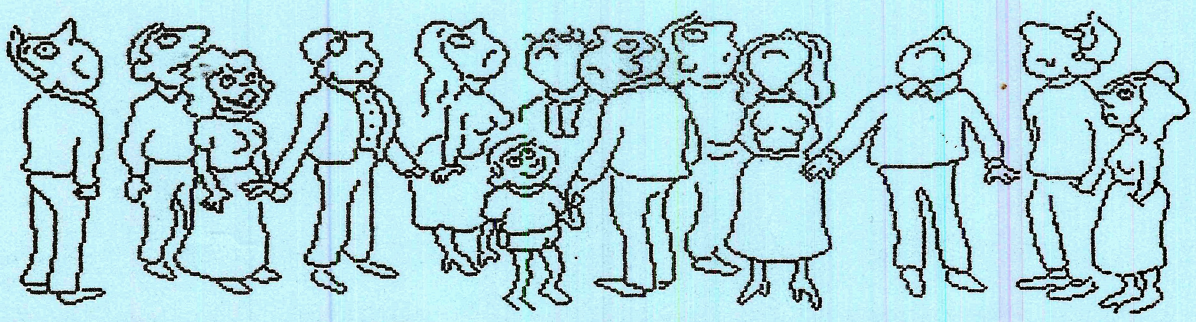
MEMOPAD

THE OFFICIAL USER MAGAZINE OF M.C.L

FOR MEMOTECH COMPUTER USERS WORLD WIDE

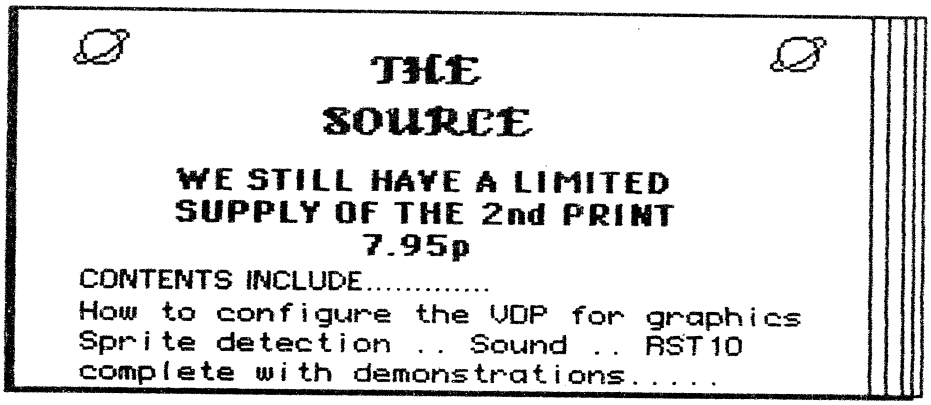
CONTENTS

VOLUME THREE ISSUE TWELVE



Cover Price 1.45p PUBLISHED BY ORION SOFTWARE

MEMOPAD



THE DISC OF PROGRAMS FROM *THE SOURCE* NOW AVAILABLE ....*LET YOUR FINGERS HAVE A REST .....GET IT RIGHT FIRST TIME !* 9.95p

CONTACT III
NOW AVAILABLE

NO ADD ON HARDWARE IS REQUIRED

CONTACT *PRESTEL : MICRONET 800 ETC....*

SUITABLE FOR ANY MODEM WHICH UTILISES 1200/1200 AT FULL DUPLEX .... TANDATA, WS2000 ETC.

HANDLES ALL GRAPHIC CODES EXCEPT DOUBLE HEIGHT AND WILL RECEIVE IN COLOUR OR MONOCHROME. AUTO-DIAL FUNCTION & MANY MORE USEFUL UTILITIES SUCH AS FACILITY TO LOCALLY EDIT SCREEN AND SAVE YOUR FAVOURITE SCREENS TO DISC.

CPM MODELS ONLY. PLEASE STATE CONFIGURATION 14.95p **INCLUSIVE.**

FOR SALE.....

SHOP SOILED TANDATA 110 MODEM *TRANSMITS AT 1200 FOR PRESTEL COMMUNICATION* **69.00 PLUS 5.00 POSTAGE & PACKING**

TANDATA 510 MODEM NORMALLY SELLS AT 259.00 AUTO DIAL & HAYES COMPATIBLE. 300/1200/1200/1200/75 ETC BRAND NEW ! 160.00p PLUS 5.00 POSTAGE & PACKING.

MEMOPAD

Editorial

Hello Readers,

Here I am again! Did you miss me?

At last! All outstanding orders of The Source have been despatched. Those of you who have not ordered a copy, and would like to do so, can still order as we have a limited supply on our shelves. It is recommended that all members have the book as articles and programs published in the magazine will be using The Source as a point of reference as will the technical department.

The new modem software Contact 3 is now available, we will be publishing a full review in the next issue or you can ring the technical department and they will tell you everything you need to know.

You may be wondering how we have managed to produce the smart new layout used on the front cover and advertisements, the program is the Fleet Street Editor but before you all rush to order your copy I must tell you that the software is only available for the IBM, it does give you an idea of what may come in the future should some large and kindly software house smile upon us.

I am also hoping to publish a review of Pro-Word in the next issue, this has received an excellent reception and is of the highest standard - considering the limitations of tape based software.

I make no apologies for the fact that I am plugging the new releases but we are proud that we have delivered the goods. I hope it has convinced members that we at Orion keep our promises!

Sue

ORION SOFTWARE
THE NORTHBRIDGE CENTRE
ELM STREET, BURNLEY
LANCS. BB10 1PD
TELEPHONE - 0282 831695

AVAILABLE BY SUBSCRIPTION ONLY

ACCESS AND BARCLACARD WELCOME

Video Display Panel

Have you ever been trying to develop a program and find it just doesn't do as it should? You try using Panel to trace through but get stuck when it comes to the video display section, you are unable to see if the correct bits have been set or if your character set is in the correct place and is looking OK!!

Well here is a utility which may be able to help you out. It is rather long and could be improved but the original idea was to cause as little intrusion to the system as possible, hence the extra routines for reading in values etc.

The program works by copying into RAM (CPU) a small section of the video RAM, the program reads this as if it were video RAM and copies it back afterwards. This space in the video RAM is then used to set up the VDP in 'Graphics Mode 1' (the one not supported by MTX basic), the program then works along the lines of Panel by displaying a block along the bottom half of the screen with the associated make up of those bytes on the last column ie. if you were looking at the data for the '£' sign this would appear at the end of the 8 bytes, this allows a quick scan through the data.

Options included in this version are 'Cell' and 'Display' these being the basic ones allowing for any inspection or alterations users may wish to do, it is easy to add extra commands by inserting compare instructions followed by respective jump at the main program loop. Others such as Find, Move and Register setup may prove may prove useful. Display performs the same as panel, you enter an address and the program responds with a value which you may then change if you wish. Cell allows you, by use of the cursor keys or joystick, to modify or create any VDP data such as sprites, character sets etc, a blown up cell is displayed with a cursor which you can move around, pressing the fire or home key will toggle the respective bit. The cell displayed may be either a single cell or 4 cells together in a block, this block may represent a sprite block, Graphics mode 2 block, user defined or sequential block of cells.

Well that is basically it, try it our, if you find it useful you could install it to initiate under Panel as an extension or a user or error command.

The version shown has been done using the EDASM on an MTX512 if you do not have the EDASM program (BUY IT!) it could possibly be done under Basic Assembly but it will not have the portability and after all that typing it seems a waste.

Dave Buck.

```
          ORG 0D000H
          LOAD 0D000H


 ;     Vidio Display Processor - panel type program
 ;         (TMS 9929A)
 ;      to try it out exit and enter
 ;       10 RAND USER(53248)
 ;      in basic compile this code exit and type RUN
 ;      use the 'ESC' key to return to basic


 ;Initalise the screen
 ;  1. copy all VRAM from 3800H to RAM
 ;  2. blank scren and setup VDP registers
 ;       Reg 0 = 00000000B = 00H  Graphic 1
 ;       Reg 1 = 10000000B = 80H  initial vals
 ;       Reg 2 = 0FH  set@ 3C00H  Name table
 ;       Reg 3 = 0FEH set@ 3F80H  Colour @ 3F80H
 ;       Reg 4 = 07H  set@ 3800H  Pattern
 ;       Reg 5 = 7EH  set@ 3F00H  Sprite att
 ;       Reg 6 = 07H  set@ 3800H  Sprite pattern
 ;       Reg 7 = 3CH LT&DRK GREEN Backdrop
 ;       Reg 8 ------STATS-REGISTER----
 ;  3. copy ASCII paterns to VRAM
 ;  4. clear screen copy SPACE No. to name table
 ;  5.
 ;Routine to set up VDP
          PUSH DE
          PUSH HL
          PUSH BC
          PUSH AF
 ; basic setup pointers
          LD HL,1918H       ;data table to display @ ASCII data in VRAM (MTX)
          LD (BYTEST),HL
          LD (BASE1),HL     ;cell also as above
          LD A,1            ;initial enlarged cellblock set to single ASCII (not 4)
          LD (SIZE),A
          LD A,8
          LD (POINT),A      ;init > in data table
          CALL SETUP        ;setup  VDP registers, save workspace to ram,copy in
                            ;character set from rom (for MTX version)

 ;=============================================================================
 ;Main start of program, scan for command
 LKB:     CALL 79H          ;look @ keyboard
          JP Z,LKB
          LD HL,(BYTEST)
          CP 27             ;ESC exit from prog
          JP Z,EXIT
          CP 11             ;up arrow !              update data table
          JP NZ,K1B
          CALL ADD8
          JP KBD
```

```
K1B:    CP  10          ;down arrow !
        JP NZ,K2B
        CALL SUB8
        JP KBD
K2B:    CP 8            ;back arrow !
        JP NZ,K3B
        CALL INCHL
        JP KBD
K3B:    CP 25           ;forward arrow !
        JP NZ,K4B
        CALL DECHL
KBD:    LD (BYTEST),HL
        CALL WRITETBL
        JP LKB
K4B:    AND 5FH         ;lower case for A-F
        CP "D"          ;Display function !
        JP NZ,K5B
        CALL CLSTH      ;clear top of screen
        LD DE,PTVDP
        CALL DPRINT     ;print text string at DE pointer
        CALL DISPLAY    ;show data table
        JP KBMENU
K5B:    CP "C"          ;Cell block edit/disp
        JP NZ,K6B
        CALL BLKCONT
        JP KBMENU
K6B:

;End of input function option loop

KBMENU: CALL CLSTH      ;clear top screen
        LD DE,MENU
        CALL DPRINT
        JP LKB
;return data
EXIT:   LD HL,STRAM     ;start address from VRAM
        CALL VAWRITE    ;set it up
        LD BC,ENDVRAM-STRAM+2    ;bytes to get
        LD HL,VINRAM
L4:     LD A,(HL)
        CALL VWRITE
        INC HL
        DEC BC
        LD A,C
        OR B
        JR NZ,L4
        CALL VAFREE
        LD A,(STATUS)   ;Restore status
        LD (0FF58H),A
        POP AF
        POP BC
        POP HL
        POP DE
        RET             ;exit back to basic
;================================================================================
```

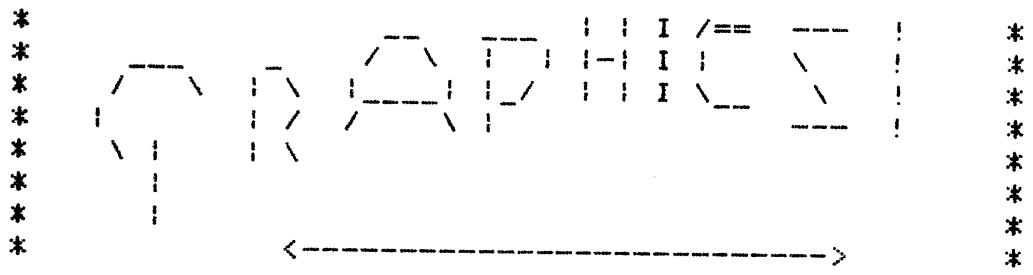
```
;Routine to setup VDP registers and screen
;on entry to VDP
SETUP:  IN A,(2)          ;dummey read to reset logic
        LD A,(0FF58H)     ;VDP STATUS STORE
        LD (STATUS),A
        LD (0FF58H),A
;stage 1 save existing VDP data to ram
        LD HL,STRAM       ;start address from VRAM
        CALL VASAFE       ;stop interupt corupting VDP
        CALL VAREAD       ;set VDP for a read
        LD BC,ENDVRAM-STRAM+2 ;bytes to get
        LD HL,VINRAM      ;copy VRAM to ram @ HL
L1:     CALL VREAD
        LD (HL),A
        INC HL
        DEC BC
        LD A,C
        OR B
        JR NZ,L1
;stage 2 blank VDP and load registers
        LD HL,REGDTA      ;data table
        LD BC,0880H       ;count,bit-set
L2:     LD A,(HL)         ;get REG data
        OUT (2),A         ;send it
        LD A,C            ;get REG No.
        OUT (2),A         ;send it
        INC C             ;next reg
        INC HL            ;next data
        DJNZ L2
;stage 3 ASCII paterns to VRAM
        CALL ASCTBL       ;setup ASCII characters
        CALL COLTBLE      ;setup coloures
;end into sprite attribute table
        LD HL,3F00H
        CALL VAWRITE
        LD A,208          ;clear all sprites
        CALL VWRITE
;clear name table by sending space chr
        LD HL,3C00H       ;base of name table
        CALL VAWRITE
        LD BC,0003H
        LD A,20H          ;Clear screen
L3:     CALL VWRITE
        DJNZ L3
        DEC C
        JP NZ,L3
        CALL WRITETBL
        LD DE,PTDTA
        CALL DPRINT
        LD DE,MENU
        CALL DPRINT
;now unblank screen
        LD A,11000000B
        OUT (2),A
        LD A,81H
        OUT (2),A
        RET
```

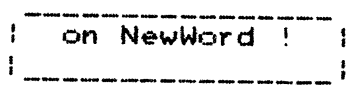
```
;=============================================================================
;Routine to setup VDP screen character set

ASCTBL: LD A,(PAGE)      ;memory set up
        PUSH AF          ;save it
        AND 0FH          ;clear existing ROM/CPM
        OR 10H           ;set ROM 1
        LD (PAGE),A
        OUT (0),A
        LD HL,3880H      ;Pattern 70H
        CALL VAWRITE
        LD DE,CELLNO1
        LD B,0FFH        ;Cell patterns
ASC1:   LD A,(DE)        ;NO's 10-1FH
        CALL VWRITE
        INC DE
        DJNZ ASC1
        LD HL,3800H      ;sprite generator table
        CALL VAWRITE
        LD HL,SPRTDTA
        LD B,8
ASC2:   LD A,(HL)        ;set up sprite cursor
        CALL VWRITE      ;for block function
        INC HL
        DJNZ ASC2
        LD HL,3900H      ;address of section pattern table to use
        CALL VAWRITE     ;setup address to send it
        LD B,5FH         ;now do ASCII
        LD DE,0          ;start @ " " = pat no. 0
        CALL CHAR
        POP AF
        LD (PAGE),A
        OUT (0),A
        RET
;Routine to get 5 bytes from ROM which represent
;the 8 byte ASCII character held in DE
;then rotate and store in vram at previousley
;set up VRAM address
CHAR:   PUSH BC
        PUSH DE
        LD HL,ASCII      ;point to ROM base address of data
        LD B,5
CHAR1:  ADD HL,DE
        DJNZ CHAR1
        POP DE
        POP BC
CHAR2:  PUSH BC
        PUSH HL
        LD C,(HL)        ;load 1st byte
        INC HL
        LD E,(HL)        ;2nd byte
        INC HL
        LD D,(HL)        ;3rd byte
        INC HL
        LD A,(HL)        ;4th byte
        INC HL
```

```
        LD H,(HL)       ;5th byte
        LD L,A
        LD B,8          ;number of bytes to send
CHAR8:  XOR A           ;clear A
        RL H            ;rotate bits through carry
        RR A
        RL L
        RRA
        RL D
        RRA
        RL E
        RRA
        RL C
        RRA
        OUT (1),A       ;send to VDP
        DJNZ CHAR8      ;do for all 8 bytes
        POP HL          ;prev start
        LD C,5          ;inc by 5
        ADD HL,BC
        POP BC          ;count to do
        DJNZ CHAR2      ;exit if all done
        RET
;=============================================================================
;Routine to set up colour table
COLTBLE:; set up VDP
        LD HL,3F80H
        CALL VAWRITE
        LD A,13H
        LD B,32
COL1:   CALL VWRITE     ;set all table to BLACK letters on
        DJNZ COL1
        LD HL,3F81H     ;set 2 bytes
        CALL VAWRITE    ;Patterns 8 - FH
        LD A,1AH        ;cell block
        CALL VWRITE
        LD A,0FAH
        CALL VWRITE
        CALL VWRITE
        RET
;end of setup
;=============================================================================
VAREAD: ;set up VRAM for read at (HL)
        PUSH AF
        LD A,L
        OUT (2),A
        LD A,H
        OUT (2),A
        POP AF
        RET
;-----------------
VREAD:  IN A,(1)
        RET
;-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

CONTINUED IN NEXT ISSUE

MEMOPAD





No,it won't win any prizes for art-work,but it will let you type in charts,tables,graphs and simple diagrams of the "push A into B and then screw it to C" variety.Since the only alternatives end up with a lot of expensive photocopying it may save time and money and also look better if your hand-drawing isn't very good.

The trick is to start off with a page full of ASCII space characters (rather then the ASCII nulls which normally represent an empty Newword page).You then turn the INSERT off (^V) and can then use the cursor keys or a trackball to move freely over the screen adding characters where you like - rather like the DSI command of MTX basic.If your printer supports special graphic symbols you can assign them to the 4 Newword user-defined characters.Alternatively you can create a small *.COM file to switch character sets or to download a special set and call it with Newword's R(un) command.

Typing a page full of blanks is VERY BORING!I would suggest saving a copy before you use it.If you are likely to do this often in different formats you may find the following short C language program useful - it takes a command-line filename,asks for the margins and page length and creates a formatted file of spaces with a preceding automatic ruler line.

```
/* filename  MAKEPAGE.C                    */
/* B.L.HOUGHTON Last update      5/9/87          */
/* creates a Wordstar file of ASCII blanks with startng
   automatic  ruler line                        */
/* Usage :-  MAKEPAGE >FILENAME                                  */

#include  "stdio.h"

main()
{

int left,right,lines,j;
register int i;
char   ruler_line[135], blank_line[135];
```

MEMOPAD

```
        do
{
        fprintf (stderr,"Input left margin,right margin ");
        scanf ("%d,%d",&left,&right);
        if (left >= right) fprintf(stderr,"?!!\n");
}
        while (left >= right);
        fprintf(stderr,"How  many lines per page? ");
        scanf ("%d",&lines);

        for (i=0;i<right;i++)
 {
        if (i < left) ruler_line[i] = ' ';
        else
 {
        if (i %  5) ruler_line[i] = '-';
        else  ruler_line[i] = '!';
 };
 }
        ruler_line[left] = 'L';
        ruler_line[right] = 'R';
        ruler_line[0] = '.';
        ruler_line[1] = 'R';
        ruler_line[2] = 'R';
        ruler_line[right + 1] = '\0';
        printf ("%s\n\n",ruler_line);
        for (i=0;i < right;i++)
        blank_line[i] = ' ';
        for (i=0;i < left;i++)
        blank_line[i] = 0xa0;  /* Newword margin is chr(#20 OR #80) */
        blank_line[right] = 0;
        for (i=0;i<lines;i++)
        printf("%s\n",blank_line);
        putchar ('\n');
}
```

Floating Point Numbers

The two excellent articles by John Hudson[1] are well-thumbed and dog-eared by now. They describe f.p. number storage in the MTX as well as ROM calls for carrying out calculations, and their reading is highly recommended. Herewith some further notes made during a recent brain-bend.
Although the MTX uses 5-bytes to store f.p. numbers, it uses a sixth byte for some calculations. The transfer of numbers from any location to ACC or OP1 is accomplished by calls to 12A7 and 12B9 respectively. Inspection of these codes shows that the sixth byte is initialised to zero:

ACC FDCB	ACC1 FDCC	FDCD	FDCE	FDCF	FDD0	OP1 FDD1	OPII FDD2	FDD3	FDD4	FDD5	FDD6
S	M4	M3	M2	M1	E	S	M4	M3	M2	M1	E

Fig 1 Two numbers lodged in ACC and OP1 respectively. E is the exponent, M1-M4 are the 4 bytes of the mantissa, and S is the sixth byte.

Moving numbers from one location to another is normally achieved by use of available ROM calls. If,however,there is a need to reduce processing time (during iterative operations perhaps) then it is speedier to use:

```
LD HL,TEMP1
LD DE,FDCBh    ;ACC
LD BC,6
LDIR

TEMP1: DB 0
TEMP1A:DS 5
```

Note how the temporary area is declared as six bytes.

Comparisons
There is a comparison routine at 1163h which is very powerful. The result appears in Register <A> as follows:

(DE) > (HL)	<A> = 1
(DE) = (HL)	<A> = 0
(DE) < (HL)	<A> =-1

Examination of this code reveals that DE and HL must point to M1 of the respective numbers before the routine is called. This is because the routine first examines the sign bits (msb of M1), then the exponents (E), then, if necessary, compares the bytes of the mantissae. If you are using the routine to compare numbers in your own locations, you must ensure that DE,HL point at the 2nd bytes (M1) and that provision is made for a sixth byte. An effective and fast way of doing this is to provide a label at M1:

```
LD HL,TEMP1B    ; first number          TEMP1:  DB 0 ; sixth byte
LD DE,TEMP2B    ; second number         TEMP1A: DS 3 ; M4,M3,M2
CALL 1163h      ; result in <A>         TEMP1B: DS 2 ; M1,E
```

There are two lines before 1163h which point HL,DE at the M1's of ACC and OP1. Thus, if the two numbers are already present in these locations, one simply makes a call to 115Dh.

Fractions

John Hudson explained the method of encoding integers into the storage format used by the MTX. Fractions are dealt with in a similar manner, by extending the binary representation to include negative powers of 2 :

2^6	2^5	2^4	2^3	2^2	2^1	2^0	ϕ 2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}
0	0	1	0	0	1	1	0	1	0	0	0	0

Fig 2 Representation of the decimal number 19.25
ϕ = position of the implied decimal point

2^4 = 16.00
2^1 = 2.00
2^0 = 1.00
2^{-2} = 0.25
19.25

The MTX stores in five bytes:

a) Place the binary representation of the decimal number as in Fig 2
b) Count the number of positions from the implied decimal point (ϕ) to the leading "1". This count is signed (+) if leftwards, (-) if rightwards. It is zero if the leading "1" corresponds to 2^{-1}. Add 128 to this count, giving the exponent E.
c) Start the 4-byte mantissa with the leading "1" and extends rightwards for 32 bits.
d) Reset the leading "1" if the number is positive. Leave it set if negative.
e) Reverse the order of all bytes: M4, M3, M2, M1, E.

Here is an example:

10.75

2^4	2^3	2^2	2^1	2^0	ϕ 2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}
0	1	0	1	0	1	1	0	0	0	0	0	0

E=128+4=132=84h

0	0	1	0	1	1	0	0	0	0	0	0	→

M4=44=2Ch M2=M3=M4=0

Storage = 00,00,00,2C,84

~~~~~~

Reference

(1) "RST 28's The Facts" John Hudson Memopad Issue 12
"RST 28's The Final Conflict" John Hudson Memopad Vol 0010 No 1

~~~~~~

Database

```
320 CLS : CSR 30,0: PRINT "View Records": CSR 29,1: PRINT "=============="
330 LET N=0
340 FOR I=7 TO 16 STEP 3
350 LET N=N+1
360 CSR 20,I: PRINT F$(N);D$(N)
370 NEXT I: PRINT
380 GOSUB 1000
390 GOSUB 900
400 IF A<128 OR A>131 THEN  GOTO 390
410 IF A=128 THEN  GOTO 500
420 IF A=129 THEN  GOTO 740
430 IF A=130 THEN  GOTO 1500
440 IF T=1 THEN  GOSUB 1250
450 CLS : CSR 28,10: PRINT "Returning to Main Menu": CSR 28,12: PRINT "Please wait....."
460 USER  LOAD "MAINMENU.BAS"
500 REM Record Directory
510 CLS : CSR 31,0: PRINT "Record Directory": GOSUB 1000
520 FOR I=1 TO Z
530 PRINT I;". ";RECORD$(I,1);"  ";RECORD$(I,2): PAUSE 100
540 IF I/15<>INT(I/15) THEN  GOTO 610
550 GOSUB 1000
560 CSR 10,20: PRINT F$(1);"Continue Directory        ";F$(2);"View Record": CSR 10,21: PRINT F$(3);"Return to Menu"
570 GOSUB 900: IF A<128 OR A>130 THEN  GOTO 560
580 IF A=128 THEN  GOSUB 1100: CLS : GOTO 610
590 IF A=129 THEN  GOTO 620
600 GOTO 320
610 NEXT I: GOTO 560
620 CLS : CSR 30,0: PRINT "View Record": CSR 29,1: PRINT "============="
630 CSR 20,4: INPUT "Number of record to view? >";F: IF F<1 OR F>Z THEN  GOTO 620
640 CLS : GOSUB 1350: CSR 24,0: PRINT P$;" Record Number ";F
650 GOSUB 1000
660 FOR I=1 TO W
670 CSR 10,I+2: PRINT HEAD$(I);":": CSR 32,I+2: PRINT RECORD$(F,I)
680 NEXT I: PRINT : GOSUB 1000
690 CSR 10,20: PRINT F$(1);"View another record      ";F$(2);"Consult Directory": CSR 10,21: PRINT F$(3);"Return to Menu"
700 GOSUB 900: IF A<128 OR A>130 THEN  GOTO 690
710 IF A=128 THEN  GOTO 620
720 IF A=129 THEN  GOTO 510
730 GOTO 320
740 CLS : CSR 25,5: INPUT "Number to start? > ";S: IF S<1 OR S>Z THEN  GOTO 740
750 GOSUB 1350
760 FOR I=S TO Z: CLS : CSR 25,0: PRINT P$;" Record Number ";I
770 GOSUB 1000
780 FOR J=1 TO W: CSR 22,J+3: PRINT RECORD$(I,J): NEXT J
790 GOSUB 1000
800 CSR 24,18: PRINT F$(1);"Next Record": CSR 24,20: PRINT F$(2);"Last Record": CSR 24,22: PRINT F$(3);"Return to Menu"
810 GOSUB 900: IF A<128 OR A>130 THEN  GOTO 800
820 IF A=128 THEN  GOSUB 1300
830 IF A=129 THEN  LET I=I-2: IF I<0 THEN  LET I=I-1
```

MEMOPAD

```
840 IF A=130 THEN  GOTO 320
845 NEXT I
850 CLS : CSR 24,10: PRINT "E N D    O F   F I L E": PAUSE 2000
860 CSR 25,14: PRINT F$(1);"Return to View Records": CSR 25,16: PRINT F$(2);"Return to Menu"
870 GOSUB 900: IF A<128 OR A>129 THEN  GOTO 860
880 IF A=128 THEN  GOTO 740
890 GOTO 320
900 LET A$=INKEY$: LET A=ASC(A$): RETURN
950 LET A$=INKEY$: IF A$<>"" THEN  GOTO 950
960 LET A$=INKEY$: IF A$="" THEN  GOTO 960
970 IF A$<>"Y" AND A$<>"y" AND A$<>"n" AND A$<>"N" THEN  GOTO 950
980 RETURN
1000 FOR X=0 TO 79: PRINT "-";: NEXT X
1010 RETURN
1050 CLS : CSR 32,0: PRINT O$: CSR 31,1: PRINT "=========="
1060 CSR 25,5: PRINT "Please wait....."
1070 RETURN
1100 IF (I-1)=Z THEN  CLS : CSR 26,10: PRINT "E N D   O F   F I L E": PAUSE 2000: CLS : GOTO 560
1120 RETURN
1150 CLS : CSR 23,10: PRINT "Is file a search list?  (Y/N)": GOSUB 950
1160 IF A$="Y" OR A$="y" THEN  GOTO 1190
1170 USER OPEN£1,O$,"I"
1180 GOTO 1210
1190 USER OPEN£1,FILE$(1),"I"
1200 LET T=1
1210 RETURN
1250 CLS : CSR 22,2: PRINT "Finished with ";FILE$(1);"?  (Y/N)": GOSUB 950
1260 IF A$="n" OR A$="N" THEN  GOSUB 1400: GOTO 1280
1270 USER ERAFILE$(1)
1280 RETURN
1300 IF I=Z THEN  GOTO 850
1310 RETURN
1350 IF T=0 THEN  LET P$=O$ ELSE  LET P$=FILE$(1)
1360 RETURN
1400 CLS : CSR 20,10: PRINT "Name to store search file? ": CSR 20,12: INPUT "Maximum 8 chars ) ";NAME$: IF LEN (NAME$)>8 THEN  GOTO 1400
1410 CLS : PRINT CHR$(5): CSR 20,10: PRINT "Please wait.....": CSR 20,12: PRINT "Changing file name"
1420 USER RENNAME$=FILE$(1)
1430 USER OPEN£1,NAME$+".HDS","O"
1440 FOR X=1 TO W
1450 USER  PRINT £1,HEAD$(X)
1460 NEXT X
1470 USER CLOSE£1
1480 RETURN
1500 CLS : CSR 30,0: PRINT "View Record": CSR 29,1: PRINT "============="
1505 FOR X=5 TO 4+W: CSR 0,X: PRINT X-4;".": CSR 4,X: PRINT HEAD$(X-4): NEXT X
1510 CSR 20,10: INPUT "Which field to search? ) ";FIELD: IF FIELD<1 OR FIELD>W THEN  GOTO 1510
1520 CSR 20,12: INPUT "Enter data to search for )";SEARCH$: IF LEN (SEARCH$)>25 THEN  GOTO 1520 ELSE  LET SER=LEN (SEARCH$)
1530 FOR X=5 TO 4+W: CSR 0,X: PRINT CHR$(5): NEXT X
1540 CSR 20,10: PRINT "Please wait.....": CSR 20,12: PRINT "Searching for ";SEARCH$
1550 FOR I=1 TO Z
1560 IF LEFT$(RECORD$(1,FIELD),SER)=SEARCH$ THEN  GOTO 1600
1570 NEXT I
1580 CLS : CSR 20,10: PRINT "E N D  O F  F I L E": PAUSE 2000
1590 GOTO 320
1600 CLS : CSR 20,0: IF T=1 THEN  PRINT FILE$(1) ELSE  PRINT O$
1610 CSR 40,0: PRINT "Record No. ";I
```

```
1620 FOR X=3 TO 2+W
1630 CSR 20,X: PRINT HEAD$(X-2);":": CSR 32,X: PRINT RECORD$(I,X-2)
1640 NEXT X: GOSUB 1000
1650 CSR 10,20: PRINT "Is this the correct ";SEARCH$;" ? (Y/N)"
1660 GOSUB 950
1670 IF A$="N" OR A$="n" THEN  GOTO 1570
1680 CSR 0,20: PRINT CHR$(5): CSR 5,20: PRINT F$(1);"View another record          ";F$(2);"Return to menu"
1690 GOSUB 900
1700 IF A<128 OR A>129 THEN  GOTO 1690
1710 IF A=128 THEN  GOTO 1500
1720 GOTO 320
10 REM ################################################################################
20 REM ############################### SEARCH.BAS ###############################
30 REM ############################### VERSION 2  ###############################
40 REM ################################# D. WEMYSS  #################################
50 REM ################################  OCT 1986  ################################
60 REM ################################################################################
70 USER  SAVE "SEARCH.BAS"
80 CLEAR : VS 5: CLS : CSR 25,10: PRINT "Please wait.....": CSR 25,12: PRINT "Setting up variables"
90 DIM F$(8,14),A$(3),FILE$(2,12),C$(2,16),D$(2,20),S$(8,20),HEAD$(10,8),RECORD$(100,10,25),P$(12),O$(12),SEARCH$(25)
100 LET W=0: LET Y=0: LET T=0: LET Z=0: LET SL=0: LET SN=0
110 CLS : CSR 20,10: INPUT "Which file to search? ) ";O$: IF LEN (O$)>8 THEN  GOTO 110
120 LET FILE$(1)=O$+".SER": LET FILE$(2)=O$+".HDS"
130 GOSUB 1000
140 CSR 25,7: PRINT "Loading heading number ";W
150 USER OPEN£1,FILE$(2),"I"
160 FOR I=1 TO 10
170 USER EOF£1,210
180 LET W=W+1: CSR 48,7: PRINT W: PAUSE 100
190 USER  INPUT £1,HEAD$(I)
200 NEXT I
210 USER CLOSE£1
220 GOSUB 1050
230 GOSUB 1000
240 CSR 25,7: PRINT "Loading record number ";Y
250 FOR I=1 TO 200
260 USER EOF£1,310
270 LET Y=Y+1: CSR 47,7: PRINT Y
280 FOR N=1 TO W
290 USER  INPUT £1,RECORD$(I,N)
300 NEXT N: NEXT I
310 LET Z=Y
320 USER CLOSE£1
330 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$(7)="F7.....": LET F$(8)="F8....."
340 LET D$(1)="Search Options": LET D$(2)="Return to Main Menu"
350 LET C$(1)="Search for Data": LET C$(2)=O$+" by D.W."
360 LET S$(1)="Make a Search List": LET S$(2)="View Search List": LET S$(3)="Modify Search List"
370 LET S$(4)="Sort Search List": LET S$(5)="Print Search List": LET S$(6)="Search another file": LET S$(7)="Search same file": LET S$(8)="Return to Menu"
380 CLS : CSR 30,0: PRINT C$(1): CSR 30,2: PRINT C$(2): PRINT : GOSUB 1200
390 LET N=0
400 FOR I=7 TO 10 STEP 3
410 LET N=N+1
420 CSR 25,I: PRINT F$(N);D$(N)
430 NEXT I: PRINT : GOSUB 1200
440 GOSUB 1300: IF A<128 OR A>129 THEN  GOTO 440
```

```
450 IF A=128 THEN  GOTO 500
460 CLS : CSR 30,10: PRINT "Returning to Main Menu": CSR 30,12: PRINT "Please wait....."
470 USER  LOAD "MAINMENU.BAS"
500 CLS : GOSUB 1550
510 LET N=0
520 FOR I=3 TO 17 STEP 2: LET N=N+1
530 CSR 20,I: PRINT F$(N);S$(N)
540 NEXT I: CSR 19,20: PRINT "NOTE: Work existing search list before making a new one": PRINT : GOSUB 1200
550 GOSUB 1300: IF A<128 OR A>135 THEN  GOTO 550
560 IF A=128 THEN  GOTO 600
570 IF A=129 OR A=130 OR A=131 OR A=132 OR A=133 OR A=134 THEN  GOSUB 1350: GOTO 1600
580 GOTO 380
600 CLS : GOSUB 1550
610 FOR X=1 TO W
620 CSR 26,X+3: PRINT X;".": CSR 31,X+3: PRINT HEAD$(X)
630 NEXT X: GOSUB 1200
640 CSR 20,20: INPUT "Which field to search? > ";HEAD: IF HEAD<1 OR HEAD>W THEN  GOTO 640
650 CLS : GOSUB 1550
660 CSR 20,5: INPUT "Type data to search for > ";SEARCH$
670 LET SER=LEN (SEARCH$): IF SER>25 THEN  GOTO 660
680 GOSUB 1200
690 CSR 28,8: PRINT "Method of Search"
700 LET N=0
710 FOR I=11 TO 21 STEP 2
720 LET N=N+1
730 CSR 28,I: PRINT F$(N)
740 NEXT I: CSR 42,11: PRINT "=": CSR 42,13: PRINT ")": CSR 42,15: PRINT "<": CSR 42,17: PRINT ")=": CSR 42,19: PRINT "(=": CSR 42,21: PRINT "<)": GOSUB 1200
750 GOSUB 1300: IF A<128 OR A>133 THEN  GOTO 750
760 GOSUB 1150
770 CLS : CSR 30,3: PRINT O$: CSR 29,4: PRINT "==========": CSR 30,6: PRINT "Record No:": CSR 30,8: PRINT "Matched:": CSR 39,8: PRINT SN
780 IF A<>128 THEN  GOTO 850
790 FOR I=1 TO Z: CSR 40,6: PRINT I
800 FOR N=1 TO W
810 IF HEAD=N THEN  IF SEARCH$=LEFT$(RECORD$(I,N),SER) THEN  GOSUB 1400
820 NEXT N
830 NEXT I
840 GOTO 910
850 FOR I=1 TO Z: CSR 41,6: PRINT I
860 FOR N=1 TO W: LET V=0: LET S=0
870 LET V=VAL(RECORD$(I,N)): LET S=VAL(SEARCH$)
880 IF HEAD=N THEN  GOSUB 1450
890 NEXT N
900 NEXT I
910 CSR 25,12: PRINT "E N D   O F   F I L E": PAUSE 4000
920 USER CLOSE#1
930 LET SL=1
940 GOTO 500
1000 CLS : CSR 30,0: PRINT O$: CSR 29,1: PRINT "=========="
1010 CSR 25,5: PRINT "Please wait....."
1020 RETURN
1050 CLS : CSR 20,10: PRINT "Is file a search list? (Y/N)": GOSUB 1250
1060 IF A$="N" OR A$="n" THEN  GOTO 1090
1070 USER OPEN#1,FILE$(1),"I"
1080 LET T=1: GOTO 1100
1090 USER OPEN#1,O$,"I"
1100 RETURN
1150 IF T=0 THEN  GOTO 1170
```

```
1160 IF T=1 THEN  USER ERAFILE$(1)
1170 USER OPEN£1,FILE$(1),"O"
1180 RETURN
1200 FOR X=0 TO 79: PRINT "-";: NEXT X
1210 RETURN
1250 LET A$=INKEY$: IF A$<>"" THEN  GOTO 1250
1260 LET A$=INKEY$: IF A$="" THEN  GOTO 1260
1270 IF A$<>"N" AND A$<>"n" AND A$<>"Y" AND A$<>"y" THEN  GOTO 1250
1280 RETURN
1300 LET A$=INKEY$: LET A=ASC(A$): RETURN
1350 CLS : IF SL=0 THEN  CSR 25,5: PRINT "No search list made": PAUSE 2000: GOTO 500
1360 RETURN
1400 FOR J=1 TO W
1410 USER  PRINT £1,RECORD$(I,J)
1420 NEXT J
1430 LET SN=SN+1: CSR 39,8: PRINT SN: LET SL=1
1440 RETURN
1450 IF A=129 THEN  IF V<S THEN  GOTO 1510
1460 IF A=130 THEN  IF V<S THEN  GOTO 1510
1470 IF A=131 THEN  IF V>=S THEN  GOTO 1510
1480 IF A=132 THEN  IF V<=S THEN  GOTO 1510
1490 IF A=133 THEN  IF V<>S THEN  GOTO 1510
1500 RETURN
1510 GOSUB 1400: GOTO 900
1550 CLS : CSR 28,0: PRINT "Search Options": CSR 27,1: PRINT "==============="
1560 RETURN
1600 CLS : LET A=ABS(A-129)
1610 GOSUB 1000: CSR 25,7
1620 ON A GOTO 1630,1650,1670,1690,1710,1730
1630 PRINT "View File Module loading"
1640 USER  LOAD "VIEWFILE.BAS"
1650 PRINT "Update Module loading"
1660 USER  LOAD "UPDATE.BAS"
1670 PRINT "Sort Module loading"
1680 USER  LOAD "SORT.BAS"
1690 PRINT "Print Module loading"
1700 USER  LOAD "PRINT.BAS"
1710 IF T=1 THEN  USER ERAFILE$
1720 GOTO 500
1730 PRINT "Resetting variables"
1740 IF T=1 THEN  USER ERAFILE$(1)
1750 GOTO 80

10 REM ################################################################################
20 REM ##################################  SORT.BAS  ##################################
30 REM ################################## VERSION 2  ##################################
40 REM ################################## D. WEMYSS  ##################################
50 REM ##################################  OCT 1986  ##################################
60 REM ################################################################################
70 USER  SAVE "SORT.BAS"
80 CLEAR : VS 5: CLS : CSR 25,10: PRINT "Please wait.....": CSR 25,12: PRINT "Setting up variables"
90 DIM F$(3,14),A$(3),FILE$(3,12),C$(2,17),D$(2,20),HEAD$(10,8),RECORD$(100,10,25),O$(8),B$(12),E$(3,15),TEMP$(10,25)
100 LET W=0: LET T=0: LET Y=0: LET Z=0: LET V=0: LET B$=""
110 CLS : CSR 20,10: INPUT "Which file to sort? > ";B$: IF LEN (B$)>8 THEN  GOTO 110 ELSE  LET O$=B$
120 LET FILE$(1)=O$+".SER": LET FILE$(2)=O$+".HDS": LET FILE$(3)=O$+".SOR"
130 GOSUB 1000
140 CSR 20,7: PRINT "Loading heading number ";W
150 USER OPEN£1,FILE$(2),"I"
```

MEMOPAD

```
160 FOR I=1 TO 10
170 USER EOF£1,210
180 LET W=W+1: CSR 43,7: PRINT W: PAUSE 200
190 USER  INPUT £1,HEAD$(I)
200 NEXT I
210 USER CLOSE£1
220 GOSUB 1050
230 GOSUB 1000
240 CSR 20,7: PRINT "Loading record number ";Y
250 FOR I=1 TO 200: USER EOF£1,290
260 LET Y=Y+1: CSR 42,7: PRINT Y: PAUSE 200: FOR N=1 TO W
270 USER  INPUT £1,RECORD$(I,N)
280 NEXT N: NEXT I
290 LET Z=Y: USER CLOSE£1
300 LET F$(1)="F1.....": LET F$(2)="F2.....": LET F$(3)="F3....."
310 LET D$(1)="Sort Data": LET D$(2)="Return to Main Menu"
320 LET C$(1)="Sort File Routine": LET C$(2)=@$+" by D.W."
330 LET E$(1)="Numeric Sort": LET E$(2)="Alphabetic Sort": LET E$(3)="Return to Menu"
340 CLS : CSR 25,0: PRINT C$(1): CSR 25,2: PRINT C$(2): GOSUB 1150
350 LET N=0
360 FOR I=7 TO 10 STEP 3: LET N=N+1
370 CSR 20,I: PRINT F$(N);D$(N)
380 NEXT I
390 PRINT : GOSUB 1150
400 GOSUB 1250: IF A<128 OR A>129 THEN  GOTO 400
410 IF A=128 THEN  GOTO 450
420 CLS : CSR 25,10: PRINT "Returning to Main Menu": CSR 25,12: PRINT "Please wait....."
430 USER  LOAD "MAINMENU.BAS"
450 GOSUB 1300
460 LET N=0
470 FOR I=7 TO 13 STEP 3
480 LET N=N+1
490 CSR 20,I: PRINT F$(N);E$(N)
500 NEXT I: PRINT : GOSUB 1150
510 GOSUB 1250: IF A<128 OR A>130 THEN  GOTO 510
520 IF A=128 THEN  GOTO 550
530 IF A=129 THEN  LET V=1: GOTO 550
540 GOTO 340
550 GOSUB 1300: PRINT : GOSUB 1150
560 FOR X=1 TO W
570 CSR 20,X+6: PRINT X;".": CSR 26,X+6: PRINT HEAD$(X)
580 NEXT X: PRINT : GOSUB 1150
590 CSR 20,20: INPUT "Which heading to sort by? > ";F: IF F<1 OR F>W THEN  GOTO 590
600 CLS : GOSUB 1300: CSR 20,10: PRINT "Please wait.....": CSR 20,12: PRINT "Sort in progress....."
610 FOR K=1 TO Z-1
620 FOR I=1 TO Z-1: FOR M=1 TO W: LET TEMP$(M)="                    ": NEXT M
630 IF V=1 THEN  GOTO 660
640 LET P=VAL(RECORD$(I,F)): LET Q=VAL(RECORD$(I+1,F))
650 IF P<=Q THEN  GOTO 760 ELSE  GOTO 670
660 IF RECORD$(I,F)<=RECORD$(I+1,F) THEN  GOTO 760
670 FOR M=1 TO W
680 LET TEMP$(M)=RECORD$(I,M)
690 NEXT M
700 FOR M=1 TO W
710 LET RECORD$(I,M)=RECORD$(I+1,M)
720 NEXT M
730 FOR M=1 TO W
740 LET RECORD$(I+1,M)=TEMP$(M)
```

TO BE CONTINUED..

MEMOPAD

NODDY PAGE COMP
* DC.
*R
NODDY PAGE C

THIS IS THE SMALLER BOARD FOR ONE PLAYER AGAINST THE COMPUTER. THE BOARD IS A 7 x 7 SQUARE WITH CONNECTIONS HORIZONTAL AND VERTICAL. (VALUES AT THEIR LIMIT ARE SHOWN AS COLOURED SQUARES.)

PRESS ANY KEY TO START
NODDY PAGE INSTRUCTIONS

*D INSTR. *R

NODDY PAGE INSTR

EXPLODE

THE BOARD IS MADE UP OF 64 SECTORS WHICH ARE LINKED TOGETHER. THEY HAVE A LIMIT OF HOW MUCH THEY CAN HOLD WHICH IS DETERMINED BY HOW MANY SECTORS ARE TOUCHING. WHEN THIS LIMIT IS REACHED THE SECTOR WILL EXPLODE AND CHANGE THE AJOINING SECTORS TO YOUR COLOUR. A PLAYER IS KNOCKED OUT WHEN ALL OF HIS COLOUR IS CAPTURED. THE GAME IS WON WHEN ONLY ONE PLAYER REMAINS.

(1 - 6 PLAYERS)

```
4694        LD A,126
4696        LD (MINIMAX),A
4699        LD (BESTP),A
469C LP1:   LD A,(CCPOS)
469F        LD L,A
46A0        LD H,0
46A2        LD C,L
46A3        LD B,H
46A4        LD DE,IDT
46A7        ADD HL,DE
46A8        LD A,(HL)
46A9        CP 1
46AB        JP NZ,CONT3
46AE        LD L,C
```

```
46AF        LD H,B
46B0        LD DE,BRD
46B3        ADD HL,DE
46B4        INC (HL)
46B5 EXP1:  LD L,C
46B6        LD H,B
46B7        PUSH HL
46B8        CALL EXP
46BB        LD A,(PERSS)
46BE        CP 0
46C0        POP HL
46C1        LD A,L
46C2        JP Z,WIN
46C5        LD A,(EXF6)
```

MEMOPAD

```
46C8        CP 0
46CA        JR NZ,EXP1
46CC PER:   LD A,0
46CE        LD (PCPOS),A
46D1        LD (MAXSC),A
46D4        LD A,2
46D6        LD (COLOUR),A
46D9        LD BC,98
46DC        LD DE,TEMP2
46DF        LD HL,BRD
46E2        LDIR
46E4 GPM:   LD A,(PCPOS)
46E7        LD L,A
46E8        LD H,0
46EA        LD C,L
46EB        LD B,H
46EC        LD DE,IDT
46EF        ADD HL,DE
46F0        LD A,(HL)
46F1        CP 2
46F3        JR NZ,CONT1
46F5        LD L,C
46F6        LD H,B
46F7        LD DE,BRD
46FA        ADD HL,DE
46FB        INC (HL)
46FC EXP2:  CALL EXP
46FF        LD A,(COMPS)
4702        CP 0
4704        JR Z,CONT2
4706        LD A,(EXF6)
4709        CP 0
470B        JR NZ,EXP2
470D        LD A,(PERSS)
4710        LD B,A
4711        LD A,(MAXSC)
4714        SCF
4715        CCF
4716        CP B
4717        JR NC,CONT1
4719        LD A,(PERSS)
471C        LD (MAXSC),A
471F CONT1: LD BC,98
4722        LD DE,BRD
4725        LD HL,TEMP2
4728        LDIR
472A        LD A,(PCPOS)
472D        INC A
472E        LD (PCPOS),A
4731        CP 49
4733        JR NZ,GPM
4735        LD A,(MAXSC)
4738        LD B,A
4739        LD A,(MINIMAX)
473C        SCF
473D        CCF
473E        CP B
473F        JR C,CONT2
4741        LD A,(MAXSC)
4744        LD (MINIMAX),A
4747        LD A,(CCPOS)
474A        LD (BESTP),A
474D CONT2: LD BC,98
4750        LD DE,BRD
4753        LD HL,TEMP1
4756        LDIR
4758        LD A,I
475A        LD (COLOUR),A
475D CONT3: LD A,(CCPOS)
4760        INC A
4761        LD (CCPOS),A
4764        CP 49
4766        JP NZ,LP1
4769        JR L11
476B L10A:  CALL GETP
476E WIN:   LD (BESTP),A
4771 L11:   LD A,(BESTP)
4774        LD B,A
4775        LD A,48
4777        CP B
4778        JR NC,L13
477A        LD A,0
477C        LD (COUNT),A
477F L12:   LD A,(COUNT)
4782        LD L,A
4783        LD H,0
4785        LD DE,WGT
4788        ADD HL,DE
4789        LD A,(HL)
478A        LD L,A
478B        LD H,0
478D        LD C,L
478E        LD B,H
478F        LD DE,IDT
4792        ADD HL,DE
4793        LD A,(HL)
4794        CP 0
4796        JR Z,L10A
4798        LD A,(COUNT)
4798        INC A
479C        LD (COUNT),A
479F        CP 49
47A1        JR NZ,L12
47A3        RET
47A4 L13:   LD BC,98
47A7        LD HL,TEMP1
47AA        LD DE,BRD
47AD        LDIR
47AF        LD A,(BESTP)
47B2        LD L,A
47B3        LD H,0
47B5        PUSH HL
47B6        LD DE,IDT
47B9        ADD HL,DE
```

MEMOPAD

```
47BA          LD A,1
47BC          LD (HL),A
47BD          POP HL
47BE          LD DE,BRD
47C1          ADD HL,DE
47C2          INC (HL)
47C3          JP DIS
47C6 GETP:    LD A,(COUNT)
47C9          LD L,A
47CA          LD H,0
47CC          LD DE,WGT
47CF          ADD HL,DE
47D0          LD A,(HL)
47D1          RET
47D2          RET
```

Symbols:

```
BOARD   4011    IDENT   4051
LIMIT   4091    XPOS    40D1
YPOS    4111    EXDIR   4151
SCORES  4198    EXF6    41A6
COLOUR  41A8    COUNT   41AA
INCR    41AC    START   41D6
LOOP    41DE    EXPLODE 41EB
NEXT    4243    LB      4206
LB1     4216    LB2     421F
SCORE   424F    LOOP1   4261
COLOURS 4191    LB3     4228
LB4     4231    LB5     423A
XP      42C8    YP      42C9
CHAR    42CA    INK     42C6
SET     42BE    GETBD   41C3
AD      41D0    GETID   41C8
GETLT   41CD    GTZ     42AD
GTL     42B7    LTL     42B8
BRD     42D7    IDT     4308
LMT     4339    WGT     436A
EXPD    439B    XPO     43CC
YPO     43F0    TEMP1   442E
TEMP2   4490    BESTP   44F2
MINIMAX 44F3    MAXSC   44F4
CCPOS   44F5    PCPOS   44F6
COMPS   44F7    PERSS   44F8
INC     4509    GBRD    4520
GIDT    4526    GLMT    452C
EXP     4532    L0      453A
EX      4548    NXT     4591
L1      4566    L2      4576
L3      457F    L4      4588
L5      45A8    Z       45BF
DISPLAY 45D6    L6      45D8
X1      4664    X3      466C
X4      4670    Y1      4665
X2      4668    Y2      4669
Y3      466D    Y4      4671
BLACK   4614    PERS    4610
```

```
SCOL    4616    PAPER   4662
C1      4666    C2      466A
C3      466E    C4      4672
V0      44F9    V2      4501
V1      44FD    VL      4505
L7      4626    SETC    4645
L8      4630    L9      4636
L10     4641    CMOVE   4680
LP1     469C    EXP1    46B5
WIN     476E    PER     46CC
EXP2    46FC    CONT1   471F
CONT2   474D    L11     4771
GPM     46E4    L13     47A4
L12     477F    ENTER   45CB
DIS     45D0    GETP    47C6
L10A    476B    CONT3   475D
```

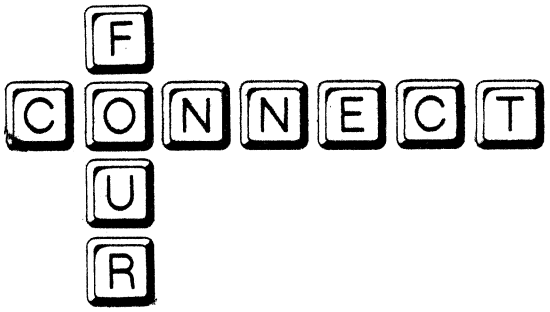
```
1000 PAPER 0: INK 12: PLOT "INSTRUCTIONS"
1010 LET COL=16808: LET EXF6=16806
1020 LET BOARD=16401
1030 LET IDENT=BOARD+64: LET LIMIT=IDENT+64
1040 LET XPOS=LIMIT+64: LET YPOS=XPOS+64: LET EXDIR=YPOS+64
1050 LET COLOURS=EXDIR+64
1060 LET SCORES=COLOURS+7
1070 FOR I=0 TO 63: POKE BOARD+I,0: NEXT
1080 FOR I=0 TO 63: POKE IDENT+I,0: NEXT
1090 FOR I=0 TO 5: POKE SCORES+I,0: NEXT
1100 LET BRD=17111: LET IDT=BRD+49: LET XPO=17356: LET YPO=XPO+49: LET V0=YPO+252: LET PERSS=17656
1110 LET TURNS=0
1600 GENPAT 0,46,0,0,0,24,24,0,0,0
1700 GENPAT 1,130,126,129,129,129,129,129,129,126
1710 GENPAT 1,131,255,129,129,153,153,129,129,255
1720 GENPAT 1,132,0,129,189,165,165,189,129,0
1730 GENPAT 1,133,255,255,195,195,195,195,255,255
1740 GENPAT 1,134,255,255,195,219,219,195,255,255
1750 GENPAT 1,135,255,255,255,255,255,255,255,255
1760 GENPAT 1,136,195,255,126,126,126,126,255,195
1770 GENPAT 1,137,0,16,16,16,16,16,16,0
1780 GENPAT 1,138,0,0,0,0,2,12,48,64
1790 GENPAT 1,139,0,0,0,0,64,48,12,2
1800 GENPAT 0,97,0,127,127,127,127,127,127,127
1810 GENPAT 0,98,0,254,254,254,254,254,254,254
1820 GENPAT 0,99,127,127,127,127,127,127,127,0
1830 GENPAT 0,100,254,254,254,254,254,254,254,0
1840 GENPAT 0,101,0,0,1,3,3,1,1,1
1850 GENPAT 0,102,0,0,128,128,128,128,128,128
1860 GENPAT 0,103,1,1,1,1,3,3,0,0
1870 GENPAT 0,104,128,128,128,128,192,192,0,0
1880 GENPAT 0,105,0,0,3,6,4,0,0,0
1890 GENPAT 0,106,0,0,192,96,32,32,96,192
1900 GENPAT 0,107,1,3,2,6,7,7,0,0
1910 GENPAT 0,108,128,0,0,0,224,224,0,0
1980 LET NP=ASC(INKEY$)-48: IF NP=1 THEN  GOTO 6000
1990 IF NP<2 OR NP>6 THEN  GOTO 1980
```

MEMOPAD

```
2000 MS 4: COLOUR 4,1: PAPER 1: INK 14: CLS
2700 FOR I=0 TO 7: INK 14
2710 CSR PEEK(XPOS+X)-1,PEEK(YPOS+X)-1: PRINT CHR$(65+X);
2720 CSR PEEK(XPOS+X+56)+1,PEEK(YPOS+X+56)+1: PRINT CHR$(65+X);
2730 CSR PEEK(XPOS+X*8)-1,PEEK(YPOS+X*8)+1: PRINT CHR$(48+X);
2740 CSR PEEK(XPOS+7+X*8)+1,PEEK(YPOS+7+X*8)-1: PRINT CHR$(48+X);
2750 INK PEEK(COLOURS)
2760 FOR Y=0 TO 7
2770 LET XP=XPOS+X+Y*8
2780 LET YP=YPOS+I+Y*8
2790 CSR PEEK(XP)-1,PEEK(YP)
2800 PRINT CHR$(138);CHR$(130);CHR$(139);
2810 NEXT Y
2820 NEXT I
2830 FOR I=0 TO 6
2840 FOR Y=1 TO 7
2850 LET XP=XPOS+X+Y*8
2860 LET YP=YPOS+I+Y*8
2870 CSR PEEK(XP),PEEK(YP)-1
2880 PRINT CHR$(137);
2890 NEXT Y
2900 NEXT I
2910 LET TURNS=0
2920 FOR X=56 TO 63
2930 CSR PEEK(XPOS+X)+1,PEEK(YPOS+X)
2940 PRINT " ";
2950 NEXT X
2960 FOR Y=0 TO 56 STEP 8
2970 CSR PEEK(XPOS+Y)-1,PEEK(YPOS+Y)
2980 PRINT " ";
2990 NEXT Y
3000 FOR MOVE=1 TO NP
3020 IF PEEK(SCORES+MOVE*2)+PEEK(SCORES+MOVE*2+1)=0 AND TURNS>NP THEN  GOTO 4990
3030 LET TURNS=TURNS+1
3040 INK 1: PAPER PEEK(COLOURS+MOVE)
3100 CSR 14,1
3110 PRINT " - - ";: INK 1
3120 LET KEYL=ASC(INKEY$)
3130 IF KEYL<65 OR KEYL>72 THEN  GOTO 3100
3140 CSR 15,1
3150 PRINT CHR$(KEYL);" ";
3160 LET KEYN=ASC(INKEY$)
3170 IF KEYN=127 OR KEYN=8 THEN  GOTO 3100
3180 IF KEYN<48 OR KEYN>55 THEN  GOTO 3160
3190 PRINT CHR$(KEYN);
3200 LET KEY=ASC(INKEY$)
3210 IF KEY=127 OR KEY=8 THEN  GOTO 3100
3220 IF KEY<>13 AND KEY<>10 AND KEY<>32 THEN  GOTO 3200
3230 LET POSITION=(KEYN-48)*8+(KEYL-65)
3240 IF PEEK(IDENT+POSITION)<>0 AND PEEK(IDENT+POSITION)<>MOVE THEN  GOTO 3100
3320 POKE BOARD+POSITION,PEEK(BOARD+POSITION)+1
3330 POKE IDENT+POSITION,MOVE
3340 POKE COL,MOVE
3350 POKE SCORES+MOVE,PEEK(SCORES+MOVE)+1
3360 CSR PEEK(XPOS+POSITION),PEEK(YPOS+POSITION): PRINT CHR$(ASC("X"));
3500 LET BC=USR(16854)
3510 IF PEEK(SCORES+MOVE*2)+PEEK(SCORES+MOVE*2+1)*255=TURNS AND TURNS>NP THEN  GOTO 5000
```

MEMOPAD

```
3520 IF PEEK(EXF6)>0 THEN  GOTO 3500
4990 NEXT MOVE: GOTO 3000
5000 CSR 2,0: COLOUR 4,PEEK(COLOURS+MOVE): PAPER PEEK(COLOURS+MOVE): PRINT "PLAYER NO.";MOVE;" WINS"
5010 PRINT "  PRESS ANY KEY TO START AGAIN"
5020 IF INKEY$="" THEN  GOTO 5020
5030 RUN
5040 SAVE "EXPLODE"
5050 GOTO 5030
6000 PLOD "COMP"
6005 IF INKEY$="" THEN  GOTO 6005
6010 VS 4: COLOUR 4,1: PAPER 1: INK 1: CLS : PAPER 14
6020 FOR Y=0 TO 6
6030 FOR X=0 TO 6
6040 LET P=X+Y*7
6050 CSR PEEK(XPO+P),PEEK(YPO+P)
6060 PRINT CHR$(PEEK(V0));CHR$(PEEK(V0+1));
6070 CSR PEEK(XPO+P),PEEK(YPO+P)+1
6080 PRINT CHR$(PEEK(V0+2));CHR$(PEEK(V0+3));
6090 NEXT X
6100 NEXT Y
6110 FOR X=0 TO 6
6120 CSR 9+X*2,3
6130 PAPER 1: INK 14: PRINT CHR$(65+X);
6140 CSR 9+X*2,20
6150 PRINT CHR$(65+X);
6160 CSR 7,5+X*2
6170 PRINT CHR$(48+X);
6180 CSR 24,5+X*2
6190 PRINT CHR$(48+X);
6200 NEXT X
6300 FOR P=0 TO 48: POKE BRD+P,0: POKE IDT+P,0: NEXT P
7000 REM PMOVE
7010 LET TURNS=TURNS+1
7020 INK 1: PAPER 7
7100 CSR 14,1
7110 PRINT " - - ";
7120 LET KEYL=ASC(INKEY$)
7130 IF KEYL<65 OR KEYL>71 THEN  GOTO 7120
7140 CSR 15,1
7150 PRINT CHR$(KEYL);" ";
7160 LET KEYN=ASC(INKEY$)
7170 IF KEYN=127 OR KEYN=8 THEN  GOTO 7100
7180 IF KEYN<48 OR KEYN>54 THEN  GOTO 7160
7190 PRINT CHR$(KEYN);
7200 LET KEY=ASC(INKEY$)
7210 IF KEY=127 OR KEY=8 THEN  GOTO 7100
7220 IF KEY<>13 AND KEY<>10 AND KEY<>32 THEN  GOTO 7200
7230 LET POSITION=(KEYN-48)*7+(KEYL-65)
7240 IF PEEK(IDT+POSITION)<>0 AND PEEK(IDT+POSITION)<>2 THEN  GOTO 7100
7250 POKE BRD+POSITION,PEEK(BRD+POSITION)+1
7260 POKE IDT+POSITION,2
7270 POKE COL,2
7280 POKE PERSS,PEEK(PERSS)+1
7290 CSR PEEK(XPO+POSITION),PEEK(YPO+POSITION): PRINT CHR$(ASC("X"));CHR$(ASC("X"));
7295 CSR PEEK(XPO+POSITION),PEEK(YPO+POSITION)+1: PRINT CHR$(ASC("X"));CHR$(ASC("X"));
7300 LET BC=USR(17667): REM PMOVE
7310 IF PEEK(PERSS)=TURNS AND TURNS>2 THEN  GOTO 8000
7320 IF PEEK(EXF6)>0 THEN  GOTO 7300
7325 LET TURNS=TURNS+1: CSR 14,1: PAPER 6: PRINT "MY GO";
7330 LET BC=USR(18048): REM CMOVE
7335 LET BC=USR(17872): REM DIS/EXP
7340 IF PEEK(PERSS+1)=TURNS AND TURNS>2 THEN  GOTO 8000
7350 IF PEEK(EXF6)>0 THEN  GOTO 7335
7360 GOTO 7000
8000 CSR 2,0: PRINT "THE WINNER IS ";: IF PEEK(COL)=1 THEN  PRINT "ME" ELSE  PRINT "YOU"
8010 PRINT "  PRESS ANY KEY TO START AGAIN"
8020 IF INKEY$="" THEN  GOTO 8020
8030 RUN
```




```
                        ;########################################
                        ;Computer Routine ....
                        ;########################################
47AA                    L6010:

47AA  DD 36 00 05               LD      (IX+00H),5          ;Make sure
47AE  DD 36 01 16               LD      (IX+01H),22         ;Message line is cleared
47B2  21 4152                   LD      HL,SPC              ;before printing
47B5  CD 48E2                   CALL    PRINT               ;next message

47B8  DD 36 00 05               LD      (IX+00H),5          ;Then reposition
47BC  DD 36 01 16               LD      (IX+01H),22         ;the cursor
47C0  21 40F5                   LD      HL,THINK            ;Thinking ...
47C3  CD 48E2                   CALL    PRINT

47C6  0E 01                     LD      C,01                ;Loop counter FOR P=1 TO 8
47C8  21 403E                   LD      HL,DIMR

47CB                    FORP:
47CB  E5                        PUSH    HL                  ;Save HL for next round of loop
47CC  7E                        LD      A,(HL)              ;L6025
47CD  3C                        INC     A
47CE  32 4078                   LD      (RV),A              ;R=R(P)+1
47D1  FE 09                     CP      9
47D3  D2 4A46                   JP      NC,L6181            ;IF R>8

47D6  21 0001                   LD      HL,01               ;EV = 16 bits so
47D9  22 407F                   LD      (EV),HL             ;use double register to clear MSB
47DC  3A 4069                   LD      A,(C$)
47DF  32 4068                   LD      (X$),A              ;C$=X$
47E2  AF                        XOR     A                   ;Clear A reg
47E3  32 4079                   LD      (FV),A              ;F=0
47E6  79                        LD      A,C
47E7  32 4072                   LD      (XV),A              ;X=P
47EA  CD 4A45                   CALL    L60                 ;Go evaluate

47ED  06 04                     LD      B,04                ;FOR L = 1 TO 4
47EF  21 404A                   LD      HL,DIMJ
47F2  AF                        XOR     A
47F3                    L6050:
47F3  77                        LD      (HL),A              ;LET J(L)=0
47F4  23                        INC     HL                  ;Bump to next element
47F5  10 FC                     DJNZ    L6050               ;Next L
```

```
47F7  C5              PUSH  BC          ;Save C counter
47F8  21 403A         LD    HL,DIMA
47FB  11 403E         LD    DE,DIMR
47FE  06 01           LD    B,01        ;Loop counter

4800                L6060:
4800  D5              PUSH  DE
4801  E5              PUSH  HL          ;Save for next time round loop

4802  7E              LD    A,(HL)
4803  32 4071         LD    (AV),A      ;A=A(I)
4806  21 4079         LD    HL,FV
4809  4E              LD    C,(HL)
480A  91              SUB   C           ;A-F
480B  FE 04           CP    4
480D  D2 4A47         JP    NC,L6210    ;>3
4810  1A              LD    A,(DE)
4811  4F              LD    C,A
4812  3A 4071         LD    A,(AV)
4815  81              ADD   A,C
4816  32 4075         LD    (QV),A      ;Q=A+K(I)
4819  FE 04           CP    4
481B  D2 4831         JP    NC,L6090    ;>3
481E  21 407F         LD    HL,EV
4821  34              INC   (HL)
4822  34              INC   (HL)
4823  34              INC   (HL)
4824  34              INC   (HL)        ;E=E+4

4825  21 404A         LD    HL,DIMJ
4828  3A 4071         LD    A,(AV)
482B  5F              LD    E,A
482C  1D              DEC   E           ;Must be dec'd for array access
482D  16 00           LD    D,0
482F  19              ADD   HL,DE
4830  34              INC   (HL)        ;J(A)=J(A)+1

4831                L6090:
4831  E1              POP   HL
4832  D1              POP   DE
4833  13              INC   DE
4834  23              INC   HL
4835  04              INC   B
4836  78              LD    A,B
4837  FE 05           CP    05
4839  20 C5           JR    NZ,L6060    ;Next I
483B  C1              POP   BC          ;All done so clear stack

483C  06 01           LD    B,01        ;Loop counter
483E  21 404A         LD    HL,DIMJ
```

```
4841                    L6100:
4841    E5                      PUSH    HL              ;Save it
4842    7E                      LD      A,(HL)
4843    3D                      DEC     A
4844    32 48A1                 LD      (WV),A          ;W=J(I)-1
4847    3C                      INC     A
4848    28 4E                   JR      Z,L6130         ;If W=1 then L6130 ...
                                                        ;If A reg = -1 and is the inc'd
                                                        ;it will then = 0 ... O.K ?

484A    3A 4079                 LD      A,(FV)
484D    B7                      OR      A               ;Test for zero
484E    28 02                   JR      Z,EPR2
4850    3E 08                   LD      A,8

4852                    EPR2:
4852    F5                      PUSH    AF              ;Save it for a mo
4853    3A 48A1                 LD      A,(WV)
4856    B7                      OR      A               ;Test it for zero
4857    28 02                   JR      Z,EPR3
4859    3E 04                   LD      A,4
485B                    EPR3:
485B    80                      ADD     A,B             ;SGN(W)+I
485C    D1                      POP     DE              ;Get AF bac into DE
485D    82                      ADD     A,D
485E    32 48A0                 LD      (ZV),A
4861    21 401A                 LD      HL,DIM6         ;E=E+6(Z)
4864    5F                      LD      E,A
4865    1D                      DEC     E               ;Sub 1 for access to array
4866    16 00                   LD      D,0
4868    19                      ADD     HL,DE           ;DIM6 is a WORD array so first
4869    5E                      LD      E,(HL)
486A    23                      INC     HL              ;aligned for address then
486B    56                      LD      D,(HL)          ;get value out into DE remember
                                                        ;LSB First!
486C    2A 407F                 LD      HL,(EV)
486F    19                      ADD     HL,DE           ;EV+DIM6(ZV)
4870    3A 48A1                 LD      A,(WV)
4873    B7                      OR      A               ;Test for zero
4874    28 02                   JR      Z,XL2
4876    3E 08                   LD      A,8
4878                    XL2:
4878    80                      ADD     A,B
4879    21 401A                 LD      HL,DIM6
487C    5F                      LD      E,A
487D    1D                      DEC     E               ;Dec for array access
487E    16 00                   LD      D,0
4880    19                      ADD     HL,DE
4881    5E                      LD      E,(HL)
4882    23                      INC     HL
4883    56                      LD      D,(HL)
4884    3A 48A1                 LD      A,(WV)
4887    3D                      DEC     A
4888    28 06                   JR      Z,XL4
488A    6B                      LD      L,E
488B    62                      LD      H,D
```

MEMOPAD

```
488C                    XL3:
488C    19                      ADD     HL,DE
488D    3D                      DEC     A
488E    20 FC                   JR      NZ,XL3          ;WV%G(B%F+I)
4890                    XL4:
4890    EB                      EX      DE,HL
4891    2A 407F                 LD      HL,(EV)
4894    19                      ADD     HL,DE           ;WV%G(B%F+I)+
4895    22 407F                 LD      (EV),HL
4898                    L6130:
4898    E1                      POP     HL
4899    23                      INC     HL
489A    04                      INC     B
489B    7B                      LD      A,B
489C    FE 05                   CP      05

489E    20 A1                   JR      NZ,L6100

                        ;%%%%%%%%%%%%%%%
                        ;Continue here next month
```

MEMOPAD

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

MEMOPAD

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 hardware.

** 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

MANUALS

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

ALL CHEQUES SHOULD BE MADE PAYABLE TO ORION SOFTWARE

****

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			
DMX 80 PRINTER RIBBONS		- £8.96			

MEMOPAD

SOFTWARE

U = Utility E = Educational L = Language
G = Game J = Joystick Compatible 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
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
CRIBBAGE	G	512	5.95
CRIBBAGE (SDX)	G	512	9.95
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
PRO WORD	B	512	12.95
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
SELF TEST	U	ANY	7.95
SELF TEST (SDX)	U	ANY	9.95
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

Abridged Terms & Conditions (Downloads)

Disclaimer

www.primrosebank.net, (***the website***) is provided by Dave Stevenson as a service to the public, is provided "as is" and carries no warranties, expressed or implied, of any kind.

Dave Stevenson is not responsible for, and expressly disclaims all liability for, damages of any kind arising out of use, reference to, or reliance on any information contained within the website or made available for download. Whilst the information contained within the website site is periodically updated, no guarantee is given that the information provided on the website is correct, complete, and up-to-date.

A number of articles on the website contain technical data and practical guidance which may be of use in testing and maintaining various items of vintage computer and electronics hardware. Such articles are not intended to cover all aspects of the tasks involved and may omit essential information, including necessary safety precautions. Performance of the tasks described may risk damage to equipment and/or people. The reader is responsible for ensuring that he/she is capable of performing the tasks described and well as assessing the inherent risks involved and taking appropriate measures to mitigate such risks.

Dave Stevenson expressly disclaims all liability for, damages to equipment or injury of any kind arising out of use of such technical data and guidance.

As far as possible, information included on the website from other sources has been credited to the respective author and/or publisher. The majority of content on the website is derived from material first published in the 1980s. ***This material is likely still under copyright of the original author and/or publishers.*** The authors and/or publishers may not have given express permission to copy, transmit or make this information available for download, but I believe that they would have no objection to this archive information being placed into the public domain.

However, should the author and/or publisher of the original material find any content on the website for which they wish to assert their rights, they should notify Dave Stevenson (by e-mail to: webmaster@primrosebank.net) who would be pleased to enter into a dialogue to agree a satisfactory resolution of their concerns.

If you obtained this file as part of a paid-for package, you have been scammed! I suggest that you request a refund from the seller, please also advise Dave stevenson at the e-mail address above.

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

****

W*A*N*T*E*D

5.25" DISC DRIVE - ANY REASONABLE PRICES CONSIDERED.
Telephone : Nigel Carr on 0532 580 488
(or leave message on answerphone)

****

ATTENTION ALL NEWWORD USERS .....
WE HAVE A SMALL QUANTITY OF BOOKLETS THAT DESCRIBE ALL YOU NEED TO KNOW ABOUT CUSTOMISING NEWWORD FOR YOUR OWN NEEDS.
THIS BOOKLET LISTS ALL *THE USER AREA PATCHES* AND WHAT IS EXPECTED FROM THE OPERATOR TO INCLUDE CUSTOMISED COMMANDS
WHILE STOCKS ARE AVAILABLE 3.00p
OTHER BOOKS ON NEWWORD WILL BE AVAILABLE SHORTLY .....
