MINDBENDING GAMES for the AMSTRAD CPC 464

Philip Laird

MINDBENDING GAMES

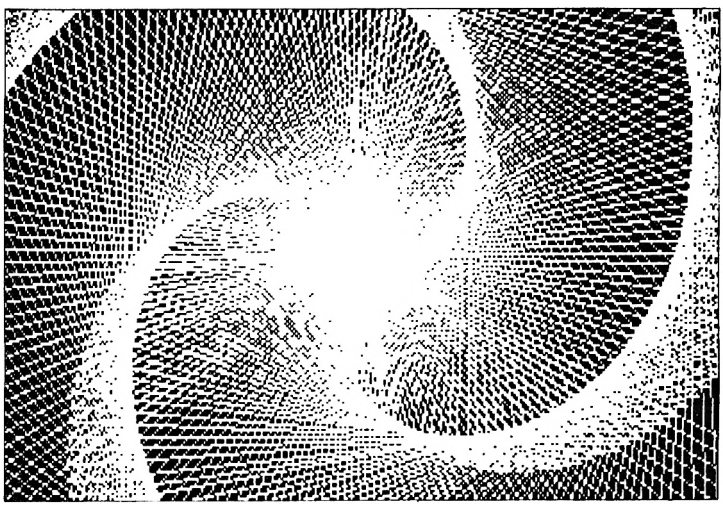
for the

AMSTRAD CPC464

Philip Laird

EXCALIBUR

MINDBENDING GAMES



First published in 1984
by EXCALIBUR PUBLICATIONS
The Glass House, 9-13 Wensum Street
Norwich NR3 1LA
and distributed by Andrew Beshara

ISBN 0 948102 00 4

Illustrations by Gillian Frances

Typeset and produced by
Margaret Helps & Associates, Norwich

Printed in Great Britain by
Morris Printing Co Ltd, Norwich

CONTENTS

INTRODUCTION

The original listings in this book should provide you with many hours of fun and amusement. Great care has been taken with presentation to ensure that the quality of the programs, their screen displays, colour, sound and error-trapping match in full measure the standards you have come to expect from more expensive cassette-based software.

The programs have been fully tried and tested to exacting standards to ensure that they provide excellent and broadly based entertainment free from programming errors. In short, I believe that if you take the time and trouble to type in each listing you will be pleasantly surprised at the results.

Each game or puzzle is fully described and illustrated, and notes are provided to guide you as to what each program does and how it works.

The main listings in this book employ many multi-statement lines and single-letter variables. This is done to save memory and for the sake of programming efficiency. The final chapter is devoted to those who seek to develop their programming skills. In that endeavour I have offered a number of suggestions designed to help plan a program and see it develop in a structured and intelligible manner. In that chapter an additional and less complex program listing has been included, with full-name variables and few multi-statement lines, to facilitate a fuller understanding of the methods and techniques described.

If you have familiarised yourself with the CPC464 microcomputer you will have discovered just how fast and versatile its BASIC is. Some interesting commands have been included, hitherto unavailable on most home micros.

The programs to be found in this volume are in many ways machine-specific. The WINDOW command, for example, is frequently employed for text formatting; interrupt commands are used to produce various effects, along with a number of other unusual and useful features of Locomotive BASIC.

All that remains now is for you to turn over the page, switch on your micro and start typing. If, however, you are new to home computing you should first read the notes that follow. The tips that you will find there should save you hours of unnecessary frustration.

Philip Laird
Norwich

October 1984

FOR NEWCOMERS TO HOME COMPUTING

Your Amstrad CPC464 manual gives full instructions on how to set up the computer and monitor. If you have not already familiarised yourself with your micro it is strongly recommended that you start by reading the Foundation Course in your manual, at least as far as page F2.4. In addition, the following notes should prove helpful.

When typing in programs accuracy is essential. Far from being inconsequential, punctuation and spelling are very important. Each comma, semi-colon, colon, quote mark, etc. is in fact an instruction to your computer to perform a specific task, and inaccurate typing will lead either to a syntax error or it will scramble the screen display. To eliminate errors in typesetting each program listing has been reproduced directly from a working program via a computer printer, and what you type should match *exactly* the listings that appear on the following pages.

When first switched on your computer will be in MODE 1, that is to say it will display up to 40 characters in each line of text. The programs in this book are presented in the same manner to facilitate checking.

Computers make a distinction between the number Ø (zero) and the letter O, and between 1 (one) and the letters I and l (lower case L). Do not confuse these. To assist you, this is how these characters appear in the listings:

```
Ø ... zero          O,o ... letter [O]
1 ... one           I,i ... letter [I]
                    L,l ... letter [L]
```

One last cautionary note before you start. Even the very best of computers is known to 'crash', that is to say it can

get such a bad headache that only a full reset (switching off and on) will restore its sanity. Occurrences of this kind should be rare, but remember that even momentary failure of the electricity supply would have the same effect. Be sure therefore to SAVE an unfinished program periodically and thereby minimise the risk of much wasted effort.

RED KNIGHT

You would be hard pressed to find many games more challenging than this. I have known several people who have spent days (literally!) trying to solve this puzzle without success, though managing to come very close to a solution.

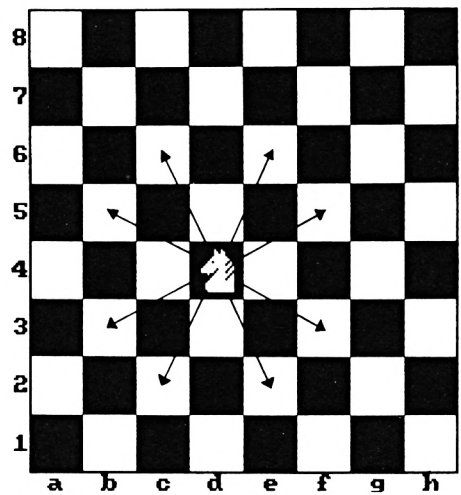
The game is a derivative of chess. The computer will display a chess board, but instead of the traditional sixteen pieces and pawns you have just one piece, a red knight.

You choose the square on which the knight will start and from then on, using the knight's move in a game of chess, you try to visit each square on the board once (and once only!).

Easy? Not a bit of it! It would not be surprising if this puzzle kept you up into the early hours tearing out your hair with frustration. Be assured, though, it can be done, and to prove it, a demonstration option has been included.

As with all the games in this book the program is very user-friendly. Full instructions are given within the program, and the computer will at all times display messages to indicate the sort of information it expects. Moves are effected very simply by keying a letter followed by a number (or *vice versa*) to denote the active square. Squares already visited are marked so that you may see at a glance which squares remain to be covered. All illegal moves will be rejected, and the computer will recognize a situation in which you cannot make a valid move. The moves you make will be recorded on screen so that if you do succeed you could make a note of the order in which you made them.

For those unfamiliar with the game of chess the diagram overleaf illustrates the way in which a knight moves.



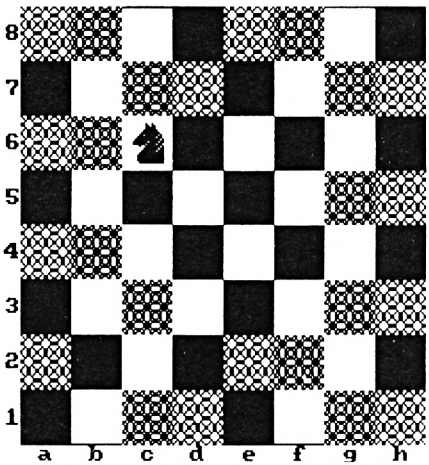
From the diagram you will see that the piece moves two squares, horizontally or vertically, and then another square, vertically or horizontally.

PROGRAM NOTES

Lines 90-200 set up the screen display; 220-370 form the main program loop; 390-540 the end-of-game routine and 560-690 the routine for entering a move and checking its validity.

The routine at 700-830 initialises the main variables. The important one to note here is array S(11,11) which is used to store information relating to the status of each square on the board and to enable the computer to recognize areas off the board. Note also KN$ which stores the user-defined graphics for the knight, read from DATA statements at the end of the listing.

Lines 840-930 form the Introduction and Instructions, and 940-1000 comprise a routine for drawing the title border.




```
  M O V E S

a8  e2
b6  g1
a4  h3
c3  g5
d1  h7
f2  f8
h1  d7
g3  b8
h5  c6
g7
e8
c7
a6
b4
a2
c1
```

RED KNIGHT

NO. MOVES: 25

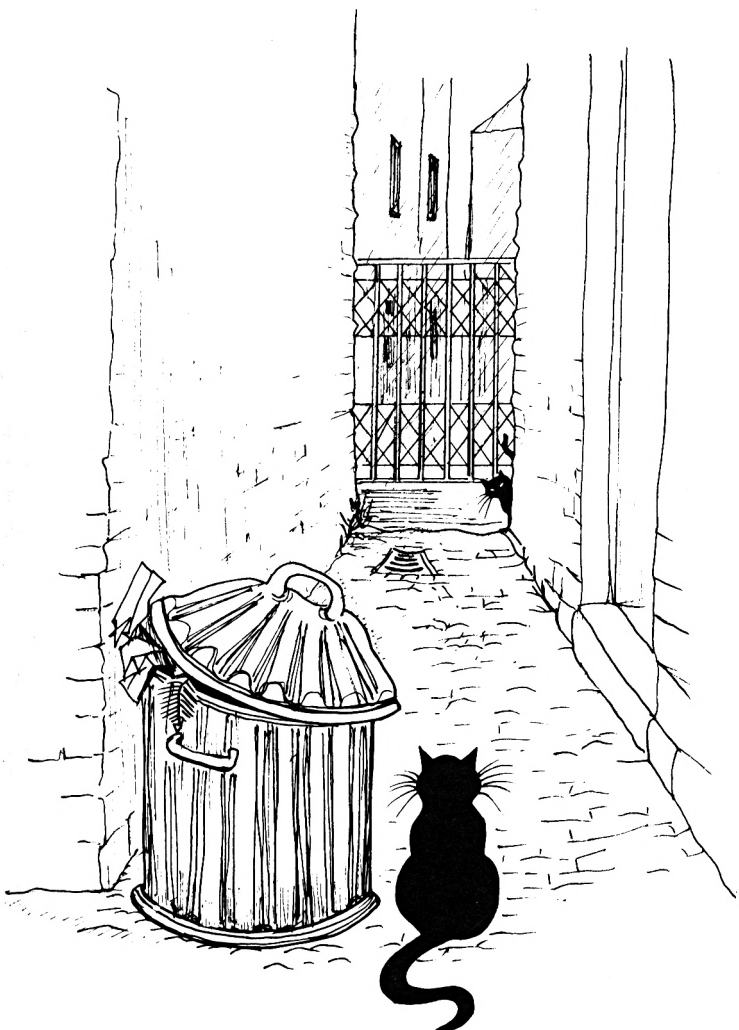
```
10 REM ------------------------------------
20 REM              RED  KNIGHT
30 REM ------------------------------------
40 GOSUB 700
50 GOSUB 840
60 INK 3,0:PEN 3:LOCATE 1,8:PRINT"WOULD
YOU LIKE A DEMONSTRATION?":PRINT TAB(7)"
Press  [Y]  or  [N]":INK 1,15:INK 3,21:P
RINT CHR$(7)
70 D$=INKEY$:D$=UPPER$(D$):IF D$<>"Y" AN
D D$<>"N" THEN 70
80 REM
90 REM ________ SCREEN DISPLAY _________
100 WINDOW 1,40,1,25:BORDER 6,0:INK 0,6:
INK 2,6:INK 3,6:CLS:SQU=0
110 FOR I=1 TO 22 STEP 3
120 Q$=STR$(8-(I\3)):PEN 2:WINDOW 1,1,I+
1,I+1:PRINT MID$(Q$,2,1);
130 FOR J=2 TO 23 STEP 3:SQU=NOT SQU
140 WINDOW J,J+2,I,I+2:IF SQU THEN PRINT
 STRING$(9,CHR$(&8F)); ELSE PRINT SPC(9)
;
150 NEXT:SQU=NOT SQU:NEXT
160 FOR I=1 TO 8:WINDOW I*3,I*3,25,25:PR
```

```
INT CHR$(96+I);:NEXT
170 MOVE 16,399:DRAW 16,16,2:DRAW 399,16
,2:DRAW 399,399,2:DRAW 16,399,2
180 MOVE 432,399:DRAW 432,88,2:DRAW 639,
88,2:DRAW 639,399,2:DRAW 432,399,2
190 WINDOW 29,39,2,22:PEN 3:PRINT " M O
V E S":WINDOW 28,40,21,25:PAPER 1:PEN 0:
PRINT " RED  KNIGHT ";:PAPER 0:PRINT:PEN
 3:PRINT "NO. MOVES:  0"SPC(13);
200 BORDER 0:INK 0,0:INK 1,15:INK 2,25:I
NK 3,21:PAPER 0:PEN 3:PRINT CHR$(7)
210 REM
220 REM _____ PLAY _____
230 IF D$="Y" THEN FOR I=0 TO 300:NEXT
240 IF M THEN F1=F:R1=R:P1=P:Q1=Q
250 IF D$="Y" THEN IF fx>0.5 THEN I$=MID
$(DEMO$,2*(M+1)-1,2) ELSE I$=MID$(DEMO$,
127-M*2,2)
260 IF D$="N" THEN GOSUB 560 ELSE GOSUB
620
270 E$=INKEY$:E$=LOWER$(E$):IF E$="q" OR
 Q$="q" THEN 390
280 S(Q,P)=1
290 PEN 1:IF M>1 THEN WINDOW F1,F1+2,R1,
R1+2:PAPER -2*((P+Q)/2=(P+Q)\2):PRINT"**
*******";
300 WINDOW F,F+2,R,R+2:IF (P+Q)/2=(P+Q)\
2 THEN PAPER 0 ELSE PAPER 2
310 PRINT KN$;:PEN 3:PAPER 0:WINDOW x,x+
1,y,y:PRINT I$;:WINDOW 39,40,23,23:PRINT
 USING "##";M;
320 SOUND 2,INT(125000/(440*(2↑(o+(10-n)
/12)))+0.5),7,5
330 n=n-1:IF n=0 THEN o=o+1:n=12
340 y=y+1:IF (y+12)/16=(y+12)\16 THEN x=
x+3:y=4
350 REM
360 REM  CHECK IF FURTHER MOVE POSSIBLE
```

```
370 IF M<65 AND (S(Q+2,P+1)=0 OR S(Q+2,P
-1)=0 OR S(Q+1,P+2)=0 OR S(Q+1,P-2)=0 OR
 S(Q-1,P+2)=0 OR S(Q-1,P-2)=0 OR S(Q-2,P
+1)=0 OR S(Q-2,P-1)=0) THEN 220
380 REM
390 REM __________ END OF GAME _________
400 WINDOW 28,40,24,25:PEN 0:CLS:PRINT C
HR$(7);:IF E$="q" OR Q$="q" THEN 460
410 PAPER 2:IF D$="Y" THEN PRINT "EVERY
 SQUARE  COVERED !  ";:GOTO 450
420 IF M<48 THEN PRINT" NEVER MIND!   Tr
y again  ";:GOTO 450
430 IF M<64 THEN PRINT" A GOOD TRY!"SPC(
14);:GOTO 450
440 PRINT" WELL DONE !"SPC(14);
450 FOR I=0 TO 4000:NEXT:PRINT CHR$(7)
460 PAPER 1:PRINT"Another game?[key  Y o
r N]";
470 Q$=INKEY$:Q$=UPPER$(Q$):IF Q$<>"N" A
ND Q$<>"Y" THEN 470
480 IF Q$="N" THEN PAPER 0:PEN 2:MODE 1:
END
490 PAPER 2:PRINT"INSTRUCTIONS?[KEY  Y O
R N]"CHR$(7);
500 Q$=INKEY$:Q$=UPPER$(Q$):IF Q$<>"N" A
ND Q$<>"Y" THEN 500
510 PAPER 0:PEN 3:PRINT"ANOTHER  GAMEFOL
LOWS   ..."CHR$(7);:IF Q$="Y" THEN RUN
520 CLEAR:GOSUB 770
530 GOSUB 940
540 GOTO 60
550 REM _____ S U B R O U T I N E S ____
560 REM __________ INPUT OF MOVE _______
570 I$="":WINDOW 28,40,24,25:CLS:PEN 2:I
F M=0 THEN PRINT "START SQUARE?"; ELSE P
RINT "NEXT SQUARE ?";
580 IF INKEY$<>"" THEN 580
590 Q$=INKEY$:IF Q$="" THEN 590
```

```
600 Q$=LOWER$(Q$):I$=I$+Q$:PRINT Q$;:IF
Q$="q" THEN RETURN
610 IF LEN(I$)<>2 THEN 580
620 IF VAL(I$)=0 THEN P=ASC(I$)-95:Q=VAL
(RIGHT$(I$,1))+1
630 IF VAL(I$)<>0 THEN Q=VAL(I$)+1:P=ASC
(RIGHT$(I$,1))-95:I$=RIGHT$(I$,1)+LEFT$(
I$,1)
640 IF P<2 OR P>9 OR Q<2 OR Q>9 THEN 570
650 F=P*3-4:R=28-Q*3:IF M=0 THEN M=1:RET
URN
660 REM
670 REM ___ CHECK IF SQUARE IS FREE ____
680 IF S(Q,P) OR (NOT(ABS(Q1-Q)=2 AND AB
S(P1-P)=1) AND NOT(ABS(Q1-Q)=1 AND ABS(P
1-P)=2)) THEN PEN 1:CLS:PRINT"NOT POSSIB
LE!"CHR$(7):FOR J=0 TO 999:NEXT:GOTO 570
690 M=M+1:RETURN
700 REM __________ INITIALISE __________
710 RANDOMIZE TIME
720 SYMBOL AFTER &F7
730 SYMBOL &F8,0,0,0,18,27,31,63,127:SYM
BOL &F9,0,0,0,0,0,0,128,192
740 SYMBOL &FA,0,1,3,7,7,0,0,0:SYMBOL &F
B,223,255,255,247,239,159,63,127
750 SYMBOL &FC,160,96,208,176,96,208,176
,112:SYMBOL &FD,0,1,1,1,0,0,0,0
760 SYMBOL &FE,255,255,255,255,255,0,0,0
:SYMBOL &FF,240,240,240,240,240,0,0,0
770 DIM S(11,11):M=0:fx=RND:n=3:o=-3:x=2
9:y=4
780 DEMO$="a8b6a4c3d1f2h1g3h5g7e8c7a6b4a
2c1e2g1h3g5h7f8d7b8c6a7b5a3b1d2f1h2g4h6g
8e7c8d6b7a5b3a1c2e1g2h4g6h8f7d8e6f4d3b2c
4e5f3d4f5e3d5f6e4c5":KN$=" "
790 FOR I=248 TO 255:KN$=KN$+CHR$(I):NEX
T
```

```
800 FOR I=0 TO 11:FOR J=0 TO 11:IF I<2 O
R I>9 OR J<2 OR J>9 THEN S(I,J)=-1
810 NEXT:NEXT
820 BORDER 0:INK 0,0:INK 1,0:INK 2,0:INK
 3,0:MODE 1
830 RETURN
840 REM __________ INTRODUCTION _________
850 GOSUB 940
860 PEN 2:PRINT"*      R E D   K N I G H
T     *"
870 PEN 3:PRINT"In  this  puzzle  you  h
ave oneknight on an empty chess board."
880 PRINT"Using  the  knight's  move in
agame of chess  you  must try tovisit  e
ach square on the boardonce, and once on
ly.":PRINT
890 PRINT"The game will end when you can
-not make a legal move,  but youmay quit
 at any time during thegame by pressing
[Q].":PRINT
900 PEN 2:PRINT"*   NOW PRESS A KEY TO P
LAY    *";
910 INK 1,15:INK 2,25:INK 3,21:PRINT CHR
$(7);
920 IF INKEY$="" THEN 920
930 CLS:RETURN
940 REM _______ DRAW TITLE BORDER ______
950 PAPER 0:PEN 1
960 FOR I=2 TO 38 STEP 3
970 WINDOW I,I+2,2,4:PRINT KN$;:WINDOW I
,I+2,23,25:PRINT KN$;
980 IF I<23 THEN WINDOW 2,4,I,I+2:PRINT
KN$;:WINDOW 38,40,I,I+2:PRINT KN$;
990 NEXT:WINDOW 6,36,6,21
1000 RETURN:REM ________________________
```



ALLEYCAT

This delightful game was written with younger children in mind, but the charm and the fascination that it holds will surely appeal to all ages.

The player is invited to choose a number of cats (between two and eight) which the computer will display in an ordered row. To the left of this row of cats is a space (or a dustbin, if you prefer to imagine it that way!).

The cats themselves are a rather motley crew, some black, some ginger, some a dirty brown, but all of them toms, and each with its distinctive number.

The idea of the game is to reverse the order in which the cats sit so that number 1 ends up on the right and the highest number on the left, with the space finishing up on the far left. For example, if you start with eight cats your starting position would be:

space	cat	cat	cat	cat	cat	cat	cat	cat
—	1	2	3	4	5	6	7	8

and the final position:

space	cat	cat	cat	cat	cat	cat	cat	cat
—	8	7	6	5	4	3	2	1

The rules of the game are straightforward, though solving the puzzle is not as easy as you might think.

A cat may jump into the space if it is sitting next to it, or a cat may jump over one other cat to occupy the space. So taking the example printed above you would start by moving either cat number 1 or cat number 2 into the space. No other cat may move at this stage because cats may not

jump over more than one of their number. Suppose therefore that you wanted to move cat number 2. Simply press 2 on the keyboard and the new position would look like this:

cat	cat	space	cat	cat	cat	cat	cat	cat
2	1	—	3	4	5	6	7	8

Now the second move gives you a wider choice: you may move any of cats 2, 1, 3 or 4, again by pressing the chosen number on the keyboard. The computer will reject illegal moves, and throughout the game the number of moves taken will be displayed.

Full instructions are included with the listing, and if you so require, the computer will demonstrate a solution to the puzzle.

PROGRAM NOTES

Lines 60–140 set up the screen display; 150–210 offer the choice between the demonstration and player mode, and line 240 checks the validity of an entered move.

Lines 210–310 move the individual cats to an appropriate sound accompaniment. At line 330 the computer establishes whether the puzzle has been solved and, if not, returns execution of the program to the beginning of the main program loop. The end-of-game sequence is held in lines 340–400.

Lines 420–670 initialise the main variables. Note the array n(8) which stores the position of the cats. The strings k1$ and k2$ together store the image of the large tomcat whilst c$ holds the graphics for the smaller (and more active) cats. Note too the use of DEFINT to define integer variables to promote the speed of program execution.

Lines 690–930 set up the title page, introduction and game instructions. Lines 1140 *et seq* hold data used for defining the various cats.

```
MOVE  WHICH  CAT  ?
MOVES  SO  FAR:   6
```

(Press [Q] to Quit)

```
10 REM --------------------------------
20 REM  ALLEYCAT: A program for children
30 REM --------------------------------
40 RUN 420
50 n=0:q=0:v=0
60 FOR i=1 TO 8:n(i)=i:NEXT
70 WINDOW 19,40,1,4:CLS:PEN 1:IF z$="P"
THEN PRINT"How many cats?";:PEN 2:PRINT,
,"Choose a number in therange  2 to 8."C
HR$(7) ELSE i$="8":GOTO 90
80 i$=INKEY$:IF i$="" THEN 80
90 c=ASC(i$)-48:IF c<2 OR c>8 THEN 80
100 CLS:PEN 1:PRINT:PRINT"   "STRING$(&1
3,CHR$(&9A))
110 FOR i=40 TO (41-4*c) STEP -4
120 PEN 3-((i\4-(10-c)) MOD 3):WINDOW i-
3,i,19,23:PRINT c$;i\4-(10-c);
130 NEXT
```

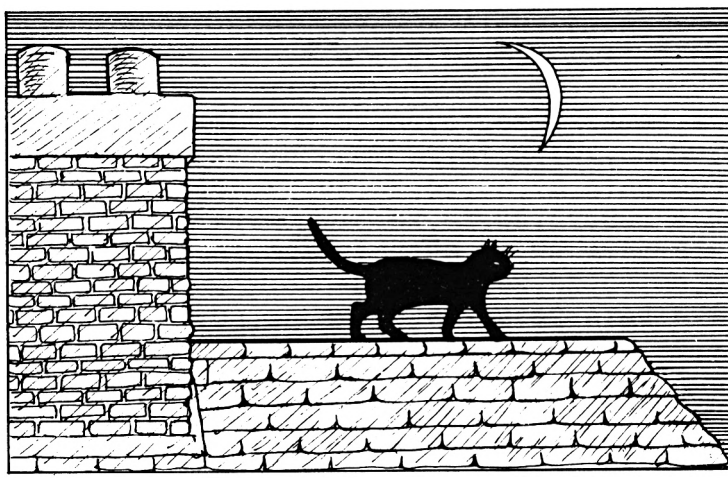
```
140 WINDOW i-3,i,19,23:PEN 1:PRINT s$;:I
F z$="P" THEN 190
150 WINDOW 19,40,1,1:PRINT"   ** DEMONST
RATION **";:WINDOW 22,40,9,13:ZONE 9:PRI
NT"Choose a speed  ...",,"PRESS    [1] F
ast",,"[2] Medium","[3] Slow";
160 i$=INKEY$:IF i$="" THEN 160
170 v=ASC(i$)-49:IF v<0 OR v>2 THEN 160
ELSE CLS
180 IF z$="D" THEN i$=MID$(m$,n+1,1):FOR
 i=0 TO v*200:NEXT:GOTO 220
190 WINDOW 22,40,24,25:PEN 1:PRINT LEFT$
(j$,19)"(Press [Q] to Quit)";:WINDOW 19,
40,1,1:PRINT"   MOVE  WHICH  CAT  ?";
200 i$=INKEY$:i$=UPPER$(i$):IF i$="" THE
N 200
210 IF i$="Q" THEN 350
220 k=ASC(i$)-48:IF k<1 OR k>c THEN 200
230 FOR i=0 TO v*1000:NEXT:IF z$="P" THE
N CLS
240 IF ABS(n(k)-q)>2 THEN PEN 2:PRINT"
 ** NOT  POSSIBLE **";:FOR i=0 TO 2000:N
EXT:CLS:PEN 1:GOTO 180
250 WINDOW 10,14,6,8:PEN 1:PRINT STRING$
(&F,CHR$(&8F));
260 WINDOW 37+4*q-4*c,40+4*q-4*c,19,23:P
EN 3-k MOD 3:PRINT c$;k;
270 WINDOW 37+4*n(k)-4*c,40+4*n(k)-4*c,1
9,23:PEN 1:PRINT s$;
280 SOUND 2,k*50,20,7
290 WINDOW 10,14,6,8:PRINT b$;
300 n=n+1:r=n(k):n(k)=q:q=r:p=0
310 WINDOW 22,40,3,3:PEN 2:PRINT"MOVES
SO  FAR: ";:PRINT USING "###";n;
320 FOR j=1 TO c:IF (n(j)+j)<>(c+1) THEN
 p=1
330 NEXT:IF p OR q>0 THEN 180
340 IF z$="P" THEN CLS:WINDOW 19,40,1,1:
```

```
PEN 1:PRINT"   YOU DID IT IN ..";:PRINT
USING "###";n;
350 WINDOW 10,40,24,25:PRINT RIGHT$(j$,3
1)"ANOTHER GAME?  (Key [Y] or [N])"CHR$(
7);:IF i$="Q" THEN WINDOW 19,40,1,2:CLS
360 i$=INKEY$:i$=UPPER$(i$):IF i$<>"Y" A
ND i$<>"N" THEN 360
370 CLS:IF i$="N" THEN 390 ELSE WINDOW 1
,40,24,25:PEN 2:PRINT j$x$CHR$(7);
380 z$=INKEY$:z$=UPPER$(z$):IF z$<>"D" A
ND z$<>"P" THEN 380
390 CLS:WINDOW 1,40,19,25:CLS:WINDOW 22,
40,1,3:CLS:IF i$="Y" THEN 50
400 WINDOW 26,40,13,14:PRINT"G O O D B Y
 E !":FOR i=0 TO 8000:NEXT:PEN 1:MODE 1:
END
410 REM
420 REM __________ INITIALISE __________
430 SYMBOL AFTER &EE
440 SYMBOL &EF,&0,&1C,&F,&F,&7,&3,&3,&3
450 SYMBOL &F0,&0,&0,&0,&BF,&FF,&FF,&FF,
&FF
460 SYMBOL &F1,&0,&1C,&38,&78,&F8,&F0,&F
0,&F0
470 SYMBOL &F2,&7,&7,&F,&71,&7,&7,&3,&0
480 SYMBOL &F3,&8C,&FF,&F3,&FF,&F3,&CC,&
FF,&FF
490 SYMBOL &F4,&78,&F8,&F9,&E7,&F9,&F9,&
F1,&C1
500 SYMBOL &F5,&60,&C0,&80,&80,&80,&80,&
80,&C0
510 SYMBOL &F6,&3,&F,&1F,&3F,&3F,&3F,&3F
,&3F
520 SYMBOL &F7,&F1,&FC,&FE,&FF,&FF,&FF,&
FF,&FF
530 SYMBOL &F8,&E0,&F0,&70,&70,&70,&F0,&
F0,&E0
540 SYMBOL &F9,&3F,&1E,&C,&7,&0,&0,&0,&0
```

```
550 SYMBOL &FA,&FF,&7F,&3E,&FF,&0,&0,&0,
&0
560 SYMBOL &FB,&FF,&3F,&1F,&F0,&0,&0,&0,
&0
570 SYMBOL &FC,&E0,&C0,&0,&0,&0,&0,&0,&0
580 SYMBOL &FD,&FF,&FF,&FF,&0,&FF,&FF,&F
F,&FF
590 SYMBOL &FE,&FF,&FF,&FF,&FF,&AA,&55,&
AA,&55
600 SYMBOL &FF,&AA,&55,&AA,&55,&FF,&FF,&
FF,&FF
610 DIM n(8):DEFINT c-r
620 k1$=SPACE$(12):k2$="":s$=k1$+CHR$(&8
C)+CHR$(&8C)+CHR$(&8C)+CHR$(&84)+"     "
630 FOR i=0 TO 15:READ k$:k$="&"+k$:c$=c
$+CHR$(VAL(k$)):NEXT
640 FOR i=13 TO 136:READ k$:k$="&"+k$:i$
=k1$:j$=CHR$(VAL(k$)):k1$=i$+j$:NEXT
650 FOR i=137 TO 267:READ k$:k$="&"+k$:i
$=k2$:j$=CHR$(VAL(k$)):k2$=i$+j$:NEXT
660 FOR i=1 TO 15:READ k$:k$="&"+k$:b$=b
$+CHR$(VAL(k$)):NEXT
670 j$=STRING$(&28,CHR$(&9A)):m$="213414
561678164235424762682643578433463648578 6
57":x$="PRESS [D] DEMONSTRATION  OR  [P]
 TO PLAY"
680 REM
690 REM __________ INTRODUCTION __________
700 MODE 0:BORDER 0:INK 0,0:INK 1,0:INK
2,0:INK 3,0
710 WINDOW 2,18,1,25:PAPER 0:PEN 1:PRINT
:PRINT"ALLEYCAT":PRINT:PRINT:GOSUB 880
720 WINDOW 2,20,25,25:PRINT"a children's
 puzzle";:INK 3,15
730 FOR i=0 TO 99
740 j=INT(RND*27):k=INT(RND*27):m=INT(RN
D*27):IF j=k OR j=m OR k=m THEN 740
750 BORDER INT(RND*27):INK 0,j:INK 1,k:I
```

```
NK 3,m:NEXT
760 BORDER 16:INK Ø,16:INK 1,Ø:INK 3,3
770 FOR i=Ø TO 5ØØØ:NEXT:INK Ø,26:INK 1,
26:INK 2,26:INK 3,26:BORDER 15,Ø
780 MODE 1:WINDOW 1,17,1,25:GOSUB 88Ø
790 WINDOW 19,4Ø,1,25:PEN 3:PRINT "ALLEY
CAT":PRINT
8ØØ PRINT "This  puzzle  presentsyou wit
h a row of catsnumbered 1 to 8, whichyou
  must  try  to  rearrange in the revers
eorder.  Cat  number 1,therefore, will e
nd upon the right,  and catnumber 8  on
the left.":PRINT
81Ø PRINT "To  the  left  of  thecats is
 a space. A catnext to this space mayjum
p into it, or a catmay jump over  1 othe
rcat to fill the space.This space ends o
n theleft, where it starts.":PEN 2
82Ø FOR i=1 TO 13 STEP 4:WINDOW i+1,i+4,
18,23:PEN 3-i MOD 3:PRINT c$;:NEXT
83Ø WINDOW 1,4Ø,24,25:PEN 1:PRINT j$x$;
84Ø INK 1,Ø:INK 2,15:INK 3,3:BORDER 26:P
RINT CHR$(7);
85Ø z$=INKEY$:z$=UPPER$(z$):IF z$<>"D" A
ND z$<>"P" THEN 85Ø
86Ø WINDOW 19,4Ø,1,23:CLS:WINDOW 2,17,18
,23:CLS:WINDOW 1,4Ø,24,25:CLS:GOTO 5Ø
87Ø REM
88Ø REM ___ PRINT BIG TOM  [k1$/k2$] ___
89Ø PRINT k1$;
9ØØ FOR i=1 TO 131
91Ø k$=MID$(k2$,i,1):IF ASC(k$)>&8F AND
ASC(k$)<>&FD THEN PEN 3 ELSE PEN 1
92Ø PRINT k$;:NEXT
93Ø RETURN
94Ø REM _ DATA FOR LITTLE CATS (UDGs) _
95Ø DATA EF,FØ,F1,2Ø,F2,F3,F4,F5,F6,8F,F
7,F8,F9,FA,FB,FC
```

```
960 REM
970 REM ___ DATA FOR BIG TOM (UDGs) ___
980 DATA 87,20,20,20,20:REM
                              k1$
990 DATA 84,20,20,20,20,20,20,20,20,20,2
0,88,85,20,20,20,20
1000 DATA 89,84,20,20,20,20,20,20,20,20,
20,8F,85,20,20,20,20
1010 DATA 82,8F,8B,8C,20,20,20,20,8C,8C,
8F,8F,8D,20,20,20,20
1020 DATA 20,8F,85,8F,8F,8C,8C,8F,87,83,
8F,8F,8F,84,20,20,20
1030 DATA 20,8B,8F,86,8B,8F,8F,8F,89,8E,
8F,8F,87,8F,20,20,20
1040 DATA 20,8A,8F,8F,8A,8F,8F,87,8A,8F,
8B,8F,85,8F,85,20,20
1050 DATA 20,8A,8F,88,8E,8F,8F,8C,8F,85,
20,8F,85,8A,8D,20,20
1060 DATA 20,8F,87,8A,8F,8F,8F,8F,8F,8D,
84,8A,8F,8A,8F,8F,84:REM k2$
1070 DATA 20,8F,89,8F,8F,8F,8F,8F,8F,8F,
8D,CF,E7,CF,8F,8F,20
1080 DATA 8A,89,8F,8F,8F,FD,FD,FD,FE,CF,
FF,FF,CF,CF,FF,FD,2D
1090 DATA 20,8F,8F,8F,8F,8F,8F,8F,D8,CF,
FE,FE,D8,8F,87,20,20
1100 DATA 20,82,8F,8F,8F,8F,8F,8F,CF,DA,
D8,DA,CF,8F,81,20,20
1110 DATA 20,20,83,8F,8F,8F,8F,8F,CF,CF,
CF,CF,CF,8F,20,20,20
1120 DATA 20,20,20,20,8B,8F,8F,8F,8F,CF,
CF,CF,FF,81,20,20,20
1130 DATA 20,20,20,20,20,20,82,83,8F,8F,
87,81
1140 DATA 8F,8F,8F,87,8F,8F,8B,8F,85,8F,
85,20,8F,85,8A:REM          b$
1150 REM ________________________________
```



SOLITAIRE

Several variations exist of this popular game, but perhaps the best known and loved is this version which many of you will remember from your early days. Time is always a good test of a game's addictive quality, and I think it fair to say that with the time this puzzle has been around SOLITAIRE has passed that test *par excellence.*

A peg occupies each of the 32 holes bar the centre position. You move by selecting a peg to jump over another into an unoccupied position, and in doing so you remove the intervening peg. The object is to remove all but the last peg in this fashion and, if like the computer you are a perfectionist, you should aim to deposit the last peg in the centre hole.

A cursor will flash to indicate the active square, and you position this over the peg you wish to jump. Cursor movement is effected simply by using the four cursor keys. To make your move you press a cursor key to indicate the direction of your jump *whilst at the same time pressing SHIFT.* This method of communicating with the computer should prove far less tiresome than entering letters and numbers for each move. Moves will be recorded on screen and an option to exit from the program has been included. Sound is also there to enliven the proceedings! A demonstration mode is incorporated to show you that solutions do exist.

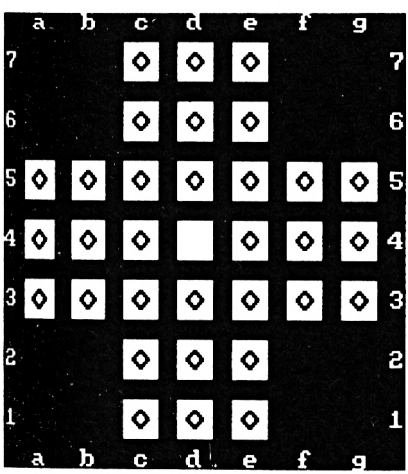
PROGRAM NOTES

Lines 50-260 set up the screen display. Lines 360-610 contain two subroutines to control the computer's and the player's moves.

Lines 610-670 check the validity of a move, and 690-780 execute it. The routines between lines 800-860 detect whether any valid moves remain.

A subroutine at line 870 causes the cursor to flash, and is called by the interrupt command EVERY at line 180.

Lines 920-990 initialise the main variables. Note the array a(8,8) which is used to identify the status of each 'hole' on the board, and to enable the computer to recognize areas which lie outside the playing zone. The dimensioned string D$(3) stores four different correct sequences for the demonstration mode.




```
10 REM __________ SOLITAIRE __________
20 REM ____________________________________
30 MODE 1:BORDER 16:INK 0,16:INK 1,26:IN
K 2,0:INK 3,6
40 GOSUB 920
50 REM _______ SCREEN DISPLAY __________
60 REM ________ PRINT  BOARD ___________
70 WINDOW 1,23,1,23
80 PAPER 0:CLS:PAPER 2:PEN 1
90 FOR i=1 TO 23 STEP 22:FOR j=1 TO 23
```

```
100 LOCATE j,i:IF (j+3)/3=(j+3)\3 THEN P
RINT CHR$(96+j\3) ELSE PRINT" ";
110 LOCATE i,j:IF (j+3)/3=(j+3)\3 THEN P
RINT USING "#";8-j\3; ELSE PRINT" ";
120 NEXT:NEXT
130 FOR i=2 TO 20 STEP 3:FOR j=2 TO 20 S
TEP 3
140 WINDOW j,j+2,i,i+2
150 IF (i<8 OR i>14) AND (j<8 OR j>14) T
HEN PAPER 2:CLS ELSE PAPER 1:PEN 2:PRINT
 CHR$(&87)CHR$(&83)CHR$(&8B)CHR$(&85)CHR
$(&CA)CHR$(&8A)CHR$(&8D)CHR$(&8C)CHR$(&8
E);
160 NEXT:NEXT
170 WINDOW p,p,q,q:PAPER 3:PRINT" "
180 EVERY 20 GOSUB 870
190 WINDOW 26,40,1,23:PAPER 2:CLS:PEN 1:
PRINT R$"   SOLITAIRE",R$:WINDOW 26,40,5
,23:LOCATE 1,18
200 g=g+1:IF g=1 THEN GOSUB 1000
210 REM
220 REM  CHOICE OF MODE: COMPUTER/PLAYER
230 PRINT "DO YOU WANT THE  COMPUTER TO
 MAKE THE MOVES?","KEY  [Y] OR [N]"CHR$(
7);
240 I$=INKEY$:I$=UPPER$(I$):IF I$<>"N" A
ND I$<>"Y" THEN 240
250 CLS:PRINT"   M O V E S":LOCATE 1,14:
PRINT R$"  GAME"SPC(5)CHR$(&18);:PRINT U
SING "##";g;:PRINT CHR$(&18)R$"  PEGS","
  REMOVED  "CHR$(&18);:PRINT USING "##";
t;:PRINT CHR$(&18)R$CHR$(&18);
260 c=1-(I$="N"):ON c GOSUB 360,480
270 REM _________ END OF GAME ___________
280 v=1:WINDOW 1,40,25,25:PAPER 0:EI:IF
e THEN 310
290 IF I$="N" THEN PEN 3:IF t<31 THEN PR
INT"* NO LEGAL MOVES LEFT *" ELSE PRINT"
```

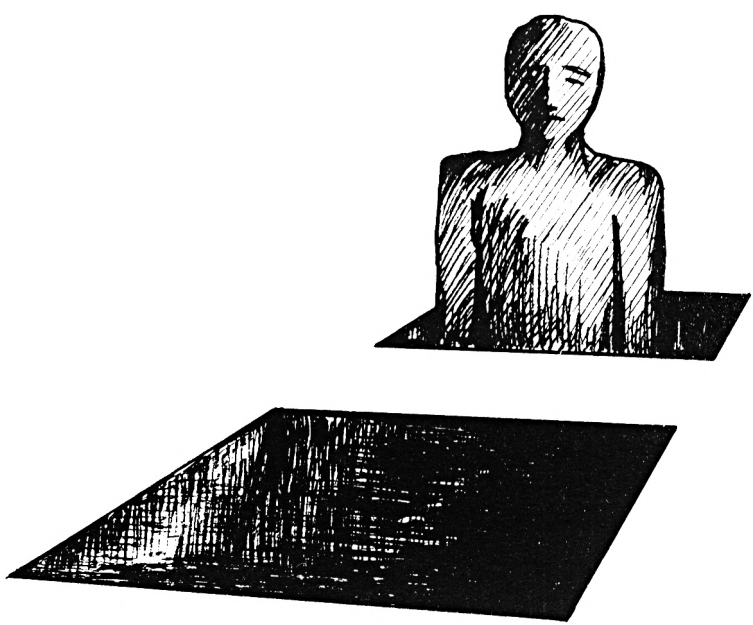
```
* W E L L   D O N E ! *"
300 FOR i=0 TO 9:FOR j=1 TO 6:SOUND 1,s(
j),7,5:NEXT:NEXT
310 CLS:PEN 2:PRINT"WOULD YOU LIKE ANOTH
ER GAME?  [Y] OR [N]"CHR$(7);
320 I$=INKEY$:I$=UPPER$(I$):IF I$<>"Y" A
ND I$<>"N" THEN 320
330 IF I$="N" THEN MODE 2:INK 0,0:INK 1,
25:BORDER 0:PAPER 0:PEN 1:END
340 MODE 1:INK 3,6:GOSUB 950
350 GOTO 50
360 REM _______ COMPUTER'S  MOVES ________
370 RANDOMIZE TIME:d=INT(RND*4)
380 FOR t=0 TO 30
390 WINDOW p,p,q,q:PAPER 1:PEN 2:PRINT P
$;:v=0
400 n1=n:p1=p:q1=q
410 p=(ASC(MID$(d$(d),t*4+1,1)))-66:q=(A
SC(MID$(d$(d),t*4+2,1)))-66
420 p2=(ASC(MID$(d$(d),t*4+3,1)))-66:q2=
(ASC(MID$(d$(d),t*4+4,1)))-66
430 n=a(q/3,p/3):P1$=CHR$(128-74*(n1=1)-
15*(n1=0)):P$=CHR$(128-74*(n=1)-15*(n=0)
)
440 WINDOW p1,p1,q1,q1:PAPER 1:PEN 2:PRI
NT P1$;:WINDOW p,p,q,q:PRINT CHR$(&18)P$
CHR$(&18);
450 FOR i=0 TO 500:NEXT
460 GOSUB 620
470 NEXT:RETURN
480 REM ________ PLAYER'S  MOVES _________
490 WHILE INKEY(63)=-1 AND e
500 DI:WINDOW p,p,q,q:PAPER 1:PEN 2:PRIN
T P$;:EI
510 n1=n:p1=p:q1=q:v=0
520 IF INKEY$="" THEN 520
530 p=p-3*(INKEY(1)=0)+3*(INKEY(8)=0):q=
q-3*(INKEY(2)=0)+3*(INKEY(0)=0)
```

```
540 p2=(INKEY(8)=32)-(INKEY(1)=32):q2=(I
NKEY(0)=32)-(INKEY(2)=32)
550 IF p<3 OR p<9 AND (q<9 OR q>15) OR p
>15 AND (q<9 OR q>15) OR p>21 OR q<3 OR
q<9 AND (p<9 OR p>15) OR q>15 AND (p<9 O
R p>15) OR q>21 THEN p=p1:q=q1:GOTO 520
560 IF p=3 AND p2=-1 OR p=21 AND p2=1 OR
 q=3 AND q2=-1 OR q=21 AND q2=1 THEN 520
570 n=a(q/3,p/3):P1$=CHR$(128-74*(n1=1)-
15*(n1=0)):P$=CHR$(128-74*(n=1)-15*(n=0)
)
580 DI:WINDOW p1,p1,q1,q1:PAPER 1:PEN 2:
PRINT P1$;:WINDOW p,p,q,q:PRINT P$;:EI
590 IF ABS(p2)=1 XOR ABS(q2)=1 THEN GOSU
B 620
600 WEND
610 RETURN
620 REM ____ CHECK VALIDITY OF MOVE ____
630 n2=a((q/3+2*q2),(p/3+2*p2)):n3=a((q/
3+q2),(p/3+p2))
640 IF n=1 AND n2=-1 AND n3=1 THEN 680
650 v=1:WINDOW 26,40,25,25:PAPER 0:PEN 3
:PRINT"!NOT PERMITTED!";
660 FOR i=0 TO 4:SOUND 1,s(i),7,5:NEXT:F
OR i=0 TO 500:NEXT:DI:CLS
670 RETURN
680 REM _________ EXECUTE MOVE __________
690 a(q/3,p/3)=-1:a((q/3+q2),(p/3+p2))=-
1:a((q/3+2*q2),(p/3+2*p2))=1
700 p3=p+3*p2:q3=q+3*q2:px=p2:qx=q2:IF I
$="N" THEN t=t+1
710 p2=p+6*p2:q2=q+6*q2
720 DI:WINDOW p,p,q,q:PAPER 1:PEN 2:PRIN
T" ";:WINDOW p3,p3,q3,q3:PRINT" ";:WINDO
W p2,p2,q2,q2:PRINT CHR$(&CA);
730 REM
740 REM __________ PRINT MOVE ___________
750 WINDOW x,x+2,y,y:PAPER 2:PEN 1:PRINT
```

```
 CHR$(96+p/3);:PRINT USING "#";8-q/3;:PR
INT CHR$(240-(qx<>)-1)-(ABS(px)=1)-(px=1)
);:WINDOW 37,38,22,22:PRINT CHR$(&18);:P
RINT USING "##";t-(I$="Y");
760 s(t)=INT(125000/(440*(2↑(oct+(10-nt)
/12)))+0.5):SOUND 1,s(t),7,5
770 nt=nt-1:IF nt=0 THEN oct=oct+1:nt=12
780 p=p2:q=q2:y=y+1:IF y=17-(x=36) THEN
x=x+4:y=7
790 REM
800 REM __ FURTHER LEGAL MOVES EXIST? __
810 e=0
820 FOR i=1 TO 7:FOR j=1 TO 7
830 IF a(i,j)<>1 OR e=1 THEN 850
840 IF a(i-1,j)+a(i,j)+a(i+1,j)=1 OR a(i
,j-1)+a(i,j)+a(i,j+1)=1 THEN e=1
850 NEXT:NEXT
860 RETURN
870 REM ___ FLASHING CURSOR & TEXT _____
880 IF I$="Y" THEN RETURN
890 n=a(q/3,p/3):P$=CHR$(128-74*(n=1)-15
*(n=0))
900 IF v THEN z=NOT z:INK 3,6-20*(NOT z)
 ELSE PRINT CHR$(&18)P$;
910 RETURN
920 REM _________ INITIALISE _________
930 DIM a(8,8),s(31),D$(3):s(0)=237
940 FOR i=0 TO 3:READ D$(i):NEXT
950 e=1:n=-1:nt=6:oct=0:p=12:q=12:t=0:v=
1:x=28:y=7:z=0
960 P$=" ":R$=STRING$(&F,CHR$(&9A))
970 FOR i=1 TO 7:FOR j=3 TO 5:a(i,j)=1:N
EXT:NEXT
980 FOR i=3 TO 5:FOR j=1 TO 7:a(i,j)=1:N
EXT:NEXT:a(4,4)=-1
990 RETURN
1000 REM _______ INTRODUCTION _________
1010 PRINT"  This is the    traditional
```

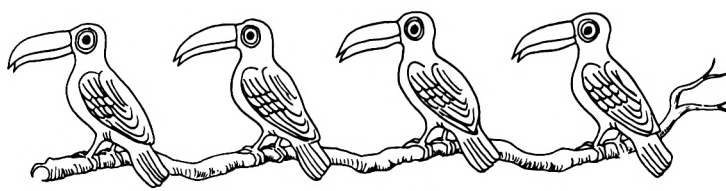
```
 game, where the object  is to remove  a
ll thepegs except thelast one (which sho
uld end up in the middle)."
1020 PRINT"You move  a pegby jumping ove
ranother  t o  a space beyond. The peg
jumped over  is then     removed.":PRINT
1030 PRINT"-PRESS ANY KEY-"CHR$(7);
1040 IF INKEY$="" THEN 1040
1050 PRINT:PRINT:PRINT" You  move the
cursor -the  flashing square -over the p
eg   you want to   move by using the cur
sor keys    ["CHR$(&F2)CHR$(&F1)CHR$(&F0
)CHR$(&F3)"]."
```

:PRINT:PRINT



```
1060 PRINT"You  then press[SHIFT] and th
erelevant cursorkey to show the directio
n of    your jump.  ":PRINT
1070 PRINT"-PRESS ANY KEY-"CHR$(7);
1080 IF INKEY$="" THEN 1080
1090 FOR i=1 TO 14:PRINT:NEXT
1100 RETURN:REM ________________________
1110 DATA NHBCTKABQEBCQNBAKECBQEBCQTBAWQ
ABNQCBWKBCWQABHQCBKWBAKNBCQWABKWBAKHBCEK
CBNKABEQBAEKCBNQBANKABHKBCHQCBNQCBTNABKN
CBQKBCTQABNTBA
1120 DATA NTBAHQCBKWBAKNBCQWABKHBCEKCBNK
ABEQBAEKCBTKABQEBCQNBAKECBQEBCQTBAWQABNQ
CBWKBCWQABNKBCNQCBTQBATKABNKABHNCBQNABKW
BAKQBAHKCBNHBC
1130 DATA TNABQTBAWQABNQCBWKBCHQCBKWBAKN
BCQWABKWBAKHBCEKCBNKABEQBAEKCBTKABQEBCQN
BAKECBQEBCKNCBQNBAQHABKHBCKNBCNTBANKBCWQ
ABQQABKTBAHNCB
1140 DATA HNCBKHBCEKCBNKABEQBATKABQEBCQN
BAKECBQEBCQTBAWQABNQCBWKBCWQABHQCBKWBAKN
BCQWABKWBAQNABKNBCKTCBQTBAQNBANHBCNQBAEK
CBKKCBQHBCTNAB
1150 REM ________________________________
```

FOUR-IN-A-ROW



FOUR-IN-A-ROW, sometimes known as 'Connect Four', is fast becoming a golden oldie. This version for the CPC464 simulates the game for two players that you may be familiar with.

Each player is provided with a number of counters (represented by distinctive orange and cyan diamonds) which in turn and one by one are 'dropped' into any of seven columns. Each counter falls to the lowest vacant spot in a column, and once there is not removed during the game. The object of the game, as the name implies, is to achieve a row of four of your own counters, and that row may be vertical, horizontal or diagonal. Naturally you will be hindered by your opponent who will be doing his or her best to block your efforts.

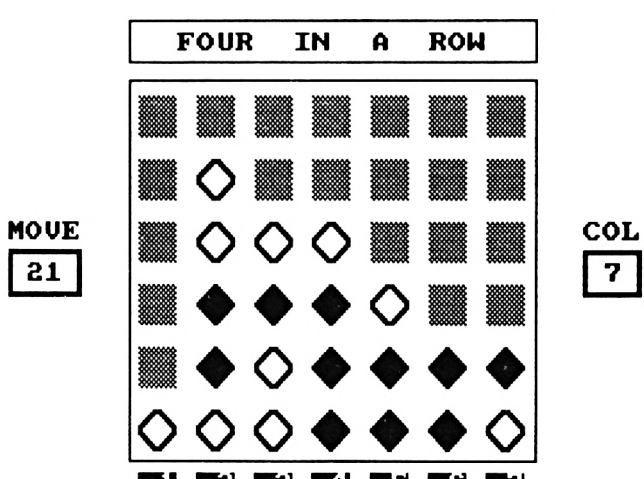
If you are unfortunate enough to have no-one to play with, you may instead pit your wits against the computer.

The program incorporates a full graphic screen display and will inform you as to whose turn it is, the number of counters dropped so far and the last column used. Each new position will be tested to recognize a winning combination. At the end of a game, use is made of the interrupt command EVERY to flash the winning player's counters.

PROGRAM NOTES

Lines 80-180 set up the screen display, and this routine is followed in lines 190-240 by the main program loop. Lines 250-370 comprise a subroutine for the player's input and a check on its validity. The routine between lines 380-450 effect the dropping of a counter with interesting sound effects. Lines 460-730 hold a check for winning combinations at every move, and is followed by the end-of-game routine at 740-900.

The lengthy routine from 910-1300 calculates the computer's next move when in the one-player mode. The subroutine at line 1310 comes into operation at the completion of a game and will flash the winning player's counters. The introduction to the game is stored at 1350-1440, and the following routine at line 1450 initialises the main variables. Note the two important arrays in line 1520: P(7) which holds the status of each column, and S(43) the status of each position on the matrix.



```
10 REM ________ FOUR IN A ROW __________
20 REM
30 GOSUB 1450
40 BORDER 0:INK 0,25:INK 1,1:PAPER 0:PEN
 1:MODE 0
50 LOCATE 3,11:PRINT LEFT$(L$,16):PRINT"
  FOUR  IN  A  ROW":PRINT TAB(3)LEFT$(L$
,16):FOR I=0 TO 3000:NEXT
60 INK 0,0:INK 1,15:INK 2,21:INK 3,10:MO
DE 1:GOSUB 1350
70 INK 1,0:INK 2,0:INK 3,0
80 REM _______ SCREEN  DISPLAY _________
90 LOCATE 13,2:PEN 2:PRINT"FOUR  IN  A
ROW"
100 MOVE 152,392:DRAW 488,392,2:DRAW 488
,360,2:DRAW 152,360,2:DRAW 152,392,2
110 MOVE 152,346:DRAW 488,346,2:DRAW 488
,56,2:DRAW 152,56,2:DRAW 152,346,2
120 WINDOW 4,7,11,14:PRINT"MOVE"CHR$(&FC
)CHR$(&9A)CHR$(&9A)CHR$(&FD)CHR$(&D3)"**
"CHR$(&D1)CHR$(&FE)CHR$(&9A)CHR$(&9A)CHR
$(&FF);
130 WINDOW 34,36,11,14:PRINT"COL"CHR$(&F
C)CHR$(&9A)CHR$(&FD)CHR$(&D3)"n"CHR$(&D1
)CHR$(&FE)CHR$(&9A)CHR$(&FF);
140 WINDOW 11,30,25,25
150 FOR I=11 TO 29 STEP 3:I$=STR$(I\3-2)
:FOR J=5 TO 23 STEP 3
160 WINDOW I,I+1,J,J+1:IF J<23 THEN PRIN
T C$; ELSE PAPER 2:PEN 0:PRINT LEFT$(I$,
2):PAPER 0:PEN 2
170 NEXT:NEXT
180 INK 1,15:INK 2,21:INK 3,20
190 GOSUB 250
200 C1=C1+1:IF C1=43 THEN 740
210 IF B1=2 THEN 190
220 GOSUB 910
230 C1=C1+1:IF C1=43 THEN 740
```

```
240 GOTO 190
250 REM ____________ PLAYER _____________
260 WINDOW 11,30,25,25:PEN 1-(2*(P1=2))
270 IF B1=1 THEN IF P1=1 THEN PRINT"
 Your turn!"; ELSE PRINT"  I'm thinking
..."
280 IF B1=2 THEN PRINT"Player"P1"- your
turn";
290 I$=INKEY$:IF I$="" THEN 290
300 A=ASC(I$)-48:IF A<1 OR A>7 THEN 290
310 IF P(A)<0 THEN PRINT"Column "A" is
full!";:FOR I=0 TO 999:NEXT:GOTO 270
320 E$=B$:IF P1=1 THEN E$=A$
330 GOSUB 380
340 S((P(A)/3)*7+A)=P1
350 GOSUB 460
360 P(A)=P(A)-3:IF G=1 THEN 740
370 J=P1:P1=0:0=J:RETURN
380 REM _________ DROP COUNTERS _________
390 FOR I=0 TO P(A):WINDOW A*3+8,A*3+9,I
+5+(I>0),21
400 IF I THEN IF (I+3)/3=(I+3)\3 THEN PR
INT D$; ELSE PEN 2:PRINT C$;
410 PEN 1-(2*(P1=2)):PRINT E$;
420 IF P1=1 THEN SOUND 1,200+I*50,5,15 E
LSE SOUND 1,950-I*50,5,15
430 NEXT
440 WINDOW 5,6,13,13:PRINT USING "*#";C1
;:WINDOW 35,35,13,13:PRINT USING "#";A;
450 RETURN
460 REM __________ WIN ROUTINE __________
470 G=0:IF P(A)<9 THEN IF S(((P(A)/3)+1)
*7+A)=P1 AND S(((P(A)/3)+2)*7+A)=P1 AND
S(((P(A)/3)+3)*7+A)=P1 THEN G=1
480 G1=1:I=0:IF A=7 THEN 510
490 I=I+1:IF S((P(A)/3)*7+A+I)=P1 THEN G
1=G1+1
```

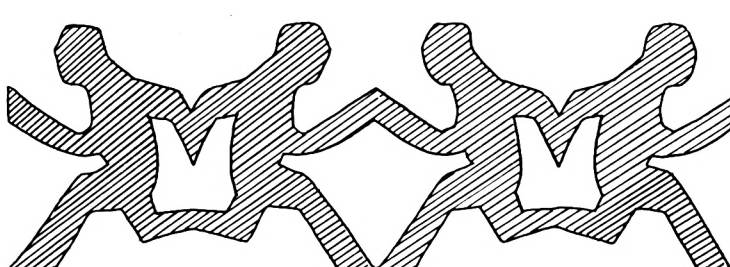
```
500 IF NOT(S((P(A)/3)*7+A+I)<>P1 OR A+I=
7) THEN 490
510 I=0:IF A=1 THEN 540
520 I=I+1:IF S((P(A)/3)*7+A-I)=P1 THEN G
1=G1+1
530 IF NOT(S((P(A)/3)*7+A-I)<>P1 OR A-I=
1) THEN 520
540 IF G1>=4 THEN G=1
550 G1=1:I=0:IF A=0 OR P(A)=15 THEN 590
560 I=I+1:IF S((P(A)*7/3)+A+8*I)=P1 THEN
 G1=G1+1
570 X=I+A+1:Y=INT(((P(A)*7/3)+A+8*(I+1)
)-I-A-1)/7)
580 IF NOT(S((P(A)*7/3)+A+I*8)<>P1 OR (N
OT((X<8) AND (Y<6)))) THEN 560
590 I=0:IF A=1 OR P(A)=0 THEN 630
600 I=I+1:IF S((P(A)*7/3)+A-I*8)=P1 THEN
 G1=G1+1
610 X=A-I-1:Y=INT(((P(A)*7/3)+A-8*(I+1)
)+I-A+1)/7)
620 IF NOT(S((P(A)*7/3)+A-I*8)<>P1 OR (N
OT((X>0) AND (Y>-1)))) THEN 600
630 IF G1>=4 THEN G=1
640 I=0:G1=1:IF A=0 OR P(A)=15 THEN 680
650 I=I+1:IF S((P(A)*7/3)+A+I*6)=P1 THEN
 G1=G1+1
660 X=A-I-1:Y=INT(((P(A)*7/3)+A+6*(I+1)
)-A+I+1)/7)
670 IF NOT(S((P(A)*7/3)+A+I*6)<>P1 OR (N
OT((X>0) AND (Y<6)))) THEN 650
680 I=0:IF A=7 OR P(A)=0 THEN 720
690 I=I+1:IF S((P(A)*7/3)+A-I*6)=P1 THEN
 G1=G1+1
700 X=A+I+1:Y=INT(((P(A)*7/3)+A-6*(I+1)
)-A-I-1)/7)
710 IF NOT(S((P(A)*7/3)+A-I*6)<>P1 OR (N
OT((X<8) AND (Y>-1)))) THEN 690
720 IF G1>=4 THEN G=1
```

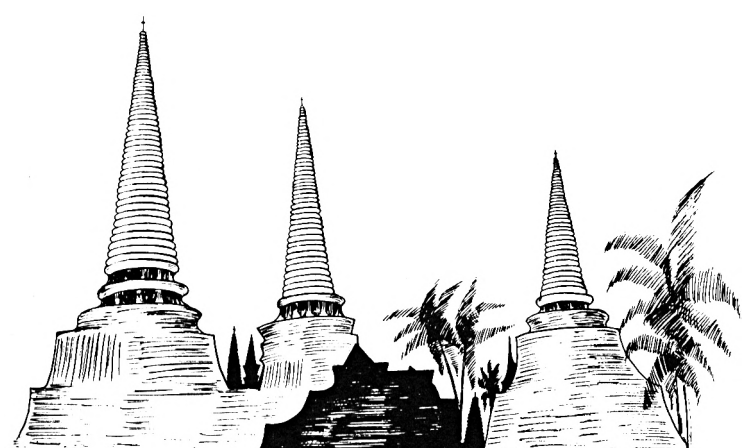
```
730 RETURN
740 REM _________ END OF GAME __________
750 WINDOW 11,30,24,25
760 IF C1=43 THEN PEN 2:PRINT "***  GAME
 DRAWN  ***";:GOTO 800
770 PEN 1-(2*(P1=2))
780 IF B1=1 THEN IF P1=1 THEN PRINT "***
   YOU WIN!   ***"; ELSE PRINT "***   I
 WIN !   ***";
790 IF B1=2 THEN PRINT"** PLAYER"P1"WINS
! **";
800 J=0:Q$="":PEN 2:PAPER 0:PRINT"ANOTHE
R GAME?  [Y/N]";
810 EVERY 20 GOSUB 1310
820 WHILE Q$<>"N" AND Q$<>"Y"
830 Q$=INKEY$:Q$=UPPER$(Q$)
840 WEND
850 PRINT REMAIN (0):INK 1,15:INK 2,21:I
NK 3,20:IF Q$="N" THEN INK 1,24:PEN 1:MO
DE 1: END
860 PEN 1:PAPER 0:PRINT"KEY [1] vs. comp
uter OR [2] vs. player  ";
870 Q$=INKEY$:IF Q$="" THEN 870
880 B1=ASC(Q$)-48:IF B1<1 OR B1>2 THEN 8
70
890 CLS:ERASE P,S:GOSUB 1450
900 PEN 2:GOTO 120
910 REM  _____ COMPUTER'S MOVES _________
920 WINDOW 11,30,25,25:PEN 3:PRINT"Think
ing ...          ";
930 I1=0:K1=0:G=0:IF P1=1 THEN E$=A$ ELS
E E$=B$
940 K1=K1+1:A=K1:IF P(K1)=-3 THEN 980
950 S((P(K1)/3)*7+K1)=P1
960 GOSUB 460
970 I1=-K1*(G=1):S((P(K1)/3)*7+K1)=0
980 IF NOT(I1>0 OR K1=7) THEN 940
990 IF G=0 THEN 1050
```

```
1000 A=K1
1010 GOSUB 380
1020 S((P(A)/3)*7+A)=P1:P(A)=P(A)-3
1030 GOTO 740
1040 GOTO 1300
1050 K1=0
1060 K1=K1+1:A=K1
1070 IF P(K1)=-3 THEN 1100
1080 J=P1:P1=0:O=J:S((P(K1)/3)*7+K1)=P1:
GOSUB 460
1090 J=P1:P1=0:O=J:I1=-K1*(G=1):S((P(K1)
/3)*7+K1)=0
1100 IF NOT(I1>0 OR K1=7) THEN 1060
1110 B=0
1120 A=K1:IF I1 THEN 1170
1130 A=INT(RND*7)+1
1140 IF C1=2 AND P(5)=15 THEN A=5
1150 IF C1=2 AND P(3)=15 THEN A=3
1160 IF P(A)=-3 THEN 1120
1170 IF I1>0 THEN 1260
1180 B=B+1:P(A)=P(A)-3:A1=A:K1=0:G=K1
1190 J=P1:P1=0:O=J:K1=K1+1:A=K1:IF P(K1)
<0 THEN 1220
1200 S((P(K1)/3)*7+K1)=P1:GOSUB 460
1210 S((P(K1)/3)*7+K1)=0
1220 J=P1:P1=0:O=J
1230 IF NOT(G=1 OR K1=7) THEN 1190
1240 P(A1)=P(A1)+3:IF G=1 THEN A1=A ELSE
 A=A1
1250 IF B<>5 AND G=1 THEN 1120
1260 GOSUB 380
1270 S((P(A)/3)*7+A)=P1:GOSUB 460
1280 IF G=1 THEN 740
1290 J=P1:P1=0:O=J:P(A)=P(A)-3
1300 RETURN
1310 REM ___ FLASH WINNING COLOUR _______
1320 IF C1<43 THEN J=NOT J:INK P1-(P1=2)
,ABS((15-5*(P1=2))*J)
```

```
1330 SOUND 1,0,40,15
1340 RETURN
1350 REM _______ INTRODUCTION __________
1360 INK 0,0:INK 1,0:INK 2,0:INK 3,10:PE
N 1
1370 LOCATE 1,4:PRINT L$"*","FOUR  IN  A
  ROW"SPACE$(12)"*"L$:PRINT
1380 PEN 2:PRINT "The object of this gam
e is to form a rowof four counters, whic
h may be vertical,horizontal or diagonal
.":PRINT
1390 PRINT "You drop one counter at a ti
me by press-ing a number from  1 to 7,
taking turnswith your opponent.":PRINT:P
RINT
1400 PEN 1:PRINT L$"NOW PRESS  [1] TO PL
AY WITH THE COMPUTER*OR PRESS  [2] TO PL
AY AGAINST A FRIEND*"L$
1410 INK 1,15:INK 2,21:INK 3,10:PRINT CH
R$(7)
1420 I$=INKEY$:IF I$="" THEN 1420
1430 B1=ASC(I$)-48:IF B1<1 OR B1>2 THEN
1420
1440 CLS:RETURN
1450 REM ________ INITIALISE ____________
1460 RANDOMIZE TIME
1470 SYMBOL AFTER &FB
1480 SYMBOL &FC,&0,&0,&0,&FF,&FF,&C0,&C0
,&C0
1490 SYMBOL &FD,&0,&0,&0,&FF,&FF,&3,&3,&
3
1500 SYMBOL &FE,&C0,&C0,&C0,&FF,&FF,&0,&
0,&0
1510 SYMBOL &FF,&3,&3,&3,&FF,&FF,&0,&0,&
0
1520 DIM P(7),S(43)
```

```
153Ø A$=CHR$(&D6)+CHR$(&D7)+CHR$(&D5)+CH
R$(&D4):B$=CHR$(&CC)+CHR$(&CD)+CHR$(&CD)
+CHR$(&CC):C$=STRING$(4,CHR$(&CF)):D$="
 ":L$=STRING$(&28,CHR$(&9A))
154Ø C1=1:O=2:P1=C1:G=Ø:ZONE 11
155Ø FOR I=Ø TO 43
156Ø S(I)=Ø:IF I<7 THEN P(I+1)=15
157Ø NEXT
158Ø RETURN:REM ________________________
```





TOWERS OF HANOI

This mindbender is fast becoming very popular amongst those who enjoy a good brainteaser which demands no small amount of concentration. As with other games of its type there are levels of skill ranging from the very easy to the unimaginably difficult.

The game is believed to have originated in Hanoi and to have been devised by Buddhist monks as an aid to concentration. The player is presented with three rods. On the first of these is placed a number of discs of different size, largest at the bottom and smallest at the top. These discs must be transferred to the third rod, one by one, and in a way that no disc is ever placed on one of smaller size. To complete the task you will find it necessary to make use of the middle rod.

Difficulty increases exponentially with each additional disc employed, and you may choose between two (very easy) to eight (extremely hard). The computer will display the minimum number of moves necessary to solve the problem, which is calculated as

2 to the power of the number of discs, less 1.

If therefore you chose the maximum eight discs the minimum number of moves necessary would be

2↑8–1 (=255).

You may of course exit from the game if you become too frustrated (key[Q] to do so), and you may if you wish watch the computer perform the moves for you. In the demonstration mode you will be offered a choice of speed. At the faster of the two, the computer will move eight discs according to the rules in well under a minute. See how long it takes you to do the same!

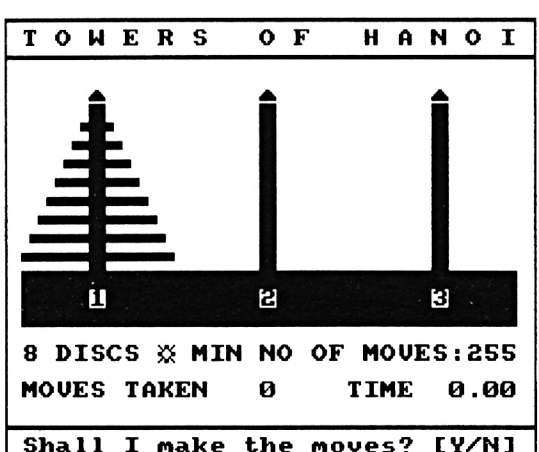
An additional timing feature has been included to demonstrate the use of *interrupts* in Locomotive BASIC. In the previous listing the command EVERY was used to perform a different task. In the running of this program you will be timed, although no time limit is imposed for completion of moves.

PROGRAM NOTES

Lines 80-330 set up the screen display and offer the user a choice between manual effort and a demonstration. Lines 350-490 control the player's inputs, and the three subroutines at 570-710 erase and print the discs and otherwise update the visual display.

In the demonstration mode lines 710-900 calculate the computer's moves. Lines 910-940 operate the clock, called by an interrupt command.

The routine at 950-980 initialise the main variables. Note the arrays n(3) and d(8,3) which are used to record the status of each rod (a maximum of eight discs on three rods). The dimensioned string d$(8) holds a graphic representation of the discs.



```
10 REM --------------------------------
20 REM TOWERS OF HANOI: a classic puzzle
30 REM --------------------------------
40 BORDER 15:INK 0,15:INK 1,1:PAPER 0:PE
N 1:MODE 0
50 LOCATE 3,12:PRINT"TOWERS OF HANOI"
60 RUN 950
70 REM
80 REM ________ SCREEN DISPLAY _________
90 INK 1,0:INK 2,0
100 WINDOW 5,35,1,17:PEN 2:PRINT 1$" T O
 W E R S   O F   H A N O I"1$
110 WINDOW 6,34,5,17:PEN 1:PRINT"     "CH
R$(&F4);TAB(15)CHR$(&F4);TAB(25)CHR$(&F4
);
120 FOR i=2 TO 10:LOCATE 5,i:PRINT CHR$(
&8F);TAB(15)CHR$(&8F);TAB(25)CHR$(&8F);:
NEXT
130 WINDOW 6,34,15,17:PEN 2:PRINT STRING
$(&21,CHR$(&8F))"1"STRING$(&9,CHR$(&8F))
"2"STRING$(&9,CHR$(&8F))"3"STRING$(&4,CH
R$(&8F))STRING$(&1D,CHR$(&8F));
140 WINDOW 5,35,23,25:PRINT 1$SPACE$(29)
1$;
150 MOVE 64,7:DRAW 64,393:MOVE 560,393:D
RAW 560,7
160 INK 1,24:INK 2,21
170 WINDOW 6,34,24,24:PEN 1:PRINT " How
 many  discs?  (2 to 8) ";CHR$(7);
180 i$=INKEY$:IF i$="" THEN 180
190 c=ASC(i$)-48:n=c:nn=0:m=0:sec=0:IF c
<2 OR c>8 THEN 180
200 WINDOW 6,34,19,19:PEN 2:PRINT i$" DI
SCS "CHR$(&EE)" MIN NO OF MOVES:";:PRINT
 USING "###";2↑n-1;
210 WINDOW 6,34,21,21:PRINT"MOVES TAKEN
  0     TIME  0.00";
```

```
220 FOR i=1 TO n:FOR j=1 TO 3:d(i,j)=0:n
(j)=0:NEXT:NEXT:y=1
230 FOR i=1 TO n
240 n(y)=i:d(n(y),y)=n+1-i
250 GOSUB 650
260 NEXT
270 WINDOW 6,34,24,24:PEN 1:PRINT "Shall
 I make the moves? [Y/N]";CHR$(7);
280 i$=INKEY$:i$=UPPER$(i$):IF i$<>"Y" A
ND i$<>"N" THEN 280
290 s=0:cc=ASC(i$):IF cc=78 THEN 340
300 PRINT"  Fast or slow speed? [F/S]  "
;CHR$(7);
310 i$=INKEY$:i$=UPPER$(i$):IF i$<>"F" A
ND i$<>"S" THEN 310
320 s=(i$="F"):CLS:IF s THEN PRINT"
See this for speed!";
330 EVERY 50 GOSUB 910:GOTO 720
340 REM
350 REM _________ PLAYER OPTION __________
360 PEN 3:PRINT" Press  [Q]  to  quit  p
lay  ";:FOR i=0 TO 2000:NEXT
370 EVERY 50 GOSUB 910
380 DI:WINDOW 6,34,24,24:CLS:PEN 1:PRINT
"Move from which rod? ...  ";:EI
390 i$=INKEY$:i$=UPPER$(i$):IF (i$<"1" O
R i$>"3") AND i$<>"Q" THEN 390
400 IF i$="Q" THEN nn=0:GOTO 500
410 x=VAL(i$):IF n(x)<=0 THEN DI:WINDOW
6,34,24,24:CLS:PEN 3:PRINT"       Rod"x"
is empty!";:EI:FOR i=0 TO 999:NEXT:GOTO
380
420 DI:WINDOW 32,34,24,24:PRINT x;:EI:FO
R i=0 TO 500:NEXT
430 DI:WINDOW 6,34,24,24:CLS:PEN 2:PRINT
"...  to which rod?         ";:EI
440 i$=INKEY$:i$=UPPER$(i$):IF (i$<"1" O
R i$>"3") AND i$<>"Q" THEN 440
```

```
450 IF i$="Q" THEN 400
460 y=VAL(i$):IF y=x THEN 440 ELSE DI:WI
NDOW 32,34,24,24:PRINT y;:EI
470 IF n(y)=0 OR  d(n(x),x)<d(n(y),y) TH
EN 480 ELSE DI:WINDOW 6,34,24,24:CLS:PEN
 3:PRINT" Not permitted!   Try again.";:
EI:FOR i=0 TO 999:NEXT:GOTO 380
480 GOSUB 570
490 IF n(3)<n THEN 380
500 REM __________ END OF GAME __________
510 WINDOW 6,34,24,24:CLS:PEN 1:PRINT"
 ANOTHER  GAME ?   [Y/N]   ";CHR$(7);
520 i$=INKEY$:i$=UPPER$(i$):IF i$<>"Y" A
ND i$<>"N" THEN 520
530 IF i$="N" THEN MODE 2:END
540 WINDOW 6,34,19,21:CLS:WINDOW 6,9,6,1
4:CLS:WINDOW 11,19,6,14:CLS:WINDOW 21,29
,6,14:CLS:WINDOW 31,34,6,14:CLS
550 FOR i=10 TO 30 STEP 10:WINDOW i,i,6,
14:PAPER 1:CLS:NEXT:PAPER 0:GOTO 170
560 REM
570 REM __________ SUBROUTINES __________
580 GOSUB 680
590 n(y)=n(y)+1:d(n(y),y)=d(n(x),x)
600 GOSUB 650
610 d(n(x),x)=0:n(x)=n(x)-1:nn=nn+1
620 DI:WINDOW 18,20,21,21:PEN 2:PRINT US
ING "###";nn;:EI
630 IF NOT s THEN FOR i=0 TO 999:NEXT
640 RETURN
650 REM __________ PRINT DISCS __________
660 d=d(n(y),y):h=y*10-4:v=15-n(y)
670 DI:WINDOW h,h+8,v,v:PEN 3:PRINT LEFT
$(d$(d),4);:PAPER 1:PRINT CHR$(&83);:PAP
ER 0:PRINT RIGHT$(d$(d),4);:EI:RETURN
680 REM __________ ERASE DISCS __________
690 d=d(n(x),x):h=x*10-4:v=15-n(x)
```

```
700 DI:WINDOW h,h+8,v,v:PEN 1:PRINT"
"CHR$(&8F)"     ";:EI:RETURN
710 REM _______ COMPUTER'S  MOVES ________
720 nn=0:IF n=1 OR n=3 OR n=5 OR n=7 THE
N 780
730 REM
740 p=1:q=2:GOSUB 840
750 p=1:q=3:GOSUB 840
760 p=2:q=3:GOSUB 840
770 IF n(3)=n THEN 500 ELSE GOTO 740
780 REM
790 p=1:q=3:GOSUB 840
800 IF n(3)=n THEN 500
810 p=1:q=2:GOSUB 840
820 p=3:q=2:GOSUB 840
830 GOTO 780
840 REM
850 x=p:y=q:IF n(p)=0 THEN x=q
860 IF n(p)=0 OR n(q)=0 THEN 880
870 IF d(n(p),p)>d(n(q),q) THEN x=q
880 IF x=q THEN y=p
890 GOSUB 570
900 RETURN
910 REM _______ TIME LAPSE ROUTINE _______
920 IF n(3)=n OR nn=0 THEN RETURN
930 sec=sec+0.01:IF sec>=0.6 THEN m=m+1:
sec=0
940 WINDOW 29,34,21,21:PEN 2:PRINT USING
 "###.##";(m+sec);:RETURN
950 REM ___________ INITIALISE ___________
960 DIM d(8,3),n(3):DIM d$(8):l$=STRING$
(&28,CHR$(&9A))
970 FOR i=1 TO 8:FOR j=1 TO 8:READ n:d$(
i)=d$(i)+CHR$(n):NEXT:NEXT
980 FOR i=0 TO 2000:NEXT
990 REM __________ INTRODUCTION __________
1000 MODE 1:INK 2,15:INK 3,15:PEN 3
```

```
1010 PRINT 1$"    T O W E R S    O F
H A N O I",1$;
1020 PEN 2:PRINT "This is a game which i
s believed to haveoriginated in Hanoi, d
evised by Buddhistmonks as an aid to con
centration.":PRINT
1030 PRINT "The rules are straightforwar
d,  althoughthe solution may not be.":PR
INT
1040 PRINT "There are three rods or colu
mns.  On thefirst  of  these  a  number
 of discs isplaced.  Each is of differen
t size, withthe smallest at the top  and
 the largestat the bottom.":PRINT
1050 PRINT "All  these  discs must be tr
ansferred tothe third rod or column, one
 by one, andin  a  way  that  no disc li
es on one ofsmaller size.   When  discs
 are  moved,therefore,  it  will be neces
sary to makeuse of the middle rod."
1060 PEN 3:PRINT 1$"-PRESS THE SPACE BAR
 WHEN YOU ARE READY-"1$;
1070 BORDER 0:INK 0,0:INK 1,24:INK 2,21:
PRINT CHR$(7);
1080 IF INKEY$<>" " THEN 1080
1090 CLS:1$=LEFT$(1$,31):GOTO 90
1100 REM ____ DATA FOR DISCS [d$()] ____
1110 DATA 128,128,128,130,129,128,128,12
8,128,128,128,131,131,128,128,128
1120 DATA 128,128,130,131,131,129,128,12
8,128,128,131,131,131,131,128,128
1130 DATA 128,130,131,131,131,131,129,12
8,128,131,131,131,131,131,131,128
1140 DATA 130,131,131,131,131,131,131,12
9,131,131,131,131,131,131,131,131
1150 REM _____________________________
```

IMPROVING YOUR PROGRAMMING SKILLS

You may well have developed some ideas of your own that you feel would be suitable subjects for your computer to handle. However much you enjoy typing-in listings from books and magazines, it is far more rewarding to write a program of your own and get it to work. If you are stumped for ideas there is no harm in taking someone else's raw material and developing that into something that bears your own hallmark. Either way you are sure to derive greater pleasure and satisfaction from your task than you would otherwise experience.

The purpose of this chapter is to help you translate an idea into a working program, and for this purpose I shall take an idea of my own which I have developed into a game called PROVERBS.

*

So how would you begin? Suppose you were writing a short note or a précis. You would no doubt pick up your pen and simply write it without the aid of any notes. If, however, you wished to write a more substantial piece – an essay or an article perhaps – you would probably want to note down the raw material with subject headings, and give some thought to an ordered presentation. Unless you possessed unusual clarity of thought coupled with a highly retentive memory, failure to draw up such a plan would almost certainly lead to an unstructured mess, difficult and confusing to read.

It is the same with writing a computer program. In fact a logical approach to the task is of supreme importance here because a computer will take absolutely nothing for

granted. Your machine will do precisely what you command it to do — no more and no less.

*

Suppose that we want to write a game in which a player has to guess a well-known proverb selected randomly from a list of such proverbs held by the computer in its memory. Step by step, this is how it might be planned:

1. Define likely variables: a turn counter and a game counter, and strings to hold a chosen proverb, player's guess, etc.
2. Decide how many proverbs the computer can choose from, and what they are going to be.
3. Select one of these proverbs randomly. Ensure that the same proverb is not chosen again during the same game.
4. Give the player a clue as to the identity of the selected proverb.
5. Invite the player to enter a guess, either the whole proverb or a single letter.
6. Examine the player's input:
   a. Has the whole proverb been entered correctly?
      Yes? Then go to step 9.
      No? Then . . .
   b. Has the player entered a letter?
      No? Then go to step 5.
      Yes? Then . . .
   c. Has this letter been chosen before?
      Yes? Then display a message to this effect and go to step 5.
      No? Then proceed to step 7.
7. Does the inclusion of this new letter complete the proverb?
      Yes? Then go to step 9.
      No? Then proceed to step 8.
8. Examine the hidden proverb, letter by letter, to see if the entered letter occurs there. If it does, every occurrence of that letter must be displayed in its correct position on screen.
9. Increment by one the number of turns taken.

10. Re-print the string holding the clue to the proverb (if the whole proverb has been guessed correctly or the last letter entered completes the proverb, this string will hold the whole proverb).
11. If the string holding the clue is different from the proverb then go to step 5 to repeat the sequence.
12. Display the number of turns (guesses) taken to complete the proverb.
13. Are all the proverbs exhausted?
    No? Then clear the display and go to step 3.
    Yes? Then . . .
14. Does the player want another game session?
    Yes? Then run the program again.
    No? Then . . .
15. End.

Here then is an outline of the program, setting down in logical sequence what we expect the computer to do. You may find it helpful to draw up a flowchart to hold this information, but this is by no means essential. It is, however, a good idea to draw up a representation of the screen display so that you have a clear idea of what a program will look like when it is RUNning. Ideally this should be done on a sheet of graph paper, or you could use the 'Text and Window Planner' to be found in Appendix VI of the user manual.

*

Now to the task of converting the plan to a form the computer will understand and execute. So back to step 1 where we have to decide what variables will be required to store changing information relating to the program. When formulating your own programs do not worry unduly if you later find that you need more variables than you originally allocated: additional variables may easily be added at a later stage in a program's development. For the purposes of this program I have designated the following key variables to hold the information stated:

proverb$	This is actually an array of 16 strings, all having the same name, but differentiated from one another by a numerical suffix *or subscript.* The numbers range from Ø to 15, and each string will hold a different proverb.
guess$	A string of dots and spaces representing the arrangement of words in the chosen proverb. This will serve as a clue for the player. As the game progresses dots will be replaced by letters as they are guessed.
try$	Holds the player's guess at each turn. This may be the whole proverb or it may consist of a single-letter guess.
go$	Stores information relating to proverbs already used to prevent duplication.
letter$	Used to record letters of the alphabet already tried. It will appear on the display to remind the player of previous guesses.
rule$	Holds 40 [—] characters and exists for the purposes of creating a neat screen display.
game	Counts the number of games.
turn	Counts the number of guesses in each game.
col	Holds column number calculated according to length of a proverb, and is used to determine print start position on the screen.
repeat	A check variable. This is set to Ø or 1 depending upon whether a condition is false (Ø) or true (1). It is employed to obviate the bad programming practice of jumping out of loops (FOR. . .NEXT, WHILE. . .WEND).
a	Is a loop control variable.
r	Holds a random number used to select a particular proverb from one of 16 held in the array proverb$.

Note that where appropriate I have used variable names which roughly correspond to the function they perform. This is particularly useful in a program's development, especially if you leave a program unfinished and return

to it at a later date. It is surprisingly easy to forget what a variable is supposed to do when work on a listing is resumed. Having said that I will also say that the longer a program is, the more memory it will occupy and, of course, it follows that SAVEing and LOADing times will be correspondingly longer. Once a program is completed and has been fully tested and errors eliminated (i.e. it has been *debugged*) you may want to shorten variable names in order to save memory. You should also bear in mind that the speed at which a program runs is determined to some extent by the amount of memory that it occupies. Moreover, speed of execution will also be improved by the use of integer variables (refer to chapter IV, page 6, of the user manual for an explanation of these). With the program in hand, as with most other programs in this book, speed is not of paramount importance, and to save repetitive typing of [%] symbols I have not made use of them.

A further point to bear in mind when writing a program is to make good use of REM statements. REM is a keyword that the computer will ignore, but it is there to help you, the programmer, remember what each part of a program does.

*

Here then is the start of our program.

```
10 REM ----------- PROVERBS ------------
20 REM
30 REM ---------- INITIALISE -----------
40 MODE 1:BORDER 0:INK 0,0:INK 1,25:PEN
1:PAPER 0
50 RANDOMIZE TIME
60 DIM proverb$(15):rule$=STRING$(40,CHR
$(154))
70 game=0:go$="Q"
80 FOR a=0 TO 15
90 READ proverb$(a)
100 NEXT a
```

Line 40 sets the display to 40 columns of text and determines the colours that are going to be used. In this mode we could use four colours in all, but two here will suffice. Line 50 sets the random number seed to ensure a different sequence of proverbs in each game. Lines 60-100 initialise variables. In line 60 sixteen strings are dimensioned (Ø to 15) and these will store the sixteen proverbs chosen. The FOR...NEXT loop in lines 80-100 'fill' these strings, one by one, with the proverbs which will be entered next in lines of DATA statements. I will adhere to the convention whereby data is stored in a block at the end of a listing, and as this is not a very long program I have chosen as a starting point line 1000.

```
1000 DATA STRIKE WHILE THE IRON IS HOT
1010 DATA PRACTICE MAKES PERFECT
1020 DATA A STITCH IN TIME SAVES NINE
1030 DATA MAKE HAY WHILE THE SUN SHINES
1040 DATA PRIDE GOES BEFORE A FALL
1050 DATA MANY HANDS MAKE LIGHT WORK
1060 DATA TOO MANY COOKS SPOIL THE BROTH
1070 DATA LOOK BEFORE YOU LEAP
1080 DATA SPARE THE ROD AND SPOIL THE CH
ILD
1090 DATA TIME AND TIDE WAIT FOR NO MAN
1100 DATA BETTER LATE THAN NEVER
1110 DATA EVERY CLOUD HAS A SILVER LININ
G
1120 DATA POUR OIL ON TROUBLED WATERS
1130 DATA EMPTY VESSELS MAKE THE MOST NO
ISE
1140 DATA THERE'S NO SMOKE WITHOUT FIRE
1150 DATA GREAT OAKS FROM LITTLE ACORNS
GROW
```

You may, if you wish, test what you have typed so far by adding

```
999 STOP
RUN
```

remembering, of course, to key ENTER after each command. If you try this, the program will stop at line 999, apparently without doing anything else. But now type, as a direct command,

? proverbs$(15)

and you should now see

GREAT OAKS FROM LITTLE ACORNS GROW

displayed on your monitor. (If you do not get this, please check your typing.)

*

Now on to the screen display and some PRINT statements.

```
110 REM -------- SCREEN DISPLAY --------
120 turn=0:guess$="":letter$=""
130 PRINT rule$;TAB(13)"P R O V E R B S"
,rule$
140 PRINT"  Can you guess the following
proverb?"
150 PRINT:PRINT"  Each dot represents a
letter.  Enter"TAB(4)"the whole proverb
if you think you"TAB(3)"know what it is,
 otherwise a letter."
160 LOCATE 1,25:PRINT rule$;
```

These lines are largely self-explanatory, but note the use of the TAB function to position text on the screen, in much the same way that you would use a typewriter tabulator.

The next step is to select a proverb.

```
170 REM -- SELECT A PROVERB AT RANDOM --
180 r=INT(RND*16):repeat=0
190 FOR a=1 TO LEN(go$)
200 IF r=ASC(MID$(go$,a,1))-65 THEN repe
at=1
210 NEXT a
220 IF repeat=1 THEN 180
```

Variable r in line 180 will now hold a random number between Ø and 15 inclusive. Line 190 signals the start of another FOR. . .NEXT loop designed to check whether the random number has been chosen before. The string go$ will store this information. If any number has previously been selected the check variable *repeat* will be set to 1 (line 200) and the computer will be referred back to line 180 at line 220. You may think this is a rather cumbersome way of doing things, but remember that computers take nothing for granted, and it is bad programming practice to jump out of incomplete loops.

If the check at line 220 is passed, execution continues at line 230.

```
230 go$=CHR$(r+65)+go$
240 col=1+(40-LEN(proverb$(r)))\2
250 FOR a=1 TO LEN(proverb$(r))
260 IF MID$(proverb$(r),a,1)>"@" THEN gu
ess$=guess$+"." ELSE guess$=guess$+MID$(
proverb$(r),a,1)
270 NEXT a
```

Line 230 adds the current random number (stored as a letter) to the string holding details of previously generated numbers (go$). Line 240 sets the variable *col* to indicate where the proverb shall be printed according to its length. Strictly speaking *col* is not necessary, but it serves to centre the text and therefore enhance the quality of the screen display. The loop that begins at line 250 inserts dots and spaces into the string that serves as a clue for the player (guess$).

Line 290 signals the start of a different type of loop.

```
280 REM ------ MAIN PROGRAM LOOP -------
290 WHILE guess$<>proverb$(r)
300 LOCATE col,13:PRINT guess$
310 LOCATE col,16:INPUT"",try$
320 LOCATE 1,16:PRINT SPACE$(40)
```

```
330 try$=UPPER$(try$)
340 IF try$=proverb$(r) THEN guess$=try$
:GOTO 490
350 try$=LEFT$(try$,1)
360 IF try$<"A" OR try$>"Z" THEN 310
370 repeat=0
380 FOR a=1 TO LEN(letter$)
390 IF MID$(letter$,a,1)=try$ THEN repea
t=1
400 NEXT a
410 IF repeat=0 THEN 450
420 LOCATE 3,25:PRINT"YOU'VE ALREADY TRI
ED '"try$"'. TRY AGAIN!";
430 FOR a=0 TO 999:NEXT a
440 LOCATE 1,25:PRINT rule$;:GOTO 310
450 letter$=letter$+try$
460 FOR a=1 TO LEN(proverb$(r))
470 IF try$=MID$(proverb$(r),a,1) THEN M
ID$(guess$,a,1)=try$
480 NEXT a
490 turn=turn+1
500 IF guess$<>proverb$(r) THEN LOCATE 1
,23:PRINT"Letters tried: "letter$;
510 WEND
```

This is the main program loop. While FOR...NEXT loops process instructions a given number of times (the number and range being stated after the control variable), WHILE... WEND instructs the computer to repeat the loop until the condition specified after WHILE becomes false. In fact, should that condition be found to be false at the outset, the loop will be skipped altogether. In line 290 two strings are being compared: guess$, which holds the player's guess(es) and proverb$(r) which holds the hidden proverb. Eventually the two will be identical, at which time control of the program will pass to the final sequence following line 510. How long that takes will depend upon the player's skill or intuition.

Line 310 invites the player's guess which is then stored temporarily in try$. The line following cancels that text on the screen because it will very shortly be transferred to guess$ in line 340 or 470. Line 330 ensures that the player's input is converted to upper case letters to facilitate a valid comparison. Line 340 makes the first of two comparisons, and establishes whether or not the player has correctly guessed the whole proverb. If he or she has, control of the program will pass very quickly to the final sequence. If the player fails to guess the whole proverb, or simply enters a single letter, line 350 ensures that only the first character of the input string is considered, and line 360 will reject any character which is not also a letter of the alphabet.

The FOR...NEXT loop between 380 and 400 checks to see if the input letter has already been used. If it has, a message to that effect is displayed and the program is referred back to the INPUT command in line 310. If the letter concerned has not been tried previously, that letter is added to the list at line 450. The loop following (460–480) examines the hidden proverb to see if the new letter occurs there. If it does, the appropriate dots are converted to the letters they represent. Line 490 increments the count of guesses (turn), and 500 re-prints the growing list of guesses.

*

Now comes the final sequence.

```
520 REM ----- END OF GAME SEQUENCE -----
530 game=game+1
540 LOCATE col,13:PRINT guess$
550 LOCATE 1,23:PRINT" YOU GOT IT IN"tur
n": PRESS THE SPACE BAR "
560 IF INKEY$<>" " THEN 560
570 CLS:IF game<16 THEN 120
580 LOCATE 1,11:PRINT "You have now comp
leted all the proverbs.","Do you want to
```

```
 try guessing them again,","     this tim
e in a different order?";TAB(11)"(Press
[Y] or [N])"
590 try$=INKEY$:try$=UPPER$(try$)
600 IF try$<>"N" AND try$<>"Y" THEN 590
610 IF try$="Y" THEN RUN
620 CLS:END
```

This last part completes the game. Line 530 increments the game counter, and if all the proverbs have been guessed the player is informed in line 580 that this is so. Line 540 ensures that the entire proverb is now displayed, and the line following informs the player as to the number of guesses taken. If further proverbs remain to be guessed then line 570 will send control back to the point where a new proverb is chosen after initialising the relevant variables (line 120).

*

If, after playing the game, you want a new set of proverbs, simply change any or all of the DATA statements. If, however, you wish to add more data you will also need to increment the number 15 in lines 60, 80 and 180, and the number 16 in line 570.

```
P R O V E R B S

Can you guess the following proverb?

Each dot represents a letter.  Enter
 the whole proverb if you think you
know what it is, otherwise a letter.


.IME .N. .I.E W.I. ,OR NO M.N


Letters tried: MCIUWPOEGNR
```

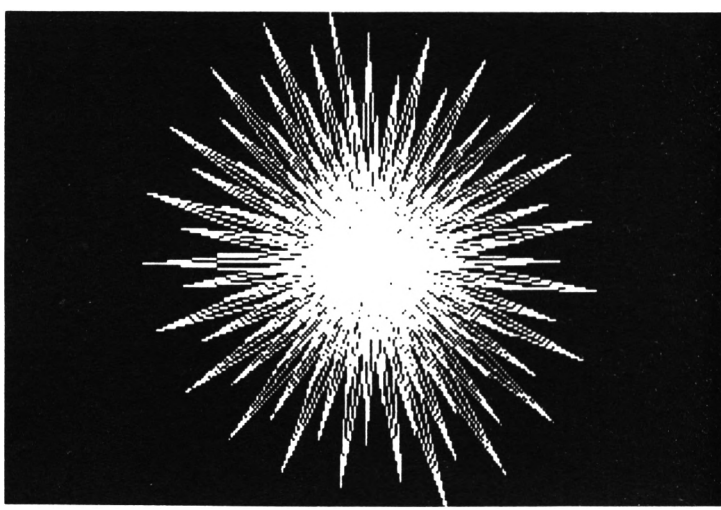
PROVERBS is not a particularly complex program, but it does, I believe, illustrate a number of useful programming techniques. You may have a more complex idea to mind, in which case a rather different approach would be appropriate. A logical approach to such a program would be to create a number of subroutines, each performing a specific function and each clearly labelled by REM statements. Each routine could be called with the command GOSUB from within a small controlling (main) program. Each subroutine would be terminated with the command RETURN. You would thus have a series of GOSUB commands issued in a sequence corresponding to the order in which you wanted to use each routine. The strong advantage of this approach lies in its clarity and the ease with which each separate area of the program may be tested.

*

One final note. When you write a program you will know what the computer expects you to do when the program is running. You cannot, however, expect others to know instinctively unless you are there personally to tell them, *unless* the program includes all the necessary instructions and appropriate prompts. You will also need to guard against erroneous inputs, whether entered accidentally or intentionally. Remember, therefore, to include the necessary checks on the validity of input data. You would be amazed at the number of 'professional' programs that lack them.

*

This chapter is little more than an introduction to what is becoming a vast subject. The world of computing is a fascinating one, and it is my hope that these pages have provided you with some knowledge and inspiration to explore further what is undoubtedly a very enjoyable and rewarding hobby.



ALL THE PROGRAMS IN THIS BOOK
are also available on cassette, price £3.50*
(including UK postage and packing).
Orders, accompanied by remittances,
should be sent to

EXCALIBUR PUBLICATIONS
The Glass House
9-13 Wensum Street
Norwich, Norfolk NR3 1LA
England

Please allow 28 days for delivery.

Overseas customers:
please add £0.50 to cover additional
postage costs. Wherever possible
remittances should be in sterling.

**Special offer price applicable to orders received by 30 June 1985. If ordering after that date please check price and availability.*

MINDBENDING GAMES for the AMSTRAD CPC 464

Superb games of originality for all the family!

Here in one volume you will find challenging programs for you to put into your microcomputer. Each game has been produced to professional standards with full colour graphics and sound. All these programs represent excellent value in themselves and will provide hours and hours of fun for all the family.
Many programming hints are included together with a chapter to help you improve your programming skills. A complete and entertaining sample game is included to illustrate structure in program-writing.

ISBN 0 948102 00 4
£2.95 net

AMSTRAD CPC

MÉMOIRE ÉCRITE
MEMORY ENGRAVED
MEMORIA ESCRITA

https://acpc.me/