAT(3X,4A4,3X,A4,85X,I5,/,8X,12HKEYWORDS ARE,
     12(10(2X,2A4),/))
  211 FORMAT(8X,18A6)
  212 FORMAT(7H RECORD,I5,2X,19HTOO BIG FOR STORAGE)
  213 FORMAT(A1,7X,4H====,10A6)
      STOP
      END
```