The Virgin Computer Games Series

Series Editor: Tim Hartnell

GAMES FOR YOUR COMMODORE 64

£££££'s of entertaining games for only £2.95

Kieron Brennan

GAMES FOR YOUR COMMODORE 64

GAMES FOR YOUR COMMODORE 64

BY KIERON BRENNAN

Virgin Books

First published in Great Britain in 1983 by Virgin Books Ltd, 61-63 Portobello Road, London W11 3DD.

ISBN 0 86369 028 9

Printed and bound in Great Britain by Richard Clay (The Chaucer Press) Ltd, Suffolk.

Production services by Book Production Consultants, Cambridge.

Designed by Ray Hyden.

Illustrated by Sue Walliker.

Typeset by QV Typesetting.

Distributed by Arrow Books.

TIM HARTNELL — THE SERIES EDITOR

Tim Hartnell is a leading computer expert and journalist, who has contributed extensively to the Technical Consumer Press. He is also the author of several books including *Getting Acquainted With Your ZX81*, *Let Your BBC Micro Teach You to Program* and *Programming Your ZX Spectrum.*

KIERON BRENNAN — THE AUTHOR

Kieron Brennan is a 17-year-old schoolboy taking his 'A' levels at Ealing Green High School. He has been working with computers for two years; he started programming on a VIC 20, but after a year he upgraded to a Commodore 64. Since there was very little software available for the 64 when he first got it, he started to write his own programs; he likes to write games that are fast and colourful, though he is starting to write more strategic games. He has also created some games for Virgin Games and done some computer graphics for a film. In his spare time he is keen on football, tennis, squash, golf, karate, swimming and skiing.

SUE WALLIKER — THE ILLUSTRATOR

Sue Walliker is a freelance illustrator.

ACKNOWLEDGEMENTS

The author would like to thank his mum for putting up with him and his computer for the past couple of years, his dad for his crazy ideas, his brother, Darren, for testing the games, John Gardner for all-round help, the Maths Department at school for his education, and Martin Maguire for a couple of the programs contained within.

THIS BOOK IS DEDICATED TO MY EGO AND URGE FOR MONEY

CONTENTS

Editor's Introduction

Your computer is waiting to challenge you. Moving graphics games, brain stretchers, word games and puzzles are all here and ready to entertain you.

A wide variety of games are included in this book. The programs have been written by some of the most talented young programmers working in this country at the moment, and represent a variety of approaches to solving programming problems.

An examination of the listings should teach you many tricks and techniques to apply to your own programming. And once you have mastered the programs in their present form, you might want to try your hand at improving them. There is no such thing as a 'perfect program', so these games are sure to benefit from your programming skill.

All that now remains is for you to turn the page and enter the programs. I can only hope that you enjoy playing the games as much as we did when preparing this volume.

Tim Hartnell, series editor
London
March 1983

Author's Introduction

The Commodore 64 is one of the 'fun' computers because it can do so many different things; each time you sit down to write a program you discover something that you didn't know before. As a result of this high level of versatility, the 64 will inspire new and totally original pieces of software.

I hope that my book may perhaps encourage you to write some of your own games programs. At any rate, I hope that you have as much fun typing in these games and playing them as I had writing them. Good luck!

Kieron Brennan
January 1984

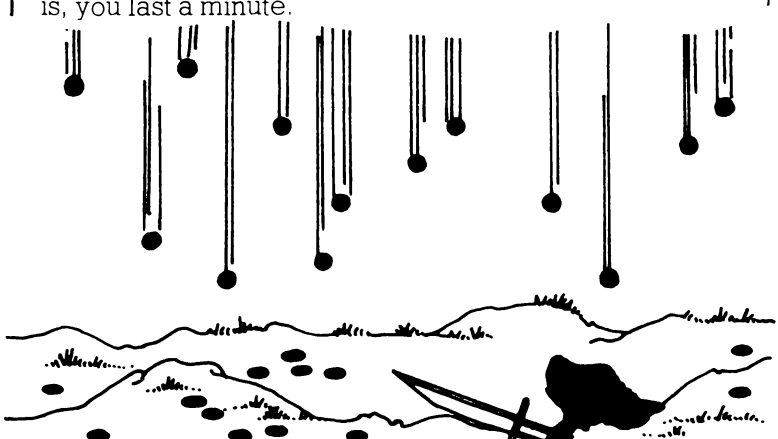
PENNIES FROM HEAVEN

You must think it's your lucky day! All this money falling from the sky — or could it be a hurricane? But the local mole finds it very annoying and is trying to stab you in the hope of stopping the thump of money falling on the ground.

Using his keen sense of direction he follows you back and forth across the screen, occasionally jumping to try and stab you. He is determined to get you, although fortunately he is not too accurate.

Whilst the mole is stabbing away, the coins keep falling. You must catch them in your hands to gain any points, but watch out, they are deadly. If one hits you on the head it will kill you; you have to keep one eye on the mole and the other on the money.

Try to collect as much as you can in one minute — if, that is, you last a minute.



```
1 M$="y u i Qaaaaaw ymjknw yQaaaaaw ypow
Qaaaaa ynmw Qaaaaa yR>r. R<rw S"
2 DIMP(40,1)
3 C$="SQQQQQQQQQQQQQQQQ"
4 L$="]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]]"
5 S=0:M=20:POKE650,255
10 GOSUB10000:GOSUB10100:GOSUB10050:GOSU
B10060
12 POKE53281,11:POKE53280,11:POKEV+39,0:
PRINT"s";:
13 S=0:M=20
20 FORA=1TO19:PRINT"
                          ::NEXT
25 TI$="000000":I=1
30 I=I+1:IFI=4THEN150
31 GETA$:IFA$=""THEN100
32 IFA$="Z"THEN36
33 IFA$="M"THEN46
34 GOTO100
36 S=S-1:IFS<0THENS=0
37 GOTO100
40 POKEV,P(M,0):PRINTC$;"QQQQ":SPC(M);"
   Qaaaa     Qaaaa     Qaaaa     S"
41 POKEV+16,P(M,1):IFPEEK(1705+S)=81ORPE
EK(1706+S)=81THEN200
42 IFPEEK(1744+S)=81ANDPEEK(1747+S)=81TH
EN250
43 IFPEEK(1704+S)=81THEN255
44 IFPEEK(1707+S)=81THEN260
45 PRINTC$;LEFT$(L$,S);M$;:GOTO30
46 S=S+1:IFS>36THENS=36
47 GOTO100
48 IFF=1THENSYS12544
49 GOTO103
100 R=R+1:IFTI>3600THEN350
101 IFR/2=INT(R/2)THEN110
102 SYS12544:GOTO40
103 NEXT
110 POKE1024+INT(40*RND(1)),81
115 SYS12544:GOTO40
```

```
150 I=1:Z=Z+1
151 IFZ>4THENZ=0:GOTO160
152 IFM<STHENM=M+1:GOTO155
153 IFM>STHENM=M-1:GOTO155
155 GOTO30
160 POKEV+1,221
162 POKEV+31,0:FORD=1TO100:NEXT
163 IFPEEK(V+31)=1THEN210
164 POKEV+1,229:Z=1:GOTO31
200 PRINT"SOYOU HAVE BEEN HIT BY A COIN
AND HAVE"
201 PRINT"LOST THE GAME."
202 PRINT"QQBUT BEFORE YOU DIED YOU MANN
AGED TO GET"
203 PRINTSC;"GOLD COINS
204 GOTO300
210 PRINT"SzYOU HAVE BEEN STABBED BY THE
 LUNATIC "
211 PRINT"MOLE "
212 GOTO300
250 SC=SC+2:POKE1744+S,77:POKE1747+S,78:
GOTO31
255 SC=SC+1:POKE1744+S,77:GOTO31
260 SC=SC+1:POKE1747+S,78:GOTO31
300 PRINT"QQDO YOU WANT ANOTHER GAME (Y/
N)?"
301 GETA$:IFA$=""THEN301
302 IFA$="Y"THENSC=0:GOTO12
303 IFA$="N"THENEND
304 GOTO301
350 PRINT"SпYOUR TIME HAS RUN OUT AND YO
U HAVE "
351 PRINT"COLLECTED";SC;"COINS"
352 GOTO300
9999 END
10000 PRINT"s   PENNIES FROM HEAVEN"
10001 PRINT"QTHE IDEA IS TO COLLECT ASMA
NY OF THE"
10002 PRINT"PENNIES THAT ARE FALLING FRO
M HEAVEN"
10003 PRINT"IN 1 MINUTE"
```

```
10004 PRINT"QBUT BE CAREFUL BECAUSE THER
E IS A MOLE"
10005 PRINT"UNDERNEATH THAT IS TRYING TO
 STAB YOU"
10006 PRINT"QQ  Z-MOVES YOU LEFT.  M-MOV
ES YOU RIGHT"
10007 PRINT"QQPRESS ANY KEY TO START"
10008 POKE198,0:WAIT198,1:POKE198,0:RETU
RN
10050 FORA=0TO62:READB:POKE832+A,B:NEXT:
V=53248:POKEV+21,1
10051 POKEV,160:POKEV+1,229:POKE2040,13:
RETURN
10060 FORA=0TO37:READP(A,0),P(A,1):NEXT:
RETURN
10100 FORA=0TO41:READB:POKE12544+A,B:NEX
T:RETURN
10200 DATA 169,96,133,87,169,7,133,88,16
0,0,177,87,201,81,240,13
10201 DATA 198,87,208,244,198,88,166,88,
224,3,208,236,96,169,32,145
10202 DATA 87,160,40,169,81,145,87,76,16
,49
10205 DATA 0,16,0,0,56,0,0,84,0,0,146,0,
0,16,0,24,56,48,6,124,192,1,255,0
10206 DATA 126,254,248,7,255,192,59,17,1
84,3,17,128,15,255,224,23,255,208
10207 DATA 39,255,200,39,255,200,35,255,
136,17,255,16,12,124,96,3,131,128
10208 DATA 0,124,0
10210 DATA 25,0,33,0,41,0,49,0,57,0,65,0
,73,0,81,0,89,0,97,0,105,0,113,0
10211 DATA 121,0,129,0,137,0,145,0,153,0
,161,0,169,0,177,0,185,0,193,0
10212 DATA 201,0,209,0,217,0,225,0,233,0
,241,0,249,0,2,1,10,1,18,1,26,1
10213 DATA 34,1,42,1,50,1,58,1,66,1
```

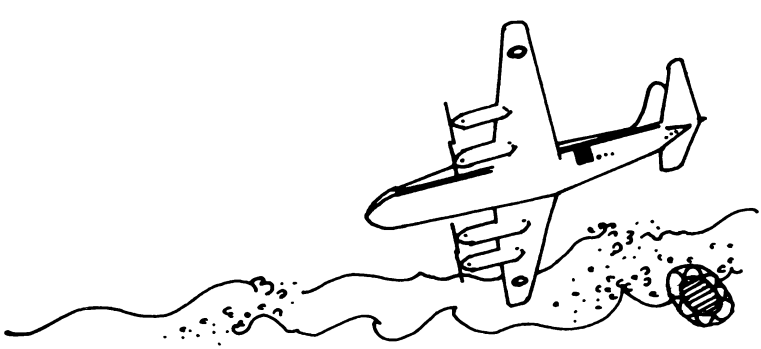
PARATROOPER

You are flying above the ocean, and down in the depths of the water is a small survival raft. You have to jump out of the plane and, guiding yourself against the wind, land on the raft. If you do so you will be praised; if you fail you will be disgraced. Apart from the wind there is one other thing that might be tricky to handle; because the raft has been in the water for a long time it has become rather slippery, so you are liable at any time to fall off.

The game is played off the keyboard. You push the space bar to jump, 'Z' to move left, and 'M' to move right.

The flagpole on the right of the screen shows you the way the wind is blowing; you can't manoeuvre against the wind very easily, and you may be carried off course if you turn with the wind.

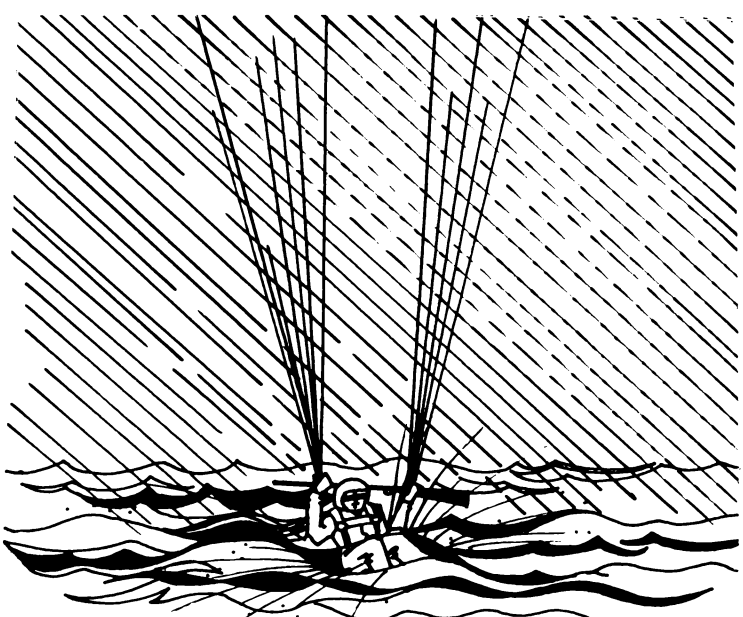
This game uses sprites, and can be made more or less difficult by changing the size of the raft (R$). You have to be wary of the edge of the screen, because if you cross over the edge you will appear on the other side which might impair your judgement. The speed of the raft can also be changed, but I will let you find that out for yourself!



```
1 GOSUB1000
2 RESTORE:GOSUB100
3 D$="SQQQQQQQQQQQQQQQQQQQQQQQQ":L$="]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]"
4 R$=" R↑          r S":PRINT"s"
5 R=1:POKEV+39,0
10 POKEV+32,14:POKEV+33,3:D=1:W=INT(2*RN
D(1)+1)
15 PRINT"SQQQQQQQQQQQQQQQQQQQQQQQQ]]]]]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]";
16 A$="R rq◣":FORA=1TO7:PRINT"p";A$;"£";
A$;:NEXT
18 POKE2040,13:POKEV,33:POKEV+1,45
19 POKE198,0
21 IFW=1THENW=-2:GOSUB200
22 IFW=2THENW=2:PRINT"11";:GOSUB200
25 Q=3:P=.5
26 C=PEEK(V)+Q:IFC>255ORC<33THENQ=-Q:C=P
EEK(V)
27 POKEV,C
28 GETB$:IFB$=""THEN30
29 IFB$=" "THEN40
30 R=R+P:IFR>280ORR<1THENP=-P:R=R+2*P
31 PRINTD$;LEFT$(L$,R);R$:GOTO26
40 POKEV+23,0:POKEV+29,0:POKE2040,14:L=4
5
41 L=L+1:IFL>206THEN150
42 POKEV+1,L
43 GETY$:IFY$=""THEN46
44 IFY$="Z"THEN60
45 IFY$="M"THEN70
46 Z=PEEK(V):Z=Z+W:IFZ<33THENZ=255
47 IFZ>255THENZ=33
49 POKEV,Z
50 R=R+P:IFR>280ORR<1THENP=-P:R=R+2*P
51 PRINTD$;LEFT$(L$,R);R$:GOTO41
60 Z=PEEK(V):Z=Z-4:IFZ<33THENZ=250
61 POKEV,Z:GOTO46
70 Z=PEEK(V):Z=Z+4:IFZ>255THENZ=35
71 POKEV,Z:GOTO46
99 END
```

```
100 POKE2040,13
102 FORA=0TO62:READB:POKE832+A,B:NEXT
104 V=53248:POKEV+21,1:POKEV+23,1:POKEV+
29,1
108 FORA=0TO62:READB:POKE896+A,B:NEXT
110 RETURN
112 DATA0,0,0,0,0,0,0,0,7,0,0,9,0,0,17,0
,0,33,7,255,221,15,128,1,31,31,129
114 DATA32,63,193,224,31,129,31,128,6,0,
127,248,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
116 DATA0,0,0,0,0,0,0,0
118 DATA0,56,0,1,255,0,15,255,224,31,255
,240,31,255,240,31,255,240,2,0,128
120 DATA2,24,128,2,60,128,2,60,128,2,24,
128,3,255,128,0,60,0,0,60,0,0,60,0,0
122 DATA24,0,0,36,0,0,66,0,0,129,0,0,129
,0,1,129,128,0,0
150 N=PEEK(V+31):IFPEEK(V+31)=0THEN160
151 PRINT"S£ WELL DONE YOU HAVE COMPLET
ED IT"
152 PRINT"TRY AGAIN":FORK=1TO3000:NEXT:G
OTO2
160 IFPEEK(V+31)<>0THEN151
161 PRINT"S←YOU HAVE FAILED TO LAND ON T
HE RAFT"
162 PRINT"YOU HAVE DIED BUT TRY AGAIN"
163 FORK=1TO3000:NEXT:GOTO2
200 PRINT"▲R▲←  9▲▲▲  r.":RETURN
1000 PRINT"sINSTRUCTIONS"
1001 PRINT"pYOU HAVE TO JUMP OUT OF YOUR
 PLANE AND "
1002 PRINT"TRY AND LAND ON THE RAFT BELO
W"
1003 PRINT"TO MAKE IT HARDER FOR YOU THE
 RAFT IS"
1004 PRINT"MOVING AND THERE IS SOME WIND
 ABOUT"
1005 PRINT"THE WIND DIRECTION IS INDICAT
ED BY THE"
1006 PRINT"FLAG. BE CAREFUL, YOU HAVE TO
 LAND ON "
```

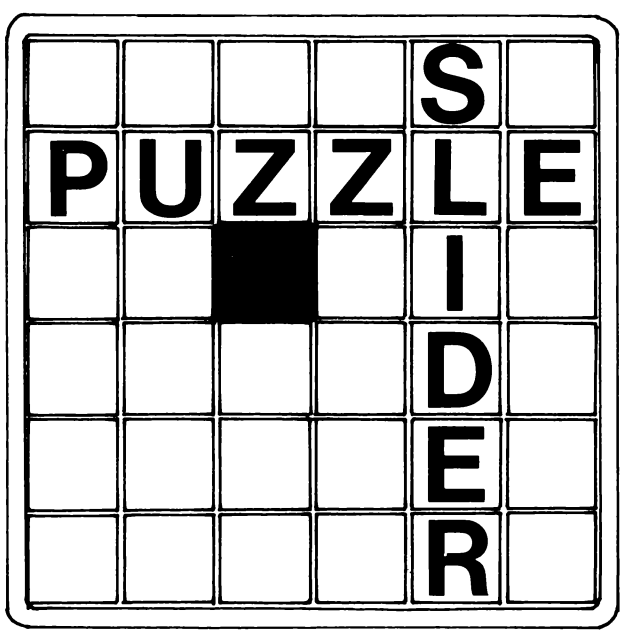
```
1007 PRINT"THE MIDDLE OF THE RAFT"
1008 PRINT"SPACE BAR RELEASES YOU FROM T
HE PLANE"
1009 PRINT"Z-MOVES YOU LEFT, M-MOVES YOU
 RIGHT"
1010 PRINT"QQPRESS ANY KEY TO START"
1011 POKE198,0:WAIT198,1
1012 PRINT"s":RETURN
```



SLIDER

This is based on the hand puzzle game in which you try and rearrange the numbers or letters, and sometimes a picture.

It is a simple program that shows how STRINGS can be used to set up a simple picture. At the start the picture is totally muddled up; you have to rearrange the pieces to reveal the picture.




```
1 POKE53281,1
2 GOSUB600
5 DIMA$(6,7)
10 FORA=5TO1STEP-1:FORB=6TO1STEP-1
20 READA$(A,B):NEXT:NEXT
25 PRINT"s"
```

```
30 PRINT"S←QQQQQ";:FORA=1TO5:PRINT"
      R▲";A"▲▲▲r RQ rq";:FORB=1TO6
35 PRINTA$(A,B);"q";:NEXT:PRINT"R Q▲ r":
NEXT
37 PRINT"SQQQQ]]]]]]]]]]]]Rm 1 2 3 4 5 6
nr"
38 PRINT"SQQQ]QQQQQQQQQQ]]]]]]]]]]R Q^
n            m▲q r"
40 PRINT"SQQQQQQQQQQQQQQQQQQQQR£
                                  r"
45 PRINT"WHICH PIECE TO MOVE E.G '1,1'"
46 INPUT"          ▲▲▲▲▲▲▲▲▲▲";B,A
47 IFB<1ORB>6THEN40
48 IFA<1ORA>5THEN40
50 IFA$(A,B)="R  Q▲▲  r"THEN40
51 IFA$(A-1,B)="R  Q▲▲  r"THEN100
52 IFA$(A+1,B)="R  Q▲▲  r"THEN200
53 IFA$(A,B-1)="R  Q▲▲  r"THEN300
54 IFA$(A,B+1)="R  Q▲▲  r"THEN400
55 GOTO40
100 A$(A-1,B)=A$(A,B):A$(A,B)="R  Q▲▲  r
"
110 GOTO500
200 A$(A+1,B)=A$(A,B):A$(A,B)="R  Q▲▲  r
"
210 GOTO500
300 A$(A,B-1)=A$(A,B):A$(A,B)="R  Q▲▲  r
"
310 GOTO500
400 A$(A,B+1)=A$(A,B):A$(A,B)="R  Q▲▲  r
"
410 GOTO500
500 Z=0:RESTORE:FORA=1TO5:FORB=1TO6
510 READT$:IFA$(A,B)<>T$THENZ=1
520 NEXT:NEXT
530 IFZ=1THEN30
540 PRINT"S↑  CONGRATULATIONS YOU HAVE C
OMPLETED IT"
545 PRINT"S←QQQQQ";:FORA=1TO5:PRINT"
      R▲";A"▲▲▲r RQ rq ";:FORB=1TO6
555 PRINTA$(A,B);"q";:NEXT:PRINT"R Q▲ r"
```

```
:NEXT
560 PRINT"SQQQQ]]]]]]]]]]]]]Rm  1 2 3 4 5
 6nr"
565 PRINT"SQQQQQQQQQQQQQQQQ]]]]]]]]]]]]]R
Qan                 maq r"
570 PRINT"ANOTHER GO ?"
571 GETD$:IFD$=""THEN571
572 IFD$="Y"THENRUN
573 IFD$="N"THENEND
574 GOTO571
600 PRINT"sQw  YOU HAVE TO TRY AND UN-SC
RAMBLE THE "
601 PRINT"  BLOCKS OF THE PICTURE AND HO
PEFULLY "
602 PRINT"          REVEAL THE HIDDEN SHAP
E"
603 PRINT"          THE GRID IS AS FOLLOWS
."
604 PRINT"Q      1 2 3 4 5 6  (X VALUES)"
610 FORA=1TO5:PRINT"    *77o7o7o7o7o74"
611 PRINTA;"*//l/l/l/l/l/4"
612 NEXT
615 PRINT"Q(Y VALUES)"
620 PRINT"QTHE PIECE IS ENTERED AS X,Y
625 PRINT"QPRESS ANY KEY TO START GAME"
630 GETP$:IFP$=""THEN630
640 GOTO5
1000 DATA"R  Qaa  r","  Qaa w","ui Qaa]k"
,"  Qaa R>r","  QaaR<r  ","  Qaaw!"
1001 DATA"  Qaa n","//Qaa1  ","//Qaa2  ","
/lQaa3  ",": Qaa4m","jkQaa  "
1002 DATA"n5Qaa07","  6Qaa77","  7Qaa77","
 8Qaa77","  9Qaa77","m Qaap  "
1003 DATA"4oQaa4l","p Qaa:  ","o7Qaa4&","
p Qaa*  ","vvQaavv","* Qaa*  "
1004 DATA"4 Qaal/","  Qaa//","4"Qaal/","
* Qaa:/","vvQaa//","* Qaa:  "
```

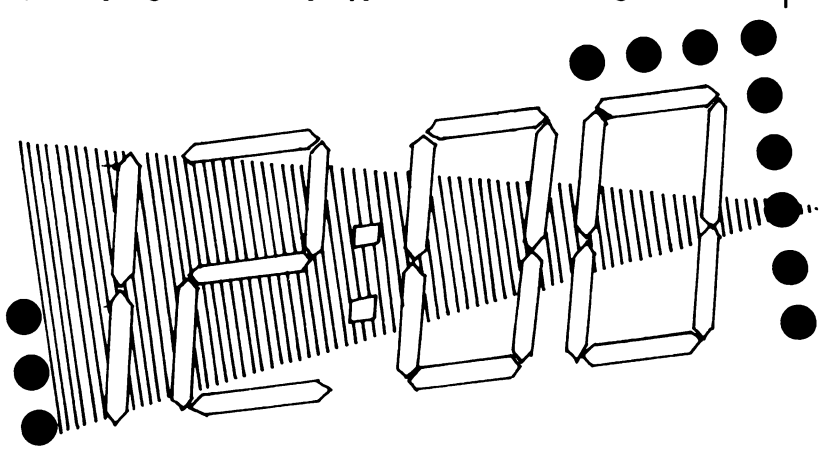
THE BIG CLOCK

This is not actually a game, but it shows a way of using the built-in clock in your Commodore 64. It also demonstrates the use of some of the STRING functions, as well as a little bit on ARRAYS.

The large numbers can easily be changed in accordance with your wishes by adjusting the DATA statements from line 1000 onwards, where line 1000 is the digit '0', 1001 '1'...1009 '9'. Sound effects can also be added to make a beep every time the clock changes.

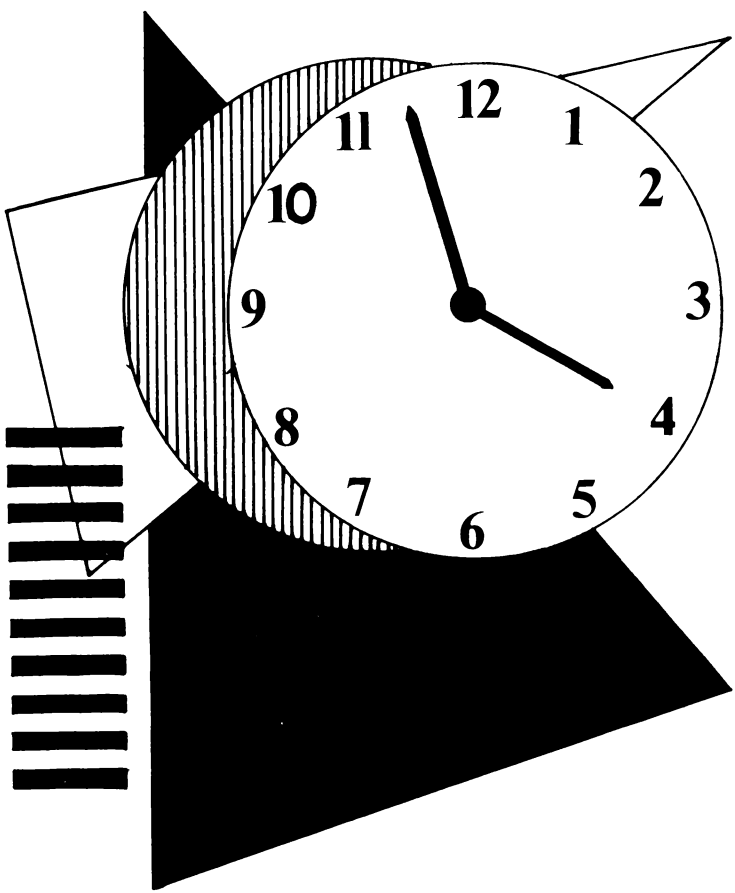
In the top left of the screen the normal clock is shown, and it is possible to see the relationship between 'TI$' and 'THE BIG CLOCK'

A problem may occur when you input the starting time. Take care to input it as a six-digit number, and do not leave out the first '0' if there is one. If you make a mistake the program will stop; type 'RUN' to start it again.



```
10 DIMA$(10)
20 FORA=0TO9:READA$(A):NEXT
30 PRINT"SET TIME PLEASE (011030=1HR 10M
IN 30SEC)"
40 INPUTTI$
50 PRINT"s←":POKE53281,3:POKE53280,1
100 T$=TI$
101 PRINT"S"TI$
110 IFT$<>TI$THEN150
120 GOTO110
150 S1=VAL(RIGHT$(TI$,1)):S2=VAL(MID$(TI
$,5,1))
155 M1=VAL(MID$(TI$,4,1)):M2=VAL(MID$(TI
$,3,1))
160 H1=VAL(MID$(TI$,2,1)):H2=VAL(MID$(TI
$,1,1))
165 IFH2=0THEN175
170 PRINT"SQQQ]]]"A$(H2);"qqqq";
175 PRINT"S]]]]]]]]]]QQQ"A$(H1);"qq^qqqq"
;A$(M2);"qqqq";A$(M1);"qq^qqqq";
180 PRINTA$(S2);"qqqq";A$(S1)
190 GOTO100
1000 DATA"u——i Q^^^^^^^   ^Q^^^^^^^   ^Q^^
^^^^   ^Q^^^^^^j——k"
1001 DATA"  u.  Q^^^^^^  ^  Q^^^^^^  ^  Q^^
^^^  ^  Q^^^^^^  —1— "
1002 DATA"u——i Q^^^^^^   ^Q^^^^^^u——kQ^^
^^^^   Q^^^^^^j——k"
1003 DATA"u——i Q^^^^^^   ^Q^^^^^^ +——3Q^^
^^^   ^Q^^^^^^j——k"
1004 DATA"0   Q^^^^^^^  0 Q^^^^^^-——|—Q^^
^^^  ^ Q^^^^^^   ^"
1005 DATA"u——i Q^^^^^^^   Q^^^^^^j——i Q^^
^^^   ^Q^^^^^^j——k"
1006 DATA"u——i Q^^^^^^^   Q^^^^^^+——i Q^^
^^^^   ^Q^^^^^^j——k"
1007 DATA"u——i Q^^^^^^   ^Q^^^^^^   ^Q^^
^^^   ^Q^^^^^^   ^"
1008 DATA"u——i Q^^^^^^^   ^Q^^^^^^+——3Q^^
```

```
1009 DATA"u                                    3
```



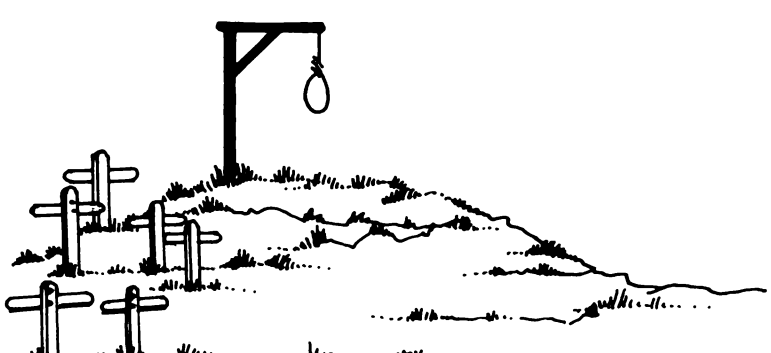
HANGMAN

In this game you have to try and guess the word that the computer has chosen at random. The only way you can save the innocent man from being hung is by guessing the hidden word.

The game allows you eight wrong guesses; on the next wrong guess the game is over. You are not penalised for trying a letter that has previously been used, but the computer will remind you that the letter has been used already. It also checks that you do not type in either numbers or non-alphabetic characters. These characters are stored in a STRING called 'G$', which is used for storing all the guesses.

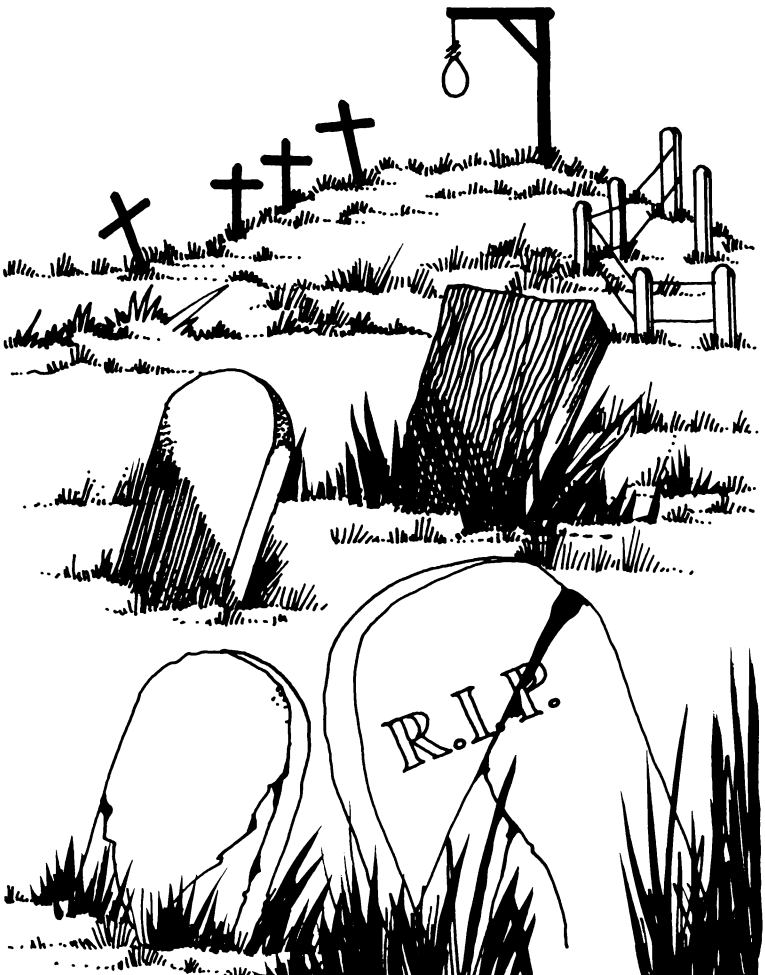
When you guess the word correctly, the computer will acknowledge your achievement, and then go onto another word if you wish. The vocabulary is read in at the start and has 50 words in it. The number of words can be changed by adding more lines of DATA after the block of words. You also have to change the line which says 'DIM W$ (50)', where '50' is the number of words in the vocabulary.

It would be a good idea to SAVE the program after you have added more words of your own.



```
5 DIMW$(50):POKE53281,3
10 GOSUB10000
20 FORA=1TO7:READA$(A):NEXT:READB$,C$:A$
(8)=B$+C$
30 FORA=1TO50:READW$(A):NEXT
31 D$="SPQQQQQQQQQQQQQQQQQQQQQ]]]"
32 L$="]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]"
100 W=INT(50*RND(1)+1)
101 G$=" 1234567890←+-£↑*@=;:/..Sefghijk
lJQ"
102 PRINTD$;"QQ]]]]£ABCDEFGHIJKLMNOPQRST
UVWXYZ"
105 FORA=1TOLEN(W$(W)):PRINTD$;LEFT$(L$,
(A*2));"/":NEXT
107 PRINTD$;"QQQNOW TYPE YOUR GUESSS"
110 REMPOKE198,0
120 GETB$:IFB$=""THEN110
125 FORV=1TOLEN(G$):IFB$=MID$(G$,V,1)THE
NX=1
130 NEXT
135 IFX=1THENPRINTD$;"QQQ  ALREADY SELE
CTEDS":FORL=1TO1000:NEXT:X=0:GOTO107
140 G$=G$+" "+B$
145 J=0:FORL=1TOLEN(W$(W))
150 IFB$=MID$(W$(W),L,1)THENK=K+1:PRINTD
$;LEFT$(L$,L*2)B$:J=1
160 NEXT
162 IFJ=0THENN=N+1
163 IFN=9THEN250
164 PRINTA$(N)
165 IFK=LEN(W$(W))THEN180
170 PRINT"S";LEFT$(L$,5);RIGHT$(G$,LEN(G
$)-30)
175 PRINTD$;"QQQ
          S":FORE=1TO1000:NEXT:GOTO107
180 PRINT"SCORRECT. THE WORD WAS "W$(W)
181 PRINT"BUT NOW TRY ANOTHER WORD."
182 PRINT"PRESS ANY KEY TO CARRY ON"
185 POKE198,0:WAIT198,1:RUN
250 PRINT"SHARD LUCK. YOU HAVE BEEN HUNG
```

```
251 PRINT"THE WORD WAS "W$(W)
252 PRINT"BUT NOW TRY ANOTHER WORD.
253 PRINT"PRESS ANY KEY TO CARRY ON"
254 POKE198,0:WAIT198,1:RUN
999 END
1001 DATA"SQQQQQQQQQQ]]]QQQQQQQQQR.
   Q▲▲▲▲▲▲▲▲▲▲▲▲                  r"
1002 DATA"SQQQ]]'Q▲'Q▲'Q▲'Q▲'Q▲'Q▲'Q▲'Q▲'Q^
'Q▲'Q▲'Q▲'Q▲'Q▲'Q▲'Q▲'Q▲"
1003 DATA"SQQ]]]/////////"
1004 DATA"SQQQQQ]]]nqnqn"
1005 DATA"SQQQQQQQQQQQQQQQ]]]mQmQm"
1006 DATA"SQQQ]]]]]]]]]]]▲Q▲▲u1!Q▲▲▲▲ ▲Q^
▲▲J2k"
1007 DATA"SQQQQQQQ]]]]]]]]u—|—|Q▲▲▲▲▲▲ ^
 ▲Q▲▲▲▲▲▲ ▵ ▲Q▲▲▲▲▲+ ▵ 3"
1008 DATA"SQQQQQQQQQQQ]]]]]]]]u—1—|Q▲▲▲^
▲▲   ▲Q▲▲▲▲▲▲   ▲Q▲▲▲▲▲▲   ▲Q▲▲▲▲▲▲"
1009 DATA"<>    <>"
1010 DATANYMPH,HYPERBOLA,MATHEMATICS,ENG
LISH,RIVER,YAUGHT,PENCIL,GNOME
1011 DATATELEVISION,PRINTER,COMPUTER,GAM
ES,PAPER,PROGRAM,WRITING,MONEY
1012 DATAMUSIC,MIRROR,PLANE,BUTTON,PLUG,
POSTER,FOLDER,SQUARE,QUEEN
1013 DATACLOCK,SHOE,LACES,LIGHT,NIGHT,DA
Y,MAY,MANUAL,RECORDER,COIN,JUMPER
1014 DATAKEY,SHELF,BOX,FOX,TAPE,ROPE,HOL
DER,BATH,MAT,SOAP,FLASH,SHEET,PENNY
1015 DATABUGGY
10000 PRINT"S←INSTRUCTIONS"
10010 PRINT"THIS IS A GAME OF HANGMAN, A
ND YOU HAVE"
10011 PRINT"TO GUESS THE HIDDEN WORD ONE
 LETTER AT "
10012 PRINT"A TIME. GOOD LUCK.
10013 PRINT"QQPRESS SPACE TO BEGIN"
10014 GETF$:IFF$<>" "THEN10014
10015 PRINT"S";:RETURN
```

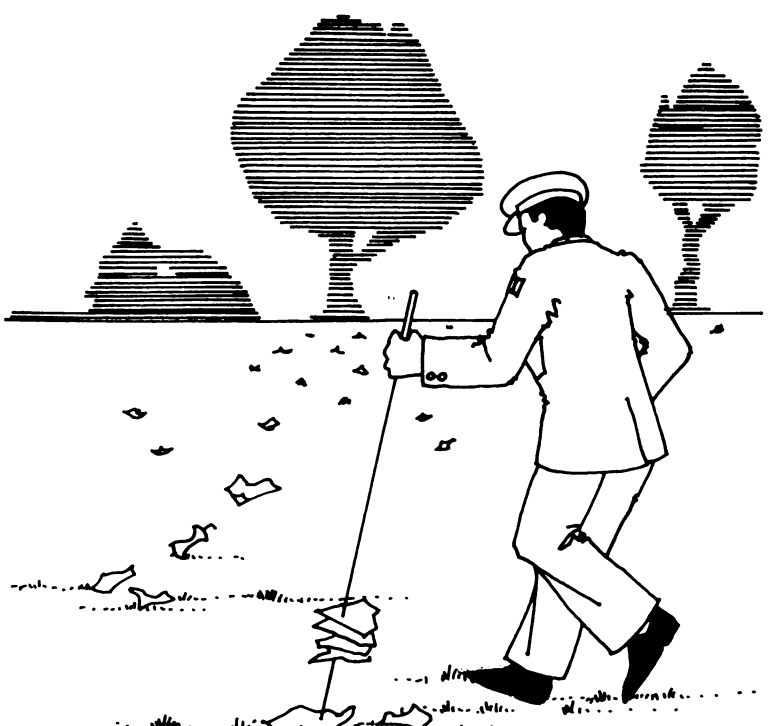

R.I.P.

PARK-KEEPER

In this game you are a park-keeper, constantly plagued by a hooligan who keeps dropping litter all over the park. You can't actually stop him from doing this because he might mug you. However, if you are very quick you can slip by him, though he may jump on you from behind.

Rather than having to clear all the litter, you have a time limit. This limit can easily be adjusted in line 51. Remember that the value shown is sixtieths of a second. The keys used to control the man are 'K' to move up, 'M' to move down, 'A' to move left and 'S' to move right.

A high-score feature could easily be added, and would be virtually the same as in some of the other games.



```
1 S=0:P1=1065:P2=1638:POKE650,255
2 POKE53281,15:POKE53280,12:PRINT"←"
3 D1(0)=1:D1(1)=-1:D1(2)=-40:D1(3)=40
4 SO=54272:POKESO+24,15:POKESO,0:POKESO+
1,0:POKESO+5,0:POKESO+6,240
10 GOSUB1000
20 PRINT"s←";
21 PRINT"&&&&&&&&&&&&&&&&&&&&&&&&&&&&"
22 PRINT"& x x x x x x x x x x x x x &"
23 PRINT"&x&&&&&&&&&&&& &&&&&&&&&&&&x&"
24 PRINT"& & x x x x x x x x x x x & &"
25 PRINT"&x&x&&&&&&&&& &&&&&&&&&&x&x&"
26 PRINT"& & & x x x x x x x x x & & &"
27 PRINT"&x&x&x&&&&&&& &&&&&&&&x&x&x&"
28 PRINT"& & & & x x x x x x x & & & &"
29 PRINT"&x&x&x&x&&&&& &&&&&&x&x&x&x&"
30 PRINT"& & & & & x x x x x & & & & &"
31 PRINT"&x&x&x&x& &&&&&&&&&x&x&x&x&x&"
32 PRINT"& & & & & &&&&K&&&& & & & & &"
33 PRINT"&x x x x x&&&&A+S&&&x x x x x&"
34 PRINT"& & & & & &&&&M&&&& & & & & &"
35 PRINT"&x&x&x&x&x&&&&&&&&&&x&x&x&x&x&"
36 PRINT"& & & & & x x x x x & & & & &"
37 PRINT"&x&x&x&x&&&&& &&&&&&&x&x&x&x&"
38 PRINT"& & & & x x x x x x x & & & &"
39 PRINT"&x&x&x&&&&&&& &&&&&&&&&x&x&x&"
40 PRINT"& & & x x x x x x x x x & & &"
41 PRINT"&x&x&&&&&&&&& &&&&&&&&&&x&x&"
42 PRINT"& & x x x x x x x x x x x & &"
43 PRINT"&x&&&&&&&&&&& &&&&&&&&&&&&x&"
44 PRINT"& x x x x x x x x x x x x x &"
45 PRINT"&&&&&&&&&&&&&&&&&&&&&&&&&&S"
:TI$="000000"
46 PRINT"SQ]]]]]]]]]]]]]]]]]]]]]]]]]
]]SCOREQ^^^^^^"S
50 POKEP1+54272,1:POKEP1,32:POKEP1+D,81:
POKEP1+54272,0:P1=P1+D:D=0
51 IFTI>3600THEN400
52 GETA$
53 IFA$="K"THEN150
54 IFA$="M"THEN200
```

```
55 IFA$="A"THEN250
56 IFA$="S"THEN300
100 TS=D2
101 R%=RND(1)*3:D2=D1(R%)
110 IFPEEK(P2+D2)=88ORPEEK(P2+D2)=32THEN
114
112 IFPEEK(P2+D2)=102THEN100
113 IFPEEK(P2+D2)=81THEN450
114 IFTS=-D2THEN101
115 POKEP2+54272,0:POKEP2,88:P2=P2+D2:PO
KEP2+54272,11:POKEP2,83
120 GOTO46
150 D=-40:IFPEEK(P1+D)=32THEN100
155 IFPEEK(P1+D)=102THEND=0:GOTO100
160 IFPEEK(P1+D)=88THENS=S+5:GOTO350
165 IFPEEK(P1+D)=83THEN450
200 D=40:IFPEEK(P1+D)=32THEN100
205 IFPEEK(P1+D)=102THEND=0:GOTO100
210 IFPEEK(P1+D)=88THENS=S+5:GOTO350
215 IFPEEK(P1+D)=83THEN450
250 D=-1:IFPEEK(P1+D)=32THEN100
255 IFPEEK(P1+D)=102THEND=0:GOTO100
260 IFPEEK(P1+D)=88THENS=S+5:GOTO350
265 IFPEEK(P1+D)=83THEN450
300 D=1:IFPEEK(P1+D)=32THEN100
305 IFPEEK(P1+D)=102THEND=0:GOTO100
310 IFPEEK(P1+D)=88THENS=S+5:GOTO350
315 IFPEEK(P1+D)=83THEN450
350 POKESO,30:POKESO+1,30:POKESO+4,33:FO
RI=1TO10:NEXT:POKESO+4,0:GOTO100
400 PRINT"SYOUR TIME IS UP AND YOU SCORE
D"S
405 PRINT"ANOTHER GAME (Y/N) ?"
410 GETF$:IFF$=""THEN410
415 IFF$="Y"THENS=0:TI$="000000":P1=1065
:P2=1638:GOTO20
420 IFF$="N"THENEND
425 GOTO410
450 PRINT"pSYOU HAVE BEEN MUGGED BY THE
LOUT AND "
460 PRINT"SCORE "S"POINTS.":GOTO405
```

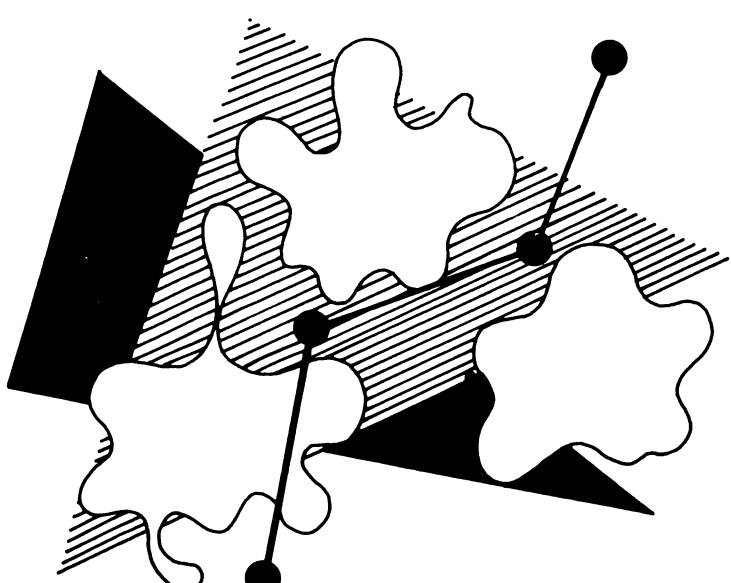
```
1000 PRINT"sp YOU ARE THE CIRCLE AND WILL
 START AT THE"
1001 PRINT"TOP LEFT OF THE MAZE TO MOVE
USE THE    KEYS AS SHOWN IN THE"
1002 PRINT"CENTRE OF THE PARK. YOU ONLY
HAVE ONE "
1003 PRINT"MINUTE TO COLLECT AS MUCH RUB
BISH AS"
1004 PRINT"POSSIBLE"
1005 PRINT"PRESS ANY KEY TO START"
1010 GETF$:IFF$=""THEN1010
1020 RETURN
```

BLOB SQUASHER

In this game you have to try and run over as many blobs in as little time as possible.

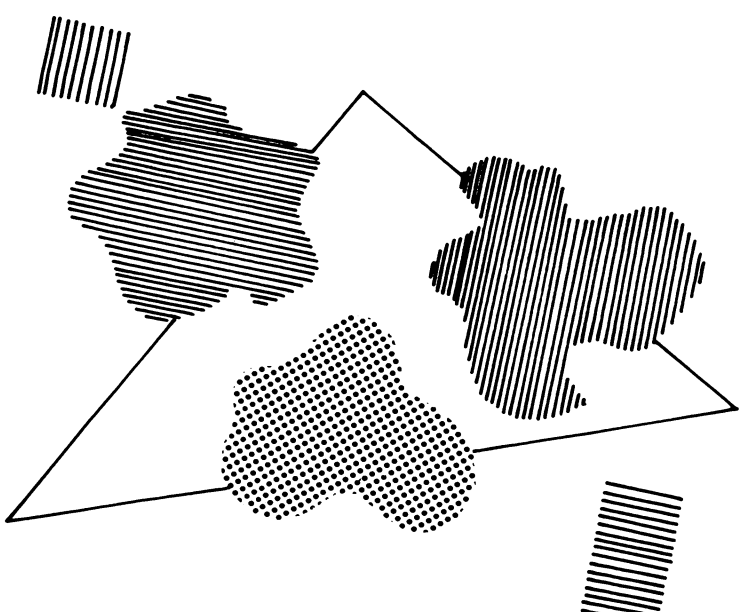
At the start, you are given a choice of which level you would like to play on. The smaller the number, the harder the game. If you have opted for a level, only to find that you cannot clear all the blobs, they will start to appear more slowly so you have a better chance of clearing the screen.

This program is run off the joystick and demonstrates a very efficient way of moving something around the screen quite quickly.



```
1 POKE54296,15:POKE54277,0:POKE54278,240
10 GOSUB1000
15 POKE53281,1
16 INPUT"INPUT LEVEL (1 - 5) 1=HARD";L
17 IFL<1ORL>5THEN16
20 PRINT"s";:FORA=1TO24:PRINT"£
                                    ";:NEXT
30 PRINT"
        S";
31 POKE56295,2
35 P=1522:D=1:TI$="000000"
40 FORA=0TO3:READDI(A):NEXT:C=0:M=0
43 J=PEEK(56320)
45 IFJ=127THEN100
46 IFJ=126THEND=0:GOTO100
47 IFJ=125THEND=2:GOTO100
48 IFJ=123THEND=3:GOTO100
49 IFJ=119THEND=1:GOTO100
100 M=M+1
101 P=P+DI(D):IFP>2023ORP<1024THENP=P-DI
(D)
103 IFPEEK(P)=42THENPOKEP,32:GOTO250
104 POKEP-DI(D),32:POKEP,81
105 IFM=(L*9)THEN200
106 GOTO43
200 M=0:Z=INT(999*RND(1))
202 IFPEEK(1024+Z)<>32THEN200
205 FORS=255TO0STEP-40:POKE54273,S:POKE5
4276,33:NEXT:POKE54273,0:POKE54276,0
210 POKE1024+Z,42:C=C+1:IFC>15THENL=L+1
220 GOTO43
250 C=C-1:IFC<1THEN350
255 FORS=0TO255STEP30:POKE54273,S:POKE54
276,17:NEXT:POKE54276,0:POKE54273,0
256 GOTO103
260 GOTO103
350 PRINT"sWELL DONE YOU HAVE CLEARED AL
L THE BLOBS"
351 PRINT"IT TOOK YOU ";TI/60;" SECONDS"
355 PRINT"QDO YOU WANT ANOTHER GO (Y/N)
356 GETA$:IFA$=""THEN356
```

```
357 IFA$="Y"THENRESTORE:GOTO16
358 IFA$="N"THEN END
359 GOTO356
999 END
1000 PRINT"{CLR}INSTRUCTIONS"
1001 PRINT"{DOWN}USING THE JOYSTICK YOU MUST
WATCH OUT   FOR INVADING BLOBS."
1002 PRINT"THESE BLOBS MAY APPEAR ANYWHE
RE ON THE  SCREEN, AND YOU MUST RUN THEM

1003 PRINT"OVER TO RID THE WORLD OF THES
E TERRIBLE CREATURES"
1008 PRINT"{DOWN}THE JOYSTICK MUST BE IN PORT
 2"
1009 PRINT"{DOWN}{DOWN}PRESS ANY KEY TO START"
1010 POKE198,0:WAIT198,1:POKE198,0:RETUR
N
1020 DATA-40,1,40,-1
```



HARMONY

This program shows how a piece of music can be converted to run on the Commodore 64. It uses all three voices even though there may be different lengths of notes.

Each voice has a separate timer and all the music DATA is pre-stored in a separate ARRAY for each voice.

The program utilises the built-in filters to create an unusual sound for the melody line of the music; this filter can be taken out quite easily if you consult the MANUAL.

I decided to use the PULSE wave form because of its versatility and because it can create a wide scope of effects.

Good luck when making up your own songs. All that needs changing is the DATA: the rest of the program will be exactly the same.

```
5 S=54272
6 POKES+2,1:POKES+3,13:POKES+9,200:POKES
+10,13:POKES+16,100:POKES+17,9
7 POKES+24,31:POKES+23,241:POKES+21,3:PO
KES+22,100
8 POKES+5,16:POKES+6,241:POKES+12,16:POK
ES+19,16:POKES+20,241:POKES+13,241
9 POKES+0,0:POKES+1,0:POKES+7,0:POKES+8,
0:POKES+14,0:POKES+15,0
10 DIM A(100,2),B(100,2),C(100,2)
11 K=0 :REM FIRST VOICE
12 READH,L,D
13 IFD=-1THEN15
14 A(K,0)=H:A(K,1)=L:A(K,2)=D:K=K+1:GOTO
12
15 ::::REM SECOND VOICE
16 I=0
17 READH,L,D
18 IFD=-1THEN20
19 B(I,0)=H:B(I,1)=L:B(I,2)=D:I=I+1:GOTO
17
```

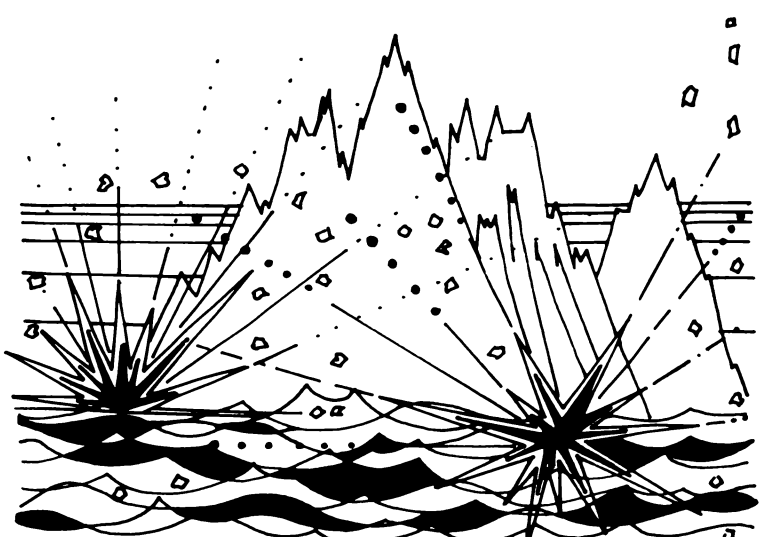
```
20 ::::REM THIRD VOICE
21 J=0
22 READH,L,D
23 IFD=-1THEN25
24 C(J,0)=H:C(J,1)=L:C(J,2)=D:J=J+1:GOTO
22
25 POKES+4,65:POKES+11,65:POKES+18,65
98 T=0:P1=-1:P2=-1:P3=-1:D1=0:D2=0:D3=0
99 :::REM MAIN TIMING ROUTINE:::
100 IFT>D1THENP1=P1+1:D1=D1+A(P1,2):POKE
S+4,64:GOSUB200
101 IFT>D2THENP2=P2+1:D2=D2+B(P2,2):POKE
S+11,16:GOSUB210
102 IFT>D3THENP3=P3+1:D3=D3+C(P3,2):POKE
S+18,16:GOSUB220
103 IFP1>KTHEN110
105 T=T+1:GOTO100
110 POKES,0:POKES+1,0:POKES+7,0:POKES+8,
0:POKES+14,0:POKES+15,0
111 POKES+4,0:POKES+11,0:POKES+18,0
115 PRINT"PRESS ANY KEY TO PLAY AGAIN"
120 POKE198,0:WAIT198,1:POKE198,0:GOTO98
199 ::::REM PLAY NEXT NOTES:::
200 POKES,A(P1,1):POKES+1,A(P1,0):POKES+
4,65:RETURN
210 POKES+7,B(P2,1):POKES+8,B(P2,0):POKE
S+11,17:RETURN
220 POKES+14,C(P3,1):POKES+15,C(P3,0):PO
KES+18,17:RETURN
1000 REM VOICE 1 DATA
1001 DATA38,126,12,34,75,12,30,141,12,34
,75,12,34,75,24,25,177,24,25,177,12
1002 DATA30,141,12,40,200,12,40,200,6,38
,126,54,38,126,12,34,75,12,30,141,12
1003 DATA34,75,12,34,75,24,25,177,24,25,
177,12,30,141,12,40,200,12,40,200,6
1004 DATA38,126,54,38,126,18,30,141,30,3
4,75,18,25,177,30,38,126,18,30,141,30
1005 DATA34,75,18,25,177,30,38,126,12,34
,75,12,30,141,8,30,141,4,34,75,12
1006 DATA34,75,18,25,177,30,0,0,12,40,20
0,12,40,200,12,40,200,6,38,126,10
```

```
1007 DATA34,75,4,30,141,50,0,0,0,0,0,-1
1008 REM VOICE 2 DATA
1010 DATA30,141,24,22,227,24,30,141,24,2
0,100,24,20,100,12,25,177,12
1011 DATA30,141,12,30,141,6,30,141,54,30
,141,24,22,227,24,30,141,24,20,100,24
1012 DATA20,100,12,25,177,12,30,141,12,3
0,141,6,30,141,54,30,141,18
1013 DATA25,177,14,25,177,4,25,177,12,25
,177,12,25,177,18,20,100,4
1014 DATA20,100,12,30,141,18,25,177,14,2
5,177,4,25,177,12,30,141,18
1015 DATA21,154,14,21,154,4,21,154,12,30
,141,24,22,227,24,21,154,18
1016 DATA21,154,30,0,0,12,25,177,12,25,1
77,12,25,177,6,22,227,10,34,75,4
1017 DATA19,63,50,0,0,0,0,0,-1
1019 REM VOICE 3 DATA
1020 DATA7,163,24,3,210,24,4,73,12,6,108
,12,8,147,12,4,73,12,5,25,12,0,0,12
1021 DATA8,147,12,5,25,12,7,163,12,5,185
,12,6,108,12,7,53,12,7,163,24
1022 DATA3,210,24,4,73,12,6,108,12,8,147
,12,4,73,12,5,25,12,0,0,12
1023 DATA8,147,12,5,25,12,7,163,12,5,185
,12,7,163,12,7,53,12,6,108,18,6,108,14
1024 DATA6,108,4,6,108,12,5,25,18,5,25,1
4,5,25,4,5,25,12,6,108,18,6,108,14
1025 DATA6,108,4,6,108,12,4,73,18,8,147,
14,8,147,4,8,147,12,7,163,24
1026 DATA3,210,24,4,73,12,6,108,12,8,147
,12,4,73,12,5,25,12,7,163,12
1027 DATA10,60,12,5,25,12,7,163,12,5,185
,12,6,108,12,7,53,22,0,0,0,0,0,-1
```

SUBMARINE BREAKOUT

You have blown out your ballast tanks, and as a result you are gradually floating to the surface. However, you have noticed through your periscope that you are underneath an iceberg and that you will therefore be unable to surface and take on more oxygen. Your only hope is to try and destroy the iceberg with your torpedoes. These are fired using the space bar. You can only fire them one at a time, and not until the previous one has been totally destroyed.

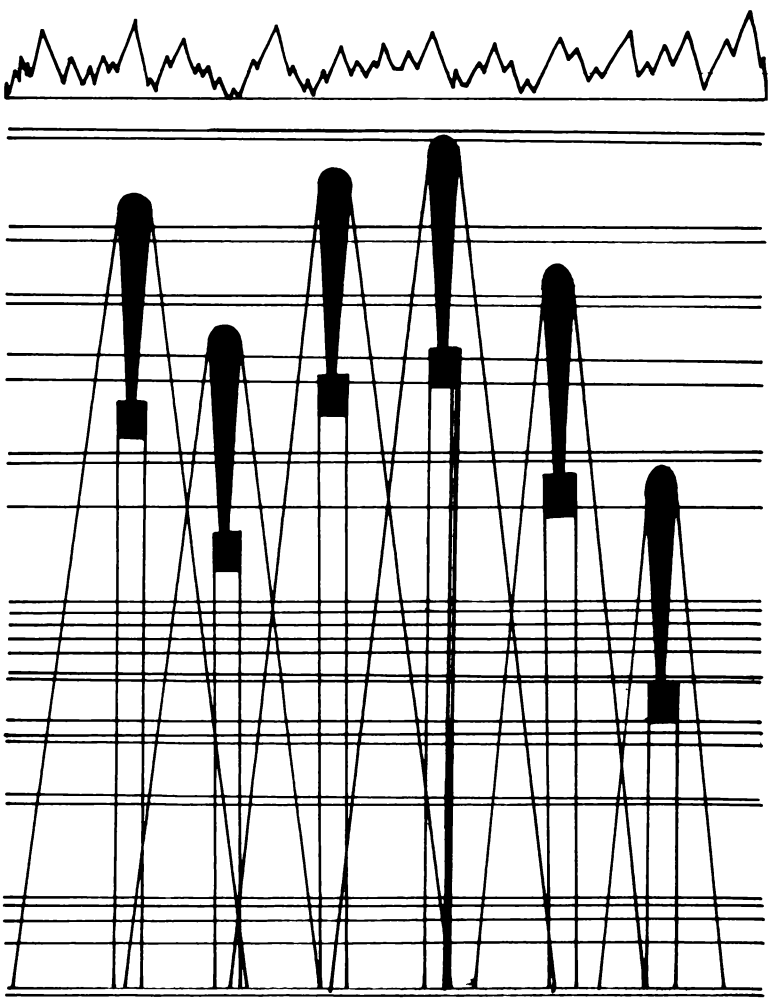
If you make it to the surface without hitting any remaining fragments of ice, the game becomes progressively more difficult and you have to be extremely accurate to survive.



```
1 POKE54276,0:POKE54278,249:POKE54277,0:
POKE54272,0:POKE54273,3
2 POKE54296,15:POKE54272,10:POKE54273,10
10 D=0:SC=0:SP=2020:POKE53281,14:POKE532
80,7
20 GOSUB1000
25 S$="]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]"
26 IFD>3THEND=3
27 PRINT"sE";:FORA=1TO32:PRINT"SQ";LEFT$
(S$,A+3);
28 R=INT(D+2*RND(1)+1):IFA<17THENR=R+(A-
2):GOTO30
29 R=R+(10-(A-20))
30 FORI=1TOR:PRINT"R ⌫Q";:NEXTI,A:PRINT"
Spr]]]]]]]]]]]]]]]]THE TIPS"
31 PRINT"S£SCORE :"SC"S]]]]]]]]]]]]]]]]
]]]]]]]]]]]]]]]]]]]]]]&S"
32 IFB>0THENGOSUB100
33 IFTI/4<>INT(TI/4)THEN33
34 GOSUB200:IFPEEK(SP)=160THEN300
35 IFPEEK(SP)=102THEN400
36 GETA$:IFA$="  "ANDB=0THENB=SP+2:GOSUB1
00
37 GOTO31
99 :REM FIRE MISSILE
100 IFB<1104THENPOKEB,32:B=0:T=0:C=0:RET
URN
102 POKEB,32:B=B-40:POKE54272+B,2:POKEB,
30
104 IFC=1THENC=0:POKEB,32:B=0:T=0:RETURN
106 IFT=0ANDC=0ANDPEEK(B-40)<>32THENT=1:
C=6:GOSUB110:RETURN
108 IFT=1THENC=C-1:SC=SC+1:RETURN
109 RETURN
110 POKE54276,0: POKE54273,8:POKE54276,1
29:POKE54276,128:RETURN
200 POKESP,60:POKESP+1,45:POKESP+2,62:PO
KESP+3,32:POKE54272+SP,0
201 POKE54273+SP,0:POKE54274+SP,0:SP=SP-
1
```

```
202 RETURN
300 A=23:SP=SP+1
302 POKESP-40,32:POKESP-39,32:POKESP-38,
32:POKE54276,129
304 POKE54273,A:GOSUB200:FOR O=1TO50:NEX
T
305 POKESP+2,42:FORO=1TO50:NEXT:SP=SP+41
306 IFSP>2023THENSP=SP-41
307 A=A-1:IFA=1THEN310
308 GOTO302
310 POKE54276,128
320 PRINT"SpHARD LUCK, YOU HAVE HIT THE
ICEBERG "
322 PRINT"AND SUNK."
324 PRINT"DO YOU WANT TO TRY AGAIN (Y/N)
?"
326 GET G$:IFG$=""THEN326
328 IF G$="Y"THENRUN
330 IFG$="N"THENEND
332 GOTO320
400 :PRINT"SCONGRATULATIONS BUT NOW TRY
AGAINN WITH"
401 PRINT"A LARGER ICE BERG":D=D+1
405 PRINT"PRESS ANY KEY"
406 POKE198,0:WAIT198,1:POKE198,0
410 B=0:SP=2020:GOTO25
999 END
1000 PRINT"sp                INSTRUCTIONS"
1001 PRINT"YOU HAVE BLOWN OUT YOUR BALLA
ST TANKS"
1002 PRINT"AND ARE STEADILY RISING. BUT
THERE IS "
1003 PRINT"AN ICEBERG IN THE WAY."
1004 PRINT"WITH YOUR TORPEDOES YOU HAVE
TO TRY AND"
1005 PRINT"TOTALLY DESTROY THE ICE TO SU
RVIVE"
1006 PRINT"QYOU CAN ONLY FIRE ONE TORPED
O AT A TIME"
1007 PRINT"QGOOD LUCK.  (YOU'LL NEED IT)
."
```

```
1008 PRINT "QQPRESS ANY KEY TO START"
1010 POKE198,0:WAIT198,1:POKE198,0:RETUR
N
```



PIANO

This program is very handy if you want to annoy someone with some unbearably weird sounds.

The strange sounds are made by typing '65' as wave form; more pleasant sounds arise if the other wave forms are used.

The main part of the program is in machine code (only the part that sets up the program is in BASIC). Although only two colours are used in this program, it is possible to use many more at the same time.

The only thing you need to know about the sound part is that '@' will stop the music (noise).

```
10 PRINT"sEVERY KEY ON THE KEYBOARD HAS
A UNIQUE"
11 PRINT"MUSICAL NOTE.
12 PRINT"QYOU CAN TYPE WHAT YOU LIKE AND
 LISTEN"
13 PRINT"TO THE SOUNDS THAT YOU MAKE"
14 PRINT"QQBUT FIRST ALL THE MACHINE COD
E MUST "
15 PRINT"BE READ IN AND THEN YOU CAN STA
RT"
16 PRINT"QQYOU WILL ALSO NOTICE THAT THE
 SCREEN IS"
17 PRINT"TWO DIFFERENT COLOURS AT THE SA
ME TIME"
18 PRINT"QTHIS IS DONE BY WHAT IS CALLED
 AN "
19 PRINT"INTERRUPT"
20 PRINT"QQPRESS ANY KEY TO CARRY ON"
21 POKE650,255
22 FORA=32768TO32993:READB:POKEA,B:NEXT
23 INPUT"sWHAT WAVEFORM (129/65/33/17)":
WF
24 POKE32888,WF
25 PRINT"sNOW YOU CAN START CREATING"
```

```
30 SYS32945
100 DATA120,169,13,141,20,3,169,128,141,
21,3
101 DATA88,96,165,87,208,20,230,87,169,6
,141,33,208,169,201,141
102 DATA18,208,169,27,141,17,208,76,54,1
28,198,87,169,7,141,33
103 DATA208,169,27,141,17,208,169,1,141,
18,208,169,1,141,25,208
104 DATA76,49,234,76,49,234,169,127,141,
13,220,32,0,128,169,1
105 DATA32,152,128,169,5,141,5,212,169,2
40,141,6,212,169,119,141
106 DATA2,212,169,2,141,3,212,165,162,10
5,2,197,162,208,252,169
107 DATA0,32,191,128,208,23,238,2,212,23
8,3,212,169,33,141,4
108 DATA212,173,28,212,141,21,212,141,22
,212,76,98,128,41,63,141
109 DATA0,212,141,1,212,169,30,141,23,21
2,76,98,128,141,26,208
110 DATA169,129,141,18,212,169,255,141,1
4,212,141,15,212,169,0,141
111 DATA19,212,141,20,212,96,169,239,141
,24,212,169,255,141,28,212
112 DATA76,65,128,0,173,18,208,141,9,212
,141,10,212,173,18,208
123 DATA141,22,212,162,16,206,3,212,202,
208,250,162,16,206,3,212
124 DATA202,208,250,32,228,255,96
```

L.E.D. DISPLAY

This idea came to me as I was looking at a moving display in a shop window and thinking how easy it would be to do a program in the same way.

It can be used to impress people, with flattering statements moving before their very eyes.

The program works on the theory of binary arithmetic which is used to build up the larger symbols. It could be expanded to use other characters rather than just the alphabet, and you could also make it do some special characters of your own. Other shapes would be added by including more DATA; the DIM statement at the start would have to be changed to take the extra shapes into account.

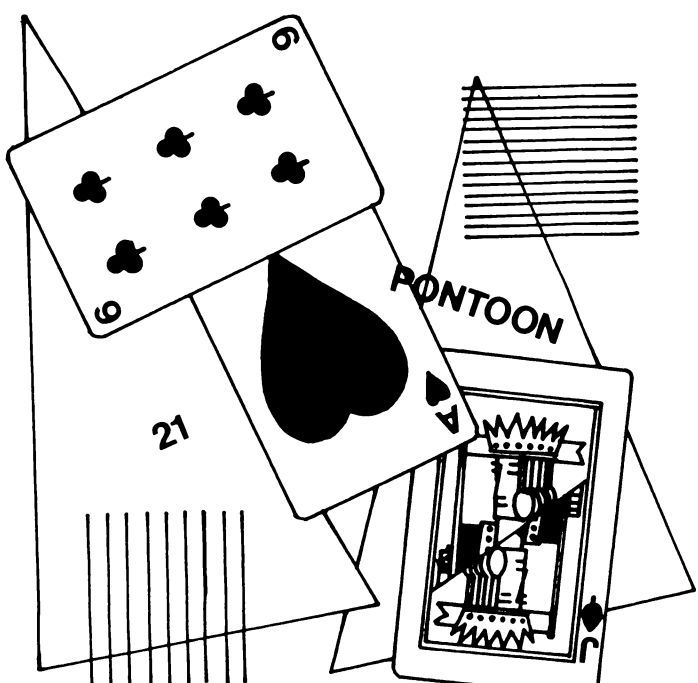
```
1 DIML(27,5),K$(5)
2 FORA=1TO27:FORB=1TO5:READL(A,B):NEXT:N
EXT
3 POKE53281,0:POKE53280,0:PRINT "s□"
10 PRINT "sTYPE IN YOUR MESSAGE IN LESS T
HAN 60
11 PRINT "LETTERS OF THE ALPHABET"
12 PRINT "AND PRESS RETURN WHEN READY"
13 INPUT "E":M$
14 IFLEN(M$)>64THEN13
15 FORA=1TO5:K$(A)="                    ":NEXT:F
ORA=1TOLEN(M$)
16 B=ABS((ASC(MID$(M$,A,1)))-64):IFB=32T
HENB=27
17 FORC=1TO5
18 N=L(B,C)
19 IFN*2>7THENN$=N$+"q":N=N-4:GOTO21
20 N$=N$+" "
```

```
21 IFN*2>3THENN$=N$+"q":N=N-2:GOTO23
22 N$=N$+" "
23 IFN/2=.5THENN$=N$+"q":GOTO25
24 N$=N$+" "
25 N$=N$+"
26 K$(C)=K$(C)+N$:N$=""
27 NEXT
28 NEXT:K$(5)=K$(5)+"q
30 PRINT"s        WATCH THIS SPACE CAREFU
LLY !"
31 PRINT"SQQQ]]]]]]]]]]]]R
   Q▵ Q▵ Q▵ Q▵ Q▵ Q▵ "
32 PRINT"SQQQQ]]]]]]]]]]]]R Q▵ Q▵ Q▵ Q▵
Q▵                    rE"
34 FORP=1TOLEN(K$(1))
35 PRINT"SQQQQ]]]]]]]]]]]]]]":MID$(K$(1),P
,15)
36 PRINT"SQQQQQ]]]]]]]]]]]]]]":MID$(K$(2),
P,15)
37 PRINT"SQQQQQQ]]]]]]]]]]]]]]":MID$(K$(3)
,P,15)
38 PRINT"SQQQQQQQ]]]]]]]]]]]]]]":MID$(K$(4
),P,15)
39 PRINT"SQQQQQQQQ]]]]]]]]]]]]]]":MID$(K$(
5),P,15)
40 NEXT
41 GOTO31
100 DATA 7,5,7,5,5,6,5,6,5,6,3,4,4,4,3,6
,5,5,5,6,7,4,6,4,7,7,4,6,4,4
101 DATA 7,5,3,1,6,5,5,7,5,5,7,2,2,2,7,1
,1,1,5,7,4,5,6,6,5,4,4,4,4,7
102 DATA 5,7,5,5,5,7,5,5,5,5,7,5,5,5,7,7
,5,7,4,4,7,5,7,1,1,7,5,7,6,5
103 DATA 7,4,7,1,7,7,2,2,2,2,5,5,5,5,7,5
,5,5,5,2,5,5,5,7,5,5,5,2,5,5
104 DATA 5,5,5,2,2,7,1,2,4,7,0,0,0,0,0
```

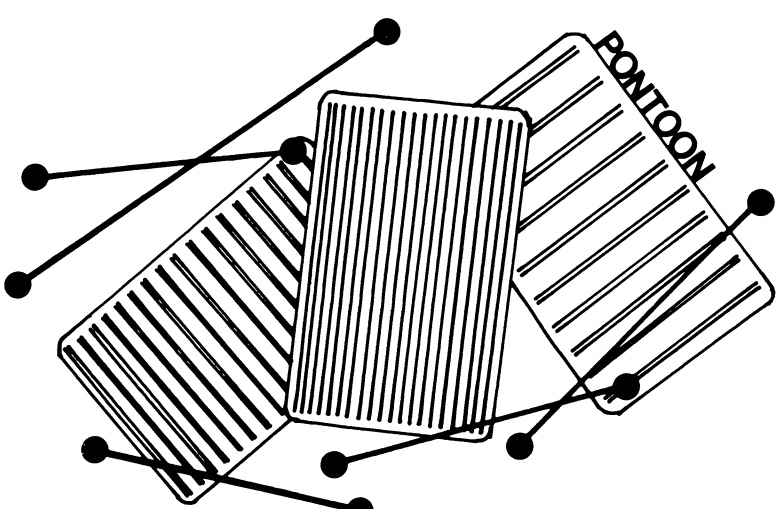
PONTOON

How about a game of cards? Just for all you laughing card sharks, here's a little practice before you take on Vegas or break the Bank of Monte Carlo.

It is a relaxing game played, without high stakes, against your friendly computer. The computer isn't out to cheat you, nor can you cheat it. The rules are the same as for normal 'pontoon', except that you can't have 'five card tricks' or actually get 'pontoon'. Also no 'burning' otherwise the computer will get angry and maybe overheat!



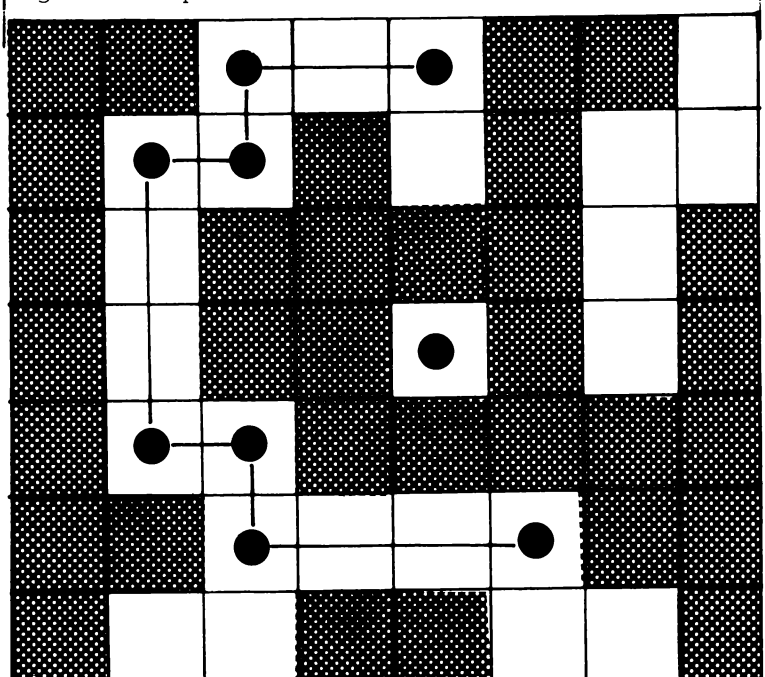
```
1008 PRINT"BY PRESSING 'S' YOU WILL STIC
K AND THEN THE COMPUTER WILL GO"
1009 PRINT"QQQPRESS ANY KEY TO START"
1010 POKE198,0:WAIT198,1:POKE198,0:RETUR
N
1100 FORA=1TO13:READC$(A,1):C$(A,1)=C$(A
,1)+" OF SPADES":NEXT:RESTORE
1101 FORA=1TO13:READC$(A,2):C$(A,2)=C$(A
,2)+" OF HEARTS":NEXT:RESTORE
1102 FORA=1TO13:READC$(A,3):C$(A,3)=C$(A
,3)+" OF CLUBS":NEXT:RESTORE
1103 FORA=1TO13:READC$(A,4):C$(A,4)=C$(A
,4)+" OF DIAMONDS":NEXT
1104 FORA=1TO4:FORB=1TO13:C(B,A)=0:NEXTB
,A:RETURN
2000 DATA ACE,TWO,THREE,FOUR,FIVE,SIX,SE
VEN,EIGHT,NINE,TEN
2001 DATA JACK,QUEEN,KING
```



MAZEY

In this simple maze game you have to get from one side of a maze to the other in a small number of moves. If you get totally blocked in, there is a blaster which will clear away one square in all directions — thus enabling you to pass. This blaster counts as extra moves so you should try and use it as little as possible. Take care, too, not to blast away your home base.

You can also be killed if you bump into a wall, or if you go off the top or bottom of the screen.



```
1 HS=999999
10 GOSUB1000:FORA=0TO3:READC(A):NEXT
11 POKE53280,12:POKE53281,9
12 PRINT"s";:FORA=0TO999
13 IFRND(1)<.38THENPOKE1024+A,102:POKE55
296+A,C(A AND 3)
```

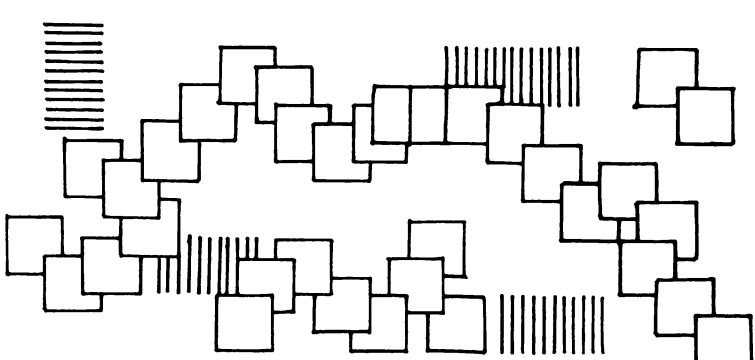
```
14 NEXT
16 P=1024:Q=2023-INT(39*RND(1))
17 POKEP,160:POKE54272+P,7:POKEQ,42:POKE
54272+Q,7
18 M=0
20 GETA$:IFA$=""THEN20
21 M=M+1:POKEP,32
22 IFA$="Y"THENP=P-40:GOTO28
23 IFA$="B"THENP=P+40:GOTO28
24 IFA$="G"THENP=P-1:GOTO28
25 IFA$="H"THENP=P+1:GOTO28
26 IFA$="e"THEN100
27 POKEP,160:POKE54272+P,7:GOTO20
28 GOTO50
30 PRINT"Sp  YOU HAVE RUN OFF THE SCREEN
 AND HAVE"
31 PRINT"LOST THE GAME"
32 GOTO253
50 IFPEEK(P)=42THEN160
51 IFPEEK(P)<>32THEN170
52 POKEP,160:POKE54272+P,7:GOTO20
100 IFPEEK(P+1)=42ORPEEK(P+40)=42ORPEEK(
P-1)=42THEN150
101 POKEP+1,32:POKEP+40,32:POKEP-1,32:PO
KEP-40,32:POKEP,160:POKE54272+P,7
102 M=M+2:GOTO20
150 PRINT"□YOU HAVE BLASTED YOUR OWN BAS
E AND LOST"
151 GOTO253
160 PRINT"sp  YOU HAVE MADE IT TO THE HO
ME BASE"
161 PRINT"AND IT TOOK YOU";M;"MOVES"
162 IFM<=HSTHENPRINT"YOU HAVE ALSO GOT T
HE HIGH SCORE":HS=M:GOTO250
163 GOTO253
170 PRINT"SE  YOU HAVE CRASED INTO THE E
LECTIFIED "
171 PRINT"WALLS AND HAVE DIED"
172 GOTO253
200 PRINT"EQ DO YOU WANT ANOTHER GAME (Y
/N)"
```

```
201 GETD$:IFD$=""THEN201
202 IFD$="Y"THENGOTO12
203 IFD$="N"THENEND
204 GOTO201
250 PRINT"■INPUT YOUR NAME (LESS THAN 10
 CHARS)"
251 INPUTN$
252 IFLEN(N$)<1ORLEN(N$)>10THEN250
253 PRINT"▷HIGH SCORE OF";HS; BY ";N$
254 GOTO200
999 END
1000 POKE53281,7:POKE53280,14:PRINT"s£
           INSTRUCTIONSQ"
1001 PRINT"YOU START AT ONE SIDE OF THE
SCREEN AND"
1002 PRINT"YOU HAVE TO TRY AND GET TO TH
E OTHER "
1003 PRINT"SIDE IN AS LITTLE MOVES AS PO
SSIBLE."
1004 PRINT"QTHERE IS A MAZE OF RANDOM CO
RRIDOORS "
1005 PRINT"IN YOUR WAY SO YOU MAY AT TIM
ES HAVE TO"
1006 PRINT"USE YOUR BLASTER TO CLEAR THE
 WALLS"
1007 PRINT"YOU HAVE TO GET TO THE '*' TO
 COMPLETE"
1008 PRINT"THE MAZE."
1009 PRINT"EVERY TIME YOU USE THE BLASTE
R IT "
1010 PRINT"COUNTS AS TWO EXTRA MOVES SO
TRY AND "
1011 PRINT"USE IT AS LITTLE AS POSSIBLE"
1012 PRINT"Q                    CONTROLS"
1013 PRINT"Q   UP-Y   DOWN-B   LEFT-G
RIGHT-H "
1022 PRINT"QQPRESS ANY KEY TO START"
1023 POKE198,0:WAIT198,1:POKE198,0:RETUR
N
1030 DATA1,0,5,14
```

PATHWAY

This is not a game, but a short program that produces a large result. It is a moving pathway that shows how colours can be used effectively. The colours are all kept in one STRING; only this STRING is changed to create the moving effect. This is done by taking the first colour off one end and putting it back on the other end.

```
10 C$="E£□ ↑←παuvwxyz|":PRINT"s";:POKE53
281,0:POKE53280,0
15 S$="##################################
####"
17 T=0:M=8:PRINT"S";
19 FORA=1TO15
20 PRINTLEFT$(C$,A);TAB(15-T);"R)";LEFT$
(S$,M)"□"
22 M=M+2:T=T+1:NEXT
25 L$=RIGHT$(C$,1)+LEFT$(C$,14):C$=L$
30 GOTO17
```

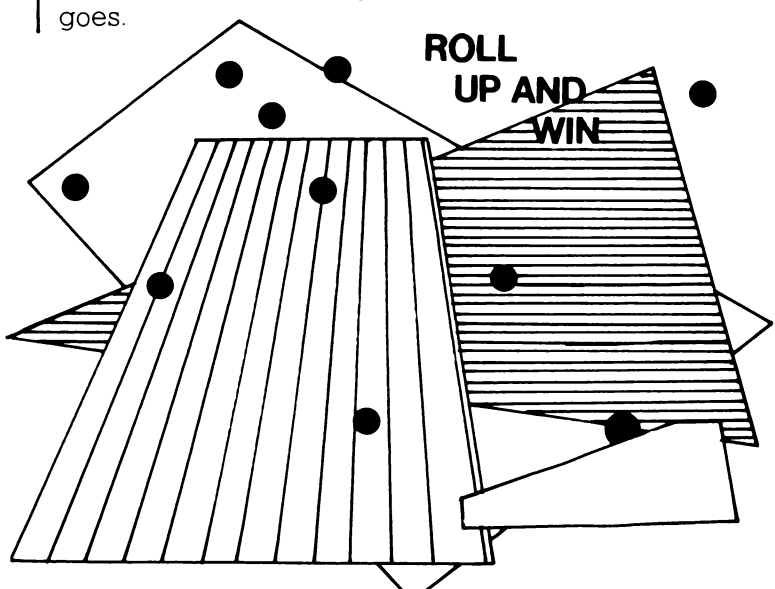


ROL-A-PENNY

This is a version of the funfair game in which you try and roll a penny (or two) down a slide and across a board of squares — hopefully to land in the middle of a square without the coin touching the sides of that square.

You take aim by lining up the arrow, fixed centrally at the bottom of the screen, with the penny, positioned above on a horizontal line. The arrow is directional, and by angling it to either the left — with the 'Z' key — or right — with the 'M' key — the coin will move correspondingly to and fro along its plane. When you are satisfied with the angle you press 'f1'. You will then be asked to control the strength of your coin's 'roll' by inputting a number between 1 and 150. The skill is in pin-pointing the correct combination of direction and strength. Press 'return' to set the coin rolling.

When the coin stops you will be told if you win or lose. The idea is to get as high a score as possible from ten goes.



```
1 V=53248:POKE650,255:POKE2040,13
5 GOSUB1000:GOSUB1020
10 N$="NO-ONE":POKE53281,1:POKE53280,15
11 S$="zopQ◣◣l :q"
12 GOSUB150:PRINT"s";:FORB=1TO9:FORA=1TO
15:PRINTS$;:NEXT:PRINT"Q":NEXT
13 PRINT"$$$$$$$$$$$$$$$$$$$$$$$$$$$$$"
:POKE1998,30:POKE54272+1998,0
14 T=0:FORG=1TO10
15 GETA$:IFA$=""THEN15
16 IFA$="Z"THEN20
17 IFA$="M"THEN30
18 IFA$="e"THEN40
19 GOTO15
20 Z=PEEK(V)-1
21 IFZ<26THENZ=255
22 POKEV,Z:GOTO15
30 Z=PEEK(V)+1
31 IFZ>255THENZ=26
32 POKEV,Z:GOTO15
40 INPUT"INPUT SPEED";SP
41 IFSP<1ORSP>150THEN12
42 L=PEEK(V):K=L:FORA=1TOSP
43 POKEV+1,PEEK(V+1)-1
44 J=(L-137)/SP:K=K+J
45 IFK<26THENK=255-(26-K)
46 IFK>255THENK=26+(K-255)
47 POKEV,K:NEXT
48 POKEV+31,255
49 PRINT"q                              ";
50 POKEV+31,0:FORD=1TO500:NEXT:IFPEEK(V+
31)=0THEN100
51 PRINT"SpNO SCORE"
52 FORD=1TO1000:NEXT:PRINT"SzopopopopQQQ
QQQQQQQQQQQQQQ"
53 RESTORE:GOSUB1020:NEXT
55 PRINT"SpAFTER 10 GOES YOU HAVE SCORED
 ";T
56 IFT>=HSTHENHS=T:PRINT"YOU HAVE GOT TH
E HIGH SCORE":GOTO60
57 GOSUB1009:RESTORE:GOSUB1020:GOTO12
```

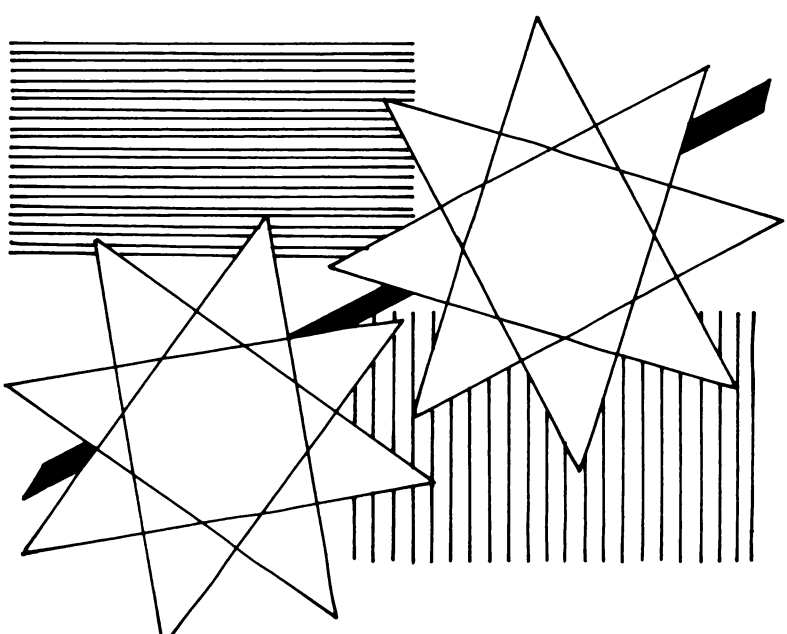
```
60 FORF=1TO3000:NEXT:PRINT"sINPUT YOUR N
AME"
61 INPUTN$
62 GOTO57
100 IFPEEK(U+31)=1THEN51
101 SC=INT(10*RND(1)+1)
102 PRINT"SzYOU HAVE WON";SC
103 T=T+SC
104 FORN=1TO2000:NEXT:PRINT Szopopopopop
opopopo";:GOTO52
150 PRINT"sHIGH SCORE OF"HS"POINTS BY ";
N$
151 FORD=1TO3000:NEXT:RETURN
999 END
1000 PRINT"sROL-A-PENNY"
1001 PRINT"QYOU ARE IN THE FAIRGROUND AN
D TRYING"
1002 PRINT"TO WIN SOME MONEY ON THE ROL-
A-PENNY"
1003 PRINT"SIDE-STALL."
1004 PRINT"USING 'Z' AND 'M' ADJUST THE
AIM"
1005 PRINT"AND BY TYPING IN A NUMBER BET
WEEN 0-150"
1006 PRINT"AS A DISTANCE GUAGE TO ROLL T
HE PENNY"
1007 PRINT"YOU HAVE 10 GOES TO GET AS HI
GH A SCORE"
1008 PRINT"AS POSSIBLE. THE POINTS ARE R
ANDOM      PRESS F1 TO GET DISTANCE"
1009 PRINT"PRESS ANY KEY TO START"
1010 POKE198,0:WAIT198,1:POKE198,0:RETUR
N
1020 FORA=0TO18:READB:POKE832+A,B:NEXT:F
ORA=19TO62:POKE832+A,0:NEXT
1021 POKEU+21,1:POKEU+39,0:POKEU+1,198:P
OKEU+0,137:RETURN
1030 DATA56,0,0,124,0,0,254,0,0,254,0,0,
254,0,0,124,0,0,56
```

DOODLER

With this program you can draw very complicated patterns or even create a work of art.

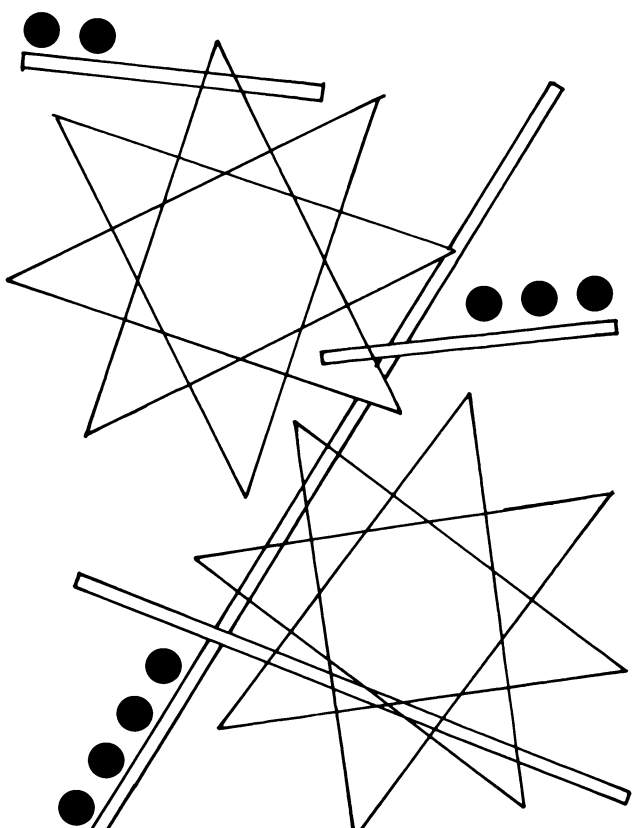
It works only with the joystick which must be in port 1; it is possible to change the program to use the keyboard, though this will make it slower. There are two machine code routines in the program. They are:
(1) To fill the colour map with the desired colour.
(2) To blank out the HI-RES screen which, if done in BASIC, will take about a minute (machine code is used to do this in less than a second).

Apart from those two routines, the program is very simple and a good example of the graphic capabilities of your Commodore 64.



```
0 FORA=0TO53:READB:POKE24448+A,B:NEXT
1 DATA 169,255,133,87,169,67,133,88,173
2 DATA 253,95,160,0,145,87,198,87,208
3 DATA 250,198,88,166,88,224,63,208,242
4 DATA 96,169,255,133,87,169,127,133,88
5 DATA 169,0,168,145,87,198,87,208,250
6 DATA 198,88,166,88,224,95,208,242,96
7 PRINT"s****SKETCH PAD****"
8 PRINT" JOYSTICK CONTROL PORT 1"
9 PRINT"FIRE BUTTON =CLEAR"
11 PRINT"PRESS ANY KEY TO START"
12 GETA$:IFA$=""THEN12
13 INPUT"sBORDER COLOUR";BC
15 INPUT"LINE COLOUR";LC
20 INPUT"SCREEN COLOUR";SC
40 LC=LC*16:IFSC*16+BC>255THEN13
50 VC=53248+13*256:V=53248
55 POKEV+24,8
60 POKEVC,PEEK(VC)AND254
70 POKEV+32,BC
80 POKEV+22,PEEK(V+22)OR200
81 POKEV+17,PEEK(V+17)OR32
90 CO=LC+SC:POKE24573,CO:SYS24448
91 POKE17152,CO:POKE32512,0
100 P=1024*24:B=128
105 POKEP,PEEK(P)ORB
110 A=PEEK(56321)AND31
120 IFA=30THEN500
130 IFA=29THEN700
140 IFA=27THEN1000
150 IFA=23THEN1300
160 IFA=15THEN1500
170 GOTO105
500 IFP/8=INT(P/8)THEN520
510 P=P-1:GOTO105
520 P=P-320+7:GOTO105
700 IF(P+1)/8=INT((P+1)/8)THEN720
710 P=P+1:GOTO105
720 P=P+320-7:GOTO105
1000 B=B*2
1010 IFB>128THEN1030
```

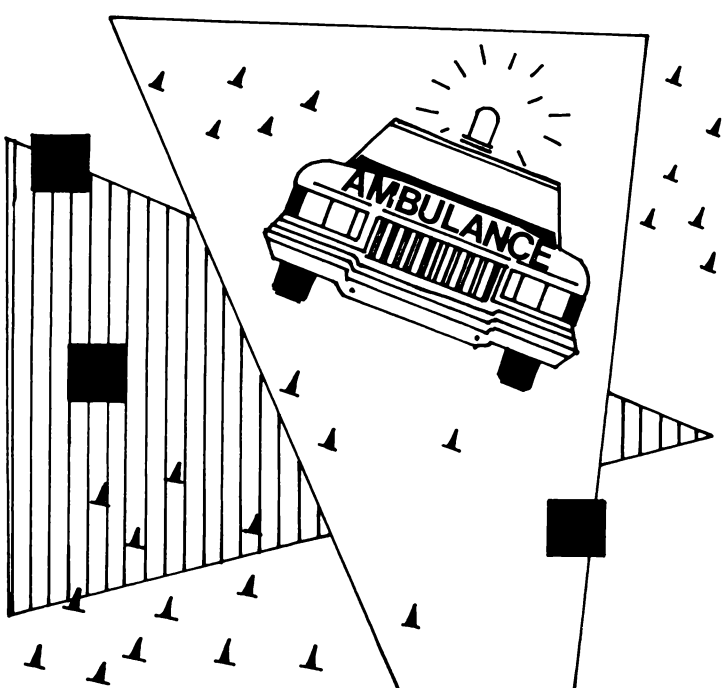
```
1020 GOTO105
1030 P=P-8:B=1:GOTO105
1300 B=B/2:IFB<1THEN1320
1310 GOTO105
1320 B=128:P=P+8:GOTO105
1500 SYS24476:GOTO105
```



AMBULANCE DRIVER

The idea is to swerve in and out of the advancing objects, consisting of people, footballs, pot-holes, and nails. If you run over any of these it will be the end of your game.

The number of objects that appear on the screen is random to a certain extent. If you wish you can change this by altering the value of 'T' — but make sure you don't change it to a negative number. Sounds can be added quite easily to the game; but I decided to leave adjustments like that to you.



```
5 GOSUB1000:P=18:T=5:POKE650,255:A=RND(-
TI)
6 FORA=1TO3:READLS(A),RS(A):NEXT
7 FORA=0TO4:READO$(A):NEXT
10 P=18:S=0:A$(0)="E      Q     7PQ    4*'
Q    PRE roQ    "
11 C$="]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]"
20 A$(1)="E      Q     7PQ    4.'Q    PR rE
oQ    "
30 POKE53281,11:POKE53280,5
40 INPUT"□INPUT SKILL LEVEL (1-3)";SL
50 IFSL<1ORSL>3THEN40
55 PRINT"s";TAB(P);A$(0)
60 L=16-5*SL:R=23+5*SL
70 FORA=0TO22:GOSUB990:NEXT
80 V=V+1:V=VAND1:S=S+1
81 FORM=1TO3:IFPEEK(1184+P+M)<>32THEN200
82 NEXT:IFS>1000THENT=2
83 IFS>600THENT=2
84 IFS>400THENT=3
85 IFS>200THENT=4
86 LL=INT(T*RND(1)):IFLL=0THENGOSUB150
87 GOSUB990
88 GETB$:IFB$=""THEN100
89 IFB$="Z"THEN250
90 IFB$="M"THEN260
100 GOSUB995
120 GOTO80
150 KK=INT(5*RND(1)):PRINT"SQQQQQQQQQQQQ
QQQQQQQQQQ";LEFT$(C$,L+1);
155 PRINTLEFT$(C$,INT((SL*10)*RND(1)));O
$(KK):RETURN
200 PRINT"SQQQQQQ]]]]]]]] &&& &&& &&& &&
& & &"
210 PRINT"]]]]]]]] &   & & & & &   & &"
211 PRINT"]]]]]]]] &   &&& &&& &&& &&&"
212 PRINT"]]]]]]]] &   &&  & &   & & &"
213 PRINT"]]]]]]]] &&& & & & & &&& & &"
215 FORQ=0TO15:FORW=0TO15:POKE53280,Q:PO
KE53281,W:NEXT:NEXT
```

```
220 POKE53281,12:POKE53280,4:PRINT"ş←HIG
H SCORE ";HS,"YOUR SCORE ";S
225 IFS>HSTHENHS=S:PRINT"A NEW HIGH SCOR
E"
227 PRINT"ANOTHER GAME (Y/N) ?"
228 GETAN$:IFAN$=""THEN228
229 IFAN$="N"THENEND
230 IFAN$<>"Y"THEN227
235 GOTO10
250 P=P-1:IFP<LS(SL)THEN200
255 GOTO100
260 P=P+1:IFP>RS(SL)THEN200
265 GOTO100
990 PRINT"SP QQQQQQQQQQQQQQQQQQQQQQQQ"TAB
(L);"?";TAB(R);"R?r":RETURN
995 PRINT"S";TAB(P)A$(VAND1);"S":RETURN
1000 PRINT"ş YOU HAVE JUST BEEN AT AN"
1001 PRINT"ACCIDENT, AND HAVE TO GET "
1002 PRINT"BACK TO THE HOSPITAL QUICKLY"
1003 PRINT"AVOIDING OBSTACLES THAT GET"
1004 PRINT"IN YOUR WAY."
1005 PRINT"IF YOU RUN SOMETHING OVER"
1006 PRINT"YOU WILL LOSE POINTS"
1007 PRINT"Z-MOVES YOU LEFT "
1008 PRINT"M-MOVES YOU RIGHT"
1010 RETURN
10000 DATA 11,24,6,29,1,36,"aqQ▵vS"
10001 DATA"p111111S","Qu1Q▵▵JkS","πwQ▵v
S"
10002 DATA"zn7mQ▵▵▵m/nS"
```

SIMON

In this game, the computer will flash up a sequence of colours; you have to copy this sequence. You may think it's easy, but as the game progresses a new colour is added to the end. You type in your answer using the first letter of the colour — thus for Red Blue Green Yellow you type in R B G Y.

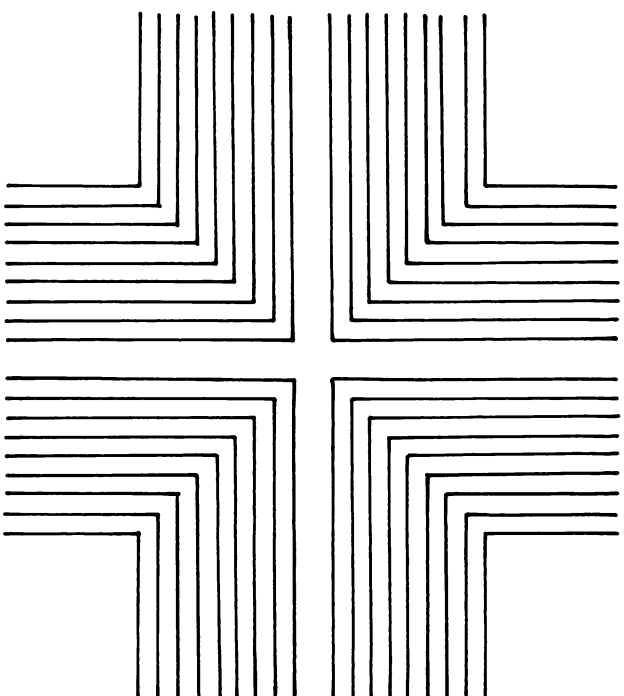
The program uses large blocks of colour and a different tone bleep for each of the four colours. To make it harder you can close your eyes and try to guess the colour by the sound.

```
10 GOSUB10000:REM INSTRUCTIONS
11 SO=54272:POKESO+24,15:POKESO+5,0:POKE
SO+6,243
12 R$="£]]]]]]]]]]]]":B$="]]]]]]]←":G$=R
$+"↑"
15 Y$="π]]]]]]]]]]]]]]]]"
20 PRINT"sQQQQQQQ";:A$=Y$:GOSUB5000 :
REM DRAW YELLOW
21 PRINT"SQQQQQQQ";:A$=B$:GOSUB5000
22 PRINT"SQQ";:A$=R$:GOSUB5000
23 PRINT"SQQQQQQQQQQQQ";:A$=G$:GOSUB5000
25 POKE53281,1:POKE53280,13
26 PRINT"SQp                    RED"
27 PRINT"QQQQQQQQQ]]]]]]BLUE]]]]]]]]]]]]
]]]]]]YELLOW"
28 PRINT"QQQQQQQQQ]]]]]]]]]]]]]]]]GREEN"
30 T=1:C$=""
32 PRINT"S
      ";:GOSUB80
33 PRINT"S NOW TYPE YOUR SEQUENCE (R,G,Y
,B)S"
34 O=1
35 GETF$:IFF$=""THEN35
37 IFF$<>MID$(C$,O,1)THEN500
38 O=O+1:IFO>LEN(C$)THEN50
```

```
39 F$="":GOTO35
50 T=T+1:GOTO32
80 FORK=1TOLEN(C$):IFMID$(C$,K,1)="R"THE
NGOSUB90
81 IFMID$(C$,K,1)="G"THENGOSUB190
82 IFMID$(C$,K,1)="Y"THENGOSUB290
83 IFMID$(C$,K,1)="B"THENGOSUB390
84 FORN=1TO300:NEXT:NEXT
85 CO=INT(RND(1)*4)+1
86 ON CO GOSUB 100,200,300,400
88 FORD=1TO500:NEXT:RETURN
90 FORA=1TO5:POKESO,100:POKESO+1,20:POKE
SO+4,33:POKESO+4,32:GOTO105
100 C$=C$+"R":FORA=1TO5:POKESO,100:POKES
O+1,20:POKESO+4,33:POKESO+4,32
105 A$=R$+"E"
110 PRINT"SQQ";:GOSUB5000:FORB=1TO50:NEX
T:A$=R$:PRINT"SQQ";:GOSUB5000
115 FORB=1TO50:NEXT:NEXT:POKESO,0:POKESO
+1,0:RETURN
190 FORA=1TO5:POKESO,50:POKESO+1,50:POKE
SO+4,33:POKESO+4,32:GOTO205
200 C$=C$+"G":FORA=1TO5:POKESO,50:POKESO
+1,50:POKESO+4,33:POKESO+4,32
205 A$=G$+"E":PRINT"SQQQQQQQQQQQQ";:GOSU
B5000
210 FORB=1TO50:NEXT:A$=G$:PRINT"SQQQQQQQ
QQQQQ";:GOSUB5000
215 FORB=1TO50:NEXT:NEXT:POKESO,0:POKESO
+1,0:RETURN
290 FORA=1TO5:POKESO,10:POKESO+1,10:POKE
SO+4,33:POKESO+4,32:GOTO305
300 C$=C$+"Y":FORA=1TO5:POKESO,10:POKESO
+1,10:POKESO+4,33:POKESO+4,32
305 A$=Y$+"E":PRINT"SQQQQQQQ";:GOSUB5000
310 FORB=1TO50:NEXT:A$=Y$:PRINT"SQQQQQQQ
";:GOSUB5000
315 FORB=1TO50:NEXT:NEXT:POKESO,0:POKESO
+1,0:RETURN
390 FORA=1TO5:POKESO,255:POKESO+1,7:POKE
SO+4,33:POKESO+4,32:GOTO405
```

```
400 C$=C$+"B":FORA=1TO5:POKESO,255:POKES
O+1,7:POKESO+4,33:POKESO+4,32
405 A$=B$+"E":PRINT SQQQQQQQ";:GOSUB5000
410 FORB=1TO50:NEXT:A$=B$:PRINT"SQQQQQQQ
";:GOSUB5000
415 FORB=1TO50:NEXT:NEXT:POKESO,0:POKESO
+1,0:RETURN
500 PRINT"sYOU HAVE MADE A MISTAKE BUT M
ANAGED TO "
501 PRINT"GET ";T-1;" GUESSES RIGHT."
502 PRINT"THE CORRECT SOLUTION WAS"
503 FORA=1TOLEN(C$)
504 IFMID$(C$,A,1)="R"THENPRINT"RED"
505 IFMID$(C$,A,1)="B"THENPRINT"BLUE"
506 IFMID$(C$,A,1)="Y"THENPRINT"YELLOW"
507 IFMID$(C$,A,1)="G"THENPRINT"GREEN"
508 NEXT
510 PRINT"DO YOU WANT ANOTHER GO (Y/N) ?
"
515 GETT$:IFT$=""THEN515
516 IFT$="N"THENEND
517 IFT$="Y"THENRUN
518 GOTO515
4999 END
5000 PRINTA$;"]]]]]]R)□"
5001 PRINTA$;"]]]]]R)  □"
5002 PRINTA$;"]]]]R)    □"
5003 PRINTA$;"]]]R)      □"
5004 PRINTA$;"]]]R        r"
5005 PRINTA$;"]]]□R      r)"
5006 PRINTA$;"]]]]□R    r)"
5007 PRINTA$;"]]]]]□R  r)"
5008 PRINTA$;"]]]]]]□)"
5010 RETURN
10000 POKE53281,1:PRINT"spINSTRUCTIONS"
10001 PRINT"QTHE COMPUTER WILL CHOSE A R
ANDOM "
10002 PRINT"SEQUENCE OF COLOURS, AND ALL
 YOU HAVE "
10003 PRINT"TO DO IS COPY EXACTLY THAT S
EQUENCE"
```

```
10004 PRINT"QEACH TIME YOU GET IT RIGHT
THERE WILL"
10005 PRINT"BE ONE MORE ADDED TO THE END
"
10006 PRINT"QSEE HOW MANY YOU CAN GET. G
OOD LUCK"
10007 PRINT"QQPRESS ANY KEY TO START"
10008 GETA$:IFA$=""THEN10008
10009 GOTO11
```



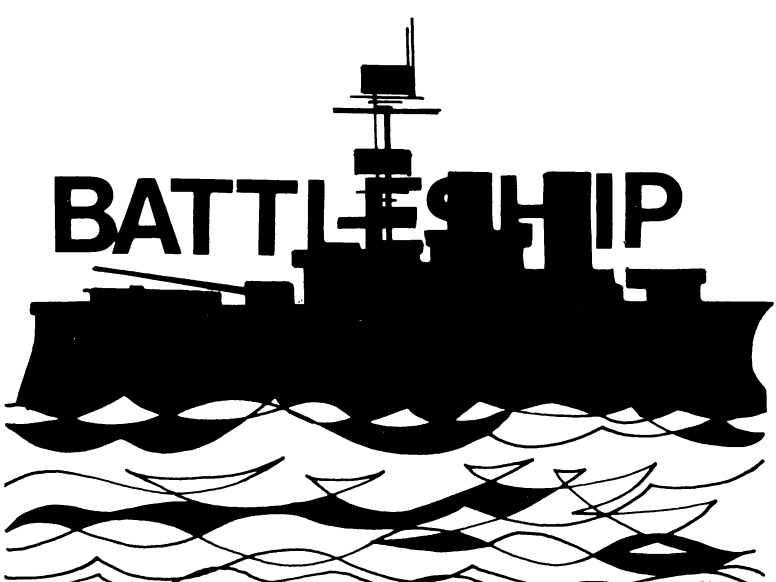
BATTLESHIP

In this new version of the old favourite you have to hunt out and destroy the enemy's fleet. The computer is the enemy, and it's pretty clever so watch out!

The computer will place its fleet at random, and when it has completed this task you are allowed to place your fleet.

The computer will show you the shape and angle of the ship, and it is up to you to place this where you want. When you have chosen the position for your vessel, the program will check that you have not placed it on top of some other ship.

The computer is not as clever as it could be, so this gives you more chance of winning the game.

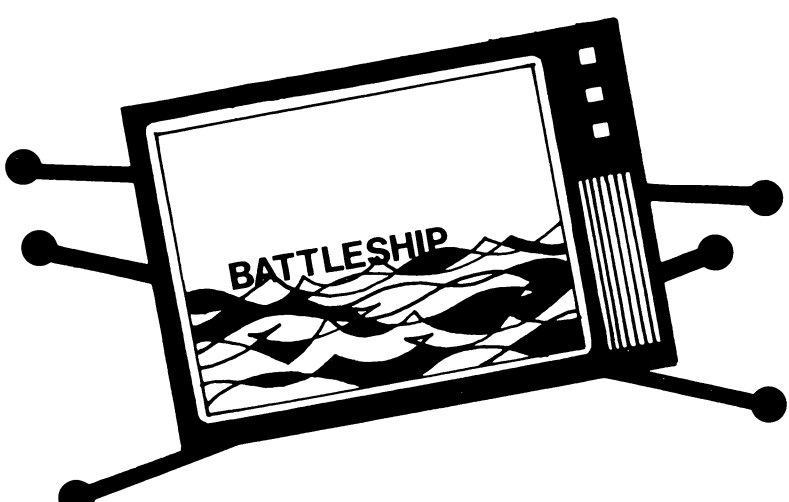


```
5 DIM A(10,10),B(10,10)
20 S=54272
30 POKES+5,0:POKES+6,249:POKES+3,10
40 POKES+24,79:POKES+23,1:POKES+1,0:POKE
S,0
50 GOSUB10000:PRINT"sPSETTING UP"
55 POKE53281,7:POKE53280,11
57 FORC=1TO10:FORD=1TO10:A(C,D)=0:B(C,D)
=0:NEXT:NEXT
58 PRINT"QQPLACING MY PIECES"
59 FORL=1TO2
60 F=INT(10*RND(1)+1):E=INT(9*RND(1)+1)
61 IFA(E,F)=1ORA(E+1,F)=1THEN60
62 A(E,F)=1:A(E+1,F)=1
63 NEXT
65 E=INT(10*RND(1)+1):F=INT(6*RND(1)+1)
66 IFA(E,F)=1ORA(E,F+1)=1ORA(E,F+2)=1ORA
(E,F+3)=1ORA(E,F+4)=1THEN65
67 A(E,F)=1:A(E,F+1)=1:A(E,F+2)=1:A(E,F+
3)=1:A(E,F+4)=1
68 E=INT(10*RND(1)+1):F=INT(8*RND(1)+1)
69 IFA(E,F)=1ORA(E,F+1)=1ORA(E,F+2)=1THE
N68
70 A(E,F)=1:A(E,F+1)=1:A(E,F+2)=1
71 F=INT(10*RND(1)+1):E=INT(7*RND(1)+1)
72 IFA(E,F)=1ORA(E+1,F)=1ORA(E+2,F)=1ORA
(E+3,F)=1THEN71
73 A(E,F)=1:A(E+1,F)=1:A(E+2,F)=1:A(E+3,
F)=1
75 PRINT"sQALL MY PIECES ARE PLACED"
76 PRINT"QNOW PLACE YOUR PIECES"
78 PRINT"QQ        123456789q10⌂0"
79 PRINT"       1Q⌂2Q⌂3Q⌂4Q⌂5Q⌂6Q⌂7Q⌂8Q⌂9
Q⌂⌂10"
80 PRINT"PIECE TO BE PLACED    R''rQ"
81 FORL=1TO2
83 INPUT"qINPUT LEFT CORNER POSITION (1,
1)      ⌂⌂⌂⌂⌂⌂";G,H
84 IFG<1ORG>9ORH<1ORH>10THEN83
85 IFB(G,H)=1ORB(G+1,H)=1THEN83
86 B(G,H)=1:B(G+1,H)=1
```

```
88 POKE1270+G+H*40,42:POKE55542+G+40*H,0
89 POKE1271+G+H*40,42:POKE55543+G+40*H,0
90 NEXT
91 PRINT"qqPIECE TO BE PLACED    R''''rQ

92 INPUT"qINPUT LEFT CORNER POSITION (1,
1)       ";G,H
93 IFG<1ORG>7ORH<1ORH>1ØTHEN92
94 IFB(G,H)=1ORB(G+1,H)=1ORB(G+2,H)=1ORB
(G+3,H)=1THEN92
95 B(G,H)=1:B(G+1,H)=1:B(G+2,H)=1:B(G+3,
H)=1
96 POKE1270+G+H*40,42:POKE55542+G+40*H,0
:POKE1271+G+40*H,42:POKE55543+G+40*H,0
97 POKE1272+G+H*40,42:POKE55544+G+40*H,0
:POKE1273+G+40*H,42:POKE55545+G+40*H,0
98 PRINT"qqPIECE TO BE PLACED          R$
qq$qq$qq$qq$qrQQQQQQ"
99 INPUT"qINPUT LEFT CORNER POSITION (1,
1)       ";G,H
100 IFG<1ORG>1ØORH<1ORH>6THEN99
101 IFB(G,H)=1ORB(G,H+1)=1ORB(G,H+2)=1OR
B(G,H+3)=1ORB(G,H+4)THEN99
102 B(G,H)=1:B(G,H+1)=1:B(G,H+2)=1:B(G,H
+3)=1:B(G,H+4)=1
103 FORK=ØTO4:POKE1270+4Ø*H+G+K*4Ø,42:PO
KE55542+4Ø*H+G+K*4Ø,Ø:NEXT
104 PRINT"qqPIECE TO BE PLACED          R
$qq$qq$qrq qq qqQQQQQQ"
105 INPUT"qINPUT LEFT CORNER POSITION (1
,1)       ";G,H
106 IFG<1ORG>1ØORH<1ORH>8THEN105
107 IFB(G,H)=1ORB(G,H+1)=1ORB(G,H+2)=1TH
EN105
108 B(G,H)=1:B(G,H+1)=1:B(G,H+2)=1
109 FORK=ØTO2:POKE1270+4Ø*H+G+K*4Ø,42:PO
KE55542+4Ø*H+G+K*4Ø,Ø:NEXT
110 PRINT"QQNOW THE GAME WILL START":P=Ø
:HI=Ø
112 PRINT"I WILL GO FIRST"
114 PRINT"PRESS ANY KEY TO BEGIN"
```

```
115 POKE198,0:WAIT198,1:POKE198,0:PRINT"
S"
117 E=INT(10*RND(1)+1):F=INT(10*RND(1)+1
)
118 IFB(E,F)=2THEN117
119 PRINT"SQQMY GUESS IS "E;F
120 PRINT"WHICH IS A ";
121 IFB(E,F)=1THENPRINT"HIT":HI=HI+1:GOT
O123
122 PRINT"MISS"
123 IFHI=16THEN150
124 B(E,F)=2
125 PRINT"QYOUR GO NOW"
126 INPUT"TYPE YOUR GUESS ?";I,J
127 IFI<1ORI>10ORJ<1ORJ>10THEN126
128 IFA(I,J)=2THENPRINT"YOU HAVE TRIED T
HAT SQUARE ":GOTO126
129 IFA(I,J)=0THENPRINT"YOU HAVE SCORED
A MISS":GOTO131
130 PRINT"YOU HAVE SCORED A HIT":P=P+1:I
FP=116THEN160
131 A(I,J)=2:FORN=1TO1000:NEXT:GOTO117
150 PRINT"I HAVE SUNK YOUR FLEET"
151 PRINT"QDO YOU WANT ANOTHER GO (Y/N)
?"
152 GETA$:IFA$=""THEN152
153 IFA$="Y"THENRUN
154 IFA$="N"THENEND
155 GOTO152
160 PRINT"YOU HAVE SUNK MY FLEET"
161 PRINT"QDO YOU WANT ANOTHER GO (Y/N)
?"
162 GOTO152
9999 END
10000 GOTO10018:POKE53280,1:PRINT"SINSTR
UCTIONS"
10001 PRINT"QQBATTLESHIPS"
10002 PRINT"YOU HAVE TO HIDE YOUR FLEET
OF SHIPS "
10003 PRINT"ON A 10 BY 10 GRID HOPEFULLY
 IN PLACES"
```

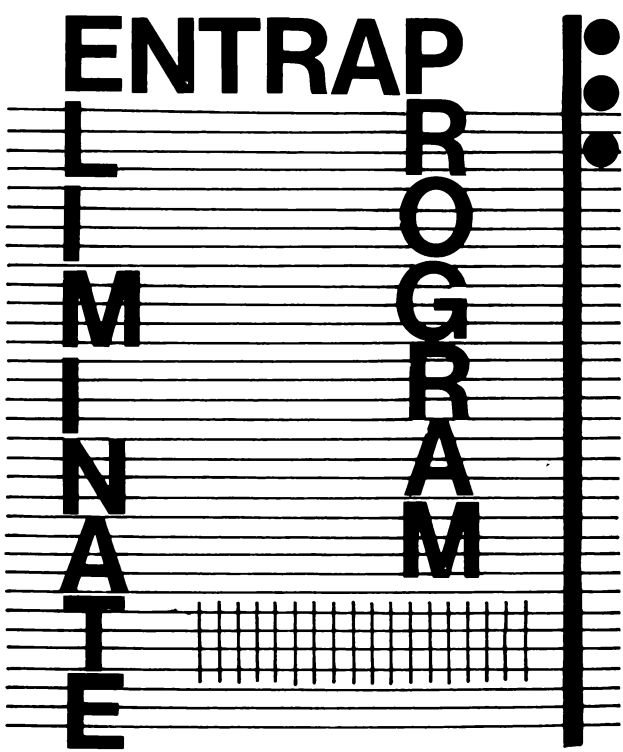
```
10004 PRINT"WHERE THE COMPUTER WILL NOT
GUESS."
10005 PRINT"QTHE COMPUTER WILL DO THE SA
ME WITH ITS"
10006 PRINT"FLEET AND YOU HAVE TO GUESS
WHERE IT IS"
10007 PRINT"HIDDEN."
10008 PRINT"QQYOU TYPE IN THE GUESS X,Y"
10011 PRINT"QPRESS ANY KEY TO START"
10012 POKES+4,65:FORL=1TO8
10013 FORA=0TO100STEP2:POKES+22,A:POKES+
1,A:NEXT:NEXT:POKES+4,64
10014 FORL=1TO1000:NEXT:POKES+1,0:POKES,
0
10018 POKE198,0:WAIT198,1:POKE198,0:RETU
RN
```



ENTRAP

You have to be good at thinking and reacting quickly in this game or it will end in your immediate elimination!

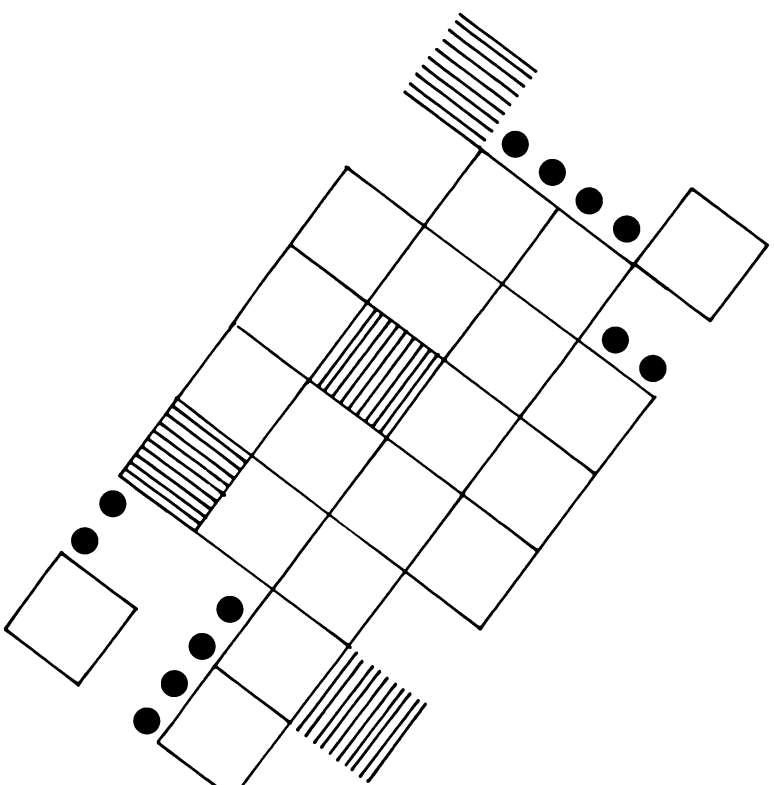
It is a game to be played by two people at the same time against one another. In a battle of tactics and sneaky manoeuvering, the idea is to try and trap, surround, or just outlast your opponent. This may be easy at first but the more you improve, the harder and better the game gets. Many a sleepless night has been caused by this game, so beware.



```
5 P1=1514:P2=1533:C1=1:C2=6:FORA=0TO3:RE
ADDI(A),M(A):NEXT:D1=3:D2=4:DI(4)=0
10 GOSUB10000
20 POKE53281,3:POKE53280,7
21 GOSUB400
22 PRINT"s";:GOSUB300
24 L=PEEK(56320)
25 IFL=126THEND1=0:GOTO29
26 IFL=125THEND1=2:GOTO29
27 IFL=123THEND1=3:GOTO29
28 IFL=119THEND1=1:GOTO29
29 POKES+1,20:POKES+4,65:POKES+4,64
30 K=PEEK(203)
31 IFK=33THEND2=0:GOTO35
32 IFK=34THEND2=2:GOTO35
33 IFK=10ORK=40=10THEND2=3:GOTO35
34 IFK=13ORK=40=13THEND2=1:GOTO35
35 POKES+1,10:POKES+4,65:POKES+4,64
40 P1=P1+DI(D1)
41 P2=P2+DI(D2):IFPEEK(P1)<>32ANDPEEK(P2
)<>32THEN230
42 IFPEEK(P1)<>32THEN100
43 IFPEEK(P2)<>32THEN200
44 POKEP1,204:POKE54272+P1,C1
45 POKEP2,204:POKE54272+P2,C2
46 FORZ=1TO50:NEXT:GOTO24
100 FORA=1TO4:FORB=2TO12:POKEP1,M(B AND
3):POKE53281,B:POKES+1,25-B*2
101 POKES+4,129:FORN=1TO10:NEXT
102 NEXT:POKE53280,A:NEXT:POKES+6,250:PO
KES+4,128
103 PRINT"PLAYER 1 HAS CRASHED"
104 PRINT"DO YOU WANT ANOTHER GAME (Y/N)
?"
105 GETA$:IFA$=""THEN105
106 IFA$="Y"THEN20
107 IFA$="N"THENEND
108 GOTO105
200 FORA=1TO4:FORB=2TO12:POKEP2,M(B AND
3):POKE53281,B:POKES+1,25-B*2
201 POKES+4,129:FORN=1TO10:NEXT
```

```
202 NEXT:POKE53280,A:NEXT:POKES+6,250:PO
KES+4,128
203 PRINT"PLAYER 2 HAS CRASHED"
204 PRINT"DO YOU WANT ANOTHER GAME (Y/N)
?"
205 GOTO105
230 FORA=1TO4:FORB=2TO12:POKEP2,M(B AND
3):POKE53281,B:POKES+1,25-B*2
231 POKEP1,M(B AND 3):POKE53281,B:POKES+
1,25-B*2
232 POKES+4,129:FORN=1TO10:NEXT
233 NEXT:POKE53280,A:NEXT:POKES+6,250:PO
KES+4,128
234 PRINT"BOTH PLAYERS HAVE CRASHED"
235 PRINT"DO YOU WANT ANOTHER GAME (Y/N)
?"

236 GOTO105
300 FORA=0TO24:POKE1024+40*A,160:POKE552
96+40*A,5:POKE1063+40*A,160
301 POKE55335+40*A,5:POKES+1,A:POKES+4,3
3:NEXT:FORX=1TO100:NEXT
302 FORA=1TO38:POKE1024+A,160:POKE1984+A
,160:POKE55296+A,5:POKE56256+A,5
303 POKES+4,65:POKES+1,40-A:NEXT:POKES+4
,128
305 RETURN
400 S=54272
401 POKES+24,15:POKES+5,0:POKES+6,244:PO
KES+3,10:POKES+2,100
402 POKES,0:POKES+1,0
403 P1=1514:P2=1533:D1=3:D2=1:RETURN
9999 END

10000 POKE53281,3:PRINT"spINSTRUCTIONS"
10002 PRINT"QTHIS IS A GAME FOR TWO PLAY
ERS, AND "
10004 PRINT"EACH PLAYER HAS TO TRY AND T
RAP THE "
10006 PRINT"OTHER PLAYER IN A CORNER OR
WITH HIS "
10008 PRINT"TAIL.
```

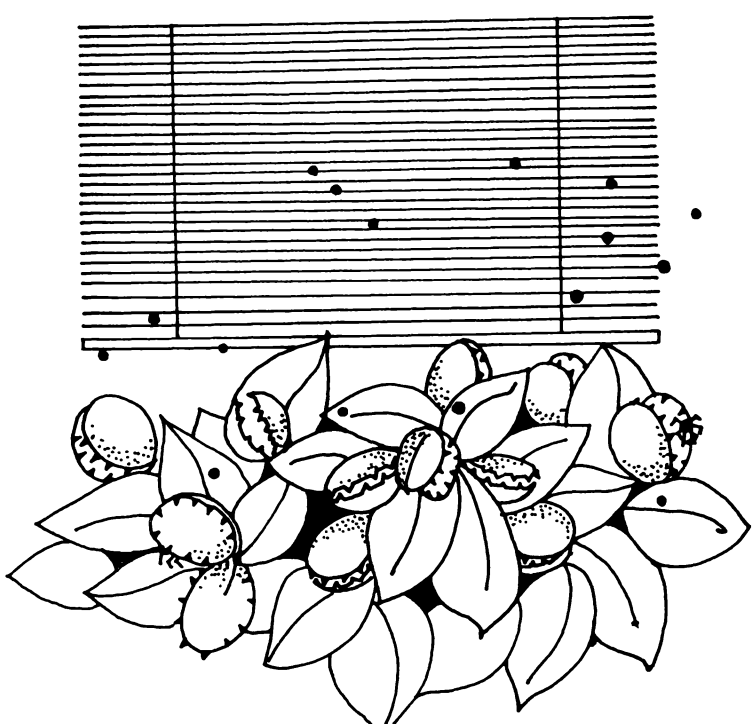
```
10010 PRINT"QONE PLAYER MUST USE A JOYST
ICK AND THE"
10012 PRINT"OTHER THE KEYBOARD."
10014 PRINT"QCONTROLS"
10016 PRINT"Q              JOYSTICK (PORT 2)

10018 PRINT"Q  UP-I      DOWN-J     LEFT-A
    RIGHT-S"
10019 PRINT"PLAYER 1 (JOYSTICK) STARTS O
N THE LEFT"
10020 PRINT"QPRESS ANY KEY TO START"
10025 POKE198,0:WAIT198,1:POKE198,0:RETU
RN
10030 DATA-40,66,1,77,40,64,-1,78
```



FLYTRAP

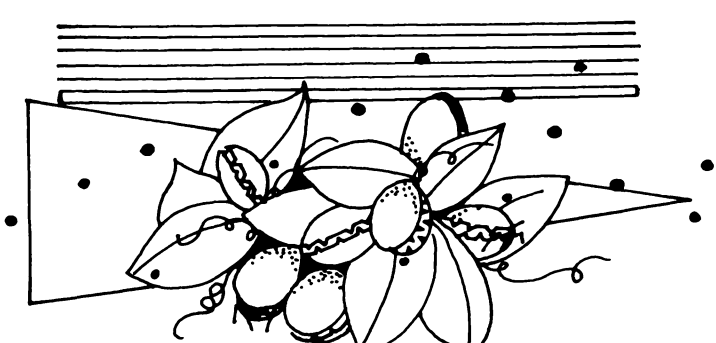
This game was inspired by the Venus Fly Trap plant, though with one difference. The plant waits for the fly to walk into its jaws, but in this version the plant has to catch its dinner in a more energetic manner.

The fly will buzz backwards and forwards at random heights. It is up to you to guess this height and, by pressing the right key at exactly the right time, help the poor plant to survive. The object of the game is to catch all ten flies in as little time as possible.



```
2 C$="SQQQQQQQQQQQQQQQQQQQQQQQQ]]]]]]]]]
]]]]]]]]]]↑R 9⌂ 9⌂ r9⌂⌂"
10 D=1:O=0:GOSUB10000
12 POKE53281,11:POKE53280,12
15 V$(0)="↑r□R r)9⌂⌂⌂⌂ R _ r 9⌂⌂⌂⌂⌂ R)_□
r 9⌂⌂⌂⌂      QQQQ⌂⌂⌂⌂⌂"
16 V$(1)="y⌂r □R r) 9⌂⌂⌂⌂⌂□R□r R)r)9⌂⌂⌂^
⌂R)r    R□9⌂⌂⌂⌂⌂r       QQQ⌂⌂⌂⌂"
17 FORA=0TO10:PRINT"sQQQ]]]]]]]]]]]]]]]]
]]"V$(A AND 1):FORJ=1TO200:NEXT
18 NEXT:PRINT"s":PRINTC$;V$(0):TI$="0000
00"
19 FORK=1TO10:H=INT(8*RND(1)+1):PO=1024+
80*H:D=1:O=0
20 POKEPO,32:O=O+D:IFO<10ORO>39THEND=-D
21 PO=PO+D:POKEPO,42:POKEPO+54272,0
30 GETA$:IFA$=""THEN100
31 T$=V$(1):A=VAL(A$):IFA=0THEN100
32 PRINTC$;:FORB=1TO2*A:PRINT"↑r R r ⌂⌂^
9";T$;:FORG=1TO50:NEXTG,B
33 POKEP,42:IFPEEK(PO+1)=233ANDPEEK(PO-1
)=223THEN110
34 T$=V$(0)
35 PRINTC$;LEFT$("9999999999999999999999
9",B);:FORB=2*ATO0STEP-1
36 FORG=1TO30:NEXT:PRINT"↑r R r ⌂⌂⌂";T$;
:NEXT
100 GOTO20
110 T$=V$(0):PRINTC$;LEFT$("999999999999
99999999999",B);:FORB=2*ATO0STEP-1
111 FORG=1TO30:NEXT:PRINT"↑r R r ⌂⌂⌂";T$
;:NEXT:NEXT
115 PRINT"S□WELL DONE YOU HAVE EATEN THE
 TEN FLIES "
116 PRINT"IN A TIME OF";TI/60;"SECONDS"
118 PRINT"QQDO YOU WANT ANOTHER GO (Y/N)
?"
120 GETA$:IFA$=""THEN120
121 IFA$="Y"THEN11
122 IFA$="N"THENEND
123 GOTO120
```

```
9999 END
10000 POKE53281,7:PRINT"s←                    IN
STRUCTIONS"
10001 PRINT"QYOU ARE A VENUS FLY TRAP AN
D YOU HAVE"
10002 PRINT"TO TRY AND CATCH THE FLIES T
HAT ARE "
10003 PRINT"FLYING ABOVE."
10004 PRINT"QTHE FLIES FLY AT DIFFERENT
HEIGHTS AND"
10005 PRINT"USING THE KEYS 1 TO 9 YOU HA
VE TO JUDGE"
10006 PRINT"VERY QUICKLY HOW HIGH THE FL
Y IS AND "
10007 PRINT"WHEN THE FLY IS DIRECTLY ABO
VE YOU LEAP"
10008 PRINT"UP AND EAT THE FLY"
10009 PRINT"QQYOU HAVE TO EAT TEN FLIES
IN AS LITTLE  TIME AS POSSIBLE TO WIN"
10010 PRINT"QQPRESS ANY KEY TO START"
10011 POKE198,0:WAIT198,1:POKE198,0:RETU
RN
```



How To Write Better Programs

How To Write Better Programs

By Tim Hartnell, series editor

There are a number of fine programs in this book, and many of the regular computer magazines contain other such ones. But no matter how good the programs from published sources are, you are certain to get more pleasure from running them if they have been partially or completely written by you. Putting your personal stamp on programs, altering them to reflect your wishes and creativity, is an excellent way to improve the programs, and eventually, of course, you'll become a better and more imaginative programmer.

Programs in magazines, and in books like this one, are ideal as starting points for your own developments. You may also find that advertisements for software packages can be fruitful 'idea-starters'. You only need to read the description of what the commercially available program does, and you will have the first step towards creating your own program. You have to be careful, of course, not to infringe copyright either in the screen displays, in the name of the program, or the names of the 'characters' within the program. However, you will probably find that at a certain point in its development the program will take on a life of its own, growing and evolving away from the original scenario, until you eventually have a completely new game concept and implementation.

Whatever you do, be careful not to pass off other people's work as your own. By all means adapt and im-

prove published programs, but do not then present them to magazines as if they were originals. I have lost count of the number of times one of my own programs, from one of my books, has been submitted to me for publication.

Always watch out for new ideas as you look through books, game and computer magazines, or wander through video game arcades. It may be worth keeping notes of ideas you come across for games, for character shapes, for sounds, for dramatic endings and so on. Thus you will never be short of ideas, and you will also be able to merge the material together to produce better games which hold the player's attention for longer.

Games tend to fall into one of three categories, and it is worth making sure of the category into which your proposed program will fall *before* you start to program, since the category of game materially alters the programming approach. This is not to say that, as you develop a program, it will not move from one category into another, nor that a particular game might not extend across two categories, but it is nevertheless useful to keep the various groups separate in your mind, just to clarify your thoughts. The three categories are:

1. Board games
2. 'Arcade' (that is, highly visual, fast moving, noisy, real time) games
3. Games of chance (such as Roulette and Snap).

In board games, the quality of play is more important than lightning-fast response, while the arcade-type programs must be kept moving at all costs, even if some 'intelligence' from your Martian intruders must be sacrificed to achieve this. Games of chance depend more on their ease of play ('user-friendly' inputs), and an approach to true randomness, than do either of the other categories.

You will find that games programs tend to fall into types, which are subdivisions of the three above mentioned categories. Many board games are variants of chess or checkers; many arcade games started off life as Space Invader-type games; and games of chance

started off in the 'real world' of dice and cards. Looking at a program description, or a games machine, and trying to categorise the game you see can help trigger new ideas which fit within that particular game's genre.

There is a school of thought within programming — generally called 'structured programming' — which believes that discipline at the beginning of the games-writing process is essential. While less interesting than sitting down at the computer right away, a much better program is produced in the end. I once wrote a program called Dome Dweller, a simulation program in which the player is in charge of a 'lunar dome' and must decide which products to manufacture and sell in order to buy oxygen and food for the station's inhabitants. (This program was used in my book *The Book of Listings*, written with Jeremy Ruston, and published by the BBC.)Once I had decided the overall scenario, I worked out the screen display, and came up with an idea as follows:

Oxygen supplies are low
There are 96 people living within your dome in year 3
Money credit is $5,693
Annual maintenance charge is $226
Oxygen tanks hold 811 units
Oxygen costs $8 per unit
Each dome dweller needs 5 units a year
Food stocks stand at 2122
Each dweller needs 3 units a year ($6 each, $576 for dome. This will last 7 years at present population.)
You can trade your unique lunar sculptures with the people who live in other domes. You use up 2 units of oxygen making each one, and sell them for $30.

As you can probably guess from this 'sample printout', the idea of the program is to decide how many 'unique lunar sculptures' you must make and sell in order to buy oxygen and food, and to pay the 'annual maintenance' charge. The problem with this particular program is that

making each sculpture uses up oxygen, so you must balance your wish to make money against the need to use the oxygen intelligently.

You may well wish to try writing such a program yourself. You should end up with an enjoyable program, and writing it will do much to help you develop your programming skills. The first thing to do is to make a list of what the program has to do:

Set up the needed variables
Tell the player the 'state of the dome'
Ask how much oxygen to be bought
Check if can afford this, if so buy it, if not go back and ask again
Ask how much food to be bought
Check if can afford this, if so buy it, if not go back and ask again
Update oxygen quantity
Update food quantity
Reduce money left total
Ask how many items of sculpture to be made
Check if there is enough oxygen to make this many, if not go back and ask again
Reduce oxygen quantity by amount needed to make the number of sculptures specified, increase money total to reflect value of sculptures made
Increase the population total slightly, add one to the 'current year'
Check if there is enough food in stocks to feed whole population
Check if there is enough oxygen for whole population
Check if there is any money
If any of these conditions are negative (eg not enough food) send action to an 'end of game' routine
If all are positive, loop back to tell the player the state of the dome, and continue to circle

You could probably write a Dome Dweller program

using the list above, together with the 'sample printout' information. There is, however, a secret I should like to share with you which unlocks programming problems almost instantly. You can actually write all the vital parts of a program in minutes, so you can see the raw framework of a program like this running long before you fill in the details. And once you have a framework you can work on it for as long as you like, knowing as you do so that — at every moment in program development — you have a working program. You do not have to wait until the end until you can run it to see how you are going. The 'secret' is to hold the entire program within a series of subroutine calls, all held within a perpetual loop. Here's how it could work with this program. The very first lines you enter in your computer are as follows:

```
10 REM DOME DWELLER
20 GOSUB 1000: REM ASSIGN VARIABLES
30 GOSUB 2000: REM PRINT OUT STATE OF
DOME
40 GOSUB 3000: REM OXYGEN
50 GOSUB 4000: REM FOOD
60 GOSUB 5000: REM SCULPTURE
70 GOSUB 6000: REM UPDATE POPULATION
80 GOSUB 7000: REM CHECK ON STATE OF
 DOME
90 IF (all conditions positive, from GOSUB 7000)
THEN GOTO 30
100 REM End of game ...
```

As you can see once you have the 'master loop' set up in this way, it is relatively simple to fill in each of the subroutines one by one, testing each as you do so, and elaborating each one so that you end up eventually with a very good program. The only thing you need now is a list of the variables which you will use with the program.

I find the best way to do this is to use explicit names for variables so that when you are programming you do not have to spend time checking, for example, whether AA stands for the population, or the number of units of oxygen used up in making each item of sculpture. To make

programs as easy as possible to transfer between different computers you can stick to two letter variable names, or you can take advantage (if your computer allows it) of long names (such as OXYUSE for the amount of oxygen used) for variables. Then you have no doubts whatsoever as to the meaning of each variable name. To show how this can work, and to illustrate a further advantage of explicit variable names, here are the variables used in Dome Dweller:

FOLK — population of dome
CASH — money in treasury
FOOD — food stocks on hand
FOODCOST — how much each unit of food costs
FOODNEED — how many units of food were consumed per person per year
ARTCOST — how much oxygen was used up making each piece of sculpture
ARTPAY — how many dollars each piece of sculpture was sold for
OXY — oxygen stocks on hand
OXYNEED — how many units of oxygen were consumed per person per year
OXYCOST — how much each unit of oxygen cost to buy
REPAIR — the cost of annual repairs to the dome
YEAR — the year of the dome's life

Using explicit variable names in this way — although they use up more memory than do single or double-letter variable names — makes it very simple to follow through a program, working out what each section of the program actually does. Moreover, and this is the further advantage mentioned, it is very easy when writing the program to insert the formulae required for calculations. By this I mean that if, for example, you wished to include (as I do in this program) an indication of how much oxygen is needed for each year, you simply multiply the number of people in the dome (FOLK) by the number of oxygen units each person needs each year (OXYNEED). You can then

include within the printouts for the state of the dome a line like:

```
PRINT "THERE ARE ";FOLK;" IN THE DOME"
PRINT "IN YEAR ";YEAR
PRINT "EACH PERSON NEEDS ";OXYNEED;"
      UNITS OF"
PRINT "OXYGEN EACH YEAR,";
      OXYNEED*FOLK;" NEEDED"
PRINT "FOR THE WHOLE DOME"
```

It also makes it very easy to check on whether purchases are possible. For example, to buy food, you could say:

```
PRINT "HOW MUCH FOOD WILL YOU BUY?"
INPUT A
IF A*FOODCOST > CASH THEN GOTO (get
another A)
```

So the suggestions given here for improving your programs by the use of 'structured programming' include the following:

- draw up a sample printout, or mock-up of the final screen display
- draw up a list of what the program has to do each time through a 'master control loop'
- change this list to a series of subroutine calls
- use explicit variable names if possible

It is useful if you are designing programs for others to use to ensure that it is quite clear what the player should do when running the program. There is little point, especially when memory is limited, in including a long set of instructions within the program, but you should certainly write such instructions down. In addition, user prompts should be explicit (such as ENTER THE NUMBER OF GOES YOU WANT) and should include warnings of the limits which will be placed on the input (HOW MANY CARDS WILL YOU START WITH: 1, 2 OR 3 ?, for instance).

You cannot assume that you will be present every time a program is run, so you should do your best to make it as foolproof as possible. If you can, add error-trapping routines to the program to ensure that a mistake in enter-

ing a choice earlier on in the program will not cause it to crash or come up with stupid results later on.

If you read through this section of the book several times and try to apply the ideas to your own programming work, you should find your work quality improves significantly, and also that you can spend more time improving and embellishing a program and less in the raw mechanical task of getting the thing running.

GLOSSARY

GLOSSARY

A

Accumulator — the place within the computer in which arithmetic computations are performed and where the results of these computations are stored.

Algorithm — the series of steps the computer follows to solve a particular problem.

Alphanumeric — this term is usually used in relation to a keyboard, as in 'it is an alphanumeric keyboard', which means that the keyboard has letters as well as numbers. It is also used to refer to the 'character set' of the computer. The character set comprises the numbers and letters the computer can print on the screen.

ALU (Arithmetic/Logic Unit) — the part of the computer which does arithmetic (such as addition, subtraction) and where decisions are made.

AND — a Boolean logic operation that the computer uses in its decision-making process. It is based on Boolean algebra, a system developed by mathematician George Boole (1815-64). In Boolean algebra the variables of an expression represent a logical operation such as OR and NOR.

ASCII — stands for American Standard Code for Information Exchange, the most widely used encoding system for English language alphanumerics. There are 128 upper and lower case letters, digits and some special characters. ASCII converts the symbols and control instructions into seven-bit binary combinations.

Assembler — a program which converts other programs written in assembly language into machine code

(which the computer can understand directly). Assembly language is a low level programming language which uses easily memorised combinations of two or three letters to represent a particular instruction which the assembler then converts so the machine can understand it. Examples of these are ADD (add), and SUB (subtract). A computer programmed in assembly language tends to work more quickly than one programmed in a higher level language such as BASIC.

B

BASIC — an acronym for Beginners All-Purpose Symbolic Instruction Code. It is the most widely used computer language in the microcomputer field. Although it has been criticised by many people, it has the virtue of being very easy to learn. A great number of BASIC statements resemble ordinary English.

Baud — named after Baudot, a pioneer of telegraphic communications. Baud measures the rate of transfer of information and is approximately equal to one bit per second.

BCD — an abbreviation for Binary Coded Decimal.

Benchmark — a test against which certain functions of the computer can be measured. There are a number of so-called 'standard Benchmark tests', but generally these only test speed. This is rarely the aspect of a microcomputer that is most of interest to the potential buyer.

Binary — a numbering system that uses only zeros and ones.

Bit — an abbreviation for Binary Digit. This is the smallest unit of information a computer circuit can recognise.

Boolean Algebra — the system of algebra developed by mathematician George Boole which uses algebraic notation to express logical relationships (see AND).

Bootstrap — a short program or routine which is read into the computer when it is first turned on. It orients the computer to accept the longer, following program.

Bug — an error in a computer program which stops the program from running properly. Although it is generally used to mean only a fault or an error in a program, the term bug can also be used for a fault in the computer hardware.

Bus — a number of conductors used for transmitting signals such as data instructions, or power in and out of a computer.

Byte — a group of binary digits which make up a computer word. Eight is the most usual number of bits in a byte.

C

CAI — Computer Assisted Instruction.

CAL — Computer Assisted Learning. The term is generally used to describe programs which involve the learner with the learning process.

Chip — the general term for the entire circuit which is etched onto a small piece of silicon. The chip is, of course, at the heart of the microcomputer.

Clock — the timing device within the computer that synchronises its operations.

COBOL — a high level language derived from the words Common Business Orientated Language. COBOL is designed primarily for filing and record-keeping.

Comparator — a device which compares two things and produces a signal related to the difference between the two.

Compiler — a computer program that converts high level programming language into binary machine code so the computer can handle it.

Complement — a number which is derived from another according to specified rules.

Computer — a device with three main abilities or functions:
1) to accept data
2) to solve problems
3) to supply results

CPU — stands for Central Processing Unit. This is the heart of the computer's intelligence, where data is handled and instructions are carried out.

Cursor — a character which appears on the TV screen when the computer is operating. It shows where the next character will be printed. On a computer there are usually 'cursor control keys' to allow the user to move the cursor around the screen.

D

Data — information in a form which the computer can process.

Debug — the general term for going through a program and correcting any errors in it, that is, chasing down and removing bugs (see Bug).

Digital Computer —a computer which operates on information which is in a discrete form.

Disk/Disc — this is a magnetically sensitised plastic disk, a little smaller than a single play record. This is used for storing programs and for obtaining data. Disks are considerably faster to load than a cassette of the same length program. The disk can be searched very quickly while a program is running for additional data.

Display — the visual output of the computer, generally on a TV or monitor screen.

Dot Matrix Printer — a printer which prints either the listing of a program or that which is displayed on the TV screen. Each letter and character is made up of a number of dots. The higher the number of dots per character the finer the resolution of the printer.

Dynamic Memory — a memory unit within the computer which 'forgets' its contents when the power is turned off.

E

Editor — this term is generally used for the routine within the computer which allows you to change lines of a program while you are writing it.

EPROM — stands for Erasable Programmable Read-Only Memory. This is like the ROM in the computer, except that it is fairly easy to load material into an EPROM and it doesn't disappear when you turn the power off. EPROMs must be placed in a strong ultra violet light to erase them.

Error Messages — the information given by a computer where there is a fault in the coding during a part of a program, usually shown by the computer stopping, and printing a word, or a word and numbers, or a combination of numbers only, at the bottom of the screen. This tells you what mistake has been made. Common mistakes include using the letter O instead of zero in a line, or leaving out a pair of brackets, or one of the brackets, in an expression, or failing to define a variable.

F

File — a collection of related items of information organised in a systematic way.

Floppy Disk — a relatively cheap form of magnetic disk used for storing computer information, and so named because it is quite flexible (see Disk/Disc).

Flow Chart — a diagram drawn up before writing a program, in which the main operations are enclosed within rectangles or other shapes and connected by

lines, with arrows to represent loops, and decisions written at the branches. It makes writing a program much easier because traps such as infinite loops, or non-defined variables can be caught at an early stage. It may not be worth writing a flow chart for very short programs, but generally a flow chart aids in creating programs.

Firmware — there are three kinds of 'ware' in computers: software 'temporary' programs; hardware like the ROM which contains permanent information; and firmware in which the information is relatively permanent, as in an EPROM (see EPROM).

Flip-Flop — a circuit which maintains one electrical condition until changed to the opposite condition by an input signal.

FORTRAN — an acronym for FORmula TRANslation, this is a high level, problem orientated computer language for scientific and mathematical use.

G

Gate — an electrical circuit which, although it may accept one or more incoming signals, only sends out a single signal.

Graphics — pictorial information as opposed to letters and numbers.

H

Hard Copy — computer output which is in permanent form.

Hardware — the physical parts of the computer (also see software and firmware).

Hexadecimal (Hex) — a numbering system to the base sixteen. The digits zero to nine are used, as well as the letters A, B, C, D, E and F to represent numbers. A

equals 10, B equals 11, C equals 12, and so on. Hex is often used by microprocessor users.

Hex Pad — a keyboard designed specifically for entering hexadecimal notation.

High Level Language — a programming language which allows the user to talk to the computer more or less in English. In general, the higher the level of the language (that is, the closer it is to English), the longer it takes for the computer to translate it into a language it can use. Lower level languages are far more difficult for human operators but are generally executed far more quickly.

I

Input — the information fed into the computer via a keyboard, a microphone, a cassette or a disk.

Input/Output (I/O Device) — a device which accepts information or instructions from the outside world, relays it to the computer, and then, after processing, sends the information out in a form suitable for storing, or in a form which could be understood by a human being.

Instruction — data which directs a single step in the processing of information by the computer (also known as a command).

Integrated Circuit — a complete electronic circuit imprinted on a semiconductor surface.

Interface — the boundary between the computer and a peripheral such as a printer.

Interpreter — a program which translates the high level language fed in by the human operator, into a language which the machine can understand.

Inverter — a logic gate that changes the signal being fed in, to the opposite one.

Interactive Routine — part of a program which is repeated over and over again until a specified condition is reached.

J

Jump Instruction — an instruction which tells the computer to go to another part of the program, when the destination of this move depends on the result of a calculation just performed.

K

K — this relates to the size of the memory. Memory is usually measured in 4K blocks. 1K contains 1,024 bytes.

Keyword — the trigger word in a line of programming, usually the first word after the line number. Keywords include STOP, PRINT and GOTO.

L

Language — computer languages are divided into three sections: high level languages, such as BASIC, which are reasonably close to English and fairly easy for humans to use; low level languages, such as Assembler, that use short phrases which have some connection with English (ADD for add and RET for return, for instance); and machine code which communicates more or less directly with the machine.

LCD — this stands for Liquid Crystal Diode. Some computers such as the TRS-80 Pocket Computer use an LCD display.

LED — this stands for Light Emitting Diode. The bright

red numbers which are often used on watch or clock displays are made up of LEDs.

Logic — the mathematical form of a study of relationships between events.

Loop — a sequence of instructions within a program which is performed over and over again until a particular condition is satisfied.

M

Machine Language or Machine Code — an operation code which can be understood and acted upon directly by the computer.

Magnetic Disk — see Disk and Floppy Disk.

Mainframe — computers are generally divided into three groups, and the group a computer falls into depends more or less on its size. The computer you are thinking of buying is a microcomputer; medium sized computers are known as minicomputers; and the giant computers that you sometimes see in science fiction movies are mainframe computers. Until 15 years ago mainframe computers were, in practical terms, the only ones available.

Memory — there are two types of memory within a computer. The first is called ROM (read-only memory); this is the memory that comes already programmed on the computer, which tells the computer how to make decisions and how to carry out arithmetic operations. This memory is unaffected when you turn the computer off. The second type is RAM (random access memory). This memory holds the program you type in at the keyboard or send in via a cassette or disk. In most computers the computer 'forgets' what is in RAM when you turn the power off.

Microprocessor — the heart of any computer. It requires peripheral unit interfaces, such as a power supply and input and output devices, to act as a microcomputer.

MODEM — stands for Modulator Demodulator. This is a device which allows two computers to talk to each other over the telephone. The computers usually use a cradle in which a telephone receiver is placed.

Monitor — this has two meanings in computer terms. One meaning is a television-like display. A monitor has no facility for tuning television programs, and usually the picture produced on a monitor is superior to that produced by an ordinary television. The second meaning of a monitor relates to ROM. The monitor of a computer is described as the information it has built in when you buy it. This information allows it to make decisions and carry out arithmetic computations.

Motherboard — a framework to which extra circuits can be added. These extra circuits often give the computer facilities which are not built-in, such as that of producing sound or of controlling a light pen.

MPU — an abbreviation for Microprocessor Unit.

N

Nano-second — a nano-second is one thousand billionth of a second, the unit of speed in which a computer or a memory chip is often rated.

Non-Volatile Memory — memory which is not lost when the computer is turned off. Some of the smaller computers such as the TRS-80 Pocket Computer have non-volatile memory. The batteries hold the program you enter for several hundred hours.

Not — a Boolean logic operation that changes a binary digit into its opposite.

Null String — a string which contains no characters. It is shown in the program as two double quote marks, without anything between them.

Numeric — pertaining to numbers as opposed to letters (that is, alphabetic). Many keyboards are described

as being alphanumeric which means both numbers and letters are provided.

O

Octal — a numbering system which uses eight as the base, and the digits 0, 1, 2, 3, 4, 5, 6 and 7. The Octal system is not used very much nowadays in microcomputer fields. The Hexadecimal system is more common (see Hexadecimal).

Operating System — the software or firmware generally provided with the machine that allows you to run other programs.

OR — an arithmetic operation that returns a 1, if one or more inputs are 1.

Oracle — a method of sending text messages with a broadcast television signal. A teletext set is required to decode the messages. Oracle is run by Independent Television Service in the UK, and a similar service — Ceefax — is provided by the BBC.

Output — information or data fed out by the computer to such devices as a TV-like screen, a printer or a cassette tape. The output usually consists of the information which the computer has produced as a result of running a program.

Overflow — a number too large or too small for the computer to handle.

P

Pad — see Keypad.

Page — often used to refer to the amount of information needed to fill one TV screen, so you can talk about seeing a page of a program, the amount of the listing that will appear on the screen at one time.

PASCAL — a high level language.

Peripheral — anything which is hooked onto a computer, for control by the computer, such as a disk unit, a printer or a voice synthesiser.

Port — a socket through which information can be fed out of or in to a computer.

Prestel — the British telecom name for a system of calling up pages of information from a central computer via the telephone and displaying them on a television screen. A similar commercial version in the United States is known as The Source.

Program — in computer terms program has two meanings. One is the list of instructions that you feed into a computer, and the second is used as a verb, as in 'to program a computer'.

PROM — stands for Programmable Read Only Memory. This is a device which can be programmed, and once it is then the program is permanent (also see EPROM and ROM).

R

Random Access Memory (RAM) — the memory within a computer which can be changed at will by the person using the computer. The contents of RAM are usually lost when a computer is turned off. RAM is the memory device that stores the program that you type in and also stores the results of calculations in progress.

Read-Only Memory (ROM) — in contrast to RAM, information in ROM cannot be changed by the user of the computer, and the information is not lost when the computer is turned off. The data in ROM is put there by the manufacturers and tells the computer how to make decisions and how to carry out arithmetic computations. The size of ROM and RAM is given in the unit K (see K).

Recursion — the continuous repetition of a part of the program.

Register — a specific place in the memory where one or more computer words are stored during operations.

Reserved Word — a word that you cannot use for a variable in a program because the computer will read it as something else. An example is the word TO. Because TO has a specific computer meaning, most computers will reject it as a name for a variable. The same goes for words like FOR, GOTO and STOP.

Routine — this word can be used as a synonym for program, or can refer to a specific section within a program (also see Subroutine).

S

Second Generation — this has two meanings. The first applies to computers using transistors, as opposed to first generation computers which used valves. Second generation can also mean the second copy of a particular program; subsequent generations are degraded by more and more noise.

Semiconductor — a material that is usually an electrical insulator but under specific conditions can become a conductor.

Serial — information which is stored or sent in a sequence, one bit at a time.

Signal — an electrical pulse which is a conveyor of data.

Silicon Valley — the popular name given to an area in California where many semiconductor manufacturers are located.

SNOBOL — a high level language.

Software — the program which is entered into the computer by a user which tells the computer what to do.

Software Compatible — this refers to two different computers which can accept programs written for the other.

Static Memory — a non-volatile memory device which retains information so long as the power is turned on, but does not require additional boosts of power to keep the memory in place.

Subroutine — part of a program which is often accessed many times during the execution of the main program. A subroutine ends with an instruction to go back to the line after the one which sent it to the subroutine.

T

Teletext — information transmitted in the top section of a broadcast television picture. It requires a special set to decode it to fill the screen with text information. The BBC service is known as Ceefax, the ITV service as Oracle. Teletext messages can also be transmitted by cable, for example the Prestel service in Britain or The Source in the United States.

Teletype — a device like a typewriter which can send information and also receive and print it.

Terminal — a unit independent of the central processing unit. It generally consists of a keyboard and a cathode ray display.

Time Sharing — a process by which a number of users may have access to a large computer which switches rapidly from one user to another in sequence, so each user is under the impression that he or she is the sole user of the computer at that time.

Truth Table — a mathematical table which lists all the possible results of a Boolean logic operation, showing the results you get from various combinations of inputs.

U

UHF — Ultra High Frequency (300-3000 megaHertz).

Ultra Violet Erasing — Ultra violet light must be used to erase EPROMs (see EPROM).

V

Variable — a letter or combination of letters and symbols which the computer can assign to a value or a word during the run of a program.

VDU — an abbreviation for Visual Display Unit.

Volatile — refers to memory which 'forgets' its contents when the power is turned off.

W

Word — a group of characters, or a series of binary digits, which represent a unit of information and occupy a single storage location. The computer processes a word as a single instruction.

Word-Processor — a highly intelligent typewriter which allows the typist to manipulate text, to move it around, to justify margins and to shift whole paragraphs if necessary on a screen before outputting the information onto a printer. Word-processors usually have memories, so that standard letters and the text of letters, written earlier, can be stored.

BIBLIOGRAPHY

BIBLIOGRAPHY

Compiled by Tim Hartnell

The A to Z Book of Computer Games (McIntire, Thomas C, Tab Books, Blue Ridge Summit, Pa.).

This is a fine Tab book to give you program ideas and ready-to-run programs, although some of the games are a disappointment, such as the overly long Othello program which does not even play, but simply records the moves made by two human players. Others, however, such as Fivecard and Hotshot, are well written, and well worth entering into your microcomputer.

BASIC Computer Games (ed. Ahl, David, Creative Computing Press, Morristown, New Jersey).

This is a classic work, the source of more programming ideas than any other computer games book ever published. I had a meal with David Ahl one night in London after a PCW show and discussed the book. He said that he'd been in the personal computer field almost before there were personal computers, and while many of the games in this book do not seem startling now, the fact that people could write and play games for computer interaction at all seemed quite incredible in the late seventies. The Checkers program, and Life for Two are just a couple of the treasures you will find in this splendid program and idea source book.

BASIC Computer Programs for the Home (Sternberg, Charles D, Hayden Book Company, Inc., Rochelle Park, New Jersey).

Traditionally, home computers (when first purchased) have been used for playing games. One reason why they have not been used for more serious applications

stems from the lack of a readily available, comprehensive set of home applications programs that were easy to use and understand and that satisfied the practical requirements of the home. This book provides a set of programs to make your computer start earning its keep. The programs provide a good cross-section of practical applications; these have been designed so as not to rely upon the availability of tape or disk-storage devices. The programs cover a wide field, and are divided into a number of sections: home financial programs (including household expenses and income tax recording); car related programs (including fuel use and trip planning); 'Kitchen Helpmates' (including diet and meal planning programs); scheduling programs for home use (including a reminder calendar and a couple of programs which I imagine are designed to short circuit arguments about which television programs will be watched); and 'List programs for every purpose' (including Christmas cards, music collections and three versions of an addresses program).

The BASIC Handbook (Lien, David A, Compusoft Publishing, San Diego, California).

This is an encyclopedia of the BASIC language. Now that BASIC is so firmly established throughout the microcomputer world, it is necessary to make its many dialects understandable so that programs can be transported between different computers. When you have found exactly the program you've been looking for, it is very frustrating to be unable to run it on your computer. This book addresses that problem by discussing in detail just about every commonly used BASIC statement, function, operator and command. For the most part, BASIC words mean the same thing to every computer which recognises them. If a computer does not possess the capabilities of a needed or specified word, there are often ways to accomplish the same function by using another word, or combination of words. Although the handbook requires some

application to transform the information into usable form, it is a very valuable reference work indeed. Every BASIC word you have ever heard of (and many you may not have heard of, such as LE, NE, GOTO-OF, RES and TIME) is probably in the book. It may be of limited use to you in your early days of computing, but it should become an indispensable handbook once you get more involved in the subject.

Beat the Odds, Microcomputer Simulations of Casino Games (Sagan, Hans, Hayden Book Company, Inc., Rochelle Park, New Jersey).

The book explains how to play certain casino games (trente-et-quarante, roulette, chemin-de-fer, craps and blackjack) and gives complete program listings in BASIC with commentaries on systems and optimal strategies. Professor Sagan (Professor of Mathematics at North Carolina State University) says he wrote the book in an attempt to convince people that, in the long run, they could not win — except possibly at blackjack — and to explain some popular systems and their pitfalls, and above all to provide very realistic computer simulations of the games themselves. He has succeeded in his attempt. The listings are possibly longer than other computer versions of the same games, but this is because the Sagan versions strictly duplicate the odds involved in playing the game 'in real life', and cover all the eventualities that a real game can produce. The programs are well-structured, and an examination of the listings should give you ideas for improving your own programming.

Beginner's Guide to Chess (Keene, Raymond, Pelham Books Ltd, London).

An ideal guide to simple chess-playing techniques which you can turn into algorithms if you intend to write a chess program of your own.

The Calculator Game Book for Kids of All Ages (Hartman, Arlene, Signet Books, New York).

The book's title says it all, and the names of the games

(which include Fibonacci Follies, Stretch to Sixty and Casting Out 9s) suggest the book's contents. There are some worthwhile brain-stretching puzzles, and 15 or so ideas definitely calling for conversion to computer games.

33 Challenging Computer Games for TRS-80/Apple/PET (Chance, David, Tab Books, Blue Ridge Summit, Pa.).

Even if you don't have any of the three computers named in the title, you will still find the book a goldmine of ideas for your own development, and many of them will run, with minimal alteration, on any BASIC-using computer. Particularly commendable programs are Life Support, Scrambled Eggs and Tank Assault.

Communicating with Microcomputers (Witten, Ian H., Academic Press, London).

This is an introduction to the technology of man/computer communications for the non-specialist. By placing particular emphasis on low-cost techniques associated with small systems and personal computers, the reader's attention is focused on the positive nature of the 'microprocessor revolution' — how machines can help people — rather than the negative aspects which are often highlighted in the non-technical press. The level of the book is suitable for the layman with some acquaintance with electronics. The final section, on speech communication, provides the most fascinating reading.

Computer Appreciation (Fry, T.F., Newnes, Butterworths, 1975).

A fairly 'straight' but useful overview of computer operation, and business applications. Designed to be used as a text for a course of business studies, the book covers a wide range of topics from a short account of the historical development of calculating devices, through computer hardware and programming, to the

organisation of a modern data-processing department. It concludes with a brief consideration of the applications of computers and a discussion on the effects of computers upon management matters. It is surprisingly undated, despite the extraordinary increase in hardware availability and capability since the book was written.

The Computer Book: An Introduction to Computers and Computing (Bradbeer, Robin; De Bono, Peter; Laurie, Peter; BBC Publications).

This book was published in conjunction with the BBC television series 'The Computer Programme', first transmitted on BBC2 from January 1982, and produced by Paul Kriwaczek. I discussed this book with Robin Bradbeer while it was being written, and he told me that the BBC editors were ruthless in pointing out any use of jargon. They insisted, said Robin, that nothing could be taken for granted. This insistence has resulted in a book which anyone can understand. It assumes nothing, not even the knowledge of how to use a shift key — or the effect of using it — on a typewriter. The many illustrations and photographs break up the text, which gives a detailed introduction to computers, especially micros, and their possible applications.

Computer Games for Businesses, Schools and Homes (Nahigian, J Victor and Hodges, William S. Winthrop Publishers Inc., Cambridge, Mass.).

Some of the programs are a little thin for the size and price of the book, but the best ones are well worth adapting to run on your computer. The inclusion of long, clear sample run printouts ensures that you know exactly what the programs will do before you run them. The Tennis and Star Trek programs are especially good.

Dice Games Old and New (Tredd, William E., Oleander Press, Cambridge).

This will give you enough clearly written games

explanations to keep you creating games programs on your microcomputer for a long time to come.

The Electronic Calculator in Business, Home and School (Birtwistle, Claude, Elliot Right Way Books, Kingswood, Surrey).

To get the best out of a calculator, you need to understand the mathematics which lies behind the operations. That is the purpose of the book, and in general it succeeds in this aim. The maths involved is, however, fairly simple and basic, since the book was written with a wide range of people in mind — the pupil at school, the student at college, the business person and the householder. It is a practical book which should be read and worked through with a calculator to hand.

Everyman's Indoor Games (Brandreth, Gyles, J M Dent and Sons Ltd, London).

If you're looking for games to convert into computer programs, ignore the chapters entitled Parlour Games and Children's Party Games and stick to the rest of the book, a treasure trove of games concepts which are certainly worth using as a starting point. Fox and Geese, Poker Dice and Billiards, as described in the book, are only a few of the programs you might write after reading it.

Games and Puzzles for Addicts (Millington, Roger, M and J Hobbs, Walton-on-Thames).

These games and puzzles first appeared in the weekly computer news-magazine 'Datalink', so they are especially likely to appeal to computer buffs. There are many ideas here that can be converted into games to be played with the computer.

Games for Home, Travel and Parties (Jensen, Helen, Western Publishing Company Inc., Racine, Wisconsin).

Aimed squarely at children, this book gives some games which are simple to program (these include Snakes, Lift-Off and Fish), and contains a complete chapter on how to play chess.

Home Computers, Questions and Answers, Hardware (Didday, Rich, dilithium Press).

The book has two main purposes. Firstly, it is intended to give readers a real feeling for what is involved in home computing, so that they can make rational decisions before buying equipment. Secondly, it is intended to give people who have no specialised knowledge of computing a general background to the subject, and specifically to microcomputers. The book succeeds in imparting enough information to ensure you will have little trouble understanding articles about advanced projects in computer hobbyist magazines, advertisements for home computing equipment, or other people who do have advanced computer knowledge.

Inside BASIC Games (Mateosian, Richard, Sybex).

This book is a guide, albeit a slightly overwritten one, for anyone who wishes to understand computer games. You will learn how to write interactive programs in BASIC and how the principles of systems development are applied to small computers. The book also looks at how the features of specific small computer systems have been supported in BASIC. The sections of the book include: Arithmetic Games, Guessing Games, Time Games, Date Games, Taxman, and programming in 'Free BASIC', a structured BASIC that is translated manually into the actual BASIC instructions to be entered into the computer. Free BASIC is not a language; it is a program description medium (like flowcharts) that has no line numbers, and uses symbolic names for subroutines. Additional chapters look at The Match-Up Game, Craps and Alien Life. If you can contend with the verbiage, you will find this book well worthwhile.

An Introduction to Personal and Business Computing (Zaks, Rodnay, Sybex).

I had lunch with Rodnay in London during a PCW show and he told me that he thought current American predictions on the growth of the personal computer

field were grossly pessimistic. He pointed out that the predictions current in 1978, when he wrote this book, have been proved so inaccurate that would-be prophets should take warning and assume that whatever they say will be wrong by a factor of 10 or 100. Despite its age — and computer books do age uncommonly quickly — this book is a good introduction to the field, explaining in clear, snappy English the fundamentals of computer operation. Dr Zaks also gives suggestions on what to look for when buying a computer.

Microsoft BASIC (Knecht, Ken, dilithium Press, Forest Grove, Oregon).

This book presents a complete introduction to programming in Microsoft BASIC. The concepts presented are illustrated with short, working programs. By starting with the simplest and most commonly used commands, and then progressing on to the more complex BASIC commands, Mr Knecht shows how the more powerful versions of the language can save valuable programming time and effort.

The Personal Computer Book (Bradbeer, Robin, Input Two-Nine).

The title says it all. Robin is deeply involved in the microfield in Great Britain. He started the North London Polytechnic Computer Fairs, assisted the BBC with their microcomputer television show (and co-authored *The Computer Book*, published by the BBC), and edited the monthly publication *Educational Computing*. This gave him a strong background from which to write the book. It explains what a computer is and how it works; it elucidates the mysteries inside the 'black boxes' which make up a computer; and it gives a number of very useful appendices, including bus standards, manufacturers and distributors, magazines, a selected bibliography (compiled by Richard Ross-Langley, of *Mine of Information*) and a glossary. But perhaps the most interesting and useful section of the book is the part which describes, in some

detail, the majority of computer systems available on the British market, their price and their capabilities. Overall, this is a very impressive source book.

Personal Computers: What They Are and How to Use Them (Wels, Byron G., Trafalgar House Publishing).

A great deal has happened in the computer world since this book was written in 1978, but there is still a great deal of value and interest in it. The book details some of the personal computers available, and the improvements that are likely to be made in the future. It explains, in layman terms, how a computer works, and how to make it work for you. There is also material on the construction and maintenance of small computer systems.

Play the Game (Love, Brian, Michael Joseph and Ebury Press, London).

This is a splendid book, containing 40 or so full-size reproductions of Victorian (and pre-Victorian) board games, many suitable for playing against a computer. It is even possible to use the boards in the book (the computer then tells you where it is moving on this external board) rather than write a routine within the program to display a board.

The Pocket Calculator Games Book (Schlossberg, Edwin and Brockman, John, Wilton House Publications Ltd., London).

There are many ideas here suitable for conversion into computer games.

57 Practical Programs and Games in BASIC (Tracton, Ken, Tab Books, Blue Ridge Summit, Pa.).

There are more serious programs than games (of the Chi-Square Evaluation and Fibonacci Numbers variety) in this book. They are well-programmed, and supported by adequate (if brief) documentation, and by flow-charts. The Space Wars programs (versions one and two) at the end of the book are particularly good.

Problems for Computer Solution (Rogowski, Stephen J, Creative Computing Press, Morristown, New Jersey).

This outlines over 50 simple (and a few not-so-simple) problems which can be solved by writing a program. There are both teacher and student editions of this book; the teacher edition has a suggested program and sample run printout to solve the difficulty. It is an excellent source for educational ideas.

Stimulating Simulations (Engel, C.W., Hayden Book Company, Inc., New Jersey).

Here, according to the cover, are '12 unique programs in BASIC for the computer hobbyist'. Inside you will find some fascinating programs: Forest Fire, Rare Birds and The Devil's Dungeon are three you are sure to enjoy playing, while Diamond Thief (the computer decides who has committed the crime, then challenges you to discover which of the suspects is guilty) is both well written and tightly programmed.

TAKE TWO! 32 Board Games for 2 Players (Tapson, Frank, A & C Black, London).

This book is aimed at children, but it does give many fascinating ideas that could be transformed into computer games (even if some of them are duplicated elsewhere in the book).

24 Tested, Ready-to-Run Game Programs in BASIC (Tracton, Ken, Tab Books, Blue Ridge Summit, Pa.).

Tab Books are prolific publishers in the microcomputer program field, and their books are deservedly successful. If nothing else, reading a book such as this one will give you ideas for structuring programs neatly, and for writing them to ensure the maximum compatibility between different versions of BASIC. Many of the games, such as Auto Rally and Capture the Alien, are (despite their weak titles) well thought out, carefully constructed programs.

1001 Things to Do with Your Personal Computer (Sawush, Mark, Tab Books, Blue Ridge Summit, Pa.).

I bought this book at a computer fair in Atlanta, and

read it (making notes, and turning down page corners) on the flight to London. And I still hadn't finished it on arrival. If you feel you have come to the end of possible applications for your computer, buy this book and discover that you have barely scratched the surface. It tells you about writing music and stories, aiding a mechanic or a carpenter, solving simultaneous equations, astrology, and much, much more.

The World Computer Chess Championship (Hayes, Jean E., and Levy, David N.L., Edinburgh University Press, Edinburgh).

This is a fascinating account of the world's first machine versus machine chess championship, held in 1974, when the dozen or so computer programs taking part were the only chess programs in existence. The games are analysed in detail, and the final section of the book outlines a board-numbering system which you could use if you're considering writing your own chess program. The book makes you realise how far the computer world has come in only a few years.

The Virgin Computer Games Series

GAMES FOR YOUR COMMODORE 64

More than 20 challenging programs, each one especially written for the series and guaranteed to provide hours of entertainment.

The games include SUBMARINE BREAKOUT (torpedo a path to safety – an iceberg blocks your way); PARATROOPER (practise landing on the rescue craft, but beware prevailing winds!); AMBULANCE DRIVER (a fast-moving game in which you must dodge other road users); DOODLER (create stunning works of art using your joystick); PARK KEEPER (can you rid the park of rubbish?); and PENNIES FROM HEAVEN (it's raining money . . . become a millionaire without moving from your armchair!).

GAMES FOR YOUR COMMODORE 64 will improve your programming skills as you follow the instructions to put each of the programs into your machine, and comes complete with a brief dictionary of computer terms, a selective bibliography and some hints on how to extend the programs in the book.

Virgin

Programs of originality and quality for all the family.

ISBN 0 86369 028 9

United Kingdom £2.95
Australia $9.95 (recommended)