```
=======================================
=                                     =
=      HP 2000 PART 4 (FILES)         =
=                                     =
=             CAPTURED BY             =
=                                     =
=      BLITZIOD ?? & GALACTUS **      =
=                                     =
=                 of                  =
=                                     =
=       THE ELITE HACKERS GUILD       =
=                                     =
=======================================
```

* FILES *

BASIC FORMATTED FILES ARE ESSENTIALLY THE SAME AS
DATA STATEMENTS, THEY BOTH HAVE POINTERS THAT MOVE ALONG THE DATA
HERE IS AN EXAMPLE OF A PROGRAM USING THE DATA STATEMENT   :

```
10 READ X
20 PRINT X
30 DATA 1,2,3,4,5,6,7,8,9,10
40 GOTO 10
50 END
```

WHEN THIS PROGRAM IS RUN THE DATA IS READ IN LINE 10
FROM THE DATA STATEMENT IN LINE 30.
AFTER THE '1' IS READ FROM THE DATA STATEMENT THE POINTER IS
MOVED TO THE '2' AND SO ON. WHEN THE FINAL PIECE OF DATA IS READ
AND THE POINTER IS MOVED BEYOND THE '10' THEN YOU WILL GET THE
ERROR MESSAGE: OUT OF DATA IN LINE 10
THIS IS ESSENTIALLY THE WAY FILES WORK.
BUT FILES HAVE MANY MORE CAPABILITIES THAN DO DATA STATEMENTS

LET'S LEARN HOW TO CREATE A FILE, OK?

TO CREATE A FILE ALL ONE MUST DO IS TYPE IN CRE- THEN THE
NAME OF THE FILE, AND HOW LONG IT MUST BE. FOR EXAMPLE IF I WANTED
TO CREATE A FILE NAMED 'DAVID' THAT IS 5 RECORDS LONG I WOULD TYPE IN

```
CRE-DAVID,5
```

THEN HIT 'RETURN'

WHENEVER YOU CREATE A FILE THE COMPUTER RESERVES THE NUMBER OF RECORDS YOU CREATED IT ON THE DISC. SO IF YOU CREATE A FILE THAT IS 50 RECORDS LONG, YOU HAVE USED 50 RECORDS OF DISC SPACE, WHETHER YOU USE THE WHOLE FILE OR NOT.

NOW I WILL SHOW YOU A PROGRAM THAT WILL PRINT ON YOUR FILE.

```
 5 DIM A$(20), B$(20)
10 FILES FIL1
20 PRINT 'NAME';
30 INPUT A$
40 IF END#1 THEN 85
50 READ#1,1
60 READ#1;B$
70 GOTO 60
85 IF END#1 THEN 110
90 PRINT#1;A$,END
100 STOP
110 PRINT'FILE FULL'
120 END
```

BY THIS TIME YOU ARE WONDERING WHAT THAT MESS IS, RIGHT ?
HERE IS AN EXPLANATION

```
STATEMENT #                       MEANING
10          OPENS THE FILE THAT IS TO BE USED, IN THIS CASE THE NAME OF
            THE FILE IS 'FIL1'.  THIS STATEMENT IS TELLING THE COMPUTER
            WHAT FILE WE WANT TO USE FOR THIS PROGRAM
            FIL1 IS REFERRED TO AS #1 AS SEEN IN LINES 40,50 & 60
            IF MORE THAN ONE FILE WAS USED LINE 10 WOULD LOOK LIKE
            THIS  FILES FIL1,FIL2.  THEN FIL2 WOULD BE REFERRED
            TO AS #2.  UP TO 16 FILES CAN BE USED IN ONE PROGRAM
30          INPUT NEW NAME TO BE PRINTED ON FILE.
40          THIS STATEMENT 'IF END' SETS UP THE IF END
            CONDITION.  THIS MEANS THAT WHEN THE FILE PIONTER GETS
            TO THE END OF  DATA IN THAT FILE THEN GOTO 85
50          THIS MEANS TO READ THE FIRST RECORD AND THE
            FIRST ITEM OF THE FILE, IN OTHER WORDS THIS
            THIS SETS THE FILE POINTER BACK TO THE BEGINNING OF THE
            DATA.
            BY USING THIS STATEMENT YOU CAN BE ASSURED OF STARTING
            AT THE BEGINNING OF THE DATA IN THAT FILE
60          READS DATA FROM FILE#1, SOMEWHAT LIKE THE 'READ' USED
            WITH DATA STATEMENTS.
85          SETS UP ANOTHER IF END CONDITION TO CHECK FOR THE END OF
            FILE.  IF ANOTHER EOF MARKER IS ECOUNTERED THEN GOTO
            LINE 110.
            END-OF-FILE MARKERS ARE DEFINED MORE PRECISELY LATER.
90          THIS STATEMENT PRINTS THE NEW NAME (A$) ON THE FILE,
            THEN PRINTS AN END OF FILE MARKER ON THE FILE.
            THE END OF FILE MARKER ENABLES THE COMPUTER TO TELL
            THROUGH THE READ STATEMENT (LINE 60) WHEN THE END OF
            DATA IS REACHED
```

HERE IS A FLOW CHART OF THAT PROGRAM:

```
1.  INPUT NEW NAME -----------2.  READ DATA FROM FILE

3.  IF END OF FILE #1 THEN 4 -----GO BACK TO 2.
                                 !
                    4.  IF END#1 THEN 7
                                 !
5.  TRY TO PRINT NEW NAME ON FILE, IF SUCCESSFUL (MEANING FILE IS
    NOT FULL) THEN 6.  IF UNSECCESSFUL (MEANING FILE IS FULL) THEN 7.
                                 !                            !
                                                              !
                                 !            7.  PRINT'FILE FULL'
                             6.  STOP                         !
                                                          8. END
```

OK, LET'S MOVE ON TO GET A LISTING OF YOUR FILE

```
05 DIM A$(20)
10 FILES FIL1
20 IF END#1 THEN 70
30 READ#1,1
40 READ#1;A$
50 PRINT A$
60 GOTO 40
70 END
```

THIS IS A RATHER SIMPLE PROGRAM, IT GOES LIKE THIS;
YOU SET UP THE IF END CONDITION, THEN READ STARTING AT THE
BEGINNING OF THE FILE. READ THROUGH THE FILE AND PRINT
EACH SEPARATE PIECE OF DATA (LINES 40 & 50). WHEN ALL THE

DATA HAS BEEN READ THROUGH, IT FALLS THROUGH TO LINE 70, BY THE CONDITION SET UP IN LINE #20 AND AT THIS POINT EXECUTION IS TERMINATED.

* END OF FILE DISCUSSION *

HERE IS SOME ELLABORATION ON WHAT YOU HAVE LEARNED SO FAR ABOUT BASIC FORMATTED FILES.

THE EOF MARKER IS WHAT THE IF END CONDITION IS USED WITH. THE EOF MARKER DESIGNATES THAT THE END OF DATA HAS BEEN REACHED. AS MORE DATA IS PUT INTO A FILE.