FLEX USER GROUP
NEWSLETTER

3540 STURBRIDGE COURT
ANN ARBOR, MI 48105
ISSUE 4

COMMENTS

Well, NOTES3 are in the mail, and I can start in earnest at NOTES4 as I call the disk files. Not that we are running out of Assembler programs or Utilities, but I thought we would take a little break this time and get into BASIC programming using disk files with the SWTPC BASIC and Computerware's SUPERBASIC. Along that line, I have several divergent programs for a mailing list. I've received a "set" of programs from Ed Young of Evansville Indiana that are a very straightforward approach to the problem. another set was sent to me by Ron Mahon of Masontown Pensylvania. these are a little more involved and also a bit more capable. In addition to these, I've written a set for myself for use as a name and address list, and I've written some additions to Ron's programs. As you will see, these can be very brief and easy or they can get rather complex. Depending on the size of the list different approaches are required. For example, if the file is a few dozen names you can get by with a "bubble sort" that goes from disk file to disk file, provided you are not in a hurry. If the file is larger, you might want to put a number, say 50 records in memory and sort them there, writing a temporary file of the 50 sorted properly, and repeating the process until you have exhausted the original file. Then you may read and "merge" all the temporary files into a final output file that will be all sorted as desired. Sorting in memory is a great deal faster than sorting from file to file, and in addition there are many sort routines that run faster than a simple bubble sort. Our friends at TSC have written a good sort-merge package that runs on the DMAF system and also in FLEX2. It is written in Assembler rather than BASIC, and so is very fast.

Please accept an apology concerning the programs in the last Newsletter. Two of them were very sparsely commented. I had a couple of others that I wanted to include, but they too were nearly commentless. One of the bad ones was mine, so I am not chewing anyone out without including myself. The programs I wanted to include were mine too, and they will be along when I have thoroughly commented them.

We now have 38 members, and I am happy to report that the last issue was reproduced and mailed at a cost that was within the "budget" of $1 a copy. At this rate, we won't have to do anything to subsidize the letter. I hope all of you received fully legible copies. The reduced size essentially cuts reproduction costs in half, and does the same for the postage. I'll be able to increase the content again this time and still break even.

While I sincerely appreciate your responses with utility programs and the mailing list programs, no one has yet sent anything in the line of an article, be it a review of some software or hardware, a "how to" article describing a technique, or something else. I have written nearly all of the text to date.

I don't mind, but we ought to get some divergent viewpoints here. Also, I will soon have told you everything I know about the subject (ha).

PROGRESS TOWARD THE 6809

The other day, My company was given a sample 6809 by the local Motorola people. We will be building a processor board around it for evaluation. We have a cross assembler that will run on our 6800 system and produce 6809 object code. I will keep you informed of progress.

FLEX2

I've converted quite a bit of my operating system utilities to FLEX2, and have been operating successfully. It appears that some of the software will be a major problem to convert, specifically A/BASIC compiler. It is not just a matter of changing references. Since the sectors in FLEX2 are twice as large as in MINIFLEX, the File Control Block is larger because it contains a buffer capable of holding the information from a whole sector. Any compiler or compiled program that you have that uses files will need to be looked at carefully to see that the larger FCB doesn't write over part of the program. Certainly multiple FCB's will overlap unless the compiler can be gotten into and the FCB space assignment routine modified to assign 320 bytes rather than 192. Lacking the time and initiative to do this, I will continue to run A/BASIC and STRUBAL+ in the MINIFLEX configuration. It is too bad that this causes one to have to use two operating systems. Fortunately they are very similar, so there is no problem. Have any of you had any problems with implementing FLEX2?

HELP

Gil Olsen of Fairfax VA wants some help from any of you that have business software that will run on his SWTPC MF-68 system. Gil is not a programmer, and his main interests in business are in the area of Finance. Gil tells me that he has been hearing that any serious user for business applications should be going to the Smoke Signal system. Any comments on that? If you are interested in helping Gil, his address is below:

Gilbert Olsen
10560 Main St.
Suite 407
Fairfax VA 22030

MAILING LIST PROGRAMS

Just to start this off on a light note, I have had a mailing list for some time for this User's Group, and have been using Ron Mahon's programs to maintain it. I thought it would be cute to write a couple of programs to sort the list by last name or zip code. Of course we all know that zip codes are 5 digit numbers, right? Sorting numbers is quite easy of course. One night I added a few names to the list and tried the zip code sort, and things went

west in a hurry. After a few seconds of head scratching, I realized that I had just entered a Canadian address, with the zip code (if that is what the Canadians call it) 'H9G 1C3'. Since H,G, and C are hardly valid digits in a decimal number, the program went rapidly astray when it hit this code! Oh well, I just changed the sort to consider the zip as a string. Problem solved.

What is needed for something like a mailing list? First of all of course, you need to be able to generate a file of names and addresses. Next you need to be able to delete names and add others, or change parts of the existing addresses. It would be nice.to be able to sort the files by name, and maybe by zip code, depending on the size of the file and its purpose. Of course it would be nice to be able to print the file in mailing label format and maybe also in some sort of report or list format. Usually all these functions are not done in a single program, but in a group with a "menu" that is presented by an "executive program" that lists your choices and allows you to enter any of the others by entering a number or letter code. In addition, when you are done with one of the subprograms, generally the exit will cause you to re-enter the menu program where you may exit the whole group or choose another program to perform a different function.

With appologies to Ron Mahon, what is presented here is a conglomeration of his original programs and my modifications. Ron has a DMAF system and TSC's new sort merge package, and therefore is not very interested in any "slow" sort routines in BASIC. I will attempt to explain this set of programs first, and then present my simple minded program with reasons why it is nearly useless even for a personal name and address list. Lastly, I'll present the group from Ed Young and explain its operation.

MONITOR.BAS

This program is almost self explanatory if you understand the use of the CHAIN command in BASIC. It simply lists to the terminal the choices of programs available, and CHAIN's the proper one in response to your input.

Of course, the first thing you must do is to start a mailing list file. Therefore, let's look at the GEN program next. The first lines are self explanatory. If you are new to Disk BASIC, I would suggest you review the section of your BASIC manual that covers the disk operations as an aid to understanding these programs. Note that SWTPC BASIC appends the extension .DAT to all the filenames. All that need be entered is the name. You may of course use any name you like for a filename when starting a new file. If the file already exists, and you are adding to it, you must use the existing filename. Lines 140 to 340 allow input of names and addresses, and prompt you for the information. When you have input the last entry, you may "escape" from this loop by responding "DONE" to the name or first address prompt. The remainder of the input lines accept the response "CLEAR", to wipe out the current entry and allow you to start over again if you see an error. On line 40, the dimension is set for the number of entries that will be allowed before the program takes over and adds them to the file. If you enter less than this number, your DONE response will cause the sort to begin.

You may specify sorting by last name or zip code. The entries that are in memory are sorted by a bubble sort, and written to a file if the file is a new one, or "merged" with a file if it already exists. After writing this, I was playing with the sort program, and a thought came to mind that would make a great improvement in the sort speed. In sorting several records, there is a great deal of "swaping" that takes place. Each record is made up of 8 separate items, and all of these are in arrays. In order to "swap" this many items the number of times required for a bubble sort takes a great deal of time. It occurred to me that a 9th array containing a "tag" would allow a mode of operation in which much time could be saved.

I will try to explain it here. The 9th item of each record is added to it as the records are entered. It is T(N). This number is essentially a serial number. That is, the first record has T(N)=1, the second 2, etc. When the sort takes place, rather than swaping 8 items, the tags are swapped. In other words, rather than sort the records, the tags are "scrambled" in such a way that they make up a list that indicates the order in which the records must be output in order to come out sorted. For a simple example, suppose the list contains two items, and the first is "greater" than the second. Rather than swaping the items, the tags are exchanged, and the tag sequence is then 2,1. This means that the second item should be first. When the list is read out, the second item is read first, and the output is sorted. This means that only two items are swapped rather than 8 each time the sort calls for a swap.

The net result was to decrease the sort time to about half of the original. More will be said about actual times in the description of the SORT program. While I was at it, I decided to change the simple bubble sort to a Shell-Metzner sort. I don't claim to be a computer science major, and had to work out the flow chart for this one from a sample program in one of the magazines. Its operation is logical but complex and we don't need to get into it here, but it requires much less time, fewer "passes" through the list, and many less swaps to sort a list of items.

To get back to the merge operation, consider the following: Think of a merge like the job of inserting a pile of file cards with words on them into a larger pile, both piles being sorted alphabetically first. The procedure is straightforward. In this program, an entry is read from the file and compared with the first entry in memory. Whichever comes first alphabetically or numerically for name or zip sort respectively, is written to the new file that becomes the final file. At the end, the original file is renamed by adding "O" to the beginning to signify "old file". The new file which was opened with a "T" at the front to signify "temporary", is renamed to the original file name. If an error occurs, and the files are closed in the middle of the program operation, the original file remains intact, and the working file (the one with the T appended) may be deleted and the operation started again.

I strongly recommend that you experiment with some short files until you have removed all the bugs from these programs after entering them in your system. Also, make a backup of the programs, as these have been written to have the data file on the same disk as the BASIC files. If you have a third drive, you might modify these to open the data files on disk drive 2 by appending a "2." to the beginning of all data filenames.

The SORT program will sort a file by name or zip. Of course the files will already be sorted by one or the other, and it would only make sense to start with a file sorted by name and sort it by zip and vice-versa. The sort was slow, taking about 8 minutes for our user group file of 38 names. It reads a "batch" of names and sorts them to a new file. If there are more than a "batch" in the original file, it reads another batch, sorts them, and merges them to the new file. The size of a batch has been tentatively set to 50 at line 40. You may make this larger if you have enough memory, or smaller if you do not. The 50 requires about 24K of memory. BASIC will tell you when you have made the dimension too large. You will notice that the sort routines, read, write, and swap routines are very similar in the SORT program, to the GEN program.

The time given above is for the original bubble sort. I found that with that sort, the job got done faster if I made the dimension 20 or so. This means that it is faster to sort two lists of 20 items and merge the second into the first, than to sort a single list of 38. The two batch sort ran about 3.6 minutes. After converting to the Shell-Metzner sort and the tag technique I found that a single sort of the 38 records took 1 minute and 8 seconds, and about half of that is the time to read the source file and write the sorted file. I feel that now, the sort program given here is respectable enough so that someone with a list of a few hundred names would find it useful.

The LIST program will list a file, one name to a line, provided that you have a 132 column printer, or one like my IDS where you can squeeze the format to 132 characters on 8 1/2 inch paper. You will obtain a slightly scrambled, but interpretable list on a narrower printer. This program simply reads the file and formats a list of the entries.

The LABEL program prints labels for you. If you have a tractor feed printer you may use the self adhesive labels available from Avery and others. If you have a friction feed printer you can print the labels on your regular paper and cut them up on a paper cutter and use a "glue stick" to attach them to your mailing. This program prints labels at 6 line intervals, and is thus set up for labels at 1 inch increments. Thanks to Ron Mahon's programming, it will give you an alignment test pattern to get the printer set up correctly before you start printing labels.

The DELETE and CHANGE programs are similar in operation. They allow you to enter a list of names in memory, and then they search the file for matching names. The DELETE program writes any entry that matches a name in the delete entries to a "deleted" file, and the remainder to the output file. If you make a mistake and it can't find a match, it will let you know that names couldn't be found by listing them for you. All files are kept and you may recover the original file if something goes wrong. The CHANGE program prompts you for which fields of the entry you want to change, and when you are satisfied, it writes the changed entry to the new file.

As you can see, there is a lot of file manipulation, and lots of things going on in memory. In order to do anything this complex, it is necessary to "batch" process the entries in memory. These programs are characterized by very complete prompting, again thanks to Ron Mahon's original programming. My contribution was the GEN and the SORT programs. The GEN program contains

prompts identical to Ron's original CREATE program. These programs were received from Ron over a considerable period of time, during which he continued to develop the whole system. Since he switched to a DMAF system, he eliminated some complications he had concerning a "default" city, state, and zip. This comes in handy for a very local mailing list because a very short entry is used to indicate the default condition. In the case of the User's Group, we are very spread out, and the default does no good. I've eliminated it from all the programs here.

MY DUMB FIRST TRY

The second set of programs, actually all combined in one longer program, will illustrate how not to do a name and address list. These were my first attempt at using disk files. I found out that it is not practical to sort 50 names using a bubble sort from file to file. To do this, it is necessary to read and write the entire file up to 50 times. All these programs are geared to finding a single entry and operating on it, reading and writing the entire file each time.

The lines down to 180 are the menu portion of the program. They present a list of the functions and allow a choice by the operator. Each entry has as its first item, a keyword. it is best to use last name first name with no space between. This keyword is used for sorting and finding a listing. It would have been better to separate the first and last names in two string entries as Ron Mahon has done, but that didn't occur to me at the time. The menu section uses a jump list to get to the needed routine in the program. You will find the sections separated and a REM at the start of each, describing the function of the section that follows. I do like the format of the output listing, as it makes a nice two-column listing for a personal "Phone Book".

I have my personal name and address file set up in this system, and will probably continue to maintain it that way for some time. The provision for printing out with or without phone numbers makes the list usable for address labels on Christmas cards, for which I have used it now for two Christmases. Since when a record is added, it is inserted alphabetically by keyword, the only time the SORT function would be needed would be in the case where you would use the text editor to set up a starting file, and not place the entries alphabetically. Incidentally, the way this program is set up, that is the only logical way to start a file of names. It would be frustrating to use the ADD function, since the whole file would be read and re-written for each name added. This would quickly get to be very time consuming, and the probability of the disk getting disorganized enough to cause a problem would be very high.

I did say that this program was very nearly useless as it stands. It could be modified making use of the techniques in the set of mailing list programs to be rather nice though. Maybe one of you will be stimulated to experiment a little and do the modifications. Perhaps this program is simple minded enough to get those of you who got lost in the mailing list programs started, so you will later be able to understand the more complex programs.

MAILLIST BY ED YOUNG

These programs again start with a MENU program that allows selection of the operation to be done. If you examine this set carefully you will see different techniques used here and there, the mark of a different individual programmer. For example, in the file creation program, Ed uses a FOR-NEXT loop to enter up to 1000 entries. Since you wouldn't ever enter that many at a single sitting, the keyword END gets you out of that loop and allows you to end the program. This is a more structured way of writing the program than using a GOTO to stay in the loop and a conditional branch to get out of it as I had done.

Ed's LIST program is equivalent to Ron Mahon's LABEL program. The ADD program adds to the end of a file. The update program allows changing a record as did the CHANGE program above. Of course the DELETE program deletes a record. The INSERT program is interesting because it allows the operator to specify after which record the new one is to be inserted. Thus if you have a list, you may insert records and retain the sorted nature of the file. Last but not least, the SORT program is more comprehensive than either of the above, because it allows sorting by any of the data fields in a record. Ed's SORT is identical in function to mine above. It reads a whole file and sorts essentially one record per pass by a bubble sort. It therefore has to read a 50 item file up to 50 times to sort it.

CONCLUSIONS

As you have seen by looking at these programs and the brief descriptions above, there is a limitation in sorting or modifying files if they must be read and re-written for each item. It is much faster to read a batch of entries and sort them in memory. The situation, however presents a much more difficult programming challenge. All this may change significantly with the advent of TSC's new BASIC that will allow random access files. The ability to swap records in a file will change the efficiency of sorting directly in disk files considerably, while easing the programming difficulties. Meanwhile, it appears necessary to use some sort of batch processing in memory to make the programs efficient. I encourage you to try these programs and see just what their limitations are. Perhaps some of you will be encouraged to put together a further improved package for us.

HOW MANY

Some of you have been wondering if you have missed issues of this newsletter. Except for the 10 or so of you that I owe back issues, you have not. It now looks like it will not be possible for me to be able to get 12 issues a year done. At this point, the most probable number will be 8 or 9. All subscriptions that I have accepted will run for 12 issues. I would rather put out 8 quality letters than rush to get 12 out and have some that are so-so. A few of you have written encouraging comments, and we will try to include something for everyone, though I have assumed that FLEX users for the most part are not in the novice category. One subscriber wrote and asked if the Assembler programming article that I am planning to run over several issues is available in complete form now, as it seemed to be just what he needs now. I will make this offer to anyone who can't wait. For $2 I'll send you all 18 pages, first class.

DMAF PROBLEMS

I have been offering new subscribers a way of getting all the back issues that is painless to me. I have been sending them the back issues formatted to the width specified on their disk which they have sent to me, so that they may print it out. Unfortunately I have not been specific enough, and three people have sent me large disks, which I have no way of processing. Any of you who have done this, are still waiting for back issues. Please hold on just a little longer. I will send you hard copies and your disk both in the near future. Meanwhile I have made it clear that I will supply only the minidisks for the time being. I may soon have available a system on which I can prepare DMAF compatible disks. Meanwhile, the DMAF and FLEX2 programs are quite compatible, and all Assembler programs will have the differences documented so that they will serve both operating systems. Along that line, this issue contains a cross reference chart of the jump addresses for the corresponding routines in Miniflex and FLEX2 (or the DMAF version).

LAST MINUTE NOTES

The group is now at 45 members and I owe several of you back issues. My employer has had a couple of rush projects, one of my employees quit, and I was busy programming for a couple of weeks and week-ends, and so have gotten "behinder" in the User Group business. My early issue copies have been used up, and I will be sending back issues to all as soon as I can get copies made. Meanwhile, the next issue is already in mind. I have enough material for an issue full of utilities and assembler routines again. Just before I received FLEX2, I wrote a couple of utilities for myself. One of these causes output to a disk file rather than the printer (strangely enough just like the O utility in FLEX2). I have used this utility to make copies of the user notes on disks for some of you. It was used with the TSC Text Processor to individualize your copies. It needs some considerable additions to the comments and a little debugging in that it doesn't terminate nicely at the end of the file. It does finish the output and close the file, but then it goes west. Maybe if I print them as they are, some one of you will see my dumb error and fix it.

The other utility is one that allows the reading of a file from disk when input is called by a program. The EXEC command and a file in FLEX allow you to enter a number of "tasks" for the DOS to do, but wont be read by a program to supply input. My C. command allows a file to be read in response to input prompts by a program. It really is great when working with a "foreign" assembler like Hemenway's relocatable macroassembler, or a language like STRUBAL+ for which not enough documentation is supplied to allow you to work easily with FLEX. I have used this with STRUBAL and the LINKAGE EDITOR of Hemenway's. It works great when I have to re-compile a source file 15 times to get it just the way I want it. The way it works, it outputs the information from the command file to the terminal just as though you were typing it in. This one has a bug too, it exits more gracefully to FLEX but indicates DISK ERROR 18 on exit.

I have received a major Utility from Milan Konecny. It is a disk file patch program that allows you to look at a disk file and change a byte in it without disturbing anything else. Milan sent me the assembled listing, a disk with the source file, a large amount of printout that explains its use, and a

letter describing it in detail! Just that one set of documentation would fill an issue of these notes.

Several others of you have sent programs which, unfortunately I have not had time to acknowledge. I will be including many of them in future issues. I trust that we will be able to include enough of a variety of items to make you all happy!

PLEASE NOTE:

THE PROGRAMS BY ED YOUNG HAVE SOMEHOW MANAGED TO BE WIPED OUT DURING A BACKUP OR SOMETHING, AND THEY ARE NOT INCLUDED HERE. THE FUNCTION WAS JUST ABOUT IDENTICAL TO THE NAMAD.BAS INCLUDED HERE. IF ANYONE "HAS TO HAVE THEM" I'LL MAKE COPIES FROM THE ORIGINAL LISTINGS AND SEND THEM TO YOU. JUST LET ME KNOW.

FLEX-FLEX2 CROSS REFERENCES

COLDS	AD00	7100
WARMS	AD03	7103
RENTER	AD06	7106
GETCHR	AD15	710F
PUTCHR	AD18	7112
INBUFF	AD1B	7115
PSTRNG	AD1E	7118
CLASS	AD21	711B
PCRLF	AD24	711E
NXTCH	AD27	7121
RSTRIO	AD2A	7124
GETFIL	AD2D	7127
LOAD	AD30	712A
SETEXT	AD33	712D
ADDBX	AD36	7130
OUTDEC	AD39	7133
OUTHEX	AD3C	7139
RPTERR	AD3F	713C
GETHEX	AD42	713F
OUTADR	AD45	NO EQUIVALENT
INDEC	AD48	NO EQUIVALENT
DOCMND	AD4B	7142
HEXADJ	B198	74BA USEFUL SUBR. OF GETHEX
OPREAD	B2A4	758A OPEN FOR READ SUBR.

DEFAULT CODES FOR SETEXT

0 - BIN
1 - TXT
2 - CMD
3 - BAS
4 - SYS
5 -BAK

THE FOLLOWING ARE FOR FLEX2 ONLY

6 - SCR
7 - DAT
8 - BAC
9 - DIR
10 - PRT
11 - OUT

```
*** MONITOR.BAS ***

10  X=1:REM SETS TERMINAL OUTPUT TO PORT #1
20 REM ***THIS PROGRAM NAMED MONITOR
30 REM ***IT IS A MAILING LIST PROGRAM
40 REM ***IT SELECTS SUB PROVISIONS TO CREATE,ADD,DELETE***
50 REM ***LIST,AND PRINT LABELS
60 REM ***(C) COPYRIGHT 1978 BY
70 REM ***RAM ASSOCIATES
80 REM ***TO 201 N. MAIN ST.
90 REM ***MASONTOWN,PA.  15461
100 REM ***RON MAHON, LINDA MICHALIK
110 REM ***
120 REM ***INSERT HOME UP AND ERASE TILL END OF FRAME
130 PRINT CHR$(26)
140 PRINT #X,TAB (10),"MAILING LIST PROGRAM"
150 PRINT #X,TAB (10),"********************"
160 PRINT #X,TAB (10),"REVISION 2 SWTP FLEX"
170 PRINT #X,"SELECT THE SUBPROGRAM FROM THIS LIST"
180 PRINT #X,"GENERATE OR ADD TO A FILE (G)"
190 PRINT #X,"SORT BY LAST NAME OR ZIP (S)"
200 PRINT #X,"LIST FILE (L)"
210 PRINT #X,"PRINT LABELS (P)"
220 PRINT #X,"DELETE FROM A FILE (D)"
230 PRINT #X,"CHANGE A RECORD IN A FILE (C)"
240 PRINT #X,"TO END PROGRAM TYPE END FOR NAME (END)"
250 INPUT "WHICH PROGRAM DO YOU WANT",A$
260 IF LEFT$(A$,1)="G" THEN CHAIN GEN
270 IF LEFT$(A$,1)="S" THEN CHAIN SORT
280 IF LEFT$(A$,1)="P" THEN CHAIN LABEL
290 IF LEFT$(A$,1)="E" THEN GOTO340
300 IF LEFT$(A$,1)="L" THEN CHAIN LIST
310 IF LEFT$(A$,1-)="D" THEN CHAIN DELETE
320 IF LEFT$(A$,1)="C" THEN CHAIN CHANGE
330 PRINT #X,"HUH???":GOTO 250
340 PRINT #X,"END OF MONITOR PROGRAM'
350 STOP
```

*** GEN.BAS ***

```
10 REM THIS PROGRAM NAMED GEN.BAS
20 REM IT CREATES A NEW FILE TO USE
30 REM WITH THE MAILING LIST PROGRAM
40 LET X=28:REM CHANGE NUMBER OF ENTRIES ALLOWED TO SUIT MEMORY SIZE
50 DIM N$(X),M$(X),Y$(X),A$(X)
60 DIM Q$(X),S$(X),Z$(X),C$(X),T(X+1)
70 PRINT CHR$(26):REM CLEAR SCREEN FOR ADM-3
80 PRINT TAB(10);"THIS PROGRAM CREATES A NEW FILE"
90 PRINT TAB(10);"OR ADDS TO AN EXISTING ONE"
100 PRINT TAB(10);"WHAT IS THE NAME OF THE FILE (7 CHAR. MAX)";:INPUT F$
110 OPEN #0,F$
120 INPUT "IS THIS A NEW FILE",W$
130 PRINT TAB(10);"TYPE DONE FOR NAME WHEN FINISHED"
140 FOR N=1 TO X
150 PRINT "FIRST NAME OR INITIAL";:INPUT N$(N)
160 IF N$(N)="DONE" THEN 360
170 PRINT "LAST NAME";:INPUT M$(N)
180 IF M$(N)="DONE" THEN 360
190 PRINT "ADDRESS";:INPUT A$(N)
200 IF A$(N)="CLEAR" THEN 150
210 IF A$(N)="DONE" THEN 360
220 PRINT "ADDITIONAL ADDRESS";:INPUT Y$(N)
230 IF Y$(N)="CLEAR" THEN 150
240 PRINT "CITY";:INPUT Q$(N)
250 IF Q$(N)="CLEAR" THEN 150
260 PRINT "STATE";:INPUT S$(N)
270 IF S$(N)="CLEAR" THEN 150
280 PRINT "ZIP";:INPUT Z$(N)
290 IF Z$(N)="CLEAR" THEN 150
300 PRINT "SERVICE CODE";:INPUT C$(N)
310 IF C$(N)="CLEAR" THEN 150
320 PRINT
330  T(N)=N
340 NEXT N
350 REM SHELL-METZNER SORT ROUTINE
360 PRINT "SORT BY NAME OR ZIP (N/Z)"
370 PRINT "NOTE, YOU MUST BE CONSISTENT WHEN ADDING TO A FILE."
380 INPUT V$
390 Z=N-1
400 IF N=X THEN Z=X
410  M=Z
420  M=INT(M/2)
430 PRINT :PRINT M;:REM SORT INTERVAL
440 IF M=0 THEN 570:REM EXIT SORT
450 J=1:K=Z-M:REM INITIALIZE POINTERS
460  I1=J
470  L=I1+M
480 IF V$="Z" THEN IF Z$(T(I1))§$(T(L)) THEN 540
490 IF V$="N" THEN IF Z$(T(I1))§$(T(L)) THEN 540
500  T(X+1)=T(I1):T(I1)=T(L):T(L)=T(X+1)
510 PRINT "*";:REM ONE SWAP OF TAG.
520  I1=I1-M
530 IF I1†O THEN 470
540  J=J+1
```

```
*** GEN.BAS - PAGE TWO ***

550 IF J†K THEN 420
560 GOTO 460
570 IF W$§†"Y" THEN 660
580 REM WRITE OR MERGE ROUTINES FOLLOW
590 FOR N=1 TO Z
600  N1=T(N)
610 WRITE #0,N$(N1),M$(N1),A$(N1),Y$(N1),Q$(N1),S$(N1),Z$(N1),C$(N1)
620 NEXT N
630 CLOSE #0
640 GOTO 990
650 REM MERGE INTO EXISTING FILE
660 LET L$="T"+F$
670 OPEN #1,L$
680 GOSUB 810
690 FOR N=1 TO Z
700  N1=T(N)
710 IF EOF(0)=1 THEN GOSUB 830:REM WRITE REMAINING MEMORY
720 IF V$="N" THEN GOSUB 880
730 IF V$="Z" THEN GOSUB 850
740 IF C=1 THEN GOSUB 830
750 IF C=2 THEN GOSUB 800:GOTO 710
760 NEXT N
770 IF EOF(0)=1 THEN 910
780 GOSUB 800:REM WRITE OLD FILE TO NEW FILE
790 GOTO 770
800 WRITE #1,N$(X),M$(X),A$(X),Y$(X),Q$(X),S$(X),Z$(X),C$(X)
810 READ #0,N$(X),M$(X),A$(X),Y$(X),Q$(X),S$(X),Z$(X),C$(X)
820 RETURN
830 WRITE #1,N$(N1),M$(N1),A$(N1),Y$(N1),Q$(N1),S$(N1),Z$(N1),C$(N1)
840 RETURN
850 REM ZIP COMPARE
860 IF Z$(N1)†Z$(X) THEN C=2:RETURN
870  C=1:RETURN
880 REM NAME COMPARE
890 IF M$(N1)†M$(X) THEN C=2:RETURN
900  C=1:RETURN
910 CLOSE #0:CLOSE#1
920  V$="O"+F$
930 RENAME F$,V$
940  L$="T"+F$
950 RENAME L$,F$
960 PRINT "ORIGINAL FILE JOW NAMED O";F$
970 PRINT "NEW FILE NOW NAMED ";F$
980 PRINT "CHECK NEW FILE BEFORE DELETING OLD ONE"
990 PRINT :PRINT"WOULD YOU LIKE TO ENTER MORE":INPUT W$
1000 IF W$="Y" THEN 120
1010 PRINT :PRINT"WOULD YOU LIKE A LISTING":INPUT W$
1020 IF W$="Y" THEN 1040
1030 CHAIN MONITOR
1040 CHAIN LIST
```

```
*** SORT.BAS ***

10 REM THIS PROGRAM NAMED SORT.BAS
20 REM SORT A MAILING LIST FILE BY
30 REM NAME OR ZIP

40 X=50:REM MAY BE ADJUSTED FOR YOUR MEMORY SIZE
50 DIM N$(X),M$(X),A$(X),Y$(X),Q$(X),S$(X),Z$(X),C$(X),T(X+1):REM TAG

60 LET I=1:E1=0:REM INITIALIZE PASS AND END FLAGS

70 REM PROMPT USER
80 PRINT CHR$(26):REM CLEAR SCREEN AND HOME FOR ADM-3
90 PRINT TAB(10);"THIS PROGRAM SORTS A FILE"
100 PRINT TAB(10);"BY NAME OR ZIP":PRINT
110 PRINT TAB(10);"NAME OF FILE TO BE SORTED";:INPUT F$
120 PRINT TAB(10);"NAME OF SORTED FILE";:INPUT O$
130 PRINT TAB(10);"SORT BY ZIP OR NAME (Z-N)";:INPUT V$

140 REM OPEN FILES AND READ FIRST BLOCK OF SOURCE FILE
150 OPEN #0,F$
160 OPEN #1,O$
170 FOR N=1 TO X
180 READ #0,N$(N),M$(N),A$(N),Y$(N),Q$(N),S$(N),Z$(N),C$(N)
190 IF EOF(0)=1 THEN E1=1:Z=N-1:GOTO 240
200  T(N)=N
210 NEXT N

220  REM SHELL-METZNER SORT ROUTINE

230  Z=N
240 M=Z
250  M=INT(M/2)
260 PRINT :PRINT M;:REM SORT INTERVAL. THIS LINE MAY BE DEL.
270 IF M=0 THEN 400:REM EXIT SORT
280  J=1:K=Z-M
290  I1=J
300  L=I1+M
310 IF V$="Z" THEN IF Z$(T(I1))§Z$(T(L)) THEN 370
320 IF V$="N" THEN IF M$(T(I1))§M$(T(L)) THEN 370
330  T(X+1)=T(I1):T(I1)=T(L):T(L)=T(X+1)
340 PRINT "*";:REM *=ONE SWAP OF TAG. THIS LINE MAY BE DEL.
350  I1=I1-M
360 IF I1†0 THEN 300
370  J=J+1
380 IF J†K THEN 250
390 GOTO 290
400 IF I§†1 THEN 510

410 REM WRITE FIRST BLOCK OF OUTPUT FILE
420 FOR N=1 TO Z
430 N1=T(N)
440 WRITE #1,N$(N1),M$(N1),A$(N1),Y$(N1),Q$(N1),S$(N1),Z$(N1),C$(N1)
450 NEXT N
460 CLOSE #1
470 IF E1=1 THEN 870:REM EXIT
```

```
*** SORT.BAS - PAGE TWO ***

480 LET I=I+1:REM PASS COUNT
490 GOTO 170:REM GO AROUND AGAIN

500 REM MERGE FURTHER BLOCKS WITH PREVIOUS OUTPUT FILE
510 LET L$="T"+O$
520 OPEN #1,O$
530 OPEN #2,L$
540 GOSUB 690

550 FOR N=1 TO Z
560  N1=T(N)
570 IF EOF(1)=1 THEN GOSUB 720:GOTO 620
580 IF V$="N" THEN GOSUB 780
590 IF V$="Z" THEN GOSUB 750
600 IF C=1 THEN GOSUB 720
610 IF C=2 THEN GOSUB 680:GOTO 570
620 NEXT N

630 REM RAN OUT OF SORTED ITEMS IN MEMORY
640 IF EOF(1)=1 THEN 810:REM MORE IN SOURCE FILE?
650 GOSUB 680
660 GOTO 640

670 WRITE ITEM FROM SOURCE FILE AND READ NEXT
680 WRITE #2,N$(X),M$(X),A$(X),Y$(X),Q$(X),S$(X),Z$(X),C$(X)
690 READ #1,N$(X),M$(X),A$(X),Y$(X),Q$(X),S$(X),Z$(X),C$(X)
700 RETURN

710 REM WRITE ITEM FROM MEMORY
720 WRITE #2,N$(N1),M$(N1),A$(N1),Y$(N1),Q$(N1),S$(N1),Z$(N1),C$(N1)
730 RETURN

740 REM ZIP COMPARE FOR MERGE
750 IF Z$(N1)†Z$(X) THEN C=2:RETURN
760 LET C=1:RETURN

770 REM NAME COMPARE FOR MERGE
780 IF M$(N1)†M$(X) THEN C=2:RETURN
790 LET C=1:RETURN

800 REM CLOSE, RENAME, DELETE TEMPORARY FILE
810 CLOSE #1,#2
820 KILL O$
830 RENAME L$,O$:I=I+1
840 IF E1=1 THEN 870
850 GOTO 170

860 REM EXIT PROGRAM
870 CLOSE #0
880 INPUT "WANT A LISTING",W$
890 IF W$="Y" THEN CHAIN LIST
900 CHAIN MONITOR
910 END

920 REM THIS PROGRAM SORTS 38 RECORDS IN 1 MIN:12 SEC
```

*** LIST.BAS ***

```
10  X1=1:X=1:P=7:REM INITIALIZES TERMINAL PORT 1,PRINTER PORT 7
20 LINE= 148
30 REM ***THIS PROGRAM NAMED LIST
40 REM IT LISTS A DATA FILE
50 REM WITH A MAILING LIST PROGRAM
60 REM ***(C)COPYRIGHT 1978 BY
70 REM ***RAM ASSOCIATES
80 REM ***TO 201 N.M MAIN ST.
90 REM ***MASONTOWN, PA.   15461
100 REM ***RON MAHON, LINDA MICHALIK
110 REM ***INSERT
120 PRINT #X,CHR$(26):REM CLEAR SCREEN FOR ADM-3
130 GOTO 180
140 FOR C=1 TO 130:PRINT#X,"-";:NEXT C
150 PRINT #X,:RETURN
160 FOR C=1 TO 132:PRINT#X,"*";:NEXT C
170 PRINT #X,:RETURN
180 PRINT #X,TAB(10),"THIS POROGRAM LISTS A FILE"
190 INPUT "WHAT IS THE NAME OF FILE( 8 CHARACTERS)",F$
200 OPEN #0,F$
210 INPUT "DO YOU WANT A HARD COPY Y,N",A$
220 IF A$="Y" THEN X=P:PRINT #X, CHR$(31):GOSUB 290
230 IF A$St"N" THEN PRINT#X,"HUH???":GOT0220
240 INPUT "DO YOU WANT TO DISPLAY FILE Y,N",A$
250 IF A$="Y" THEN GOSUB 290
260 IF A$§†"N" THEN PRINT#X,"HUH???":GOTO 240
270 IF A$="N" CLOSE #0
280 CHAIN MONITOR.BAS
290 REM ***THIS SUB READS FILE IN ORDER OF INPUT
300 REM @***AND EITHER PRINTS OR DISPLAYS
310 RESTORE #0
320 PRINT #X
330 PRINT #X,CHR$(14);"THIS IS A LISTING OF THE FILE NAMED ";F$
340 GOSUB 160
350 GOSUB 160
360 PRINT #X;"LAST NAME";
370 PRINT #X,TAB(30);"FIRST NAME";
380 PRINT #X;TAB(41);"ADDRESS";
390 PRINT #X;TAB(71);"ADDITIONAL ADDRESS";
400 PRINT #X;TAB(101);"ZIP";
410 PRINT #X;TAB(116);"SERVICE CODES";:PRINT#X
420 GOSUB 140
430 REM ***READS FILE PRINTS IN ORDER OF INPUT
440  G=0
450 READ #0,N$,N$(1),A$,A$(1),Q$.S$,Z$,C$
460 PRINT #X,N$(1);
470 PRINT #X,TAB(30);N$;
480 PRINT #X,TAB(41);A$;
490 IF LEN (A$(1))§2 THEN 510
500 PRINT #X,TAB(71);A$(1);
510 PRINT #X,TAB(101);Z$;
520 PRINT #X,TAB(116);C$;
530 PRINT #X
540 IF EOF(0)§†0 THEN CLOSE#0:GOTO 570
```

```
550 G=G+1
560 GOTO 450
570 PRINT #X,"END OF FILE".:PRINT#X,G;"NAMES PRINTED"
580 PRINT #X,CHR$(29)
590 INPUT "DO YOU WANT TO PRINT ANOTHER FILE Y,N",A$
600  X=X1
610 IF A$="Y" THEN 190
620 IF A$§†"No' PRINT#X,"HUH???":GOTO 590
630 CHAIN MONITOR.BAS
```

*** LABEL.BAS ***

```
10 LINE= 148
20  X=7:REM INITIALIZE FOR PRINTER
30 REM ***THIS PROGRAM NAMED LABEL
40 REM IT CREATES A NEW FILE TO USE
50 REM WITH A MAILING LIST PROGRAM
60 REM ***(C)COPYRIGHT 1978 BY
70 REM ***RAM ASSOCIATES
80 REM ***TO 201 N.M MAIN ST.
90 REM ***MASONTOWN, PA.   15461
100 REM ***RON MAHON, LINDA MICHALIK
110 REM ***INSERT
120 PRINT CHR$(26)
130 PRINT TAB(10),"THIS PROGRAM PRINTS LABELS FROM A FILE"
140 INPUT "WHAT IS THE NAME OF FILE( 8 CHARACTERS)",F$
150 OPEN #0,F$
160 INPUT "IS THE PRINTER READY Y,N",A$
170 IF A$§†"Y" THEN PRINT,"HUH???":GOTO 160
180 INPUT "DO YOU NEED TEST PATTERN Y,N",A$
190 IF A$="Y" THEN GOSUB 400
200 IF A$§†"N" THEN PRINT "HUH???":GOTO 180
210 RESTORE #0
220 REM ***READS FILE PRINTS IN ORDER OF INPUT
230 PRINT #X,CHR$(11):G=0
240 READ #0,N$,N$(1),A$,A$(1),Q$,S$,Z$,C$
250 PRINT #X,TAB(2);N$;" ";N$(1)
260 PRINT #X,TAB(2);A$:C=1
270 IF LEN (A$(1))§2 THEN C=0:GOTO 290
280 PRINT #X,TAB(2);A$(1)
290 PRINT #X,TAB(2);Q$;", ";S$;" ";Z$
300 IF C=0 THEN PRINT#X,CHR$(11)
310 IF C=1 THEN PRINT#X:PRINT#X
320 IF EOF(0)§†O THEN CLOSE#O:GOTO 350
330  G=G+1
340 GOTO 240
350 PRINT #X,"END OF FILE":PRINT#X,G;"LABELS PRINTED"
360 INPUT "DO YOU WANT TO PRINT ANOTHER FILE Y,N",A$
370 IF A$="Y" THEN 140
380 IF A$§†"N" PRINT "HUH???":GOTO 360
390 CHAIN MONITOR.BAS
400 REM ***THIS PRINTS ALIGNMENT TEST PATTERN
410 FOR L=1 TO 4
420 PRINT #X,TAB(2);"§==============================†"
430 PRINT #X,TAB(2);"§==============================†"
440 PRINT #X,TAB(2);"§==============================†"
450 PRINT: PRINT: PRINT
460 NEXT L
470 RETURN
```

```
*** DELETE.BAS ***

10  REM ***(C)COPYRIGHT 1978 BY
20  REM ***RAM ASSOCIATES
30  REM ***201 N. MAIN ST.
40  REM ***MASONTOWN, PA. 15461
50  REM ***RON MAHON, LINDA MICHALIK
60  REM ***INSERT SCREEN CLEAR HERE
70  DIM G$(50),H$(50)
80  PRINT CHR$(26)
90  LINE=80
100 INPUT "NAME OF MASTER FILE FROM WHICH DATA WILL BE DELETED",F$
110 F$(1)="D"+F$:IF LEN(F$(1))†8 THEN F$(1)=MID$(F$(1),1,8)
120 F$(2)="O"+F$:IF LEN(F$(2))†8 THEN F$(2)=MID$(F$(2),1,8)
130 F$(3)="T"+F$:IF LEN(F$(3))†8 THEN F$(3)=MID$(F$(3),1,8)
140 PRINT "INPUT NAME EXACTLY AS IN MASTER RECORD"
150 PRINT"WHEN JOB IS COMPLETE, DELETED NAMES WILL BE IN A FILE NAMED"
160 PRINT"               ";F$(1)
170 PRINT"ORIGINAL INFO WILL BE IN A FILE CALLED ";F$(2)
180 OPEN #1,F$,#2,F$(1),#3,F$(3)
190 PRINT"FILES READY"
200 PRINT"INPUT NAMES TO BE DELETED----DONE WHEN FINISHED"
210 PRINT"INPUT NAME EXACTLY AS IN MASTER RECORD"
220 L=1
230 INPUT"FIRST NAME THEN LAST NAME ",G$(L),H$(L)
240 IF G$(L)="DONE" THEN 280
250 IF H$(L)="DONE" THEN 280
260 L=L+1
270 GOTO 230
280 REM DELETED DEFAULT REFERENCES HERE
290 READ #1,N$,M$,A$,Y$,Q$,S$,Z$,C$
300 T(2)=T(2)+1
310 IF EOF(1)=1 THEN PRINT"END OF FILE REACHED IN FILE ";F$:GOTO 480
320 REM COMPARE NAMES
330 FOR R=1 TO L-1
340 IF G$(R)=N$ THEN IF H$(R)=M$ THEN GOSUB 410
350 IF N$="" THEN 290
360 NEXT R
370 REM WRITE RECORDS NOT FOUND TO F$(3)
380 WRITE #3,N$,M$,A$,Y$,Q$,S$,Z$,C$
390 T(1)=T(1)+1:REM COUNT RECORDS IN NEW MASTER FILE
400 GOTO 290
410 REM WRITE DELETED RECORDS TO FILE
420 WRITE #2,N$,M$,A$,Y$,Q$,S$,Z$,C$
430 PRINT N$,M$
440 PRINT A$,Y$,Q$,Z$,"DELETED"
450 G$(R)=" ":H$(R)=" ":T=T+1
460 N$="":M$=""
470 RETURN
480 REM END OF JOB PROCEDURE
490 CLOSE #1,#2,#3
500 PRINT T,"NAMES DELETED FROM FILE ";F$
510 PRINT"THESE NAMES WERE NOT FOUND "
520 FOR R=1 TO L-1
530 PRINT G$(R),H$(R)
540 NEXT R
```

```
550 T(2)=T(2)-1
560 PRINT T(2);" RECORDS READ FROM ORIGINAL FILE"
570 PRINT T(1);" NAMES NOW IN MASTER FILE ";F$
580 RENAME F$,F$(2):RENAME F$(3),F$
590 INPUT "DO YOU WANT TO DELETE MORE NAMES Y,N",V$
600 IF V$="Y" THEN 100
610 IF V$§†"N" THEN 590
620 CHAIN MONITOR
630 END
```

```
*** CHANGE.BAS ***

10 REM ***(C)COPYRIGHT 1978 BY
20 REM ***RAM ASSOCIATES
30 REM ***201 N. MAIN ST.
40 REM ***MASONTOWN, PA. 15461
50 REM ***RON MAHON, LINDA MICHALIK
60 REM ***INSERT CLEAR SCREEN HERE
70 DIM G$(50,H$(50)
80 PRINT CHR$(26)
90 LINE= 80
100 INPUT "NAME OF MASTER FILE TO WHICH DATA WILL BE CHANGED",F$
110  F$(2)="0"+F$:IF LEN(F$(2))†8 THEN F$(2)=MID$(F$(2),1,8)
120  F$(3)="T"+F$:IF LEN(F$(3))†8 THEN F$(#)=MID$(F$(3),1,8)
130 PRINT "INPUT NAME EXACTLY AS IN MASTER RECORD"
140 PRINT "CHANGED RECORDS WILL BE IN A FILE NAMED ";F$
150 PRINT "ORIGINAL INFO WILL BE IN A FILE CALLED 15;F$(2)
160 OPEN #1,F$,#3,F$(3)
170 PRINT "FILES READY"
180 PRINT "INPUT NAMES TO BE CHANGED----DONE WHEN FINISHED"
190  L=1
200 INPUT "FIRST NAME ,THEN LAST NAME ",G$(L),H$(L)
210 IF G$(L)="DONE" THEN 250
220 IF H$(L)="DONE" THEN 250
230  L=L+1
240 GOTO 200
250 READ #1,N$,M$,A$,Y$,Q$,S$,Z$,C$
260  T(2)=T(2)+1
270 IF EOF(1)=1 THEN PRINT"END OF FILE REACHED IN FILE ";F$:GOTO 490
280 REM COMPARE LINES
290 FOR R=1 TO L-1
300 IF G$(R)=N$ THEN IF H$(R)=M$ THEN GOSUB 370
310 IF N$="" THEN 250
320 NEXT R
330 REM WRITE RECORDS NOT FOUND TO F$(3)
340 WRITE #3,N$,M$,A$,Y$,Q$,S$,Z$,C$
350  T(1)=T(1)+1:REM COUNT RECORDS IN NEW MASTER FILE
360 GOTO 250
370 PRINT CHR$(26)
380 REM DISPLAY AND CORRECT FIELDS
390 PRINT "(1)";N$,"(2)";M$
400 PRINT "(3)";A$,"(4)";Y$
410 PRINT "(5)";Q$,"(6)";S$,"(7)'9;Z$,"(8)";C$
420 INPUT "WHAT FIELD DO YOU WISH TO CHANGE",M
430 ON M GOSUB 580,630,680,730,780,830,880,930
440 INPUT "DO YOU WISH TO CHANGE ANOTHER FIELD IN THIS RECORD ",V$
450 IF V$="Y" THEN 370
460 IF V$§†"N" THEN 440
470 GOTO 330
480 END
490 CLOSE #1,#3
500  T(2)=T(2)-1
510 PRINT T(2);" RECORDS READ FROM ORIGINAL FILE"
520 PRINT T(1);" NAMES NOW IN MASTER FILE ";F$
530 RENAME F$,F$(2):RENAME F$(3),F$
540 INPUT "DO YOU WANT TO CHANGE MORE RECORDS Y/N",V$
```

*** CHANGE.BAS - PAGE TWO ***

```
550 IF V$="Y" THEN 100
560 IF V$§†"N" THEN 540
570 CHAIN MONITOR
580 INPUT "FIRST NAME",N$
590 PRINT N$:INPUT "CORRECT Y/N ",V$
600 IF V$="Y" THEN RETURN
610 IF V$§†"N" THEN 590
620 GOTO 580
630 INPUT "LAST NAME",M$
640 PRINT M$:INPUT "CORRECT Y/N ",V$
650 IF V$="Y" THEN RETURN
660 IF V$§†"N" THEN 640
670 GOTO 630
680 INPUT "ADDRESS",A$
690 PRINT A$:INPUT "CORRECT Y/N ",V$
700 IF V$="Y" THEN RETURN
710 IF V$§†"N" THEN 690
720 GOTO 680
730 INPUT "ADDITIONAL ADDRESS",Y$
740 PRINT Y$:1NPUT "CORRECT Y/N ",V$
750 IF V$="Y" THEN RETURN
760 IF V$§†"N" THEN 740
770 GOTO 730
780 INPUT "CITY",Q$
790 PRINT Q$:INPUT "CORRECT Y/N ",V$
800 IF V$="Y" THEN RETURN
810 IF V$§†"N" THEN 790
820 GOTO 780
830 INPUT "STATE",S$
840 PRINT S$:INPUT "CORRECT Y/N ",V$
850 IF V$="Y" THEN RETURN
860 IF V$§†"N" THEN 840
870 GOTO 830
880 INPUT "ZIP CODE",Z$
890 PRINT Z$:INPUT "CORRECT Y/N ",V$
900 IF V$="Y" THEN RETURN
910 IF V$§†"N" THEN 890
920 GOTO 880
930 INPUT "SERVICE CODE",C$
940 PRINT C$:INPUT "CORRECT Y/N ",V$
950 IF V$="Y" THEN RETURN
960 IF V$§†"N" THEN 940
970 GOTO 930
```

*** NAMAD.BAS ***

```
10 REM FILENAME NAMAD
20 REM NAME-ADDRESS-PHONE NUMBER LIST
30 REM USES NAME.DAT AS LIST, SCRATCH.DAT AS WORKING FILE
40 LINE= 0
50 PRINT CHR$(26):PRINT"COMMAND LIST":PRINT
60 PRINT "CHANGE LIST","C"
70 PRINT "FIND LISTING","F"
80 PRINT "LIST FILE","L"
90 PRINT "LIST, NO PHONE","N"
100 PRINT "SELECTIVE COPY","S"
110 PRINT "SORT","SO"
120 INPUT R$
130 IF R$="C" THEN 280
140 IF R$="F" THEN 200
150 IF R$="L" THEN 870
160 IF R$="N" THEN 1120
170 IF R$="S" THEN 1140
180 IF R$="SO" THEN 1470
190 REM FIND A LISTING
200 INPUT "ENTER KEYWORD, LAST NAME FIRST, NO SPACE,IE. JONESJOHN",R$
210 GOSUB 1370:REM FILESEARCH
220 IF E=1 THEN PRINT"NAME NOT FOUND":PRINT
230 CLOSE #1
240 INPUT "COMMAND LIST",Z$
250 IF Z$="Y" THEN 50
260 END
270 REM CHANGE A LISTING
280 INPUT "ADD (A), CHANGE OR DELETE (C)",Z$:F=0
290 OPEN #1,NAME.DAT
300 OPEN #2,SCRATCH.DAT
310 IF Z$="C" THEN INPUT"KEYWORD FOR RECORD TO BE CHANGED",R$
320 IF Z$="A" THEN 640
330 REM CHANGE A RECORD
340 READ #1,I$,N$,A$
350 READ #1,C$,P$
360 IF EOF(1)=1 THEN 410
370 IF I$=R$ THEN 500
380 WRITE #2,I$,N$,A$
390 WRITE #2,C$,P$
400 GOTO 340
410 IF F=0 THEN PRINT "FILE NOT FOUND":CLOSE#1
420 IF F=0 THEN CLOSE #2
430 IF F=0 THEN KILL SCRATCH.DAT
440 IF F=0 THEN GOTO 240
450 IF F=1 THEN PRINT"DONE":CLOSE#1
460 IF F=1 THEN CLOSE #2
470 IF F=1 THEN KILL NAME.DAT
480 IF F=1 THEN RENAME SCRATCH.DAT,NAME.DAT
490 GOTO 240
500 INPUT "DELETE",Z$:F=1
510 IF Z$="Y" THEN 340
520 INPUT "CHANGE KEYWORD",Z$
530 IF Z$="Y" THEN INPUT"NEW KEYWORD",I$
540 INPUT "CHANGE NAME",Z$
```

```
*** NAMAD.BAS - PAGE TWO ***

550 IF Z$="Y" THEN INPUT"NEW NAME",N$
560 INPUT "CHANGE ADDRESS",Z$
570 IF Z$="Y" THEN INPUT"NEW ADDRESS",A$
580 INPUT "CHANGE CITY-ZIP",Z$
590 IF Z$="Y" THEN INPUT"NEW CITY",C$
600 INPUT "CHANGE PHONE",Z$
610 IF Z$="Y" THEN INPUT"NEW PHONE",P$
620 GOTO 380
630 REM ADD A NAME
640 INPUT "KEYWORD",J$
650 INPUT "NAME",O$ .
660 INPUT "ADDRESS",B$
670 INPUT "CITY" D$
680 INPUT "PHONE",Q$
690  S=0
700 READ #1,I$,N$,A$
710 READ #1,C$,P$
720 IF EOF(1)=1 THEN 780
730 IF S=1 THEN 750
740 IF LEFT$(I$,4)tLEFT$(J$,4)THEN WRITE#2,J$,O$,B$:WRITE#2,D$,Q$:S=1
750 WRITE #2,I$,N$,A$
760 WRITE #2,C$,P$
770 GOTO 700
780 CLOSE #1,#2
790 KILL NAME.DAT
800 RENAME SCRATCH.DAT,NAME.DAT
810 INPUT "MORE ENTRIES",Z$
820 IF Z$="Y" THEN OPEN #1,NAME.DAT
830 IF Z$="Y" THEN OPEN #2,SCRATCH.DAT
840 IF Z$="Y" THEN 280
850 GOTO 240
860 REM LIST THE FILE
870  Q=0
880 OPEN #1,NAME.DAT
890 INPUT "PRINTER (Y-N)",Z$
900 IF Z$="Y" THEN PORT=7
910  L=0:PRINT:PRINT:PRINT:PRINT:PRINT
920 PRINT TAB(16);"NAME AND ADDRESS LIST PG. ";"1":P1=2:PRINT
930 READ #1,I$,N$,A$
940 READ #1,C$,P$
950 IF EOF(1)=1 THEN 1080
960 READ #1,J$,O$,B$
970 READ #1,D$,Q$
980 IF EOF(1)=1 THEN 1080
990 PRINT TAB(7);N$;TAB(47);O$:L=L+1
1000 PRINT TAB(7);A$;TAB(47);B$:L=L+1
1010 PRINT TAB(7);C$;TAB(47);D$:L=L+1
1020 PRINT :L=L+1
1030 IF Q=0 THEN PRINT TAB(7);P$;TAB(47);Q$:PRINT:L=L+2
1040 IF L†=50 THEN PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
1050 IF L†=50 THEN PRINT TAB(16);"NAME AND ADDRESS PG. ";P1:P1=P1+1
1060 IF L†=50 THEN L=0
1070 PRINT :GOTO 930
1080 CLOSE #1
```

```
1090 PORT= 1:GOTO 240
1100 REM LIST, NO PHONE
1110 REM SET PHONE SWITCH OFF
1120  Q=1:GOTO 880
1130 REM SELECTIVE COPY, PROMPTED
1140 OPEN #l,NAME.DAT
1150 OPEN #2,COPY.DAT
1160  E=0
1170 READ #1,I$,N$,A$
1180 READ #1,C$,P$
1190 IF EOF(1)=1 THEN 1240
1200 PRINT I$:PRINTN$:PRINTA$:PRINTC$:PRINTP$
1210 INPUT "COPY", Z$
1220 IF Z$="Y" THEN WRITE #2,I$,N$,A$:WRITE #2,C$,P$
1230 GOTO 1170
1240 CLOSE #1
1250 CLOSE #2
1260 INPUT "LIST COPY FILE",Z$
1270 IF Z$="Y" THEN INPUT"PRINTER",Z$
1280 IF Z$="Y" THEN PORT=7
1290 OPEN #2,COPY.DAT
1300 READ #2,I$,N$,A$:READ #2,C$,P$
1310 IF EOF(2)=8 THEN 1340
1320 PRINT N$:PRINTA$:PRINTC$:PRINTP$:PRINT
1330 GOTO 1300
1340 CLOSE #2
1350 PORT= 1:GOTO 240
1360 REM SUBROUTINE FOR FIND A LISTING
1370 OPEN #1,NAME.DAT
1380  E=0:F=0
1390 READ #1,I$,N$,A$:READ #1,C$,P$
1400 IF I$=R$ THEN F=1:GOTO 1430
1410 IF EOF(1)=1 THEN E=1:RETURN
1420 GOTO 1390
1430 PRINT N$:PRINTA$:PRINTC$:PRINT:PRINTP$:PRINT:RETURN
1440 REM SORT A FILE BY NAME - BUBBLE SORT
1450 REM MUST READ AND WRITE ENTIRE FILE UP TO N TIMES
1460 REM FOR N ITEMS SORTED.
1470 OPEN #1,NAME.DAT
1480 OPEN #2,SCRATCH.DAT
1490  S=0:G=0
1500 READ #1,I$,N$,A$:READ #1,C$.P$
1510 READ #1,I$(1),N$(1),A$(1):READ #1,C$(1),P$(1)
1520 IF LEFT$(I$,4)tLEFT$(I$(1),4) THEN G=1
1530 IF G=1 THEN WRITE #2,I$(1),N$(1),A$(1):WRITE #2,C$(1),P$(1)
1540 IF G=1 THEN READ #1,I$(1),N$(1),A$(1):READ #1,C$(1),P$(1)
1550 IF G=0 THEN WRITE #2, I$,N$,A$:WRITE#2,C$,P$
1560 IF G=0 THEN READ #1,I$,N$,A$:READ #1,C$,P$
1570 IF G§†T THEN S=1
1580 IF G=0 THEN T=1
1590 IF G=1 THEN T=0
1600 IF EOF(1)=1 THEN 1620
1610  G=0: GOTO 1520
1620 IF G=1 THEN WRITE #2,I$,N$,A$:WRITE #2,C$,P$
```

```
1630 IF G=0 THEN WRITE #2,I$(1),N$(1),A$(1):WRITE #2,C$(1),P$(1)
1640 CLOSE #1,#2
1650 KILL NAME.DAT
1660 RENAME SCRATCH.DAT,NAME.DAT
1670 IF S=1 THEN 1490
1680 GOTO 240
```