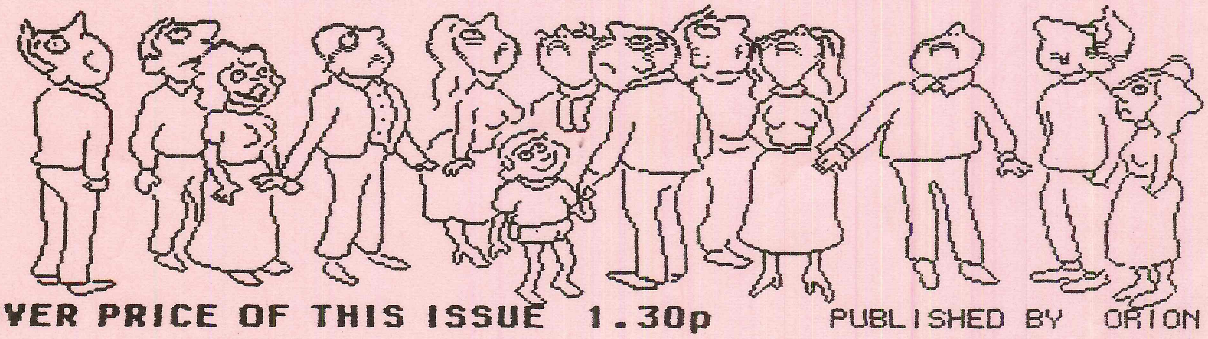
MEMOPAD

THE OFFICIAL USER MAGAZINE OF M.C.L

FOR MEMOTECH COMPUTER USERS WORLD WIDE

CONTENTS

VOLUME THREE ISSUE NUMBER 11



COVER PRICE OF THIS ISSUE 1.30p PUBLISHED BY ORION

Volume Three **MEMOPAD** Number 11

Extending Your Basic

This article is written to provide an insight into how you can add your own Basic extentions to the Memotech MTX (and RS) series of computers through the USER commands.

The program with this article will provide two new commands that will allow direct manipulation of the video RAM from basic (allowing you to read sprite positions, character definitions etc..)

The new commands are :-

USER POKE <address>, <byte> and
USER PEEK (<address>), <variable name>

Note that the PEEK command has a different format than usual. This is to allow a value to be returned by the machine code routine; but more of that later.

USER COMMANDS:

MTX basic allows for new commands in the form of:
USER <name>,<parameter>, <parameter>... etc
where <name> can be any basic keyword, or even a new word of your own. If you use a new word (Eg: FILL) your routine will need to check each character in turn; but when using a keyword it will be tokenised as the line is entered, and so only one byte must be checked. (Note: £E5 = PEEK, £AD = POKE)

When the MTX ROM encounters your new command it will provide some syntax checking depending on what has been loaded into locations £FA88 down to £FA86. The most common syntax bytes are:

0 - Numeric expression
1 - String expression
2 - Arithmetic expression
3 - List of expressions separated by "," or ";"
4 - List of numbers separated by ","
5 - List of arithmetic expressions
6 - Single number from 0-65535
7 - Check nothing ..... (most common)

Although other values can also be used.
If no errors are found then the ROM will jump to your new routine through a vector at £FA89, so the format is:

£FA85	- Return byte
£FA86 to £FA88	- Syntax checking bytes (loaded from £FA88 downwards)
£FA89	- jump (£C3)
FA8A/£FA8B	- adress of new routine (LSB/MSB)

PASSING VALUES

When the MTX finally enters your new routine the register pair DE will point to the character that follows the USER statement. Your routine can use this to read the bytes after the USER statement to obtain values. The easiest way to do this is to use the RST 3(instruction. On the MTX when DE is pointing to a variable name (or number) the RST 30 instruction will return with the BC register containing the value of the variable, DE pointing to the terminating character +1 and the zero flag set. If an error is detected the zero flag will be reset (ie: if NZ then error).

This means that by repeated use of RST 30 your routine can obtain as many values as required. Although obviously you are restricted to integers in the range 0-65535.

RETURNING VALUES

The MTX does not allow for a USER statement to be given a value (Eg: LET X=USER PEEK..) but this can easily be overcome by a call to the MTX ROM. There is a routine at £29DA (the ROM LET routine) that is used by basic when it defines a variable, so by using this your routine can return as many values as required (hence the odd format for USER PEEK).

USING THE PROGRAM

The listing should be entered using the built-in assembler and then saved to tape before running. This is because the program moves itself into the common RAM page (£C000 - £FFFF), this is essential if it is to be used on machines with more than 32K RAM.

Once run the program will NEW itself and a basic program can then be entered that uses the new command. It may be of use to know that the MTX configures VRAM as follows:

```
 ___________________________________
|                                   |--- £3FFF - 16383
| Sprite attributes                 |
|___________________________________|--- £3F00 - 16128
| ASCII name table (graphics)       |
|___________________________________|--- £3C00 - 15360
| Sprite patterns                   |
|___________________________________|--- £3800 - 14336
| Graphics colour table             |
|___________________________________|--- £2000 - 8192
| ASCII name table (text)           |
|___________________________________|--- £1C00 - 7168
| Text pattern table                |
|___________________________________|--- £1800 - 6144
| Graphics pattern table            |
|___________________________________|--- £0000 - 0
```

For more information you should refer to appendix 6 in the MTX manual.

ROM CALLS:

RST 30:

Enter with DE pointing to data
Returns with BC=value of data, and zero flag reset if an error is detected.

£0DD0:

Enter with BC = value
Returns with DE pointing to the ASCII equivalent of the value (terminating with £FF)

£29DA:

Enter with DE pointing to variable definition (terminating with £FF)
Returns after defining variable.
Note tokenised form of "=" is £D4

£FA54

Vector for the MTX error handling routine in ROM (at £18AF)
Enter with A=error code.

I hope this has proved of some use to other Memotech owners and I am sure you will agree that this is a very powerful feature of MTX basic.

Number 11

MEMOPAD

Volume Three

REVIEW REVIEW REVIEW REVIEW REVIEW REVIEW REVIEW

FOOTBALL MANAGER REVIEW by Adrian Holt

Tape	-	£6.95
CP/M & all 3.5" formats	-	£9.95
CP/M & all 5.25" formats	-	£8.25

Football Manager is a game of strategy and skill. The aim of the game is to take your selected team to the top of the first division and ultimately become league champions.

You must first select the level of play you desire, there are seven skill levels available :- BEGINNER, NOVICE, AVERAGE, GOOD, EXPERT, SUPER EXPERT and GENIUS. Pick your team from a selection of sixty four. Starting at the bottom of the fourth division you must play fifteen matches in a season, at the end of the season if you are one of the top three teams in your division you will be promoted to the next division and receive your league success bonus.

Entry for the F.A Cup is automatic and it is possible, as in the real F.A. Cup, to win this even if you do not win the league, if you do win you will be rewarded by an excellent display of the Cup while the crowds cheer.

You are allowed to top your money up by obtaining a loan. The size of the loan will depend upon which league you are in. Exceeding this loan results in you being sacked -so be careful. Another nice touch is the facility to display the league table so that you can see the number of points you have, goals scored - for and against. You can also change your team and player names so you can be a player/manager of your favourite team.

If you are finding the skill level you selected too easy you can change to a more difficult level at any time during the game, on the other hand, if the going is a little too demanding you can go down to a lower level - I found this useful!

Having selected your team, it is possible to change players for each game according to the strategy you want to adopt. You are presented with the three dimensional highlights of the game, which are very fast and far more realistic in comparison to the original Amstrad version. After each game the players energy and morale values are updated in relation to result.

Should you be forced to leave the game for some reason it is possible to save your game to tape or disc, so on returning to the game you can start where you left off - handy eh?

This is a must for football fans, business game fanatics or anyone who likes a challenge. Addiction rating is very high and playability will keep you coming back for more.

A.I. AFTERTHOUGHT

Well, here I am again, which only goes to prove what an excellent choice of editor our Keith has made. Still, down to business as they say, and what better place to start than with a couple of weaknesses which we were left with in the previous article.

Firstly, the problem with variable adjustment. Namely, to allow it to change quickly yet at the same time in an accurate manner. At this point you may be asking why it is necessary to able to control accurately what is supposed to be a random number. The point is, we are not going to control the random number, merely it's range. The reason for this is so that the variable can be quickly adjusted to the value that best enables it to play effectively. It starts off with a large range which allows for rapid slewing across the range. Then as it closes in on the best value, the range reduces so that time is not wasted in trying larger values than necessary. But what is the point? after all, isn't the whole idea that it should find the best position anyway,in time? Yes, it is but 'saving time' is the real point. If the program can be made to achieve optimum performance that much quicker, then if the optimum value changes, maybe due to some change in the playing style by its opponent, then it can adjust itself to the change that much more quickly, so maintaining it's effectiveness as an opponent. OK, so how is it done? Well the way I did it was to make the range of random values chosen vary according to how effectively the program chooses values that improve itself. So as long as they improve the effectiveness of it's play, the range of values stays large, but when it produces a series that do not improve the situation, as it will when the value approaches optimum, it reduces the range of the values of the numbers produced, so that the probability of obtaining a value that does improve the situation rises. Suppose C is twice the maximum range over which random numbers are chosen, and TOT is the total number of random values tried with the selected variable, and GOOD is the total number of those that improve it. We can change line 8030 to :-

```
8030 LET VA = RND x (GOOD/TOT) x 2 x C
```

It works as follows. Normally the number of GOOD choices should be around half the total number of TOT choices, but as the variable approaches the optimum position, TOT will increment several times while GOOD stays static, so GOOD/TOT will fall and so reduce VA.

```
22      DIM GOOD (16), TOTAL (16)
8015    LET  TOTAL  (POS)  =  TOTAL  (POS)  +  1  :  REM  TRY  ANOTHER
        CHANGE
8025    LET GOOD = GOOD (POS) : LET TOT = TOTAL (POS)
8030    LET VA = RND X (GOOD/TOT) X 2 X 2000
8110    IF RESULT > -1 THEN LET GOOD (POS) = GOOD (POS) +1
```

At this point, you may like to try changing the 2000 in line 8030 to another value, say 1000 or 500 or how about making it proportional to MEM. Those of you who know statistics may care to try VA with a normalised distribution and let GOOD\TOT control the standard deviation.

Now, what about the way we check for performance improvement. What can we do to improve that? Some sort of time factor, so that the number of moves needed is taken into consideration? Well, yes but if we give it a pre-set number of moves to win in, and within which improvements are allowed but after which none are allowed, seems little better than the present situation, while having to beat it's previous best would be far to much of a restriction. So what then? ... Wait a bit, how about letting the program decide for itself. After all, it is supposed to be intelligent. We could use a variable to indicate the number of moves within which a win must be made to qualify as an improvement, like that first improvement, like that first idea. Only instead of being preset, the program works the value out for itself. Now the minimum moves for a win is 4, while the maximum is 32, a range of only 28. So a simple offset of + or -1 chosen at random should control it.

```
23      LET SPEED = 32: REM IMPROVEMENT QUALIFYING MOVES (MAX)
1505    LET TEMPO = 0: REM PLAYER STARTS
1515    LET TEMPO = 1 REM PROGRAM STARTS
6005    LET TEMPO = TEMPO +1 : REM ANOTHER MOVE
8010    LET POS = INT (RND X 16) +1 : REM EXTRA VARIABLE (SPEED)
8012    IF POS = 17 THEN LET VA = INT (RND X 2) -1 : GOTO 8070
8060    GOTO 8090
8070    LET SPEED = SPEED + VA
8072    IF SPEED > 32 THEN LET SPEED = 32 : LET VA = 0
8074    IF SPEED < 4 THEN LET SPEED = 4 : LET VA = 0
8105    IF TEMPO > SPEED THEN LET RESULT = -1
8125    IF POS = 17 THEN LET SPEED = SPEED - VA : GOTO 8200
8520    REM YOU NOW NEED TO RESTORE AND SAVE SPEED AS WELL
```

Once Keith has converted Connect Four, you may like to convert these routines yourself. It then becomes possible to modify the program so that it can play against itself, old variable against new one (MEM v MEM + VA), over and over again, allowing you to try out different ideas over a number of games, unattended.

I think that this should be enough to be going on with, so I will sign off and I hope you found it interesting, I would be interested to hear from anyone who has any ideas for improvements or changes.

Volume Three **MEMOPAD** Number 11

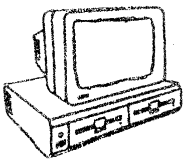
Database

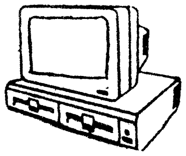
Here is another suite of programs from Dave Wemyss comprising of seven separate files which combine together to make a complete Database.

```
10 REM ****************************** MAINMENU.BAS ******************************
20 REM ****************************** VERSION 2 ******************************
30 REM ****************************** D.WEMYSS ******************************
40 REM ****************************** OCT 1986 ******************************
50 USER  SAVE "MAINMENU.BAS"
60 VS 5: CLS : CLEAR : DIM F$(7,14),D$(7,14),CONF$(3)
70 LET F$(1)="F1............": LET F$(2)="F2............": LET F$(3)="F3............": LET F$(4)="F4............": LET F$(5)="F5............": LET F$(6)="F6..
..........": LET F$(7)="F7............"
80 LET D$(1)="Start New File": LET D$(2)="Update File": LET D$(3)="Viev File": LET D$(4)="Search File": LET D$(5)="Sort File": LET D$(6)="Print File": LET D$(
7)="End Session"
90 PLOD "PROG1": CLS : CSR 30,0: PRINT "MAIN MENU": CSR 29,1: PRINT "==========="
95 LET N=1
100 FOR I=4 TO 22 STEP 3
110 CSR 23,I: PRINT F$(N);D$(N)
120 LET N=N+1: NEXT I
130 LET A$=INKEY$: LET A=ASC(A$)
140 IF A<128 OR A>134 THEN  GOTO 130
150 CLS : CSR 25,10
160 PAUSE 100
170 IF A=128 THEN  GOTO 260
180 IF A=129 THEN  GOTO 280
190 IF A=130 THEN  GOTO 300
200 IF A=131 THEN  GOTO 320
210 IF A=132 THEN  GOTO 340
220 IF A=133 THEN  GOTO 360
230 IF A=134 THEN  INPUT "Type 'YES' to quit   ";CONF$
240 IF CONF$<>"YES" THEN  CLS : GOTO 90
250 NEW
260 PRINT "New File Module loading": GOSUB 380
270 USER  LOAD "INPUTDAT.BAS"
280 PRINT "Update Module loading": GOSUB 380
290 USER  LOAD "UPDATE.BAS"
300 PRINT "Viev File Module loading": GOSUB 380
310 USER  LOAD "VIEWFILE.BAS"
320 PRINT "Search File Module loading": GOSUB 380
330 USER  LOAD "SEARCH.BAS"
340 PRINT "Sort Module loading": GOSUB 380
350 USER  LOAD "SORT.BAS"
360 PRINT "Print Module loading": GOSUB 380
370 USER  LOAD "PRINT.BAS"
380 CSR 25,12: PRINT "Please vait.............."
390 RETURN
10 REM *********************************************************************
20 REM ***************************** INPUTDAT.BAS" ******************************
30 REM ****************************** VERSION 2 ******************************
40 REM ****************************** D.WEMYSS ******************************
50 REM ****************************** OCT 1986 ******************************
60 REM *********************************************************************
70 USER  SAVE "INPUTDAT.BAS"
80 VS 5: CLS : CLEAR : DIM F$(2,14),A$(3),B$(30),FILE$(12),H$(30),O$(8),D$(2,20)
90 LET B$="": LET R=0
100 LET F$(1)="F1............": LET F$(2)="F2............": LET D$(1)="Open nev file": LET D$(2)="Return to Main Menu"
110 CSR 30,0: PRINT "DATA INPUT": CSR 29,1: PRINT "============"
```

```
120 CSR 22,10: PRINT F$(1);D$(1): CSR 22,13: PRINT F$(2);D$(2)
130 LET A$=INKEY$: LET A=ASC(A$)
140 IF A<128 OR A>129 THEN  GOTO 130
150 IF A=128 THEN  GOTO 180
160 CLS : CSR 30,10: PRINT "Returning to Main Menu": CSR 30,12: PRINT "Please wait.................."
170 USER  LOAD "MAINMENU.BAS"
180 CLS : CSR 10,5: PRINT "Name of datafile? (Max 8 chars)";: INPUT "  >";B$
190 IF LEN (B$)>8 THEN  GOTO 180 ELSE  LET O$=B$
200 CLS : CSR 10,5: PRINT "Maximum number of records =80"
210 CSR 10,7: INPUT "Number of headings? (Max 10) ";NH
220 IF NH<1 OR NH>10 THEN  GOTO 210
230 DIM HEAD$(NH,8),RECORD$(80,NH,25),R$(30)
240 CLS : CSR 36,0: PRINT O$: CSR 35,1: PRINT "=========="
250 CSR 10,5: PRINT "Enter headings (Max 8 chars)"
260 FOR X=0 TO 79: PRINT "-";: NEXT X
270 FOR I=1 TO NH
280 LET B$="          ": CSR 10,I+6: PRINT I;".": CSR 16,I+6: INPUT "> ";B$: IF LEN (B$)>8 THEN  GOTO 280
290 IF B$="" THEN  GOTO 280
300 LET HEAD$(I)=B$: NEXT I
310 CLS : CSR 36,0: PRINT O$: CSR 35,1: PRINT "=========="
320 FOR I=1 TO NH
330 CSR 10,I+4: PRINT I;".": CSR 15,I+4: PRINT HEAD$(I)
340 NEXT I
350 CSR 5,22: PRINT "Are these headings O.K.? (Y/N)": GOSUB 360: GOTO 390
360 LET A$=INKEY$: IF A$<>"" THEN  GOTO 360
370 LET A$=INKEY$: IF A$="" THEN  GOTO 370
380 IF A$<>"N" AND A$<>"n" AND A$<>"Y" AND A$<>"y" THEN  GOTO 360
385 RETURN
390 IF A$="Y" OR A$="y" THEN  GOSUB 660: GOTO 460
400 CSR 5,22: INPUT "Which heading to change?       >";F
410 IF F<1 OR F>NH THEN  GOTO 400
420 CLS : CSR 10,10: PRINT F;".  ";HEAD$(F)
430 LET B$="          ": CSR 10,12: INPUT "New heading (Max 8 chars) ";B$
440 IF B$="" OR LEN (B$)>8 THEN  GOTO 430 ELSE  LET HEAD$(F)=B$
450 GOTO 310
460 USER OPEN£1,O$,"O"
470 LET H$=O$+" Record No. "+STR$(R+1)
480 CLS : CSR 25,0: PRINT H$
490 FOR J=24 TO 26+LEN (H$): CSR J,1: PRINT "=";: NEXT J
500 FOR I=1 TO NH: CSR 10,I+3: PRINT HEAD$(I): CSR 18,I+3: PRINT "> ": NEXT I
510 CSR 10,20: PRINT F$(1);"Enter data": CSR 50,20: PRINT F$(2);"Return to menu"
520 LET A$=INKEY$: LET A=ASC(A$)
530 IF A<128 OR A>129 THEN  GOTO 510
540 IF A=129 THEN  GOTO 640
550 LET R=R+1
560 FOR I=1 TO NH: LET R$=""
570 CRVS 0,3,21,I+3,25,1,80
580 VS 0: EDITOR R$: IF R$="" OR LEN (R$)>25 THEN  GOTO 580 ELSE  LET RECORD$(R,I)=R$
590 NEXT I: GOSUB 800
600 FOR I=1 TO NH
610 USER  PRINT £1,RECORD$(R,I)
620 NEXT I
630 VS 5: GOTO 470
640 USER CLOSE£1
650 VS 5: CLS : GOTO 110
660 LET FILE$=O$+".HDS"
670 USER OPEN£1,FILE$,"O"
680 FOR I=1 TO NH
690 USER  PRINT £1,HEAD$(I)
700 NEXT I: USER CLOSE£1
710 RETURN
800 VS 5: CLS : CSR 35,0: PRINT O$: CSR 34,1: PRINT "=========="
```

```
810 FOR X=1 TO NH: CSR 20,X+4: PRINT HEAD$(X);":": CSR 30,X+4: PRINT RECORD$(R,X): NEXT X
820 CSR 20,20: PRINT "Are these entries O.K.? (Y/N)": GOSUB 360
830 IF A$="y" OR A$="Y" THEN  RETURN
840 CSR 20,20: INPUT "Which line do you wish to change? ";L: IF L<1 OR L>NH THEN  GOTO 840
850 CLS : CSR 20,10: PRINT RECORD$(R,L)
860 LET B$="": CSR 20,12: INPUT "New Record > ";B$: IF LEN (B$)>25 THEN  GOTO 860 ELSE  LET RECORD$(R,L)=B$
870 GOTO 800
10 REM **************************************************************************
20 REM ****************************** UPDATE.BAS ********************************
30 REM ****************************** VERSION 2  *******************************
40 REM ****************************** D. WEMYSS  ********************************
50 REM *****************************  OCT 1986  *******************************
60 REM **************************************************************************
70 USER  SAVE "UPDATE.BAS"
80 CLEAR : VS 5: CLS
90 CSR 20,5: PRINT "Please wait.............": CSR 20,7: PRINT "Setting up variables.............."
100 DIM H$(30),CONF$(3),FILE$(2,12),HEAD$(10,8),F$(4,14),RECORD$(80,10,25),EMP$(25),C$(2,20),D$(4,19),P$(12),O$(8),L$(12),B$(30)
110 LET W=0: LET T=0: LET Y=0: LET Z=0: LET L$="": LET EMP$="                         ": LET B$=""
120 CLS : CSR 10,5: PRINT "Which file to work? (Max 8 chars)";: INPUT " > ";L$: IF LEN (L$)>8 THEN  GOTO 120
130 LET O$=L$
140 LET FILE$(1)=O$+".SER": LET FILE$(2)=O$+".HDS"
150 GOSUB 1600
160 CSR 20,7: PRINT "Loading heading number ";W
170 USER OPEN£1,FILE$(2),"I"
180 FOR I=1 TO 10
190 USER EOF£1,230
200 LET W=W+1: CSR 43,7: PRINT W: PAUSE 100
210 USER  INPUT £1,HEAD$(I)
220 NEXT I
230 USER CLOSE£1
240 GOSUB 1650
250 GOSUB 1600
260 CSR 20,7: PRINT "Loading record number ";Y
270 FOR I=1 TO 200
280 USER EOF£1,330
290 LET Y=Y+1: CSR 42,7: PRINT Y
300 FOR N=1 TO W
310 USER  INPUT £1,RECORD$(I,N)
320 NEXT N: NEXT I
330 LET Z=Y
340 USER CLOSE£1
350 LET F$(1)="F1............": LET F$(2)="F2............": LET F$(3)="F3............": LET F$(4)="F4............"
360 LET D$(1)="Add a record": LET D$(2)="Modify a record": LET D$(3)="Delete a record": LET D$(4)="Return to Main Menu"
370 LET C$(1)="Add-Update-Delete": LET C$(2)=O$+" by D.W."
380 CLS : CSR 25,0: PRINT C$(1): CSR 25,2: PRINT C$(2)
390 GOSUB 1850
400 LET N=0
410 FOR I=7 TO 16 STEP 3
420 LET N=N+1
430 CSR 25,I: PRINT F$(N);D$(N)
440 NEXT I: GOSUB 1850
450 CSR 36,18: PRINT "WARNING!": PAUSE 500: CSR 19,20: PRINT "ADD ANY NEW RECORDS AFTER DOING ANY CHANGES!"
460 GOSUB 1850
470 GOSUB 1800
480 IF A<128 OR A>131 THEN  GOTO 470
490 IF A=128 THEN  GOTO 600
500 IF A=129 OR A=130 THEN  GOTO 850
520 CLS : CSR 28,10: PRINT "Returning to Main Menu": CSR 28,12: PRINT "please wait............."
530 USER  LOAD "MAINMENU.BAS"
600 REM Add a record
610 GOSUB 1710: LET R=0
```





```
620 IF T=0 THEN  LET P$=D$ ELSE  LET P$=FILE$(1)
630 LET H$=P$+" New Record "+STR$(R+1)
640 CLS : CSR 23,0: PRINT H$
650 FOR J=22 TO 24+LEN (H$): CSR J,1: PRINT "=";: NEXT J
660 FOR I=1 TO W: CSR 10,I+3: PRINT HEAD$(I): CSR 18,I+3: PRINT ">": NEXT I
670 CSR 10,20: PRINT F$(1);"Enter data": CSR 50,20: PRINT F$(2);"Return to Menu"
680 GOSUB 1800
690 IF A<128 OR A>129 THEN  GOTO 670
700 IF A=129 THEN  GOTO 790
710 LET R=R+1: FOR V=1 TO W: LET RECORD$(R,V)=EMP$: NEXT V: FOR I=1 TO W: LET LETB$=""
720 CRVS 0,3,21,I+3,25,1,80
730 VS 0: EDITOR B$: IF B$="" OR LEN (B$)>25 THEN  GOTO 730 ELSE  LET RECORD$(R,I)=B$
740 NEXT I: GOSUB 1900
750 FOR I=1 TO W
760 USER  PRINT £1,RECORD$(R,I)
770 NEXT I
780 VS 5: GOTO 630
790 USER CLOSE£1
800 VS 5: CLS : GOTO 380
850 REM Modify/Delete a record
860 CLS : CSR 34,0: IF T=0 THEN  PRINT D$ ELSE  PRINT FILE$(1)
870 FOR X=5 TO 4+W: CSR 0,X: PRINT X-4;".": CSR 4,X: PRINT HEAD$(X-4): NEXT I
880 CSR 20,10: INPUT "Which field to search? > ";FIELD: IF FIELD<1 OR FIELD>W THEN  GOTO 880
890 CSR 20,12: INPUT "Enter data to search for > ";SEARCH$: IF LEN (SEARCH$)>25 THEN  GOTO 890 ELSE  LET SER=LEN (SEARCH$)
900 FOR X=5 TO 4+W: CSR 0,X: PRINT CHR$(5): NEXT X
910 GOSUB 1610: CSR 20,7: PRINT "Searching for ";SEARCH$
920 CSR 20,9: PRINT "Now looking at record no. "
930 FOR I=1 TO Z: CSR 49,9: PRINT I
940 IF LEFT$(RECORD$(I,FIELD),SER)=SEARCH$ THEN  GOTO 1020
950 NEXT I
960 FOR X=5 TO 9 STEP 2: CSR 0,X: PRINT CHR$(5): NEXT X
970 CSR 20,10: PRINT "E N D  O F  F I L E": PAUSE 2000
980 CSR 10,20: PRINT F$(1);"Check directory            ";F$(2);"Return to Menu"
990 GOSUB 1800: IF A<128 OR A>129 THEN  GOTO 990
1000 IF A=128 THEN  GOSUB 2000: GOTO 850
1010 GOTO 380
1020 CLS : FOR X=5 TO 4+W: CSR 20,X: PRINT X-4: CSR 24,X: PRINT RECORD$(I,X-4): NEXT X
1030 CSR 20,20: PRINT "Is this the record to change? (Y/N)": GOSUB 1750
1040 IF A$="N" OR A$="n" THEN  CLS : GOTO 950
1050 CSR 0,20: PRINT CHR$(5): CSR 20,20: PRINT F$(1);"Modify         ";F$(2);"Delete"
1060 GOSUB 1800: IF A<120 OR A>129 THEN  GOTO 1060
1070 IF A=129 THEN  GOTO 1400
1080 GOSUB 1900
1200 CSR 0,20: PRINT CHR$(5): CSR 20,20: PRINT "Modify/Delete another record? (Y/N)": GOSUB 1750
1210 IF A$="Y" THEN  GOTO 860
1220 USER OPEN£1,"BACKUP.DAT","O"
1230 CLS : CSR 30,10: PRINT "Please wait.......................": CSR 30,12: PRINT "Saving changed data": CSR 30,14: PRINT "Now at record number "
1235 FOR I=1 TO Z
1238 CSR 51,14: PRINT I
1240 IF LEFT$(RECORD$(I,1),5)="DELXY" AND LEFT$(RECORD$(I,2),4)="XYDE" THEN  GOTO 1280
1250 FOR N=1 TO W
1260 USER  PRINT £1,RECORD$(I,N)
1270 NEXT N
1280 NEXT I
1290 USER CLOSE£1
1300 IF T=1 THEN GOTO 1340
1310 USER ERAO$
1320 USER RENO$="BACKUP.DAT"
1330 GOTO 1360
1340 USER ERAFILE$ (1)
1350 USER RENFILE$ (1) = "BACKUP.DAT"
```

```
1360 CLS : GOTO 380
1400 REM Delete Record
1410 CLS : CSR 26,0: PRINT "ARE YOU SURE YOU WANT TO": CSR 26,1: PRINT "****  D E L E T E  ****": CSR 32,2: PRINT "THIS RECORD"
1420 GOSUB 1550: PRINT : GOSUB 1850
1430 PAUSE 200: CSR 20,20: PRINT "TYPE 'YES' TO CONFIRM  -  'NO' TO BACKOUT"
1440 CSR 20,22: INPUT "> ";CONF$
1450 IF CONF$="NO" THEN  GOTO 860 ELSE  IF CONF$<>"YES" THEN  GOTO 1440
1460 LET RECORD$(I,1)="DELXY": LET RECORD$(I,2)="XYDE": FOR X=3 TO W: LET RECORD$(I,W)=EMP$: NEXT : CLS : GOTO 1200
1500 IF (I-1)=Z THEN  CLS : CSR 30,10: PRINT "END OF FILE": PAUSE 2000: CLS : GOTO 380
1510 IF (I-1)<Z THEN  GOTO 2100
1520 RETURN
1550 FOR R=1 TO W: CSR 20,R+4: PRINT RECORD$(I,R): NEXT R
1560 RETURN
1600 CLS : CSR 30,0: PRINT O$: CSR 29,1: PRINT "=========="
1610 CSR 20,5: PRINT "Please wait..........."
1620 RETURN
1650 CLS : CSR 22,5: PRINT "Is file a search list? (Y/N)": GOSUB 1750
1660 IF A$="N" THEN  GOTO 1690
1670 USER OPEN£1,FILE$(1),"I"
1680 LET T=1: GOTO 1700
1690 USER OPEN£1,O$,"I"
1700 RETURN
1710 IF T=1 THEN  USER OPEN£1,FILE$(1),"O"
1720 IF T=0 THEN  USER OPEN£1,O$,"O"
1730 RETURN
1750 LET A$=INKEY$: IF A$<>"" THEN  GOTO 1750
1760 LET A$=INKEY$: IF A$="" THEN  GOTO 1760
1770 IF A$<>"Y" AND A$<>"N" THEN  GOTO 1750
1780 RETURN
1800 LET A$=INKEY$: LET A=ASC(A$): RETURN
1850 FOR X=0 TO 79: PRINT "-";: NEXT X: RETURN
1900 VS 5: CLS : CSR 30,0: PRINT O$: CSR 29,1: PRINT "==========": CSR 50,0: PRINT "Page Number ";I
1910 FOR X=1 TO W: CSR 15,X+4: PRINT X;".": CSR 20,X+4: PRINT HEAD$(X);":": CSR 30,X+4: PRINT RECORD$(I,X): NEXT
1920 CSR 20,20: PRINT "Are these records O.K.?  (Y/N)": GOSUB 1750
1930 IF A$="y" OR A$="Y" THEN  RETURN
1940 CSR 20,20: INPUT "Which line do you wish to change?";F
1950 IF F<1 OR F>W THEN  GOTO 1940
1960 CLS : CSR 20,10: PRINT RECORD$(I,F)
1970 LET B$="": LET RECORD$(I,F)=EMP$: CSR 20,12: INPUT "New record > ";B$: IF LEN (B$)>25 THEN  GOTO 1970 ELSE  LET RECORD$(I,F)=B$
1980 CSR 20,14: PRINT "Record now changed": PRINT : GOSUB 1850: PAUSE 1000: GOTO 1900
2000 REM DIRECTORY
2010 CLS : CSR 32,0: PRINT "Record Directory": GOSUB 1850: PAUSE 500
2020 FOR I=1 TO Z
2030 PRINT I;". ";RECORD$(I,1);"  ";RECORD$(I,2): PAUSE 100
2040 IF I/15<>INT(I/15) THEN  GOTO 2100
2050 CSR 1,20: PRINT F$(1);"Continue Directory      ";F$(2);"Modify/Delete Record": CSR 1,21: PRINT F$(3);"Return to Menu"
2060 GOSUB 1800: IF A<128 OR A>130 THEN  GOTO 2060
2070 IF A=128 THEN  CLS : GOSUB 1500: GOTO 2100
2080 IF A=129 THEN  GOTO 1020
2090 GOTO 380
2100 NEXT I: GOTO 2050
10 REM ****************************************************************
20 REM ******************************* VIEWFILE.BAS *******************************
30 REM ******************************** VERSION 2  ********************************
40 REM ******************************** D. WEMYSS  ********************************
50 REM ********************************  OCT 1986  ********************************
60 REM ****************************************************************
70 USER  SAVE "VIEWFILE.BAS"
80 CLEAR : VS 5: CLS : CSR 30,10: PRINT "Please wait............": CSR 30,12: PRINT "Setting up variables"
90 DIM FILE$(2,12),O$(8),P$(12),HEAD$(10,8),F$(4,14),D$(4,19),RECORD$(100,10,25),A$(3),B$(12),NAME$(9): LET W=0: LET T=0: LET Y=0: LET Z=0: LET B$=""
100 CLS : CSR 20,10: INPUT "File to view? > ";B$: IF LEN (B$)>8 THEN  GOTO 100 ELSE  LET O$=B$
```

```
105 LET FILE$(1)=O$+".SER": LET FILE$(2)=O$+".HDS"
110 USER OPEN£1,FILE$(2),"I"
120 GOSUB 1050
130 CSR 25,7: PRINT "Loading heading number ";W
140 FOR I=1 TO 10
150 USER EOF£1,190
160 LET W=W+1: CSR 48,7: PRINT W: PAUSE 100
170 USER  INPUT £1,HEAD$(I)
180 NEXT I
190 PAUSE 1000: USER CLOSE£1
210 LET F$(1)="F1............": LET F$(2)="F2............": LET F$(3)="F3............": LET F$(4)="F4............": LET D$(1)="View directory": LET D$(2)="Vie
w all records": LET D$(3)="View a record": LET D$(4)="Return to Main Menu"
220 GOSUB 1150: GOSUB 1050
230 CSR 25,7: PRINT "Loading record number ";Y
240 FOR I=1 TO 200
250 USER EOF£1,300
260 LET Y=Y+1: CSR 48,7: PRINT Y
270 FOR N=1 TO W
280 USER  INPUT £1,RECORD$(I,N)
290 NEXT N: NEXT I
300 LET Z=Y: PAUSE 1000
310 USER CLOSE£1
```

* D INTRO.

* E

* R

INTRO

DATABASE

This suite of programs making up the database was devised by Dave Wemyss of Strathkinness. It is in FDX Basic for an 80-column VDU, but with a little bit of thought, could easily be adapted for other configurations.

It is menu-driven with mostly single key responses. Written for the 3 1/2" disc drive, the ordinary 64K memory will cope with up to 100 records, each with up to 10 fields and up to 25 characters per field. With an expanded memory, then it cope with many more records.

File names should follow the following instructions. Give each file a name of up to 8 characters with no extension. When working with a search list, the extension .SER will be added automatically by the programme. When working with a search list do not add .SER when asked for the file name.

Press <RET> to continue

MEASURE

DO YOUR HAVE PROBLEMS CONVERTING LENGTH, AREA, VOLUME, MEASURE ETC. LET YOUR COMPUTER DO THE WORK FOR YOU WITH THIS HANDY LITTLE PROGRAM.

```
10 DIM L(8),L$(8,11),A(7),A$(7,9),V(7),V$(7,12),M(6),M$(6,9),P(5),P$(5,11)
20 FOR K=1 TO 8: READ L(K),L$(K): NEXT
30 DATA  1,Inches,12,Feet,36,Yards,63360,Miles,.03937,Millimetres,.3937,Centimetres,39.37,Metres,39370,Kilometres
40 FOR K=1 TO 7: READ A(K),A$(K): NEXT
50 DATA  1,SQ.Inches,144,SQ.Feet,6272640,Acres,4.0145E9,SQ.Miles,155,SQ.Cms,1550,SQ.Metres,1.55E7,Hectares
60 FOR K=1 TO 7: READ V(K),V$(K): NEXT
70 DATA  1,Cubic Inches,1728,Cubic Feet,34.67,Pints,277.36,Gallons,.06102,Cc'S,61.024,Litres,61024,Cubic Metres
80 FOR K=1 TO 6: READ M(K),M$(K): NEXT
90 DATA  1,Ounces,16,Pounds,35840,Tons,.03527,Grams,35.27,Kilograms,35270,Tonnes
100 FOR K=1 TO 5: READ P(K),P$(K): NEXT
110 DATA  1,Psi,51.73,mmHG,6895,N/Sq.Metre,.0681,Atmospheres,68.95,Millibars
120 PAPER 4: INK 15: CLS : CSR 16,1: PRINT "MAIN MENU"
130 CSR 10,5: PRINT "0 :-  QUIT PROGRAM"
140 CSR 10,7: PRINT "1 :-  LENGTH"
150 CSR 10,9: PRINT "2 :-  AREA"
160 CSR 10,11: PRINT "3 :-  VOLUME"
170 CSR 10,13: PRINT "4 :-  WEIGHT"
180 CSR 10,15: PRINT "5 :-  PRESSURE"
190 CSR 10,17: PRINT "6 :-  TEMPERATURE"
195 CSR 12,22: PRINT "SELECT CATEGORY"
200 LET X$=INKEY$: IF X$<"0" OR X$>"6" THEN  GOTO 140
210 IF X$="0" THEN  CLS : STOP
220 CLS : ON VAL(X$) GOSUB 220,1000,1500,2000,2500,3000,3500
230 GOTO 120
1000 PAPER 12: CSR 17,2: PRINT "LENGTH": CSR 17,3: PRINT "______": FOR K=1 TO 8: CSR 12,7+K: PRINT K;" - ";L$(K): NEXT
1005 CSR 11,22: PRINT "SELECT MEASUREMENT"
1010 LET B$=INKEY$: IF B$<"1" OR B$>"8" THEN  GOTO 1010
1020 LET B=VAL(B$): CLS : CSR 5,12: PRINT " Input Number of ";L$(B);
1030 INPUT VL
1040 CLS : CSR 10,2: PRINT VL;" ";L$(B);" Equals "
1060 FOR K=1 TO 8: CSR 5,5+K: PRINT VL*L(B)/L(K): CSR 20,5+K: PRINT L$(K): NEXT
1070 GOSUB 4000
1080 GOSUB 4010
1090 LET X$=INKEY$: IF X$="" THEN  GOTO 1090
1100 IF X$=CHR$(13) THEN  RETURN  ELSE  CLS : GOTO 1000
1500 PAPER 8: CSR 18,2: PRINT "AREA": CSR 18,3: PRINT "____": FOR K=1 TO 7: CSR 12,7+K: PRINT K;" -";A$(K): NEXT
1505 CSR 11,22: PRINT "SELECT MEASUREMENT"
1510 LET B$=INKEY$: IF B$<"1" OR B$>"7" THEN  GOTO 1510
1520 LET B=VAL(B$): CLS : CSR 5,12: PRINT "Input Number of ";A$(B);
1530 INPUT VL
1540 CLS : CSR 12,2: PRINT VL;" ";A$(B);" Equals "
1560 FOR K=1 TO 7: CSR 5,5+K: PRINT VL*A(B)/A(K): CSR 20,5+K: PRINT A$(K): NEXT
```

```
1570 GOSUB 4000
1580 GOSUB 4010
1590 LET X$=INKEY$: IF X$="" THEN  GOTO 1590
1600 IF X$=CHR$(13) THEN  RETURN  ELSE  CLS : GOTO 1500
2000 PAPER 13: CSR 17,2: PRINT "VOLUME": CSR 17,3: PRINT "_____": FOR K=1 TO 7: CSR 12,7+K: PRINT K;" - ";V$(K): NEXT
2005 CSR 11,22: PRINT "SELECT MEASUREMENT"
2010 LET B$=INKEY$: IF B$<"1" OR B$>"7" THEN  GOTO 2010
2020 LET B=VAL(B$): CLS : CSR 5,12: PRINT "Input Number of ";V$(B);
2030 INPUT VL
2040 CLS : CSR 12,2: PRINT VL;" ";V$(B);" Equals "
2060 FOR K=1 TO 7: CSR 5,5+K: PRINT VL*V(B)/V(K): CSR 20,5+K: PRINT V$(K): NEXT
2070 GOSUB 4000
2080 GOSUB 4010
2090 LET X$=INKEY$: IF X$="" THEN  GOTO 2090
2100 IF X$=CHR$(13) THEN  RETURN  ELSE  CLS : GOTO 2000
2500 PAPER 7: INK 1: CSR 17,2: PRINT "WEIGHT": CSR 17,3: PRINT "_____": FOR K=1 TO 6: CSR 12,7+K: PRINT K;" - ";M$(K): NEXT
2505 CSR 11,22: PRINT "SELECT MEASUREMENT"
2510 LET B$=INKEY$: IF B$<"1" OR B$>"6" THEN  GOTO 2510
2520 LET B=VAL(B$): CLS : CSR 5,12: PRINT "Input Number of ";M$(B);
2530 INPUT VL
2540 CLS : CSR 12,2: PRINT VL;" ";M$(B);" Equals "
2560 FOR K=1 TO 6: CSR 5,5+K: PRINT VL*M(B)/M(K): CSR 20,5+K: PRINT M$(K): NEXT
2570 GOSUB 4000
2580 GOSUB 4010
2590 LET X$=INKEY$: IF X$="" THEN  GOTO 2590
2600 IF X$=CHR$(13) THEN  RETURN  ELSE  CLS : GOTO 2500
3000 PAPER 3: INK 1: CSR 16,2: PRINT "PRESSURE": CSR 16,3: PRINT "_______": FOR K=1 TO 5: CSR 12,7+K: PRINT K;" - ";P$(K): NEXT
3005 CSR 11,22: PRINT "SELECT MEASUREMENT"
3010 LET B$=INKEY$: IF B$<"1" OR B$>"5" THEN  GOTO 3010
3020 LET B=VAL(B$): CLS : CSR 5,12: PRINT "Input Number of ";P$(B);
3030 INPUT VL
3040 CLS : CSR 12,2: PRINT VL;" ";P$(B);" Equals"
3060 FOR K=1 TO 5: CSR 5,5+K: PRINT VL*P(B)/P(K): CSR 20,5+K: PRINT P$(K): NEXT
3070 GOSUB 4000
3080 GOSUB 4010
3090 LET X$=INKEY$: IF X$="" THEN  GOTO 3090
3100 IF X$=CHR$(13) THEN  RETURN  ELSE  CLS : GOTO 3000
3500 PAPER 10: INK 1: CLS : CSR 14,2: PRINT "TEMPERATURE": CSR 14,3: PRINT "__________": CSR 13,6: PRINT "CONVERT WHICH"
3510 CSR 5,11: PRINT "(C)  Centigrade to Fahrenheit": CSR 5,14: PRINT "(F)  Fahrenheit to Centigrade"
3520 LET B$=INKEY$: IF B$<>"C" AND B$<>"F" THEN  GOTO 3520
3530 IF B$="C" THEN  GOTO 3570
3540 CSR 5,19: INPUT "Input Degrees Fahrenheit? ";VL
3550 CLS : CSR 5,4: PRINT VL;" Degrees Fahrenheit Equals"
3560 CSR 9,8: PRINT (VL-32)*5/9;" Degrees Centigrade": GOTO 3595
3570 CSR 5,19: INPUT " Input Degrees Centigrade? ";VL
3580 CLS : CSR 5,4: PRINT VL;" Degrees Centigrade Equals"
3590 CSR 9,8: PRINT 32+VL*9/5;" Degrees Fahrenheit"
3595 GOSUB 4000: GOSUB 4010
3600 LET X$=INKEY$: IF INKEY$="" THEN  GOTO 3600
3610 IF X$=CHR$(13) THEN  RETURN  ELSE  GOTO 3500
4000 CSR 1,20: PRINT "PRESS ANY KEY FOR ANOTHER MEASUREMENT": RETURN
4010 CSR 7,22: PRINT "PRESS RETURN FOR MAIN MENU": RETURN
```

WRITTEN BY - MR R. SNEATH.

```
10 GOTO 1000
20 CODE

4011 BOARD:  DS 64
4051 IDENT:  DS 64
4091 LIMIT:  DB 02,04,04,04,04,04,04,03
4099         DB 04,06,06,06,06,06,06,04
40A1         DB 04,06,06,06,06,06,06,04
40A9         DB 04,06,06,06,06,06,06,04
40B1         DB 04,06,06,06,06,06,06,04
40B9         DB 04,06,06,06,06,06,06,04
40C1         DB 04,06,06,06,06,06,06,04
40C9         DB 03,04,04,04,04,04,04,02
40D1 XPOS:   DB 02,04,06,08,10,12,14,16
40D9         DB 04,06,08,10,12,14,16,18
40E1         DB 06,08,10,12,14,16,18,20
40E9         DB 08,10,12,14,16,18,20,22
40F1         DB 10,12,14,16,18,20,22,24
40F9         DB 12,14,16,18,20,22,24,26
4101         DB 14,16,18,20,22,24,26,28
4109         DB 16,18,20,22,24,26,28,30
4111 YPOS:   DB 11,10,09,08,07,06,05,04
4119         DB 12,11,10,09,08,07,06,05
4121         DB 13,12,11,10,09,08,07,06
4129         DB 14,13,12,11,10,09,08,07
4131         DB 15,14,13,12,11,10,09,08
4139         DB 16,15,14,13,12,11,10,09
4141         DB 17,16,15,14,13,12,11,10
4149         DB 18,17,16,15,14,13,12,11
4151 EXDIR:  DB 06,30,30,30,30,30,30,28
4159         DB 39,63,63,63,63,63,63,60
4161         DB 39,63,63,63,63,63,63,60
4169         DB 39,63,63,63,63,63,63,60
4171         DB 39,63,63,63,63,63,63,60
4179         DB 39,63,63,63,63,63,63,60
4181         DB 39,63,63,63,63,63,63,60
4189         DB 35,51,51,51,51,51,51,48
4191 COLOURS:DB 14,05,02,06,10,13,07
4198 SCORES: DS 14
41A6 EXFG:   DS 2
```

```
41A8 COLOUR: DS 2
41AA COUNT:  DS 2
41AC INCR:   LD A,(COUNT)
41AF         ADD A,E
41B0         LD H,0
41B2         LD L,A
41B3         PUSH HL
41B4         LD DE,BOARD
41B7         ADD HL,DE
41B8         INC (HL)
41B9         POP HL
41BA         LD DE,IDENT
41BD         ADD HL,DE
41BE         LD A,(COLOUR)
41C1         LD (HL),A
41C2         RET
41C3 GETBD:  LD DE,BOARD
41C6         JR AD
41C8 GETID:  LD DE,IDENT
41CB         JR AD
41CD GETLT:  LD DE,LIMIT
41D0 AD:     LD HL,(COUNT)
41D3         ADD HL,DE
41D4         LD A,(HL)
41D5         RET
41D6 START:  LD A,0
41D8         LD (COUNT),A
41DB         LD (EXFG),A
41DE LOOP:   CALL GETBD
41E1         LD B,A
41E2         CALL GETLT
41E5         CP B
41E6         JR Z,EXPLODE
41E8         JP NC,NEXT
41EB EXPLODE:LD A,(COLOUR)
41EE         LD (EXFG),A
41F1         CALL GETLT
41F4         LD B,A
41F5         CALL GETBD
41F8         SCF
```

```
41F9        CCF
41FA        SBC A,B
41FB        LD (HL),A
41FC        CP 0
41FE        JR NZ,LB
4200        CALL GETID
4203        LD A,0
4205        LD (HL),A
4206 LB:    LD DE,EXDIR
4209        CALL AD
420C        LD C,A
420D        BIT 0,C
420F        JR Z,LB1
4211        LD E,£F9
4213        CALL INCR
4216 LB1:   BIT 1,C
4218        JR Z,LB2
421A        LD E,1
421C        CALL INCR
421F LB2:   BIT 2,C
4221        JR Z,LB3
4223        LD E,8
4225        CALL INCR
4228 LB3:   BIT 3,C
422A        JR Z,LB4
422C        LD E,7
422E        CALL INCR
4231 LB4:   BIT 4,C
4233        JR Z,LB5
4235        LD E,£FF
4237        CALL INCR
423A LB5:   BIT 5,C
423C        JR Z,NEXT
423E        LD E,£FB
4240        CALL INCR
4243 NEXT:  LD A,(COUNT)
4246        INC A
4247        LD (COUNT),A
424A        CP 64
424C        JP NZ,LOOP
424F SCORE: LD C,13
4251        LD B,0
4253        LD HL,SCORES
4256        LD DE,SCORES
4259        INC DE
425A        LDIR
425C        LD A,0
425E        LD (COUNT),A
4261 LOOP1: CALL GETID
4264        SLA A
4266        LD E,A
4267        LD D,0
4269        LD HL,SCORES
426C        ADD HL,DE
426D        LD A,(HL)
426E        PUSH HL
426F        LD C,A
4270        INC HL
```

```
4271        LD A,(HL)
4272        LD B,A
4273        CALL GETBD
4276        LD L,A
4277        LD H,0
4279        ADD HL,BC
427A        LD C,L
427B        LD B,H
427C        POP HL
427D        LD A,C
427E        LD (HL),A
427F        LD A,B
4280        INC HL
4281        LD (HL),A
4282        LD DE,XPOS
4285        CALL AD
4288        LD (XP),A
428B        LD DE,YPOS
428E        CALL AD
4291        LD (YP),A
4294        CALL GETID
4297        LD L,A
4298        LD H,0
429A        LD DE,COLOURS
429D        ADD HL,DE
429E        LD A,(HL)
429F        LD (INK),A
42A2        CALL GETBD
42A5        CP 0
42A7        JR NZ,GTZ
42A9        LD A,136
42AB        JR SET
42AD GTZ:   LD B,A
42AE        CALL GETLT
42B1        DEC A
42B2        CP B
42B3        JR C,GTL
42B5        JR NZ,LTL
42B7 GTL:   LD A,132
42B9        JR SET
42BB LTL:   LD A,B
42BC        ADD A,48
42BE SET:   LD (CHAR),A
42C1        RST 10
42C2        DB £64,£87
42C4        DB 16,0
42C6 INK:   DS 1
42C7        DB 3
42C8 XP:    DS 1
42C9 YP:    DS 1
42CA CHAR:  DS 1
42CB        LD A,(COUNT)
42CE        INC A
42CF        LD (COUNT),A
42D2        CP 64
42D4        JR NZ,LOOP1
42D6        RET
42D7 BRD:   DS 49
```

```
4308 IDT:     DS 49
4339 LNT:     DB 2,3,3,3,3,3,2
4340          DB 3,4,4,4,4,4,3
4347          DB 3,4,4,4,4,4,3
434E          DB 3,4,4,4,4,4,3
4355          DB 3,4,4,4,4,4,3
435C          DB 3,4,4,4,4,4,3
4363          DB 2,3,3,3,3,3,2
436A WGT:     DB 00,48,06,42,02,14,34
4371          DB 46,04,20,44,28,24,16
4378          DB 32,18,30,17,31,25,23
437F          DB 03,45,27,21,08,40,12
4386          DB 36,01,07,41,47,05,13
438D          DB 43,35,10,38,26,22,09
4394          DB 15,33,39,11,19,37,29
439B EXPD:    DB 06,14,14,14,14,14,12
43A2          DB 07,15,15,15,15,15,13
43A9          DB 07,15,15,15,15,15,13
43B0          DB 07,15,15,15,15,15,13
43B7          DB 07,15,15,15,15,15,13
43BE          DB 07,15,15,15,15,15,13
43C5          DB 03,11,11,11,11,11,09
43CC XPO:     DB 9,11,13,15,17,19,21
43D3          DB 9,11,13,15,17,19,21
43DA          DB 9,11,13,15,17,19,21
43E1          DB 9,11,13,15,17,19,21
43E8          DB 9,11,13,15,17,19,21
43EF          DB 9,11,13,15,17,19,21
43F6          DB 9,11,13,15,17,19,21
43FD YPO:     DB 5,5,5,5,5,5,5
4404          DB 7,7,7,7,7,7,7
440B          DB 9,9,9,9,9,9,9
4412          DB 11,11,11,11,11,11,11
4419          DB 13,13,13,13,13,13,13
4420          DB 15,15,15,15,15,15,15
4427          DB 17,17,17,17,17,17,17
442E TEMP1:   DS 98
4490 TEMP2:   DS 98
44F2 BESTP:   DS 1
44F3 MINIMAX:DS 1
44F4 MAXSC:   DS 1
44F5 CCPOS:   DS 1
44F6 PCPOS:   DS 1
44F7 COMPS:   DS 1
44F8 PERSS:   DS 1
44F9 VO:      DB 97,98,99,100
44FD V1:      DB 101,102,103,104
4501 V2:      DB 105,106,107,108
4505 VL:      DB 32,32,32,32
4509 INC:     LD A,(COUNT)
450C          ADD A,E
450D          LD H,0
450F          LD L,A
4510          PUSH HL
4511          LD DE,BRD
4514          ADD HL,DE
4515          INC (HL)
```

```
4516          POP HL
4517          LD DE,IDT
451A          ADD HL,DE
451B          LD A,(COLOUR)
451E          LD (HL),A
451F          RET
4520 GBRD:    LD DE,BRD
4523          JP AD
4526 GIDT:    LD DE,IDT
4529          JP AD
452C GLNT:    LD DE,LNT
452F          JP AD
4532 EXP:     LD A,0
4534          LD (COUNT),A
4537          LD (EXFG),A
453A LO:      CALL GBRD
453D          CP 0
453F          JR Z,NXT
4541          LD B,A
4542          CALL GLMT
4545          CP B
4546          JR Z,EX
4548          JP NC,NXT
454B EX:      LD A,(COLOUR)
454E          LD (EXFG),A
4551          CALL GLNT
4554          LD B,A
4555          CALL GBRD
4558          SCF
4559          CCF
455A          SBC A,B
455B          LD (HL),A
455C          CP 0
455E          JR NZ,L1
4560          CALL GIDT
4563          LD A,0
4565          LD (HL),A
4566 L1:      LD DE,EXPD
4569          CALL AD
456C          LD C,A
456D          BIT 0,C
456F          JR Z,L2
4571          LD E,£F9
4573          CALL INC
4576 L2:      BIT 1,C
4578          JR Z,L3
457A          LD E,1
457C          CALL INC
457F L3:      BIT 2,C
4581          JR Z,L4
4583          LD E,7
4585          CALL INC
4588 L4:      BIT 3,C
458A          JR Z,NXT
458C          LD E,£FF
458E          CALL INC
4591 NXT:     LD A,(COUNT)
```

```
4594         INC A
4595         LD (COUNT),A
4598         CP 49
459A         JP NZ,L0
459D         LD A,0       ; ZERO SCORES
459F         LD (COMPS),A
45A2         LD (PERSS),A
45A5         LD (COUNT),A
45A8 L5:     CALL GIDT
45AB         CP 0
45AD         JR Z,Z
45AF         LD E,A
45B0         LD D,0
45B2         LD HL,PCPOS         ;PLUS OFFSET FOR SCORE
45B5         ADD HL,DE
45B6         LD A,(HL)
45B7         PUSH HL
45B8         LD B,A
45B9         CALL GBRD
45BC         ADD A,B
45BD         POP HL
45BE         LD (HL),A
45BF Z:      LD A,(COUNT)
45C2         INC A
45C3         LD (COUNT),A
45C6         CP 49
45C8         JR NZ,L5
45CA         RET
45CB ENTER:  LD A,2
45CD         LD (COLOUR),A
45D0 DIS:    CALL DISPLAY
45D3         CALL EXP
45D6 DISPLAY:LD A,0
45D8         LD (COUNT),A
45DB L6:     LD DE,XPO
45DE         CALL AD
45E1         LD (X1),A
45E4         LD (X3),A
45E7         INC A
45E8         LD (X2),A
45EB         LD (X4),A
45EE         LD DE,YPO
45F1         CALL AD
45F4         LD (Y1),A
45F7         LD (Y2),A
45FA         INC A
45FB         LD (Y3),A
45FE         LD (Y4),A
4601         CALL GIDT
4604         CP 0
4606         JR Z,BLACK
4608         CP 2
460A         JR Z,PERS
460C         LD A,6
460E         JR SCOL
4610 PERS:   LD A,7
4612         JR SCOL
4614 BLACK:  LD A,7
4616 SCOL:   LD (PAPER),A
4619         CALL GBRD
```

```
461C         CP 0
461E         JR NZ,L7
4620         LD IX,V0
4624         JR SETC
4626 L7:     LD B,A
4627         CALL GLMT
462A         DEC A
462B         CP B
462C         JR C,L8
462E         JR NZ,L9
4630 L8:     LD IX,VL
4634         JR SETC
4636 L9:     LD A,B
4637         CP 1
4639         JR NZ,L10
463B         LD IX,V1
463F         JR SETC
4641 L10:    LD IX,V2
4645 SETC:   LD A,(IX+0)
4648         LD (C1),A
464B         LD A,(IX+1)
464E         LD (C2),A
4651         LD A,(IX+2)
4654         LD (C3),A
4657         LD A,(IX+3)
465A         LD (C4),A
465D         RST 10
465E         DB £64,£93
4660         DB 16,0
4662 PAPER:  DS 1
4663         DB 3
4664 X1:     DS 1
4665 Y1:     DS 1
4666 C1:     DS 1
4667         DB 3
4668 X2:     DS 1
4669 Y2:     DS 1
466A C2:     DS 1
466B         DB 3
466C X3:     DS 1
466D Y3:     DS 1
466E C3:     DS 1
466F         DB 3
4670 X4:     DS 1
4671 Y4:     DS 1
4672 C4:     DS 1
4673         LD A,(COUNT)
4676         INC A
4677         LD (COUNT),A
467A         CP 49
467C         JP NZ,L6
467F         RET
4680 CMOVE:  LD BC,98    ; STORE BOARD
4683         LD DE,TEMP1
4686         LD HL,BRD
4689         LDIR
468B         LD A,0      ;INIT FLAGS
468D         LD (CCPOS),A
4690         INC A
4691         LD (COLOUR),A
```

FOOTBALL POOLS PREDICTOR

PART FOUR

```
10 REM ****************************************************************************
20 REM ******************** FOOTBALL POOLS FORECASTING *************************
30 REM ********************  PROFESSOR FRANK GEORGE    *************************
40 REM ********************      COMMODORE 64          *************************
50 REM ********************   ADAPTED FOR MEMOTECH     *************************
60 REM ********************     BY DAVE WEMYSS         *************************
70 REM ********************       JAN 1987             *************************
80 REM ********************     VIEWAMND.FIX           *************************
90 REM ****************************************************************************
100 DISC SAVE "VIEWAMND.FIX"
110 VS 5: CLS : CLEAR
120 DIM RECORD$(25,11,16),SAV$(12),DIV$(25),DATE$(9),WEEK$(9),LOA$(2,12),F(12,2)
,AA$(8)
130 LET SAV$="": LET DATE$="": LET WEEK$="": LET AA$="Fixtures": LET DIV$=""
220 CLS : CSR 32,0: PRINT AA$: CSR 31,1: PRINT "=========="
230 CSR 16,10: PRINT "F1..........View last Saturday's Fixtures"
240 CSR 16,12: PRINT "F2..........Input next Saturday's Fixtures"
245 CSR 16,14: PRINT "F3..........Return to Main Menu"
250 GOSUB 1100: IF A<128 OR A>130 THEN GOTO 250
260 IF A=129 THEN GOTO 600
265 IF A=130 THEN GOTO 510
270 GOSUB 1050
280 GOSUB 1150
290 LET Z=0: GOSUB 1000: PRINT "Loading fixture number ";Z
300 DISC OPEN £1,LOA$(1),"I"
310 DISC INPUT £1,WEEK$
320 FOR V=1 TO 15
330 DISC EOF £1,380
340 LET Z=Z+1: CSR 43,7: PRINT Z: PAUSE 100
350 DISC INPUT £1,F(V,1)
360 DISC INPUT £1,F(V,2)
370 NEXT V
380 DISC CLOSE £1
390 GOSUB 1250
410 FOR V=1 TO Z
420 CSR 50,V+4: PRINT "Match ";V: CSR 60,V+4: PRINT F(V,1);" v ";F(V,2)
430 NEXT V
440 CSR 7,20: PRINT "F1..........View another division     F2..........Return to
 Menu"
450 GOSUB 1100: IF A<128 OR A>129 THEN GOTO 450
470 IF A=129 THEN GOTO 220
480 FOR V=1 TO Z: LET F(V,1)=0: LET F(V,2)=0: NEXT V
490 FOR X=1 TO Y: FOR I=1 TO 11: LET RECORD$(X,I)="                ": NEXT I: NE
XT X
500 GOTO 270
510 GOSUB 1000: PRINT "Returning to Main Menu"
520 DISC LOAD "POOLS.BAS"
600 CLS : CSR 20,10: PRINT "Date for ";AA$;" > ";: INPUT WEEK$
610 GOSUB 1050: GOSUB 1150: CLS
620 GOSUB 1250
630 CSR 66,4: PRINT AA$
640 CSR 20,20: PRINT "For void matches enter 0": PAUSE 1000
```

```
650 FOR X=1 TO Y/2: GOSUB 1500: NEXT X
660 PAUSE 2000: CSR 20,18: PRINT "Are fixtures all correct? (Y/N)": GOSUB 1650
670 IF A$="Y" OR A$="y" THEN GOTO 720
680 CSR 0,18: PRINT CHR$(5): CSR 20,18: INPUT "Which match to change? >";X
690 IF X<1 OR X>Y/2 THEN GOTO 690
700 CSR 0,18: PRINT CHR$(5): GOSUB 1500
710 GOTO 660
720 GOSUB 1000: LET X=0: PRINT "Saving Fixture List.........": CSR 20,9: PRINT "
For Saturday ";WEEK$: CSR 20,11: PRINT "Now saving match number ";X
730 LET SAV$=LEFT$(LOA$(2),8)+".FIX"
735 DISC ERA SAV$
740 DISC OPEN £1,SAV$,"O"
750 DISC PRINT £1,WEEK$

755 FOR Y=1 TO Y/2: CSR 44,11: PRINT X
760 DISC PRINT £1,F(X,1)
770 DISC PRINT £1,F(X,2)
780 NEXT X
790 DISC CLOSE £1
800 CLS : CSR 8,20: PRINT "F1..........Another Division     F2..........Return t
o Menu"
810 GOSUB 1100: IF A<128 OR A>129 THEN GOTO 810
820 IF A=128 THEN GOTO 610 ELSE GOTO 220
1000 CLS : CSR 20,0: PRINT DIV$: GOSUB 1600: CSR 20,5: PRINT "Please wait.......
...": CSR 20,7
1010 RETURN
1050 CLS : PLOD "PROB!"
1060 GOSUB 1100: IF A<128 OR A>134 THEN GOTO 1060
1070 RETURN
1100 LET A$=INKEY$: LET A=ASC(A$): RETURN
1150 LET DIV$="                              "
1160 IF A=128 THEN LET LOA$(1)="ENGDIVS1.FIX": LET LOA$(2)="ENGDIVS1.TMS": LET D
IV$="English First Division": LET N=22: GOTO 1230
1170 IF A=129 THEN LET LOA$(1)="ENGDIVS2.FIX": LET LOA$(2)="ENGDIVS2.TMS": LET D
IV$="English Second Division": LET N=22: GOTO 1230
1180 IF A=130 THEN LET LOA$(1)="ENGDIVS3.FIX": LET LOA$(2)="ENGDIVS3.TMS": LET D
IV$="English Third Division": LET N=24: GOTO 1230
1190 IF A=131 THEN LET LOA$(1)="ENGDIVS4.FIX": LET LOA$(2)="ENGDIVS4.TMS": LET D
IV$="English Fourth Division": LET N=24: GOTO 1230
1200 IF A=132 THEN LET LOA$(1)="SCOTPREM.FIX": LET LOA$(2)="SCOTPREM.TMS": LET D
IV$="Scottish Premier Division": LET N=12: GOTO 1230
1210 IF A=133 THEN LET LOA$(1)="SCOTDIV1.FIX": LET LOA$(2)="SCOTDIV1.TMS": LET D
IV$="Scottish First Division": LET N=12: GOTO 1230
1220 IF A=134 THEN LET LOA$(1)="SCOTDIV2.FIX": LET LOA$(2)="SCOTDIV2.TMS": LET D
IV$="Scottish Second Division": LET N=14
1230 RETURN
1250 DISC OPEN £1,LOA$(2),"I"
1270 LET S=N/2: LET Y=0: CLS
1280 DISC INPUT £1,DATE$
1290 CSR 10,0: PRINT "Teams updated to ";DATE$;"       ";AA$;" are for ";WEEK$
1300 CSR 25,2: PRINT DIV$: GOSUB 1600
1310 FOR X=1 TO S
1320 LET Y=Y+1
1330 FOR I=1 TO 11
1340 DISC INPUT £1,RECORD$(X,I)
1350 NEXT I
1360 CSR 5,X+4: PRINT X;".": CSR 9,X+4: PRINT RECORD$(X,1)
1370 NEXT X: LET L=5
1380 FOR X=S+1 TO N
1390 LET Y=Y+1
```

```
1400 FOR I=1 TO 11
1410 DISC INPUT £1,RECORD$(X,1)
1420 NEXT I
1430 CSR 24,L: PRINT X;".": CSR 28,L: PRINT RECORD$(X,1)
1440 LET L=L+1: NEXT X
1450 DISC CLOSE £1
1460 RETURN
1500 CSR 20,18: PRINT "Match No. ";X: CSR 35,18: INPUT "Home Team No. > ";F(X,1)
1510 CSR 55,X+4: PRINT X;".": CSR 65,X+4: PRINT F(X,1);" v "
1520 CSR 35,18: PRINT "                              ": CSR 35,18: INPUT "Away Team No. > "
;F(X,2)
1530 CSR 70,X+4: PRINT F(X,2): PAUSE 1000
1540 CSR 0,18: PRINT CHR$(5)
1550 RETURN
1600 FOR X=0 TO 79: PRINT "-";: NEXT
1610 RETURN
1650 LET A$=INKEY$: IF A$<>"" THEN GOTO 1650
1660 LET A$=INKEY$: IF A$="" THEN GOTO 1660
1670 IF A$<>"Y" AND A$<>"Y" AND A$<>"N" AND A$<>"N" THEN GOTO 1650

1680 RETURN

PROG1
```

```
               * D MENU.

               * R



                 CHOOSE DIVISION MENU
                 ====================

English First Division.............................F1

English Second Division............................F2

English Third Division.............................F3

English Fourth Division............................F4

Scottish Premier Division..........................F5

Scottish First Division............................F6

Scottish Second Division...........................F7
```

* PLEASE NOTE *

Since receiving the listing for 'Football Pools Predictor' from Dave he had had to make certain modifications and changes, these are listed on the next page. You now have the complete set of programs I hope they help you win a fortune. Remember if you are successful - I want my cut!

CHANGES TO FORECASTING PROGRAM

We should explain that the first part of page 43 of Issue 8 is the introduction which is in NODDY. Line 400 of POOLS.BAS calls this up if required. Missing is PROG1" which should be:

```
    PROG 1          *  D INTRO
                    *  E
                    *  R
```

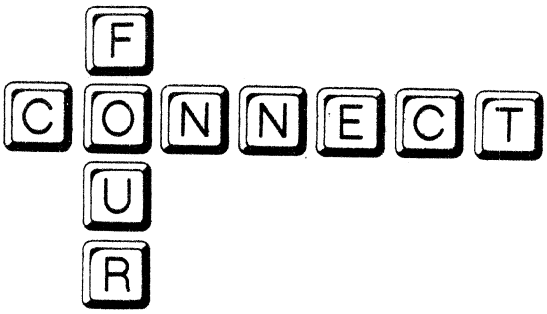
Since the English League Divisions One and Two have been re-organised this season there have to be changes in the VIEWAMND.FIX module. They are :

```
   1420  CSR  10,3:PRINT DIV$:LET N=INT (Y/2) : LET NN=MOD (Y,2)

   1455   IF  NN=1  THEN  CSR  24,X+4:PRINT  X*2  -1;  ".":CSR  28,
         X+4:PRINT RECORD$ ((X*2-1),1)
```

I later found that the change in the number of teams in the English Divisions One and Two also affected the module for re-arranging the league positions (POSITION.SOR). Below are the necessary changes to line 290 and the new line 295.

```
290 LET N=INT (Y/2): LET NN=MOD (Y,2): FOR X+1 TO N: CSR 0, X+4;
PRINT X; ".": CSR 5, X+4: PRINT RECORD$ (X,1); CSR 22, X+4: PRINT
RECORD$ (X,2): CSR 30, X+4: PRINT X+N; "."; CSR 35, X+4: PRINT
RECORD$ (X+N,1): CSR 53, X+4: PRINT RECORD$ (X+N,2): NEXT X

295 IF NN=1 THEN CSR 30, X+4: PRINT X*2-1; ".": CSR 35,X+4: PRINT
RECORD$ ((X*2-1),1): CSR 53, X+4: PRINT RECORD$ ((X*2-1),2)
```

Volume Three **MEMOPAD** Number 11



The very first task this month is to insert a forgotten variable TH$. This is used by the printing routine when the computer is thinking.

Immediately after G$: DB 00 enter the INSERT MODE and type in : TH$: DB 00 as per the first two lines of this month's listing.

Next, immediately after LD SP,STACK enter the next section of code (again in the 'insert mode') up to the line of ampersands (8). This initialises the VDP to our configuration.

This month's main code starts right after SETBRD. ENTER ALL THIS CODE. Make sure you still in the INSERT MODE so that it shoves all the other code upward in memory.

The first part of the code request an input asking : :Do you want to go first?" This section calls a subroutine that first clears the input line and then sets up the cursor position before printing the message. It then calls a rom routine that detects an input from the keyboard and does not return until a key is pressed.

The code then insert the relevant Ascii codes for computer graphic, human graphic and thinking string into variables C$, H$, TH$. These are the LEFT HAND CODES - the actual graphics are made up of two graphic blocks for computer and human. These are only used to detect whose piece is in a particular square. When we print the actual characters to screen they will, in the case of the human piece, for example, be printed as CH$ (91) + CH$ (92).

SECTION 4 is entered at the very end of the code (after RET). These are general subroutine used by the code already typed in.

If the assembler informs you that you have some undefined lables when you exit, don't worry, we will put these in with the next section of code.

DON'T FORGET TO SAVE YOUR LATEST VERSION.

If you put a loop in to test last month's listing don't forget to erase it or the program won't go anywhere except in that loop.

You cannot run this month's code because it's incomplete. Next month we will start the computer's routines.

```
                        TITLE Connect Four Assembler version for magazines only  <c> K. Hook 1987
                        .Z80
0003                    ASEG
                        .LIST
4062    00              G$:     DB      00
4063    00              TH$:    DB      00              ;****MUST BE PUT IN BY MAG      ......[1]
                        .LIST
469D    F3              START:  DI                              ;DISABLE
469E    31 43B3                 LD      SP,STACK                ;MAKE SURE STACK POINTER O.K.

                        ;*********************************************
                        ;This must be included up to the line of &&&&&                  ......[2]
                        ;
                        ;*********************************************
46A1    21 43B5                 LD      HL,REGSET               ;REGISTER DATA **
46A4    01 0880                 LD      BC,0880H                ;B=COUNT C=REGISTER NO. +80H
                                                                ;COS BIT 7 MUST BE SET
46A7                    INITLP:
46A7    7E                      LD      A,(HL)
46A8    D3 02                   OUT     (02),A                  ;SEND REGISTER DATA 1ST
46AA    79                      LD      A,C                     ;SEND REGISTER NUMBER
46AB    D3 02                   OUT     (02),A
46AD    23                      INC     HL                      ;BUMP DATA POINTER
46AE    0C                      INC     C                       ;BUMP REGISTER NUMBER
46AF    10 F6                   DJNZ    INITLP

                        ;&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

46B1    CD 4172                 CALL    62INIT                  ;INITIALISE SCREEN
46B4    CD 47AA                 CALL    CLS                     ;MAKE SURE ITS CLEARED
46B7    CD 4B66                 CALL    SETBRD          ;SET UP BOARD

46BA    CD 493F                 CALL    SOFF                    ;Make sure sound off    ......[3]
46BD    21 40DB         STR:    LD      HL,MES4                 ;'Do you ant to go first?'
46C0    CD 4918                 CALL    INPUT                   ;Returns with Ascii in A reg
46C3    FE 59                   CP      'Y'
46C5    28 1F                   JR      Z,L5060
46C7    FE 79                   CP      'y'
46C9    28 1B                   JR      Z,L5060
46CB    FE 4E                   CP      'N'
46CD    CA 47AA                 JP      Z,L6010
46D0    FE 6E                   CP      'n'
46D2    CA 47AA                 JP      Z,L6010
46D5    18 E6                   JR      STR


                        ;
46D7    3E 5B                   LD      A,91            ;Lefthand  ascii value of H$
46D9    32 4067                 LD      (H$),A
46DC    3E 5D                   LD      A,93            ;Ascii Value of TH$
46DE    32 406B                 LD      (TH$),A
46E1    3E 5E                   LD      A,94            ;Lefthand value of C$
46E3    32 4069                 LD      (C$),A          ;All The above are used to test which
                                                        ;Values are in the arrays
                                                        ;However when they are printed to the
                                                        ;the screen they will have to have their
                                                        ;Righthand value added because all chars
                                                        ;are made up of two character cells.
```

```
                      ;********************************

                      ;Human routine

                      ;********************************

46E6                  L5060:
46E6    CD 493F               CALL    SOFF              ;Turn sound off .. Just incase
46E9    21 410F               LD      HL,YRGO           ;'Pick a number ...' message
46EC    CD 4918               CALL    INPUT             ;Input routine will place csr and
                                                        ;print message and return with
                                                        ;key pressed in A register
46EF    D6 30                 SUB     30H               ;Strip ascii to leave actual number
46F1    28 04                 JR      Z,ERR             ;If zero then error must be 1-8
46F3    FE 09                 CP      9                 ;If >8 then error remember ...
46F5    38 3E                 JR      C,L5130           ;testing for C means less than 9
46F7                  ERR:
46F7    DD 36 00 05           LD      (IX+00H),5        ;X pos
46FB    DD 36 01 16           LD      (IX+01H),22       ;Y pos
46FF    21 4152               LD      HL,SPC            ;Clear message line
4702    CD 47EA               CALL    PRINT
4705    DD 36 00 05           LD      (IX+00H),5
4709    DD 36 01 16           LD      (IX+01H),22       ;PUT CSR BACK TO START
470D    11 0608               LD      DE,0608H          ;COMMAND BYTE IN D PARAMETER IN E
4710    CD 490D               CALL    COMSCR            ;This routine interfaces to Ksub1
                                                        ;All we are doing is sending ink
                                                        ;colour 8 to Vdp (6 = ink colour)
                                                        ;in Ksub1 ... well you might have
                                                        ;forgotten !
4713    21 40FF               LD      HL,ILEGAL         ;'Illegal input...'
4716    CD 47EA               CALL    PRINT
4719    3E 01                 LD      A,1
471B    32 FE14               LD      (0FE14H),A        ;Channel number
471E    21 0384               LD      HL,900            ;FREQUENCY
4721    22 FE16               LD      (0FE16H),HL
4724    3E 0F                 LD      A,15              ;Full volume
4726    32 FE18               LD      (0FE18H),A
4729    CD 08F6               CALL    08F6H             ;Make a sound with a rom call
472C    CD 4902               CALL    DELAY             ;Wait a bit
472F    CD 493F               CALL    SOFF              ;Now turn it off
4732    C3 46E6               JP      L5060             ;NOW go and get a proper input ..
                                                        ;hopefully !

                      ;********************************
                      ;When input correct jumps to here
                      ;********************************

4735                  L5130:
4735    21 403E               LD      HL,DIMR           ;R()
4738    3D                    DEC     A                 ;Dec 1 for access to arrays
4739    5F                    LD      E,A
473A    16 00                 LD      D,0

473C    19                    ADD     HL,DE             ;Now at correct place in array R(x)
473D    7E                    LD      A,(HL)            ;R(X)
473E    FE 08                 CP      8
4740    30 B5                 JR      NC,ERR            ;>7 then error remember NC  = or >
```

```
                                                          ;so test for value plus 1
4742   3C                         INC    A                ;R = R+1
4743   77                         LD     (HL),A           ;R(X)=R+1
4744   32 4078                    LD     (RV),A
4747   21 400A                    LD     HL,DIMGC
474A   19                         ADD    HL,DE            ;GC(X)
474B   7E                         LD     A,(HL)
474C   DD 77 00                   LD     (IX+00H),A       ;E=GC(X)
474F   D5                         PUSH   DE               ;SAVE DE FOR A MO
4750   3A 4078                    LD     A,(RV)
4753   5F                         LD     E,A
4754   21 4012                    LD     HL,DIMDN
4757   19                         ADD    HL,DE            ;DN(R)
4758   7E                         LD     A,(HL)           ;
4759   DD 77 01                   LD     (IX+01H),A       ;F=DN(R)
                                                          ;Screen position now set up
475C   D1                         POP    DE
475D   21 404E                    LD     HL,T6$TAB
4760   CB 23                      SLA    E                ;Must mul by 2 for address access
4762   19                         ADD    HL,DE            ;We are now getting the correct
                                                          ;address in memory of T$TAB( ,x)
4763   5E                         LD     E,(HL)
4764   23                         INC    HL
4765   56                         LD     D,(HL)
4766   3A 4078                    LD     A,(RV)
4769   6F                         LD     L,A
476A   26 00                      LD     H,0
476C   19                         ADD    HL,DE            ;T$TAB(R,X)
476D   3A 4067                    LD     A,(H$)           ;HUMAN COUNTER
4770   32 4068                    LD     (X$),A           ;X$ =H$
4773   77                         LD     (HL),A           ;T$TAB(R,X)=H$
4774   F5                         PUSH   AF               ;SAVE AF cos Ksub1 doesn't preserve
4775   11 0603                    LD     DE,0603H         ;INK 3 for human counter
4778   CD 490D                    CALL   COMSCR
477B   CD 419A                    CALL   KSUB1            ;Now print first half on screen
477E   F1                         POP    AF
477F   3C                         INC    A                ;Next half of counter
4780   CD 419A                    CALL   KSUB1            ;And print it on screen
4783   3E 01                      LD     A,1
4785   32 FE14                    LD     (0FE14H),A
4788   3E 64                      LD     A,100
478A   32 FE16                    LD     (0FE16H),A
478D   3E 0F                      LD     A,15
478F   32 FE18                    LD     (0FE18H),A
4792   CD 00F6                    CALL   00F6H
4795   CD 494D                    CALL   L60
4798   06 00                      LD     B,0              ;FOR I = 1 TO 4
479A   21 403A                    LD     HL,DIMA
479D                      DIMLP:
479D   7E                         LD     A,(HL)
479E   FE 04                      CP     04H
47A0   D2 494D                    JP     NC,HUWON         ;IF A(I)>4

47A3   23                         INC    HL
47A4   04                         INC    B
47A5   78                         LD     A,B
47A6   FE 04                      CP     4                ;NEXT I
47A8   20 F3                      JR     NZ,DIMLP
```

```
                   ;****************************************
                   ;Computer Routine ....  NEXT MONTH
                   ;****************************************
47AA               L6010:




                   .LIST
                                                                        ......[4]
4902               DELAY:
4902   D9                   EXX                 ;preserve registers by using alternatives
4903   01 6000              LD      BC,6000H    ;Delay count
4906               DEL1:
4906   0B                   DEC     BC
4907   79                   LD      A,C
4908   B0                   OR      B
4909   20 FB                JR      NZ,DEL1

490B   D9                   EXX                 ;Get standard regs back
490C   C9                   RET


490D               COMSCR:

                   ;Send command in D to Ksub1 with parameter in E

490D   F5                   PUSH    AF
490E   7A                   LD      A,D         ;Command byte 3,4,6 etc
490F   CD 419A              CALL    KSUB1
4912   7B                   LD      A,E         ;Parameter Ink, csr x,y etc
4913   CD 419A              CALL    KSUB1
4916   F1                   POP     AF
4917   C9                   RET


4918               INPUT:
4918   E5                   PUSH    HL                      ;Save message pointer
4919   DD 36 00 05          LD      (IX+00H),5              ;X POS
491D   DD 36 01 16          LD      (IX+01H),22             ;YPOS
4921   21 4152              LD      HL,SPC
4924   CD 47EA              CALL    PRINT                   ;Clear command line
4927   DD 36 00 05          LD      (IX+00H),5
492B   DD 36 01 16          LD      (IX+01H),22
492F   11 0604              LD      DE,0604H
4932   CD 490D              CALL    COMSCR
4935   E1                   POP     HL
4936   CD 47EA              CALL    PRINT
4939   CD 0079     SCAN:    CALL    0079H            ;Keyboard scan returns ascii in A
493C   28 FB                JR      Z,SCAN           ;Zero flag set if no pressed
493E   C9                   RET

493F               SOFF:

                   ;Turn sound on channel 1 off

493F   3E 01                LD      A,1
4941   32 FE14              LD      (0FE14H),A
4944   3E 00                LD      A,0
4946   32 FE1B              LD      (0FE1BH),A
4949   CD 08F6              CALL    08F6H
494C   C9                   RET
```

Number 11 **MEMOPAD** Volume Three

The Complete Price List

Hardware

DESCRIPTION	MEMBERS PRICE	NON MEMBERS PRICE	CARRIAGE
COMPLETE CP./M PACKAGE			
1 X 1 MBYTE 3.5" INDUSTRY STANDARD DISC DRIVE, 500K FAST ACCESS RAM DISC CP/M 2.2 OPERATING SYSTEM. 256K RAM. 12" GREEN SCREEN MONITOR CENTRONICS STANDARD PRINTER I/F POSITIVE ACTION KEYBOARD. COLOUR MONITOR OUTPUT. TWO JOYSTICK I/F.	359.95	399.95	20.00
PURCHASES INDIVIDUALLY (basic system)			
256K COMPUTER PLUS TAPE OPERATING SYSTEM	89.95	99.95	10.00
CP/M SYSTEM			
1 X 1 MBYTE 3.5" DRIVE + 512 SILICON DISC + 80 COL + CP/M + N.W.	237.59	264.00	10.00
HX 12" GREEN SCREEN MONITOR	85.49	95.00	10.00
TWIN RS232 INTERFACE (UPGRADE)	26.96	29.95	3.00
FDX 2 X 1 MBYTE CP/M + 2 MBYTE SILICON DISC.	877.50	975.00	10.00
32K MEMORY EXPANSION	37.95	39.95	3.00
64K MEMORY EXPANSION	47.45	49.95	3.00
128K MEMORY EXPANSION	75.95	79.95	3.00
NEWWORD ON ROM	37.95	39.95	3.00
PASCAL ON ROM	37.95	39.95	3.00
RS232 INTERFACE (FULL BOARD)	37.95	39.95	3.00

SIDISC PRICES

1 X 1 MBYTE	161.10	179.00	10.00
1 X 2 MBYTE	304.20	338.00	10.00
1 X 3 MBYTE	542.40	636.00	10.00
PRINTER CABLE	11.65	12.95	0.50

SPECIAL NOTES

Silicon Discs can be factory fitted for an extra £30.00 (U.K. Only).
FDX Twin Systems require RS232 Comms Board.
Carriage is applicable to U.K. orders only.
Always quote the type of computer owned Eg: MTX 500, 512 or series 2 when ordering add-on's.

** PLEASE NOTE ALL PRODUCTS WHICH ARE NOT MENTIONED ON THIS LIST ARE NO LONGER AVAILABLE **

THE ISSUE OF THIS PRICE LIST CANCELS ALL PREVIOUS OFFERS

ANUALS

CRIB CARDS	£1.50	ROM CALLS INFO SHEET	0.50
RST10 CALLS INFO SHEET	0.50	INTERRUPTS INFO SHEET	0.50
MTX SERVICE MANUAL	£9.95	MTX NEW USER MANUAL	£8.95
V.D.P. MANUAL	£7.95	D.D.T. MANUAL	£2.50

DISC BASED LISTINGS

MTX ROM LISTING	5.25"	-	£15.95	3.5"	-	£21.95
SDX DISC CONTROLLER	5.25"	-	£9.95	3.5"	-	£12.95

ADVERTISING IN THE MEMOPAD

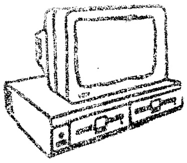
SMALL ADVERT.	£5.00	1/8 PAGE	£27.50
1/4 PAGE	£45.00	1/2 PAGE	£80.00

THE COMPLETE HOME COMPUTER MAINTENANCE KIT

CASSETTE DEMAGNETISER
SCREEN + KEYBOARD CLEANER CLOTHS
CASSETTE HEAD CLEANER
DUST BLOWER
CLEANING WANDS

PACKAGED IN A PRESENTATION CASE
(MAKES A SUPER GIFT)

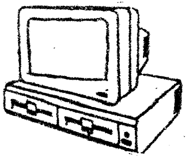
£21.50 + 50P P+P



Number 11

MEMOPAD

Volume Three



Software

U = Utility
G = Game

E = Educational
J = Joystick Compatible

L = Language
B = Business

3D TACHYON FIGHTER	G+J	ANY	6.95
AGROVATOR	G	ANY	5.95
ALICE	G	ANY	6.95
ASTROMILON	G+J	ANY	6.95
ASSEM LANGUAGE COURSE	E	512	8.95
ASTROPAC	G+J	ANY	6.95
KILLER TOMATOES	G	512	7.95
BLOBBO	G+J	ANY	6.95
BOUNCING BILL	G+J	ANY	5.95
BRIDGE	G	ANY	6.95
CAVES OF ORB	G	512	5.95
CHAMBEROIDS	G+J	512	6.95
CHESS	G	512	8.95
COMBAT	G	512	3.95
CRYSTAL	G	512	6.95
DISASM	U	512	7.95
DOODLEBUG	G+J	ANY	5.95
DOWNSTREAM DANGER	G+J	512	6.95
DR. FRANKIE	G+J	ANY	5.95
DRIVE THE CEE 5	G+J	ANY	6.95
EDASM	U	512	7.95
EDASM SDX (DISC)	U	512	8.95
EMERALD ISLE	G	ANY	6.95
ESCAPE FROM ZARCOS	G+J	512	6.95
EXTENDED BASIC	U	ANY	7.45
F1 SIMULATOR	G+J	512	4.95
FATHOMS DEEP	G+J	512	6.95
FIG FORTH	L	512	15.75
FIG FORTH SDX (DISC)	L	512	15.75
FIREHOUSE FREDDIE	G+J	512	6.95
FIRST LETTERS 1	E	512	8.95
FLUMMOX	G+J	512	6.95
FOOTBALL MANAGER	G	ANY	6.95
FOOTBALL MAN. 3.5"	G	512	9.95
FOOTBAL MAN. 5.25"	G	512	8.25
GHOSTLY CASTLE	G	ANY	3.45
GOLDMINE	G+J	ANY	6.95
GRAPHICS	U	ANY	5.95
HEI-MATHS	E	512	7.95
HIGHWAY ENCOUNTER	G+J	512	7.95
HUNCHY	G+J	ANY	5.95
ICEBERG	G+J	ANY	5.95
JUMPING JACK FLASH	G+J	512	5.95
KARATE KING	G+J	ANY	6.95
KILOPEDE	G+J	ANY	6.95
KNUCKLES	G+J	ANY	7.95
LITTLE DEVILS	G+J	ANY	5.95
MISSILE COMMAND	G+J	ANY	5.95
MATHS 1	E	512	8.95
MAXIMA	G+J	ANY	6.95
MEMOCHEQUE	U	512	6.95
MEMOSKETCH	U+J	ANY	7.95
MEMOSKETCH SDX	U+J	512	8.95
MISSION ALPHA.	G+J	ANY	5.95
MISSION OMEGA	G+J	ANY	5.95
NEMO	G+J	ANY	6.95
OBLIT. ZONE	G+J	ANY	6.95
OBLOIDS	G+J	ANY	6.95
PAINTBOX	U	512	5.95
PAYROLL	B	512	21.25
PHAID	G+J	ANY	6.95
PONTOON	G	ANY	6.95
POT HOLE PETE	G+J	ANY	6.95
PURCHASE LEDGER	B	512	12.75
QOGO	G+J	ANY	6.95
QOGO 2	G+J	512	6.95
QUANTUM	G+J	ANY	5.95
QUAZZIA	G+J	512	6.95
QUEST 1	G	ANY	5.20
REVERSI	G	512	7.95
ROLLA BEARING	G+J	512	6.95
RUTHLESS B.	G	512	3.45
SALES LEDGER	B	512	15.75
SALTY SAM	G+J	ANY	5.95
SEPULCRI	G	512	6.95
SMG	G+J	512	6.95
SNAPPO	G+J	ANY	6.95
SNOOKER	G	512	7.95
SON OF PETE	G+J	ANY	6.95
SUPA CODER	U	512	7.95
SUPER BIKE	G+J	ANY	5.95
SUPER MINEFIELD	G	ANY	6.95
SURFACE SCANNER	G+J	512	6.95
TAPE TO DISC	U	512	6.95
TAPEWORM	G+J	ANY	6.95
TARGET ZONE	G+J	512	6.95
THE WALL	G+J	512	5.95
TOADO	G+J	ANY	6.95
TURBO	G+J	ANY	6.95
USER BASIC	U	512	8.96
USER BASIC (SDX)	U	512	9.95
UTILITIES (SDX)	U	512	9.95
VERNON & VAMPS.	G	ANY	5.95
WORD AND PICTURE	E	512	8.95

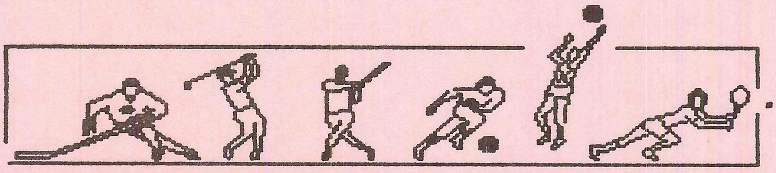
PLEASE NOTE It is not always possible to put tape based software on disc, please check with our sales staff before ordering any of the above on 5.25" or 3.5". If the software will transfer a charge of £1.50 for 5.25" and £3.00 for 3.5" will be added to the tape price.

DISC BASED SOFTWARE

MEMOSKETCH	3.5"	(CPM)	£9.95	5.25"	(CPM)	£7.95
ROLLA BEARING	3.5"		£9.95	5.25"		£8.99
FAB. FIVE	3.5"	(CPM)	£12.95	5.25"	(CPM)	£7.99
FRANTIC FREDDIE	3.5"		£9.95	5.25"		£8.99
DISC ONE	3.5"	(CPM)	£11.95	5.25"	(CPM)	£7.99
DISC TWO	3.5"	(CPM)	£11.95	5.25"	(CPM)	£7.99
DISC THREE	3.5"	(CPM)	£10.95	5.25"	(CPM)	£7.50
DISC FOUR	3.5"	(CPM)	£9.50	5.25"	(CPM)	£7.50

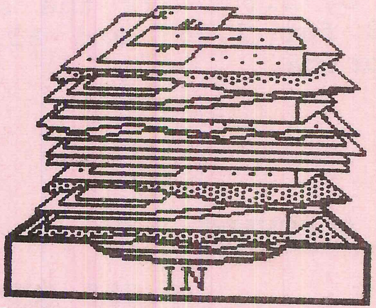
SUNDRIES

DISC HEAD CLEANER	3.5"	-	£18.25	5.25"	-	£16.95
HOME COMPUTER MAINTAINANCE KIT		-	£21.50	+ 50p	P+P	
SMALL DISC BOX (HOLDS 40)		-	£16.00	+ 75p	P+P	
LARGE DISC BOX (HOLDS 80)		-	£19.95	+ 75p	P+P	



TOURNAMENT SNOOKER............

SPECIAL OFFER TO ALL MEMBERS
FOR THIS MONTH ONLY BUY THIS EXCELLENT
GAME FOR ONLY 4.95p and save 3.00p !!!



NOW AVAILABLE
FOR TAPE BASED
SYSTEMS.... ....

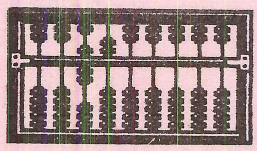
PRO-WORD

WORD PROCESSOR

A must for all serious MTX user.

Full Editing Facilities

MTX 512 only 12.95p The Best !



SELF TEST

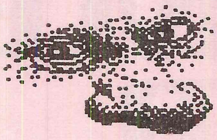
FOR TAPE & DISC SYSTEMS ....

This utility will allow you to carry out tests on your equipment ......ideal for finding those hidden faults ......

DISC VERSION 9.95p
TAPE VERSION 7.95p

PLEASE STATE DISC SYSTEM & FORMAT
WHEN ORDERING e.g 5.25 500k cpm.

ORION SOFTWARE THE NORTHBRIDGE CENTRE
ELM STREET, BURNLEY BB10 1PD 0282831695



FOR SALE

MTX 500 WITH 32k MEMORY EXPANSION
NEWWORD ON ROM : ASSORTED GAMES
USER MANUAL : CASSETTE RECORDER
PLUS AN ASSORTMENT OF GAMES
ANY REASONABLE OFFER CONSIDERED
Call *Olive* on 091 5286839 (after 6pm)