

THE DELL
COMPUTER GAMES SERIES

 DELL • 52789 • U.S. \$5.95
CAN. \$7.50

GAMES FOR YOUR VIC20

MORE THAN 25 FASCINATING, ORIGINAL
PROGRAMS FOR ARCADE-STYLE GAMES,
BRAIN TEASERS, WORD GAMES,
PUZZLES, AND MUSIC TO CHALLENGE
AND ENTERTAIN YOU

Alastair Gourlay

1870

1871

1872

1873

1874

1875

1876

1877

1878

1879

1880

1881

1882

1883

1884

1885

1886

1887

1888

1889

1890

1891

1892

1893

1894

1895

1896

1897

1898

1899

1900



***GAMES
FOR YOUR
VIC20***





THE DELL COMPUTER GAMES SERIES

GAMES FOR YOUR TIMEX-SINCLAIR 1000

GAMES FOR YOUR TIMEX-SINCLAIR 2000

GAMES FOR YOUR VIC20

GAMES FOR YOUR ATARI COMPUTER





THE
DELL
COMPUTER
GAMES
SERIES

***GAMES
FOR YOUR
VIC20***

Alastair Gourlay

Series editor: Tim Hartnell

A DELL TRADE PAPERBACK

A DELL TRADE PAPERBACK

Published by
Dell Publishing Co., Inc.
1 Dag Hammarskjold Plaza
New York, New York 10017

GAMES FOR YOUR VIC 20 was first published in Great Britain by
Virgin Books Ltd. as part of the Virgin Computer Games Series.

Copyright © 1983 by Interface/Virgin Books

All rights reserved. No part of this book may be reproduced or
transmitted in any form or by any means, electronic or
mechanical, including photocopying, recording or by any
information storage and retrieval system, without the written
permission of the Publisher, except where permitted by law.

Dell ® TM 681510, Dell Publishing Co., Inc.

Printed in the United States of America

First U.S.A. printing—November 1983

Library of Congress Cataloging in Publication Data

Gourlay, Alastair.
Games for your VIC20.

(The Dell computer games series)

Bibliography.

1. Computer games. 2. VIC20 (Computer)—Programming.

I. Title. II. Series.

GV1469.2.G68 1983

794.8'2

83-14357

ISBN 0-440-52789-9

ALASTAIR GOURLAY—THE AUTHOR

Alastair Gourlay is studying computer science at Paisley College, near Glasgow. He has also written *34 Amazing Games for the 1KZX81* published in the UK by Interface.

SUE WALLIKER—THE ILLUSTRATOR

Sue Walliker is a freelance illustrator.

ACKNOWLEDGEMENTS

The author would like to thank Ken Staples, Ric Kube, Mike O'Neill, Adam Burbidge and Kevin Smith, who have all contributed material to this book. Especial thanks to John Cade who spent many hours testing, debugging and running the programs contained herein.



CONTENTS

Editor's Introduction	11
Author's Introduction	13
Control characters used	15
New Kid in Town	19
Combat	21
Trail Blazer	24
Piano 2001	26
Sketchpad	27
Amaze	28
Wumpus	30
Digit-Dodge	33
Tail Gunner	35
Space Dock	37
Golf	39
Breakout	41
Moonbase	43
Life	45
Fruit Machine	47
Spots from Outer Space	49
Destroyer	51
Car Race	54
Sub Chase	56
Reversi (Othello)	58
Apollo II	61
Sound Effects	62
Hangman	64
Simon	67
Blackjack	69

10 • GAMES FOR YOUR VIC20

Ski Run	72
Diamond Glory	73
Zombies	75
Micro Music	79
Rat Trap	81
Jupiter Run	84
Vogon	87
How to Write Better Programs	91
Glossary	101
Bibliography	117

Editor's Introduction

Your computer is waiting to challenge you. Arcade-style games, brain teasers, word games and puzzles are all here and ready to entertain you.

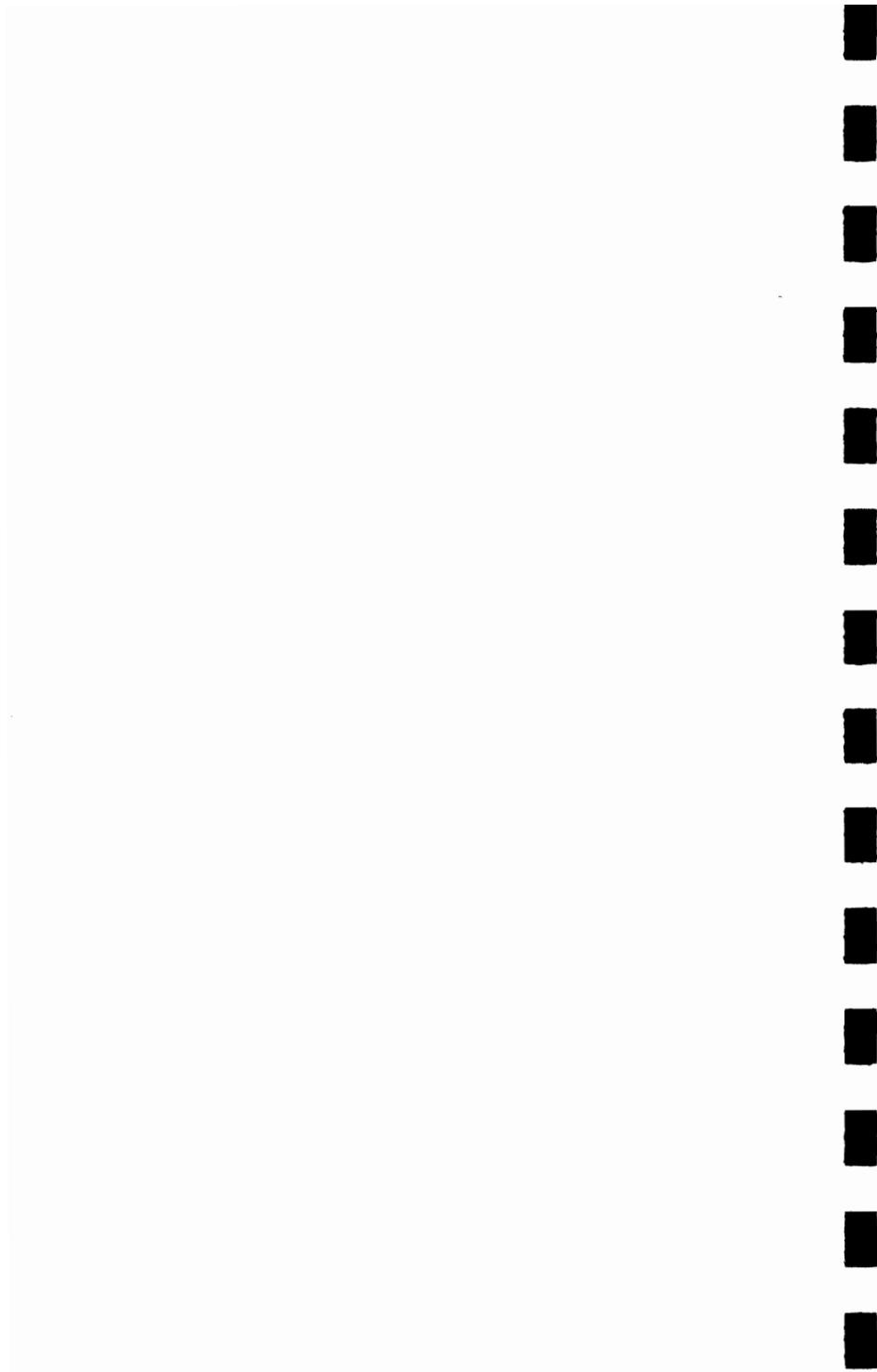
A wide variety of games are included in this book. The programs have been written by some of the most talented young programmers around at the moment, and represent a variety of approaches to solving programming problems.

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 VIC 20 is a remarkably flexible computer, yet many collections of programs designed for the VIC 20 seem to ignore this fact. In this group of programs I have looked for ones which demonstrate convincingly just how flexible and agile a computer the VIC 20 can be.

From music making to space fights, from golf to tank battles on minefields, these programs should give you hours of fun with your computer, as well as teaching you more about its fascinating possibilities.

I do not want you, however, to regard these programs as Holy Writ, final versions not to be tampered with. Rather they should be used as jumping-off points for your own games, your own developments, your own gradual discovery of the flexibility—and even companionship—of a fully stretched VIC 20.

ALASTAIR GOURLAY

Glasgow, Scotland
September 1982



THE FOLLOWING CONTROL
CHARACTERS HAVE BEEN
USED IN ALL LISTINGS-

F1-■ F2-■

F3-■ F4-■

F5-■ F6-■

F7-■ F8-■

CURSOR-

UP-⌈ DOWN-⌋

LEFT-⌊ RIGHT-⌋

CLEAR SCREEN-⌋

CURSOR HOME-⌋

BLK-■

WHT-■

RED-■

CYN-▲

PUR-■

GRN-■

BLU-■

YEL-■

***NOTE-IN THE GOLF
GAME TWO LISTINGS HAVE
BEEN GIVEN. THIS WILL
HELP IN THE ENTRY OF
THE LOWER CASE TEXT.





***GAMES
FOR YOUR
VIC20***



NEW KID IN TOWN

This is the program to use to show off the best features of your VIC when you first get it. Written by Ken Staples, no instructions are needed. Just type in the program, and then use it to impress your friends.

READY.

```
10 SL=7680:CL=38400:S1=36874:S2=36875:S3=36876:S4=36877:V=36878
11 DIMD(8)
12 PRINT"##### DEMONSTRATION TAPE":PRINT"### A LIGHT HEARTED":PRINT"### LOOK A
T A FEW";
13 PRINT"▲ OF THE":PRINTSPC(5)"###CAPABILITIES":PRINTSPC(6)"###OF THE VIC":FORI=1T
010000:NEXT
20 PRINT"###SPC(6)"###HI THERE!":PRINT"### MY NAME IS VIC.":PRINT"###SPC(9)"YOUR
30 PRINT"### FAITHFUL SERVANT":PRINT"###WOULD YOU LIKE TO SEE SOME OF THE WONDERF
UL"
40 PRINT"###THINGS I CAN DO":PRINT"###SPC(9)"YES!!!"
50 PRINT"###GOOD! ARE YOU SITTING### COMFORTABLY?":FORI=1T010000:NEXT
60 PRINT"#####WATCH ME CHANGE BORDER":PRINTSPC(1)"###AND SCREEN COLOUR IN"
65 PRINTSPC(2)"###255 DIFFERENT WAYS"
70 FORI=1T05000:NEXT:FORI=1T0255:POKE36879,I:PRINT"###I:FORJ=1T0300:NEXTJ,I
75 FORI=1T05000:NEXT:FORI=1T0255:POKE36879,I:NEXT
80 PRINT"##### JUST LET ME CHANGE"
90 PRINT"###TO MY ORIGINAL COLOUR":FORI=1T05000:NEXT:POKE36879,27
95 FORI=1T01000:NEXT
100 PRINT"###NOW LET US PLAY THE PIANO. JUST PRESS THE FOLLOWING KEYS"
110 PRINT"###1 1 5 5 6 6 5"SPC(9)"4 4 3 3 2 2 1 5 5 4 4 3 3 2"
120 PRINT"###5 5 4 4 3 3 2"SPC(9)"1 1 5 5 6 6 5"SPC(9)"4 4 3 3 2 2 1 8 9"
130 FORI=1T08:READD(I):NEXT
131 DATA223,227,230,231,234,236,238,239,223,223,234,234
135 DATA236,236,234,231,231,238,238,227,227,223,234,234,231,231
136 DATA238,238,227,234,234,231,231,238,238,227,223,223,234,234,236,236,234,231,
231,238
137 DATA238,227,227,223,239
140 POKEY,15
150 GETA$:IFA$=""THEN150
160 I=VAL(A$):IFI=0ORI=9THENPOKES2,0:GOTO100
170 POKES2,0:FORJ=1T025:NEXT:POKES2,D(I):GOTO150
180 PRINT"##### WHY NOT TRY AGAIN. ###ALTERNATIVELY, WHY ###NOT TRY TO COMP
OSE"
181 PRINT"### YOUR OWN TUNE. JUST ###PRESS 'Y' OR 'N'"
185 GETZ$:IFZ$<"Y"ANDZ$>"N"THEN185
190 IFZ$="Y"THENRESTORE:GOTO100
250 PRINT"###LET US SEE IF I CAN ###REMEMBER THAT TUNE ###YOU WERE TRYING TO
PLAY"
```

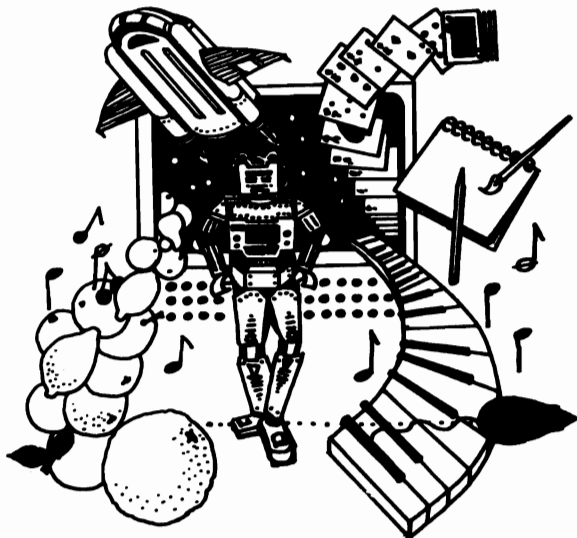
20 • GAMES FOR YOUR VIC20

```

260 FORI=1T0400:NEXT
300 POKEV,15:FORI=1T043:READD(0):POKE$2,D(0):FORJ=1T0400:NEXT:POKE$2,0
301 A=250:FI=70RI=140RI=280RI=35THENA=500
305 FORK=1TOR:NEXT
310 NEXTI:POKE$2,0:POKEV,0
320 PRINT"XXXXXXXXXX NOT EXACTLY BACK BUT I'M TAKING LESSONS!!!!"
340 FORI=1T0500:NEXT:PRINT"XXXXXXXXXX NOW LET US HAVE A FEW SOUND EFFECTS"
345 FORI=1T0500:NEXT
350 POKE$4,220:FORL=15T00STEP-1:POKEV,L:FORM=1T0300:NEXTH,L:POKE$4,0:POKEV,0
360 POKEV,15:FORL=15T00STEP-1:POKE$4,0:FORI=1T0100:NEXTI,0
370 FORO=233T0248STEP-1:POKE$4,0:FORI=1T0100:NEXTI,0:POKE$4,0:POKEV,0
380 POKEV,3:FORL=230T0120STEP-1:POKE$3,L:FORM=1T020:NEXTH,L:POKE$3,0:POKE$4,200
390 FORL=15T00STEP-.05:POKEV,L:NEXT:POKE$4,0
400 POKEV,15:FORL=1T010:POKE$2,200:FORM=1T0500:NEXT:POKE$2,0:POKE$3,200:FORM=1T0
500:NEXT
410 POKE$3,0:NEXT:POKEV,0
420 POKE$4,133:FORL=1T010:D=INT(RND(1)*90)+30:FORM=0T010:POKEV,M:FORM=1T0D:NEXTH
,M
430 FORM=15T00STEP-1:POKEV,M:FORM=1T0D:NEXTH,M,L:POKEV,0:POKE$4,0
440 POKEV,15:FORL=1T015:FORM=200T0220+L#2:POKE$3,M:NEXTH,L:POKEV,0:POKE$3,0
450 POKEV,15:0=FORL=120T0255STEP11:POKE$1,L:FORM=1T010:NEXT:0=0+1
460 IF0=3THEN0=POKE$1,0
470 NEXTL:POKEV,0:POKE$1,0
480 PRINT"ON FOR A VARIATION OF KNOCKING DOWN THE WALL GAME"
490 PRINT"OR WHY NOT USE IT TO DESIGN YOUR NEXT LOUNGE CARPET!!!"
495 FORI=1T010000:NEXT
500 PRINT"O":POKE36879,9:POKEV,15
510 FORI=1T0500:POKE7679+INT(RND(1)*506)+1,102:NEXT
520 X=1:Y=1:DX=1:DY=1
530 POKE7679+X+22*Y,01:FORI=1T010:NEXT:POKE7679+X+22*Y,32
540 X=X+DX:IFX=0ORX=21THENDX=-DX:POKE$3,220
550 Y=Y+DY:IFY=0ORY=22THENDY=-DY:POKE$3,230
560 POKE$3,0:IFPEEK(7679+X+22*Y)=102THENDX=-DX:POKE$3,100
565 A=A+1:IFA=500THENPOKE$3,0:POKEV,0:POKE36879,27:RUN
567 0=0+1:IF0=15THEN0=B1+1:0=0
568 IF0=16THEN0=0
570 POKE36879,B1:0T0530

```

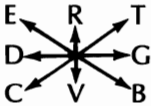
READY.



COMBAT

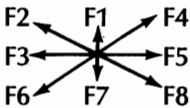
In this two player game, written by Ric Kube of South Australia, two tanks battle it out to the death. Both tanks are well protected—they can sustain up to ten hits before being destroyed. Deadly mines (shown as diamonds) are scattered around the battlefield. If a tank hits one of these it will lose one of its ten lives. There are other obstacles (solid squares) which, after being hit once, change (to dotted squares). Hitting the obstacle in its latter form will destroy the tank totally. The first player to lose ten lives loses the battle.

The controls for the left player are:



F to fire

The controls for the right player are:



Use the up arrow to fire

The object of the game is to score ten points. You score a point in one of two ways, either by shooting your opponent's tank, or when your opponent moves onto a mine. Missiles will fly over the mines and shatter on hitting the outer walls.

22 • GAMES FOR YOUR VIC20

```

100 REM##COMBAT
110 REM##RIC KUBE
120 GOSUB850
130 GOSUB848:PRINT "J"
140 POKE36879,93
150 FORX=1TO20
160 POKE7600+X,100:POKE8096+X,99
170 NEXT
180 FORY=1TO10
190 POKE7600+Y*22,103:POKE7701+Y*22,101
200 NEXTY
210 FORV=1TO10
220 FORX=1TO20
230 R=RND(1)*10
240 IFR>9.3ANDRC<9.75THENPOKE7600+22*Y+X,160
250 IFR>9.75THENPOKE7600+22*Y+X,90
260 NEXT:NEXT
270 LV=10:LH=3:RV=10:RH=36
280 M=32
290 POKE7903,01:POKE7936,07
300 POKE8126,SL+48:POKE8133,SR+48
310 IFSL=5ORSR=5THENPOKE36879,91
320 IFSL=9ORSR=9THENPOKE36879,90
330 IFSL=10ORSR=10THEN740
340 R#=""
350 GETA$:IFA#=""THEN350
360 IFA#="C"ORR#=""THENR=-1:D=1
370 IFA#="V"ORR#=""THENR=0:D=1
380 IFA#="B"ORR#=""THENR=1:D=1
390 IFA#="D"ORR#=""THENR=-1:D=0
400 IFA#="F"ORR#=""THENR=70
410 IFA#="Q"ORR#=""THENR=1:D=0
420 IFA#="E"ORR#=""THENR=-1:D=-1
430 IFA#="R"ORR#=""THENR=0:D=-1
440 IFA#="T"ORR#=""THENR=1:D=-1
450 IFR=0ANDD=0THEN340
460 IFASC(R#)>130THENRR=R:RD=D:V=RV:H=RH:P=07:S=8L:M=RM
470 IFASC(R#)<130THENLR=L:LD=D:V=LV:H=LH:P=01:S=6R=M+LM
480 C=7600+H+22*V
490 N=7600+H+22*(V+D)+R
500 IFPEEK(N)>00ANDPEEK(N)<90THEN300
510 IFPEEK(N)=32THENPOKEC,M:M=32:POKEN,P
520 IFPEEK(N)=90THENPOKEC,M:M=90:POKEN,P:GOSUB600:S=S+1
530 H=H+R:V=V+D
540 IFASC(R#)>130THENRV=V:RH=H:SL=S:RM=M
550 IFASC(R#)<130THENLV=V:LH=H:SR=S:LN=M
560 GOTO300
570 IFA#="@"THENV=RV:H=RH:R=RR:D=RD:Q=07
580 IFA#="F"THENV=LV:H=LH:R=LR:D=LD:Q=01
590 C=7600+H+22*V
600 N=7600+H+R+22*(V+D)
610 IFPEEK(N)=32THENPOKEC,Q:POKEN,46:H=H+R:V=V+D:Q=32:GOTO590
620 IFPEEK(N)=102THENPOKEC,Q:GOSUB600:POKEN,32:GOTO300
630 IFPEEK(N)=160THENPOKEC,Q:GOSUB600:POKEN,102:GOTO300
640 IFPEEK(N)=90THENPOKEC,Q:POKEN,46:H=H+R:V=V+D:Q=90:GOTO590
650 IFPEEK(N)=81THENS=SR+1:POKEC,Q:GOSUB600:GOTO300
660 IFPEEK(N)=07THENS=SL+1:POKEC,Q:GOSUB600:GOTO300
670 IFPEEK(N)>90THENPOKEC,Q:GOTO300
680 Z1=PEEK(N+1):Z2=PEEK(N-1):Z3=PEEK(N+22):Z4=PEEK(N-22)
690 POKEN+1,42:POKEN-1,42:POKEN+22,42:POKEN-22,42
700 FORX=0TO50:NEXT
710 POKEN+1,Z1:Z1:POKEN-1,Z2:POKEN+22,Z3:POKEN-22,Z4
720 POKE36878,15:POKE36877,255:FORD=1TO50:NEXT:POKE36878,0
730 RETURN
740 IFSL=10THENR#="LEFT"
750 IFSR=10THENR#="RIGHT"
760 POKE36879,93

```

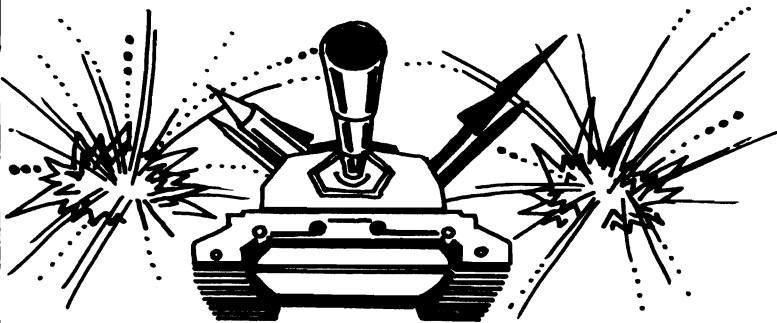


```

770 PRINT"THE GAME WAS WON BY":PRINT" THE "A$" PLAYER"
780 PRINT"ANOTHER GAME?(Y/N)"
790 GETA$:IF A$="" THEN 790
800 IF A$="Y" THEN 830
810 IF A$="N" THEN END
820 GOTO 790
830 OOSUB 840:GOTO 130
840 A$="":R=0:D=0:SL=0:SR=0:LH=32:RM=32:RETURN
850 PRINT"COMBAT"
860 PRINT"THE OBJECT OF THE GAME IS TO SCORE 10 POINTS"
870 PRINT"YOU MAY SCORE A POINT IN TWO WAYS-"
880 PRINT"1)SHOOTING YOUR OPPONENT'S TANK":PRINT"2)YOUR OPPONENT MOVING ON TO A
"
890 PRINT"MISSILES WILL FLY OVER MINES AND DISINTEGRATE ON HITTING THE OUTER WALL
"
900 PRINT"EACH PLAYER HAS 9 CONTROLS.":PRINT" PRESS ANY KEY TO CONTINUE"
910 GETA$:IF A$="" THEN 910
920 PRINT"LEFT PLAYER"
930 PRINT" E R T"
940 PRINT" M D F G"
950 PRINT" M C V B"
960 PRINT"RIGHT PLAYER"
970 PRINT" F2 F1 F4"
980 PRINT" F3 ↑ F5"
990 PRINT" F6 F7 F8"
1000 PRINT"PRESS ANY KEY TO CONTINUE"
1010 GETA$:IF A$="" THEN 1010
1020 PRINT"TO MOVE PRESS THE KEY FOR THE DIRECTION YOU WANT TO GO"
1030 PRINT"TO FIRE PRESS THE 'F' OR '↑' KEYS"
1040 PRINT"THE MISSILE FIRES IN THE DIRECTION YOU LAST MOVED"
1050 PRINT"LEFT TANK-'L'"
1060 PRINT"RIGHT TANK-'R'"
1070 PRINT"PRESS ANY KEY TO BEGIN"
1080 GETA$:IF A$="" THEN 1080
1090 RETURN

```

READY.



TRAIL BLAZER

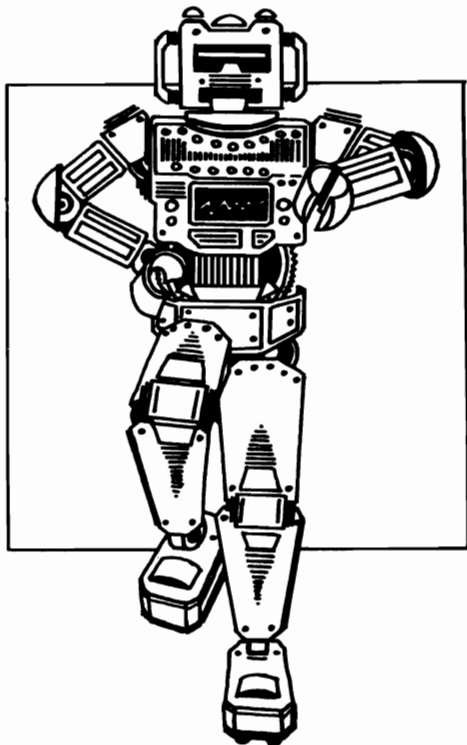
You are trapped in a square, dot filled area, with a computer controlled robot. As you and the robot move around, you lay trails behind you. If you hit the walls of the arena, or either of the trails, you will be electrocuted. Points are scored as follows:

Trapping the robot (the green O)—200 points

Hitting purple power point—50 points

Hitting while dot—10 points

The speed of the game increases with each round.



```

10 RESTORE
13 GOSUB3000:GOSUB2000
15 X=RND(-T1):DEFFNR(X)=(PEEK(7663+X)=0)
20 D(0)=1:D(1)=22:D(2)=-1:D(3)=-22
30 L=150+(KJ$="J")*50
40 PP=46:PC=437:DP=1:DC=2:SC=7600:CL=38400:POKE36878,15
45 GOSUB1000
50 IFKJ$="J"THEN62
55 GETA$:"J"198,0
60 ND=22*(R$="P")-(R$=".")+(R$="L")-(R$=":"):IFND<0THENDP=ND
61 GOTO70
62 SYS7600:X1=0:FORX=0TO3:X1=X1-FNR(X):NEXT:IFX1>1THEN70
65 ND=22*(FNR(0)-FNR(1))+FNR(2)-FNR(3):IFND<0THENDP=ND
70 NP=PP+DP:X1=PEEK(SC+NP):IFX1<46THEN700
75 POKE36874,128:S=S+10
80 POKESC+PP,160:POKECL+PP,2:POKESC+NP,42:POKECL+NP,2:PP=NP
85 PRINT"SCORE "S":POKE36874,0:POKE36876,0
90 NC=PC+D(DC):IFPEEK(SC+NC)<46ORRND(1)<.15THEN200
100 POKESC+PC,160:POKECL+PC,5:POKESC+NC,15:POKECL+NC,5:PC=NC
110 FORX=0TOL:NEXT:GOTO50
200 FORX=1TO19:DC=INT(RND(1)*4):IFPEEK(SC+PC+D(DC))=46THENNC=PC+D(DC):GOTO100
210 NEXT
220 S=S+200:L=L/2
230 FORX=1TO4:FORY=128TO160:POKE36875,Y:NEXT:NEXT:POKE36875,0:POKE198,0:GOTO40
700 IFX1=81THENPOKE36876,200:S=S+50:GOTO80
800 POKE36877,250:FORX=1TO2000:NEXT:POKE36877,0
810 PRINT"GAME OVER":FORX=1TO5000:NEXT:POKE198,0:S=0:GOTO10
1000 PRINT" ";FORX=1TO20:PRINT" ";NEXT
1010 PRINT" ";
1020 FORX=1TO5:X1=INT(RND(1)*20)+1+(INT(RND(1)*20)+1)*22
1030 POKE7680+X1,160:POKE38400+X1,0:NEXT
1040 FORX=1TO5:X1=INT(RND(1)*20)+1+(INT(RND(1)*20)+1)*22
1050 POKE7680+X1,81:POKE38400+X1,4:NEXT:POKESC+46,46:POKESC+437,46
1060 POKECL+46,1:POKECL+437,1:RETURN
2000 POKE36879,25
2010 PRINT"TRAIL BLAZER"
2020 PRINT"MOVE AROUND THE SCREEN USING THE JOYSTICK OR-"
2030 PRINT" "
2040 PRINT"SCORING-"PRINT",-10 PTS":PRINT"50 PTS":PRINT"TRAP-200 PTS"
2360 PRINT"PRESS ANY KEY"
2370 GETA$:"A"297,0
2390 POKE36879,110:IFKJ$<>" "THENRETURN
2390 PRINT"KEYBOARD OR JOYSTICK?":PRINT" (K/J)"
2391 GETKJ$:IFKJ$<>"K"ANDKJ$<>"J"THEN2391
2392 RETURN
3000 POKE31,176:POKE52,29:POKE55,176:POKE56,29
3010 FORX=7600TO7661:READA:POKEA,A:NEXT
3020 RETURN
3030 DATA169,0,141,19,145,169,127,141,34,145,173,17,145,141,238,29,169,4,45,238,
29
3040 DATA141,239,29,169,8,45,238,29,141,240,29,169,16,45,238,29,141,241,29,169,3
2
3050 DATA45,238,29,141,243,29,173,32,145,41,128,141,242,29,169,255,141,34,145,96

```

READY.

PIANO 2001

This program turns the VIC into a weird, futuristic piano. The piano uses the middle and high music generators of the VIC, allowing you to play two notes at once. The keyboard is used as follows:

High Notes:

Q W E R T Y U I F3—off

Middle Notes:

A S D F G H J K F5—off

F1 is used to effect the move between low and high octaves, increasing the piano's range.

READY.

```
1 FORX=36875T036877:POKEX,127:NEXT
3 POKE36879,8:PRINT"#####"
5 DIMNO(2,9)
6 OC=1
10 FORX=1T09:READNO(1,X):NEXT
15 FORX=1T09:READNO(2,X):NEXT
100 DATA128,141,153,159,170,179,187,191,127,191,198,204,207,213,217,221,223,127
200 GETA$
205 IFR$="!" THENOC=OC+1:IFOC=3THENOC=1
220 B$="RSDFGHJKM":GOSUB1000:IFXC=1THENT=36875:GOSUB2000
230 B$="QMERTYUI":GOSUB1000:IFXC=1THENT=36876:GOSUB2000
235 FORX=15T06STEP-3:POKE36878,X:NEXT
236 FORX=6T015STEP3:POKE36878,X:NEXT
240 GOTO200
1000 FORX=1TOLEN(B$):IFR$=MID$(B$,X,1)THENX=NO(OC,X):RETURN
1010 NEXT X=-1:RETURN
2000 IFX=PEEK(T)THENPOKET,X-10
2010 FORN=PEEK(T)T0XSTEPSON(X-PEEK(T))#ABS(X-PEEK(T))/99:POKET,N:NEXT:POKET,X:RE
TURN
```

READY.



SKETCHPAD

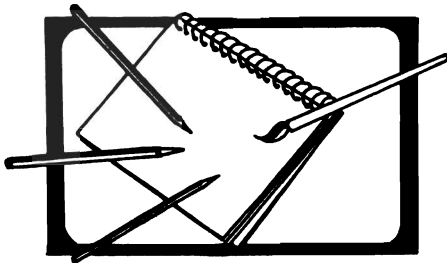
Let your imagination run wild with this multi-colored sketch program. You move the cursor (the checkerboard square) around the screen using the following keys: "P"—up, "."—down, "L"—left, and ":"—right. Hold any shift key to draw, or press the shift lock key while moving the cursor. If the shift key is not pressed, the cursor will erase as it moves.

To clear the screen, just hold down the "." key until the picture moves off the top of the screen. To change the drawing color, press the keys one to eight, depending on the color required. Note that shift and shift lock should not be pressed when changing color.

READY.

```
10 POKE36879,136:PRINT"SKETCHPAD":CL=6:CL$="SKETCHPAD":POKE36878,13
20 X=PEEK(197)
25 A$=" "
30 IFPEEK(653)08THENA$=" B"
40 IFX=13THENPRINTA$:"B";
50 IFX=37THENPRINTA$:"B";
60 IFX=21THENPRINTA$:"B";
70 IFX=45THENPRINTA$;
75 GETA$:A=VAL(A$):IFR08THENCL=A
80 PRINT"SKETCHPAD";MID$(CL$,CL,1);"B";:POKE36875,168+CL*10
90 FORX=1TO200:NEXT:PRINT"SKETCHPAD":POKE36875,0:GOTO20
```

READY.



AMAZE

This is a simple but addictive maze game, written by Mike O'Neill, in which you pit your skills against the computer. You control a circular piece, the computer has a diamond. There are ten levels of play (different speeds) and three different mazes. Your time is displayed at the bottom of the screen by a horizontal bar which gets smaller as time runs out. The keys used are "A"—up, "S"—down, "K"—left, and "L"—right.

The mazes may be changed by altering the subroutines between lines 5000 and 7010. More mazes may be added if a memory expansion cartridge is available. The keys used may be changed by altering lines 350 to 380.

```
10 PRINT"AMAZE " :PRINT" BY MIKE O'NEILL" :POKEV,0
20 R=RND(-TI):S=36876:V=36878:PRINT
30 INPUT" ENTER LEVEL (0-9)":R:IFR>9ORR<0THEN30
40 IR=200-R*20
50 PRINT
60 INPUT" WHICH MAZE (1-3)":I%
70 IFI%<1OR I%>3THEN60
80 POKE36879,57:PRINT"
90 ONI%GOSUB7000,5000,6000
100 PRINT" I "
140 X1=INT(RND(1)*20)
150 X2=INT(RND(1)*20):IFABS(X1-X2)<6THEN150
160 Y1=INT(RND(1)*10)
170 Y2=INT(RND(1)*10):IFABS(Y1-Y2)<6THEN170
180 P1=7702+X1+Y1*22:IFPEEK(P1)<>32THEN140
190 P2=7702+X2+Y2*22:IFPEEK(P2)<>32THEN140
200 POKEP1+30720,:FORJ=1TO10
21: POKEP1,32:FORI=1TO50:NEXT
220 POKEP1,81
230 POKES,202:POKEV,15:FORI=1TO50:NEXT:POKEV,0:NEXT
250 POKEP2+30720,2:FORJ=1TO10
260 POKEP2,32:FORI=1TO50:NEXT
270 POKEP2,90
280 POKES,151:POKEV,15:FORI=1TO50:NEXT:POKEV,0:NEXT
290 I%=0
300 I%=I%+1:GOSUB4000:A$="":IFI%>100THEN8000
310 POKES,150+I%:POKEV,15
320 GETB$:IFB$=""THEN340
330 A$=B$:GOTO320
340 IFR$=""THEN400
```

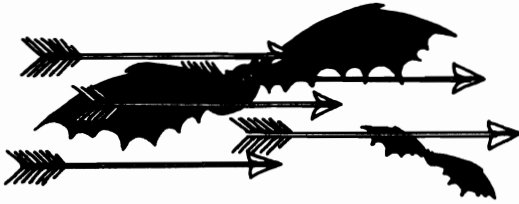
WUMPUS

You are trapped in the Wumpus Caves, with two foul-smelling wumpi. Scattered around the caves are a number of bottomless pits. If you fall into one of these, you will be lost for good. Your only aid is a set of five magic arrows; these can be used to kill a wumpus, but only when you are close enough.

You will be warned of the proximity of a wumpus by your trusty VIC, the only ally you have in the caves. Your VIC will also give you a warning if you wander too close to any of the bottomless pits. The cave system is numbered as follows:

	92	94	96	98	100
81	83	85	87	89	
	72	74	76	78	80
61	63	65	67	69	
	52	54	56	58	60
41	43	45	47	49	
	32	34	36	38	40
21	23	25	27	29	
	12	14	16	18	20
1	3	5	7	9	

Enter the number of the cave in which you wish to move. You can move one cave up, down, left or right. There are one or two additional problems. You must not take more than 20 minutes to find and kill the two wumpi, and you will be in



real trouble if you use up all of your arrows. There are also the caves inhabited by superbats which can carry you to any other cave in the system. A final warning: it is inadvisable to disturb the wumpi, they can turn quite nasty.

```

100 POKE36879,127:PRINT"J"
110 DEFFNR(X)=INT(RND(1)*X)
120 P=1:MA=5
130 NP=FNR(5)+3:FORN=1TONP:PT(N)=FNR(99)+2:NEXT
140 W1=FNR(99)+2:W2=FNR(99)+2:IFW1=W2THEN140
150 TI$="000000"
160 TH=VAL(MID$(TI$,3,2))
170 PRINT"CAVE"PT" ATLINE"TH"MIN."
180 PRINT"MAGIC ARROWS:"MA
190 IFABS(W1-P)>1ANDABS(W1-P)>10ANDABS(W2-P)>1ANDABS(W2-P)>10THEN370
200 PRINT"### THERE IS A WUMPUS NEARBY"
210 PRINT"WIRE AN ARROW?(Y/N)":POKE190,0
220 GETA$:IFR$<"Y"ANDR$<"N"THEN220
230 IFR$="N"THEN470
240 INPUT"WHERE TO";CH
250 MA=MA-1
260 IFABS(CH-P)>1ANDABS(CH-P)>10THENPRINT"CAN'T SHOOT HERE":GOTO240
270 IFW1<CNANDW2<CNTHENPRINT"### YOU MISSED IT"
280 IFW1=CNTHENW1=-100
290 IFW2=CNTHENW2=-100
300 PRINT"### YOU KILLED A WUMPUS":FORN=1TO1000:NEXT
310 IFW1<100ORW2<100THEN470
320 POKE36879,25:PRINT"##### YOU HAVE SURVIVED WITH"20-TH"MINUTES LEFT"
330 PRINT"### ANOTHER GAME(Y/N)":POKE190,0
340 GETA$:IFR$<"Y"ANDR$<"N"THEN340
350 IFR$="N"THENPOKE36879,27:PRINT"###":POKE190,0:END
360 RUN
370 FORN=1TONP
380 IFABS(PT(N)-P)>1ANDABS(PT(N)-P)>10THEN400
390 PRINT"### IT'S DRAUGHTY HERE- PITS NEARBY!!!"
400 NEXT
410 IFFNR(30)>0THEN470
420 PRINT"### SUPERBATS HAVE CARRIED YOU AWAY!!!!!"
430 CH=FNR(100)+1:IFCN=PTHEN430
440 P=CH:PRINT"THEY DROPPED YOU IN CAVE"PT:FORN=1TO3000:NEXT
450 GOTO510
460 IFR$=0THEN620
470 IFR$=0THEN620
480 INPUT"### WHERE TO NOW";CN
490 IFCN<1ORCN>100ORABS(CH-P)>1ANDABS(CH-P)>10THENPRINT"CAN'T GO THERE!!":GOTO
470
500 P=CN

```

32 • GAMES FOR YOUR VIC20

```
510 FORN=1TONP:IFP=PT(N)THEN560
520 NEXT
530 IFTM=20THEN600
540 IFW1=POR42=PTHE500
550 GOTO160
560 POKE36879,8:PRINT"#####YOU FELL DOWN A PIT"
570 GOTO330
580 POKE36879,42:PRINT"#####YOU WERE EATEN BY A   WUMPUS!!!!!!!"
590 GOTO330
600 POKE36879,110:PRINT"#####YOU RAN OUT OF TIME   "
610 GOTO330
620 POKE36879,76:PRINT"#####YOU HAVE NO MAGIC           ARROWS   "
GOTO330
```

DIGIT-DODGE

In this graphics game you have 30 seconds to maneuver your patented Digit-Dodger around the screen, eating up the displayed numbers in the correct order, starting with 1. Once you have completed each set of numbers correctly, you will be given a new stage, with an added number. The game ends when you fail to complete a stage within the 30-second time limit.

The game also ends if you crash into the obstacles, or hit a number out of sequence three times in a round. To move your Digit-Dodger use the "P" (up), "." (down), "L" (left) and ":" (right) keys.

READY.

```
100 POKE36879,31:PRINT"J";
110 S=2:SC=0:POKE36878,15
120 DEFFNR(X)=INT(RND(1)*X):DEFFNK(X)=K-INT(X/22)*22
130 PRINT"J":FORN=1TO20+S*20:K=FNR(462)+44:POKEX+7688,168:POKE38400+X,0:NEXT
140 FORN=1TO5
150 X=FNR(462)+44:IFPEEK(X+7688)O32THEN150
160 GOSUB440
170 POKEX+7688,N+176:POKEX+38400,N+1:NEXT
180 CP=22:TI#="000000":CS=1:CR=0
190 POKE36874,130
200 NP=CP+22*(PEEK(197)=13)-(PEEK(197)=37)+(PEEK(197)=21)-(PEEK(197)=45)
210 POKE36874,0
220 IFNP=CPORNP<22ORNP>585ORABS(FNK(NP)-FNK(CP))>1THENNP=CP:GOTO200
230 N=PEEK(NP+7688)
240 IFX=CS+176THENGOSUB440:SC=SC+(48-VAL(TI#))*CS:CS=CS+1:GOTO270
250 IFXOCS+176ANDXO32THENGOSUB450:NP=CP:CR=CR+1:IFCR=3THEN370
260 PRINT"###TIME:"RIGHT$(TI#,2)"###SCORE"SC"###"
270 IFC>6THEN400
280 POKENP+38400,4:POKE7688+CP,32:POKE7688+NP,81:CP=NP
290 IFTI#<"000030"THEN190
300 FORN=1TO5:GOSUB450:NEXT
310 PRINT"#####TIME UP ###"
320 PRINT"#####ANOTHER GAME(Y/N) ###"
330 POKE198,0
340 GET# :IF#O"Y"AND#O"N"THEN340
350 IF#="N"THENPOKE36879,27:PRINT"J":END
360 RUN
```

34 • GAMES FOR YOUR VIC20

```
370 PRINT"!!!!!!!!!!!!!! !! TOO MANY CRASHES !!"  
380 PRINT"!!!!!!!!!!!!!! GAME OVER !!"  
390 GOTO320  
400 FORN=128TO248:POKE36874,N:POKE36875,N:NEXT:POKE36874,0:POKE36875,0  
410 C8=1:S=S+1:IFC8>5THEN8=5  
420 GOTO130  
430 GOTO430  
440 FORN1=200TO128STEP-1:POKE36876,N1:NEXT:POKE36876,0:RETURN  
450 FORN1=200TO148STEP-1:POKE36877,N1:NEXT:POKE36877,0:RETURN  
  
READY.
```

TAIL GUNNER

You are the tail gunner in a Federation Starfighter, which is being pursued by a fleet of enemy spy planes. You must destroy them as they enter your sights. Each plane must be destroyed as quickly as possible to score the maximum number of points. If you exceed the time limit shown at the top of the screen, the enemy will inform their base and the game will be lost.

To move your sights onto the alien, use the same keys as in Digit-Dodge ("P"—up, "."—down, "L"—left and ":"—right). Hit the "=" to fire when the alien is in the center of your sights.

READY.

```
100 CL=30720:DEFFNR(X)=INT(RND(1)*X):DEFFNX(X)=X-7680-INT((X-7680)/22)*22:TT=30:
DL=1.8
110 POKE36879,12:PRINT"J";
120 POKE36878,15
130 PRINT"TIME 00:"TT:PRINT"XXXXXXXXXXSCORE"SC"X";
140 FORN=1TO22
150 PRINT:PRINTTAB(11)"X"X";:NEXT:PRINT"XXXXXXXXXXXXX"X"X"X"
160 PA=7782:TI$="000000"
170 NA=PA-22*(PEEK(197)=13)-(PEEK(197)=37)-(PEEK(197)=21)+(PEEK(197)=45)
180 IFNR>PATHENPOKE36877,130:POKE36874,130
190 GOSUB380
200 POKE36877,0:POKE36874,0
210 IFPEEK(197)=46THENGOSUB200
220 IFFNR<INT(DL)=0THEN250
230 NA=PA+22*(FNR(3)-1)+FNR(3)-1
240 GOSUB380
250 PRINT"XXXXXXXX"RIGHT$(TI$,2)
260 IFVAL(TI$)=TTTTEN420
270 GOTO170
280 NA=20:FORN=8121TO7932STEP-21:POKE36877,250:POKE36876,250
290 POKEN,78:POKEN+CL,2:POKEN+NA,77:POKEN+NA+CL,2
300 POKE36877,0:POKE36876,0:NA=NA-2:NEXT
310 NA=20:FORN=8121TO7932STEP-21
320 POKEN,32:POKEN+NA,32
330 NA=NA-2:NEXT:IFPR<7911THENRETURN
340 TI=VAL(TI$)
350 POKE36877,130:FORN=15TO0STEP-1:POKE36878,N:FORNA=1TO50:NEXT:NEXT
```

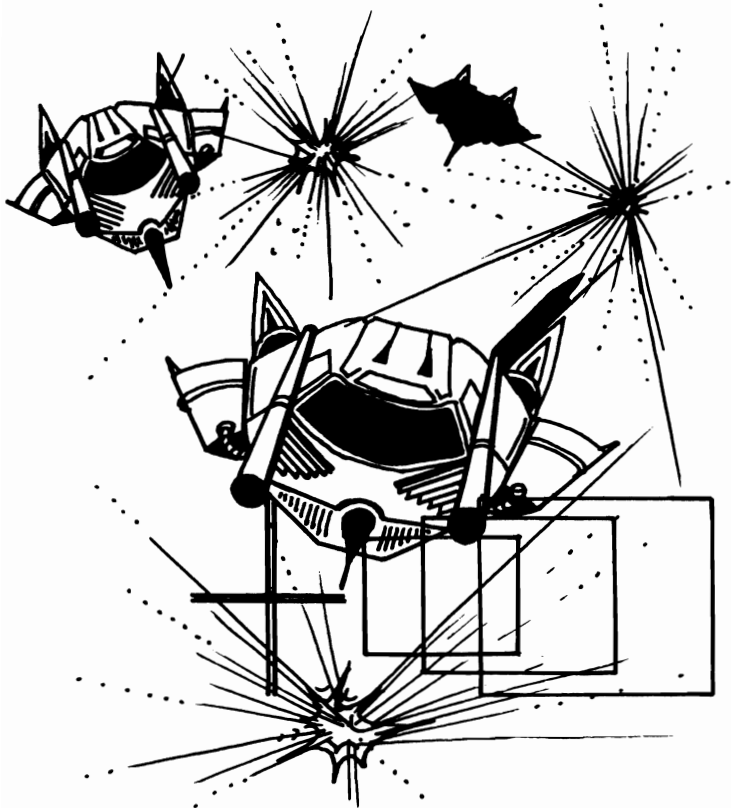
36 • GAMES FOR YOUR VIC20

```

360 SC=SC+TT-T1:DL=DL+.2:TT=TT-2:IFTT<10THENTT=10
370 GOTO110
380 IFNR<7020RNR>01850RABS(FNK(NR)-FNK(PA))>1THENNA=PA:RETURN
390 IFPEEK(PA)<160THENPOKEPA,32
400 IFPEEK(NR)<160THENPOKECL+NR,1:POKENR,01
410 PA=NR:RETURN
420 FORN=1TO8:POKE36876,ABS(SGN(PEEK(36876))-1)*190:FORNA=1TO50:NEXT:NEXT
430 PRINT"#####"
440 PRINT"##### YOU RAN OUT OF TIME #####"
450 PRINT"#####ANOTHER GAME (Y/N)#####"
460 POKE198,0
470 GETA$:IFA$<"Y"ANDR$<"N"THEN470
480 IFA$="N"THENPOKE36879,27:PRINT"#####":POKE198,0:END
490 RUN
500 GOTO500
510 RETURN

```

READY.

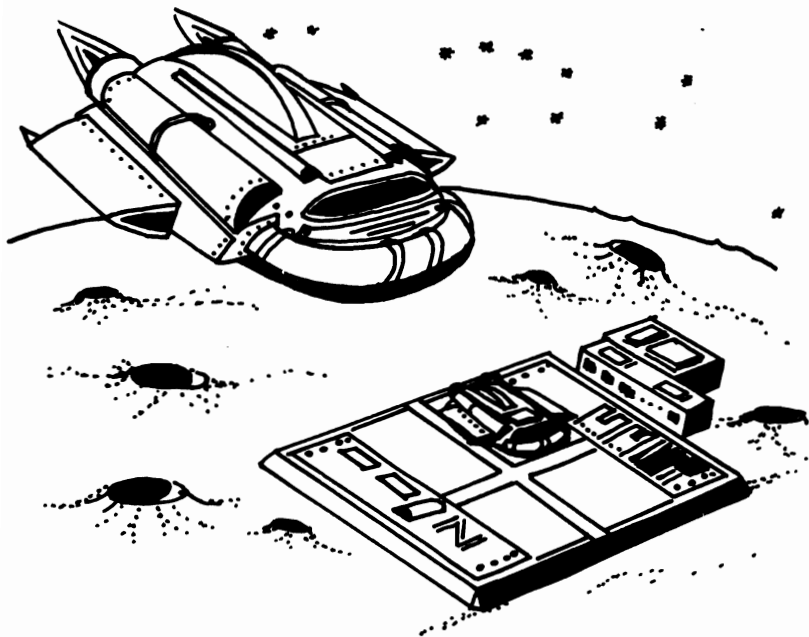


38 • GAMES FOR YOUR VIC20

```

1035 PRINT"MPRESS ANY KEY TO PLAY"
1036 GETA$:IFA$=""GOTO1036
1040 RO=4:P=22:S=1
1050 PRINT" "
1053 D=INT(RND(1)*43)+410
1055 POKE$C+D,01:POKECL+D,2
1060 PRINT"MM"
1070 GOTO10
2000 POKE36879,30:PRINT"MM SPACE DOCKING!"
2010 PRINT"MM DO YOU WANT INSTRUCTIONS MM"
2020 PRINT"MM [Y OR N]"
2030 GETA$:IFA$<"Y"AND$<"N"THEN2030
2035 POKE198,0
2040 IFA$="N"THEN900
2060 POKE36879,127:PRINT"MM SPACE DOCKING"
2070 PRINT"MM THE OBJECT OF THE GAME IS TO GUIDE YOUR SPACE SHUTTLE TO THE"
2080 PRINT"RED DOCKING PORT.":PRINT" YOU MUST AVOID THE WHITE STARS AND ALSO
THE TOP AND ";
2090 PRINT"BOTTOM OF THE SCREEN.IT IS SAFE TO LEAVE ONE SIDE OF THE SCREEN AND
";
2100 PRINT"REAPP-EAR ON THE OTHER."
2110 PRINT"MPRESS ANY KEY"
2115 GETA$:IFA$=""GOTO2115
2117 POKE198,0
2120 PRINT"MMYOUR SCORE IS SHOWN ON THE TOP LEFT OF THE SCREEN."
2130 PRINT"MM CHANGE THE DIRECTION OF THE SHUTTLE WITH:--"
2140 PRINT"MM-Rotate CLOCKWISE"
2143 PRINT"MM-Rotate ANTICLOCKWISE"
2150 PRINT"MPRESS ANY KEY"
2160 GETA$:IFA$=""GOTO2160
2170 POKE198,0:GOTO900

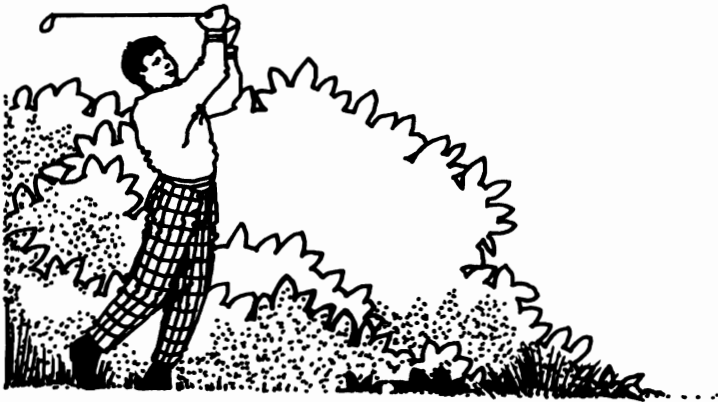
```



GOLF

Prepare yourself for the 18 holes of the Commodore Golf Course. You have two sets of clubs to help you get around the course under par. The first set is for use on the tee and the fairway, this being the stronger of the two sets of clubs; the other set is to use on, or near, the green. You enter a number in the range one to nine to select the strength of each shot.

The first, stronger set of clubs has a maximum range of around 300 yards. The weaker set has a maximum range of about 60 yards. You will be told the par for each hole as you play it, and you will also be reminded of your current total and the related par. At the end you will be told your score, and the par for the 18 holes. You will hear the ball being hit, and bouncing to a stop as each shot is played.



40 • GAMES FOR YOUR VIC20

```

100 N=RND(-TI):POKE36878,15
110 DEFFNR(X)=INT(RND(1)*X):POKE36879,93:PRINT"CHR$(14)
120 GOSUB470
130 HO=1:TS=0:TP=0
140 S=0:PRINT"YOU ARE ON THE TEE AT HOLE"HO
150 D=FNR(250)+100:P=INT(D/60):IFP<2THENP=2
160 TP=TP+P
170 PRINT"MIT IS"D"YDS LONG,":PRINT"WITH A PAR OF"P
180 IFS>0THENPRINT"DISTANCE TO THE HOLE:"D:PRINT"SHOTS SO FAR:"S:IFD>40THENPRI
NT"ON THE FAIRWAY"
190 IFD<20THENPRINT"ON THE GREEN":GOTO210
200 IFD<40THENPRINT"IN SIGHT OF THE GREEN"
210 PRINT"SELECT STRENGTH(1-9)":POKE198,0
220 GETD$:IFD<"1"ORD$>"9"THEN220
230 S=S+1
240 D1=VAL(D$)*(<25+FNR(15)):IFD<40THEND1=INT(D1/6)
250 IFD<5ANDD<"3"THEND1=D
260 IFD1=0THEND1=1
261 POKE36877,150:POKE36876,200:FORN=1TO50:NEXT
262 POKE36876,0:POKE36877,0:FORN=1TOD1#7:NEXT
263 IFD1<20THEN270
264 FORN=1TOFNR(4)+1
265 FORN=1TO50#(5-N):NEXT
266 POKE36877,150:POKE36874,200:FORN=1TO50:NEXT:POKE36877,0:POKE36874,0
267 NEXT
270 PRINT"YOU HIT"D1"YDS"
280 D=ABS(D-D1):IFD<=1THEN310
290 FORN=1TO2000:NEXT
300 GOTO180
310 POKE36874,130
320 PRINT" PLOP"
330 IFS>1THEN340
333 PRINT" WOW!!!! A HOLE IN ONE"
335 FORN=1TO8:POKE36874,ABS(SGN(PEEK(36874))-1)*150:FORN=1TO300:NEXT:NEXT:GOTO35
0
340 PRINT"YOU TOOK"S"SHOTS"
350 PRINT"THE PAR WAS"P
360 TS=TS+S
370 PRINT"SHOTS SO FAR:"TS:PRINT"TOTAL PAR IS"TP
380 POKE36874,0
390 HO=HO+1:IFHO=19THEN410
400 GOSUB540:GOTO140
410 PRINT"TP=TS-TP:IFTP=0THENPRINT"YOU EQUALLED THE PAR"GOTO450
420 PRINT" YOU COMPLETED THE COURSE"ABS(TP);
430 IFTP>0THENPRINT"OVER."
440 IFTP<0THENPRINT"UNDER."
450 GOSUB540
460 RUN
470 PRINT" VIC GOLF "
480 PRINT" WHEN YOU ARE ON THE FAIRWAY YOUR CLUBS AREPOWERFUL ENOUGH TO HITOV
ER 300YDS";
490 PRINT" (CLUB 9)
500 PRINT" WHEN YOU ARE ON,OR IN SIGHT OF,THE GREEN THEY ONLY HIT AROUND 60
MAXIMUM."
510 PRINT" IT WILL TAKE SOME PRACTICE IN CLUB SEL- ECTION TO GET UNDER OR EQU
AL PAR ";
520 PRINT"SCORES."
530 GOTO540
540 PRINT" PRESS ANY KEY ":POKE198,0
550 GETD$:IFD$=""THEN550
560 RETURN

```

READY.

42 • GAMES FOR YOUR VIC20

```
460 PRINT"*****PRESS ANY KEY TO PLAY"
470 FORN=1TO1000:NEXT
480 IFPEEK(197)=64THEN480
490 RETURN
500 POKE36879,8:PRINT"*****";
510 FORN=1TO20:PRINTSP$;:NEXT
520 PRINT"*****";
530 FORN=7TO1STEP-1:PRINTMID$("*****",N,1)"*****";:NEXT
540 PRINT"*****SCORE"S"*****BALL"LV
550 RETURN
```

READY.

MOONBASE

You are on laser duty on the moonbase, and your job is to protect the blue city on the screen from an alien bomber. You are aided by a two-level concrete barrier. The alien will try to bomb his way through this to reach the city below. Your base serves two purposes:

1. To shoot down the alien
2. To cover up holes in the concrete barrier

You score points just for surviving, and gain a bonus if you manage to shoot the alien down. If you beat the high score, you can enter your name (up to eight letters). "Z" moves you left and "C" moves you right. Press F7 to fire.

```
0 00T096
1 POKE36874,150:FORX=P-22T07944STEP-22:CH=PEEK(X):POKEX,103:POKECL+X,7
2 IFCH=95THENSC=SC+500:POKE36874,0:GOSUB2000:Q=Q+.1:00T0100
3 POKEX,32:NEXT:POKE36874,0
4 GOTO230
10 POKE36877,150:FORX=PA+22T08164STEP22:CH=PEEK(X):POKEX,103:POKECL+X,7
15 IFCH=160THENPOKEX,32:POKE36877,0:00T0250
20 IFCH=105ORCH=102THENGOSUB2000:00T03000
25 POKEX,32:NEXT
96 HI=3000:CL=38400-7680:NA#="VIC-20"
97 DEFFN(X)=X-7680-22*INT((X-7680)/22)
98 POKE36879,9
99 LL=3:SC=0:Q=.2
100 GOSUB5000
105 POKE36870,15:GOSUB1000
110 PA=7778:Q=0
120 P=0054
200 IFPEEK(197)=63THEN1
210 N=P+2*(PEEK(197)=33)-(PEEK(197)=34)):IFN(0054ORN)0074THENN=P
220 POKEP,160:POKEP+1,160:POKECL+N,2:POKEN,105:POKECL+N+1,2:POKEN+1,95:P=N
230 SC=SC+1:PRINT"SCORE"SC
240 Q=Q+Q:IFQ<1THEN200
244 POKE36876,140
245 IFRND(1)<.15THENGOTO10
246 Q=0
250 NA=PA+2*(SGN(RND(1)-RND(1)))+22*INT(RND(1)+.2)
260 IFABS(FNK(NA)-FNK(PA))>2THENNA=PA
```

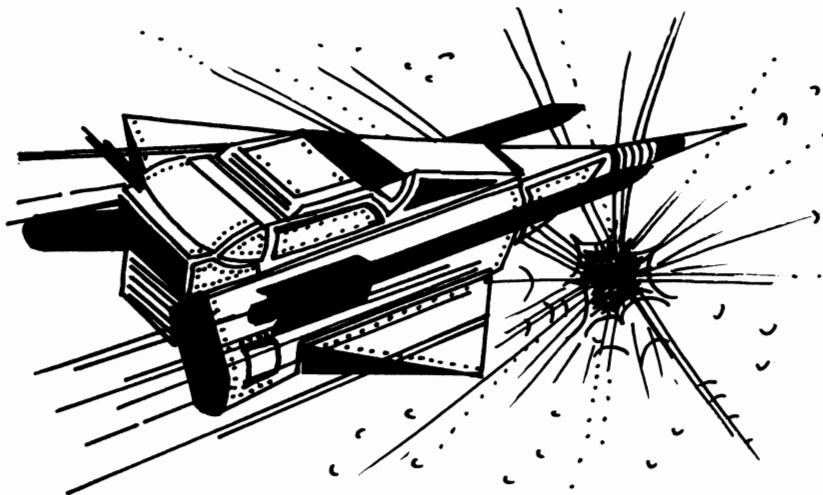
44 • GAMES FOR YOUR VIC20

```

270 IFNA=8054THENGOSUB2000:GOTO1200
280 POKEPA,32:POKEPA+1,32:POKECL+NA,4:POKENA,95:POKENA+CL+1,4:POKENA+1,105:PA=NA
290 POKE36876,0:GOTO200
997 POKE198,0
998 GETA$:IFA$=""THEN998
999 END
1000 PRINT"XXXXXXXXXXXXXXXXXXXX"
1010 PRINT"XXXXXXXXXXXXXXXXXXXX";
1020 PRINT"SCORE"SC:PRINT"HI"HI;NA$;"LL
1030 PRINT"XXXXXXXXXXXXXXXXXXXX":RETURN
1200 PRINT"XXXXXXXXXXXXXXXXXXXXGAME OVER"
1210 .IFSC>HITHENHI=SC:POKE198,0:INPUT"NAME";NA$:NA$=LEFT$(NA$,8)
1240 FORX=1TO3000:NEXT
1250 GOTO999
2000 POKE36877,100:FORX=15TO0STEP-1:POKE36878,X:FORN=1TO100:NEXT:NEXT:POKE36877,
0:RETURN
3000 LL=LL-1:IFLL>0THEN105
3010 GOTO1200
5000 PRINT"MOONBASE"
5010 PRINT"YOUR MISSION IS TO PROTECT YOUR MOONBASE (MOON) FROM THE ALIEN"
5020 PRINT"BOMBER HOVERING ABOVE"
5030 PRINT"THE MOONBASE IS PROTECTED BY TWO SHIELDS (SHIELD).YOU CAN HELP IN
TWO WAYS-"
5040 PRINT"1) REPAIR THE TOP SHIELD BY MOVING YOUR CANNON(CANNON) OVER IT"
5050 PRINT"2) BY DESTROYING THE ALIEN WHEN IT IS LOW ENOUGH"
5060 PRINT"PRESS ANY KEY":POKE198,0
5070 GETA$:IFA$=""THEN5070
5080 PRINT"TO MOVE -"
5090 PRINT"LEFT-SC":PRINT"RIGHT-SC":PRINT"FIRE-SF"
5100 PRINT"THE LONGER YOU SURVIVE,THE MORE YOU SCORE. 500 POINTS ARE SCORED FO
R ";
5110 PRINT"HITTING THE ALIEN"
5120 PRINT" A LIFE IS LOST IF THE ALIEN HITS THE MOON- BASE(MOON)OR YOUR CANNON"
5130 PRINT"THE GAME ENDS IF THE ALIEN LANDS"
5140 PRINT"PRESS ANY KEY TO PLAY"
5150 POKE198,0
5160 GETA$:IFA$=""THEN5160
5170 RETURN

```

READY.



LIFE

This is one of the most famous computer simulations, in which the evolution of a colony of cells is created on the screen. The life, death and birth of cells follows the rules laid down by John Conway, who invented the game:

1. If a cell is surrounded by less than two other cells it will die of loneliness
2. If a cell has two or three neighbors it will survive to the next generation
3. If an empty space has three neighbors a new cell will be born in that space in the next generation
4. If a cell has four or more neighbors it will die from overcrowding

You enter your own pattern of cells at the start of the game. To add a cell, you enter the row and column of the position in which you want it to appear. If you change your mind, you can delete that cell by entering the coordinates again. Enter "0" for the "across" number to start the process of LIFE.

You have three controls which you can use while the generations are evolving (note that these keys must be held down until the required reaction is seen). The controls, and their effects, are:

F1—this delays generation development until the key is released. It is useful if you wish to study a certain pattern

F3—this clears all cells and restarts the game

F5—this restarts the game with the current pattern intact

46 • GAMES FOR YOUR VIC20

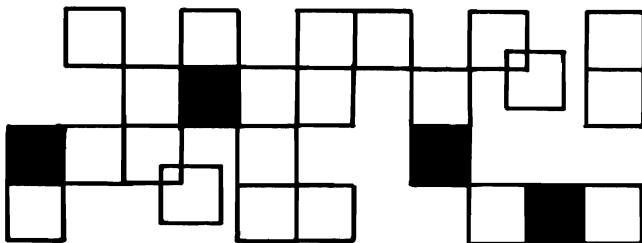
READY.

```

100 POKE36879,93:PRINT"J";
110 DIMM(1,10,10)
120 G=0
130 GOSUB250
140 INPUT"MACROSS":X:IFX=0THEN180
150 IFX<10R<9THEN140
160 INPUT"DOWN":Y:IFY<10R<9THEN160
170 M(1,X,Y)=ABS(M(1,X,Y)-1):GOTO130
180 FORV=1TO9:FORX=1TO9:C=M(0,X-1,V-1)+M(0,X-1,V)+M(0,X-1,V+1)+M(0,X,Y-1)+M(0,X,
Y+1)
190 C=C+M(0,X+1,V-1)+M(0,X+1,V)+M(0,X+1,V+1)
200 IFC=3THENM(1,X,Y)=1
210 IFC<2ORC<3THENM(1,X,Y)=0
220 NEXT:NEXT
230 G=G+1:GOSUB250:GOTO180
240 END
250 IFG<2THENPRINT"J";
260 PRINT"      LIFE  "
270 IFG>0THENPRINT"      GENERATION:  "G
280 PRINT"      123456789 "
290 FORV=1TO9:X$=" "RIGHT$(STR$(V),1)+" ":PRINTX$;
300 FORX=1TO9:IFM(1,X,Y)=1THENPRINT" ";:GOTO320
310 PRINT" ";
320 M(0,X,Y)=M(1,X,Y):NEXT:PRINTX$:NEXT
330 PRINT"      123456789 "
340 C=PEEK(197):IFC>64THEN360
350 RETURN
360 POKE198,0:IFC=39THEN340
370 IFC=47THENRUN
380 IFC=55THEN120
390 GOTO350

```

READY.



FRUIT MACHINE

In this fruit machine simulator, you start with ten coins. These are shown in the "tray" at the bottom of the screen. Each shot costs you one coin, and there are two ways you can win:

1. Two of a kind, such as BELL-BELL-ANY
2. Three of a kind, such as SWORD-SWORD-SWORD

The amount you win depends on the winning fruit. BAR is the highest, and CHERRY is the lowest. Your winnings are paid into the tray at the bottom. The game ends if the tray fills or empties. There are two aids you can employ:

1. HOLD—when this occurs you can hold any or all of the reels by pressing the numbers one to three (left reel is one). You can cancel a hold on a reel by pressing the reel number a second time. Press the space bar when you wish to restart.
2. NUDGE—when this occurs you will be given one to three nudges. To nudge a reel, use the same key as you used for HOLD. If you do not wish to use all your nudges, press SPACE.

Note that the SPACE BAR is used as a "start button."

```
100 DEFFNR(X)=INT(X*8ND(1))
110 POKE36879,56:POKE36878,15
120 PRINT"FRUIT MACHINE"
130 PRINT"*****"
140 PRINT"*****"
150 PRINT"*****"
160 MC=10:DC=0:GOSUB610:DIR$(7):FORN=0T07:READ$(N):A$(N)=A$(N)+"IT":NEXT
170 DC=-1:GOSUB610
180 POKE36874,140:POKE36877,140
190 PRINT"*****":FORN=1T03:IFH(N)=0THENPRINT$(0)+" "":GOTO210
200 PRINT"*****":
210 NEXT
220 FORN=1T01000:NEXT
230 POKE36874,0:POKE36877,0
240 FORN=1T03:IFH(N)=0THENR(N)=FNR(7)+1
```

SPOTS FROM OUTER SPACE

In this arcade-style game (sounding like a particularly unpleasant disease) you must protect your home planet from invading spots. If you are hit by a missile dropped by the spots, or if the spots land, you will be killed. When you are killed, or you defeat all the spots, the game will automatically restart. Your laser base is controlled by the "Z" (left), "X" (right) and "*" (fire) keys.

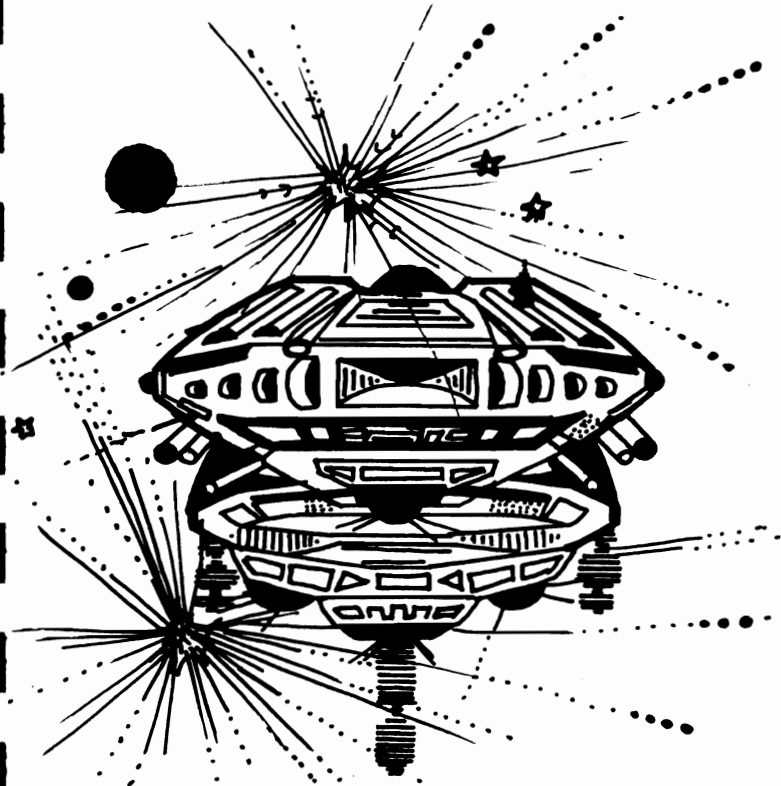
```
5 POKE650,255
10 PRINT"J"
15 F=38
16 M=12
20 POKE36867,34
25 H=INT(RND(1)*22)+7790
30 POKE36879,42
31 POKE198,8
32 PRINT"  SPOTS FROM OUTER SPACE"
33 PRINT"SKILL LEVEL?":PRINT"1-40"
34 INPUTG
38 PRINT"J"
40 A=7998
50 PRINT"  . . . . . "
60 PRINT"  . . . . . "
70 PRINT"  . . . . . "
80 PRINT"  . . . . . "
200 REM GENERAL ROUTINE
210 POKER,65
220 D=INT(RND(1)*8)+1
230 IFD=1 THENF=F+1:POKE36865,F
231 IFD=1 THENA=A-22:POKER+22,96
232 IFD=1 THENM=M+INT(RND(1)*3)-1
233 POKE36864,M
234 POKEH,96:H=H+22:POKEH,93
236 IFH>8010 THENH=INT(RND(1)*22)+7790
237 IFPEEK(H+22)=65 THEN700
240 IFPEEK(A)=81 THEN700
245 IFR<7680 THEN700
250 REM CONTROL
260 GETA$
270 IFA$="Z" THENA=A-1:POKER+1,32
280 IFA$="X" THENA=A+1:POKER-1,32
290 IFA$="*" THEN500
```

50 • GAMES FOR YOUR VIC20

```
300 GOTO200
500 REM FIRE
510 B=A-22
520 POKEB,32:B=B-22
530 POKEB,34
535 POKEH,96:H=H+22:POKEH,93
536 IFH>8032 THENH=INT(RND(1)*22)+7790
537 IFPEEK(H+22)=65 THEN700
540 IFPEEK(B-22)=81 THEN500
542 IFB=H THENH=INT(RND(1)*22)+7790
543 IFB=H THEN200
545 IFB<7678 THEN200
550 GOTO520
580 POKEB,32:POKEB-22,32
582 C=C+1:IFC=42 THEN600
590 GOTO200
600 PRINT"JIMMELL DONE!!"
610 POKE36867,46
620 POKE36865,38
630 GOTO720
700 PRINT"THE'VE KILLED YOU!!!!"
710 POKE36867,46:POKE36865,38
720 FORM=1TO2000:NEXT:RUN
```

DESTROYER

This program was written by Adam Burbidge. A spaceship (a greater-than symbol) will move from left to right against a background of deepest space. The object of the game is to destroy hostile aliens who will blast your ship to smithereens



unless you are very careful. An alien can only be destroyed if it lies within five spaces directly in front of you. Points are scored according to the color of the aliens. "A" moves your ship down, "B" moves it up, and you press "P" to fire.

Warning: the aliens move very quickly, and their advanced technology allows them to become invisible from time to time. They can still damage you while invisible.

```

100 DIMZ(5),A(9)
110 REM##DESTROYER
120 REM##ADAM BURBIDGE
130 I=3:D=INT(RND(1)*10)
140 K(1)=0:Q=7900
150 PRINT"J"
160 B=B+1
170 C=INT(RND(1)*506)+7600
180 A(B)=C
190 D=D-1:IFD<1THENZ10
200 GOTO160
210 POKE36879,0
220 POKE0,62
230 E=0
240 F=A(E)
250 E=E+1
260 POKEF,35
270 Z(2)=INT(RND(1)*0)
280 POKEF+30720,Z(2)
290 IFE=9THENZ10
300 GOTO240
310 Z(0)=INT(RND(1)*7)
320 GOSUB1070
330 POKE0,32:FORT=1TO100:NEXT
340 POKE0,32:FORT=1TO100:NEXT
350 G=0+1:K(0)=K(0)+1
360 IFK(0)>22THENK(0)=0
370 IFPEEK(Q-22)=35THENZ20
380 IFPEEK(Q+1)=35THENZ20
390 GETH#
400 IFQ<7600THENQ=0165
410 IFQ>8165THENQ=7600
420 IFH#=""THENH#470
430 IFH#="A"THENQ=Q+22
440 IFH#="D"THENQ=Q-22
450 POKE0,62
460 IFH#="P"THENQ20
470 POKE0,62
480 GOTO490
490 B=INT(RND(1)*9)
500 POKEA(B),32
510 C=INT(RND(1)*4)+1
520 IFC=4THENA(B)=A(B)+22
530 IFC=3THENA(B)=A(B)-22
540 IFC=2THENA(B)=A(B)-1
550 IFC=2THENA(B)=A(B)+1
560 IFA(B)<7600THENA(B)=7600
570 IFA(B)>8165THENA(B)=A(B)+22
580 IFA(B)>8165THENA(B)=A(B)-22
590 POKEA(B)+30720,INT(RND(1)*7)
600 POKEA(B),35
610 GOTO330

```

```

620 IFPEEK(G+1)=35THENPOKEG+1,102:GOTO760
630 IFPEEK(G+1)=210THENPOKEG+1,102:SC=SC+20:GOTO700
640 IFPEEK(G+2)=35THENPOKEG+2,102:GOTO700
650 IFPEEK(G+2)=210THENPOKEG+2,102:SC=SC+20:GOTO720
660 IFPEEK(G+3)=35THENPOKEG+3,102:GOTO800
670 IFPEEK(G+3)=210THENPOKEG+3,102:SC=SC+20:GOTO820
680 IFPEEK(G+4)=35THENPOKEG+4,102:GOTO840
690 IFPEEK(G+4)=210THENPOKEG+4,102:SC=SC+20:GOTO860
690 GOTO330
700 SC=SC+PEEK(G+30721)*100
710 GOTO850
720 SC=SC+PEEK(G+30722)*100
730 GOTO850
740 SC=SC+PEEK(G+30723)*100
750 GOTO850
760 SC=SC+PEEK(G+30721)*10
770 GOTO850
780 SC=SC+PEEK(G+30722)*10
790 GOTO850
800 SC=SC+PEEK(G+30723)*10
810 GOTO850
820 SC=SC+PEEK(G+30724)*10
830 GOTO850
840 SC=SC+PEEK(G+30725)*10
850 POKE36877,120
860 POKE36878,15
870 A(B)=0
880 FORT=1T0300:NEXT
890 POKE36877,0
900 SC=SC+10
910 GOTO330
920 POKEG,102
930 POKE36877,220
940 FORL=15T00STEP-1
950 POKE36876,INT(RND(1)*120)+120
960 POKE36878,L
970 FORM=1T0100
980 NEXT
990 NEXT
1000 POKE36877,0
1010 POKE36876,0
1020 PRINT"SCORE:"SC
1030 I=I-1
1040 IFI=0THENPRINT"GAME OVER" :CLR:I=3:D=INT(RND(1)*10):FORT=1T01000:N
EXT
1050 FORT=1T01000:NEXT
1060 GOTO140
1070 Z(1)=INT(RND(1)*22)+0143
1080 Z(3)=INT(RND(1)*3)+4
1090 POKEZ(1),210
1100 POKEZ(1)+30720,Z(3)
1110 RETURN

```

READY.

CAR RACE

Kevin Smith's game puts you at the wheel, the aim being to drive your car for as long as you can before the inevitable crash when your attention wavers.

The controls you have on your VIC-mobiles are:

"Z"—right

"X"—left

Cursor down—accelerate

Cursor right—brake

First gear—F1

Second—F3

Third—F5

Fourth—F7

```
1 PRINT"J CAR RACE",,"M2=LEFT:X=RIGHT"
2 PRINT"DEERS",,"F1=FIRST:F4=FOURTH":PRINT"ROAD THICKNESS?"
3 PRINT"[1,2,3,4]"
4 GETC$:IFC$=""THEN4
5 C=7690:B=8
6 POKE36879,40
7 I=100
8 POKE50,255
9 PRINT"J"
10 K=36074
11 A=INT(RND(1)*3)-1
12 POKEK,94
13 POKEK,0
14 POKEK,94
15 POKEK,0
16 POKEK,INT(RND(1)*10)+106
17 E=C+D
18 POKEK,0
19 POKE36070,15
20 POKEK,109+INT(RND(1)*6)
21 POKEE,00
22 POKE36074,0
23 GETB$
24 IFB$="Z" THEND=D-1
25 IFB$="X" THEND=D+1
26 IFB$="C" THENB1
27 GOTO82
28 GETB$:IFB$=""THENB1
29 IFB$="M" THENI=100
```



```

84 IFB$="I" THENI=70
86 IFB$="II" THENI=55
88 IFB$="III" THENI=25
90 IFB$="M" THEN500
92 IFB$="J" THENI=1-5
100 B=B+A
102 POKEK,93
110 IFB<I THEN B=B+1
111 IFB>15 THEN B=B-1
115 O=O+1
116 POKEK,0
117 IFC$="1" THENPRINTTAB(B)"  "
118 IFC$="2" THENPRINTTAB(B)"  "
119 IFC$="3" THENPRINTTAB(B)"  "
120 IFC$="4" THENPRINTTAB(B)"  "
121 POKE36870,15:POKE36874,193
123 IFPEEK(E)=102 THEN300
125 POKEE,00
126 POKE36874,0
128 POKEK,192
129 POKEK,0
130 FORI=1TOI:NEXT
132 POKEK,95
150 GOTO20
300 POKE36870,12
305 POKE36874,0
310 FORL=1TO10
320 FORM=250 TO 240 STEP-1
330 POKE36876,M:NEXTM
340 FORM=240TO250
350 POKE36876,M:NEXTM
360 POKE36876,0:NEXTL
370 POKE36870,0
375 PRINT"!"
380 PRINT"YOU KEPT GOING FOR ";O/40;" MILES!!"
390 FORM=1TO2000:NEXT:RUN
500 I=I+35
505 POKE36870,13
510 FORL=235TO245
520 POKE36876,L
530 FORM=1TO10:NEXTM
540 NEXTL
550 POKE36876,0
560 GOTO100

```

READY.



SUB CHASE

You are the captain of the HMS Victory, a CBM class sub hunter, and your mission is to destroy the enemy subs lurking below you. You are armed with a limitless supply of depth charges; but stay alert, because the subs can hit back with their missiles.

A running total is kept of the number of subs sunk, as well as the number of hits the subs have scored on your ship. The two totals are shown at the top of the screen. Use the following controls for your ship:

“Z”—Move left

“C”—Move right

“B”—Drop depth charge

Note that only one depth charge can be in view at a time.

```
1 REM#SUB CHASE
2 REM#K. STAPLES
20 S3=36876: S2=36875: S=36877: V=36878: OS#=""
100 PRINT"CHOOSE HOW HARD YOU WANT THE GAME(0-1) M1 IS THE HARDEST"
110 INPUT"CHOOSE GAME";G: IFG<1000>GTHEN110
120 PRINT"### SUB CHASE ###"
130 PRINT"Z-LEFT, C-RIGHT, B-D/CS#"
140 FORI=1TO374: POKE7811+I, 102: POKE38531+I, 3: NEXT
150 GOSUB1000: GOSUB2000: U(1)=T3: DI(1)=T2: GOSUB2000: U(2)=T3: DI(2)=T2
160 IFPEEK(197)=34THENR=A+1: GOSUB1000
170 IFPEEK(197)=33THENR=A-1: GOSUB1000
180 IFDI(1)=1THEN230
190 U(1)=U(1)+1: T(1)=T(1)+1
200 IFT(1)=20THENU=U(1): GOSUB2100: T(1)=0: GOSUB2000: U(1)=T3: DI(1)=T2
210 POKEU(1)-2, 102: POKEU(1)-1, 60: POKEU(1), 149: POKEU(1)+1, 62
220 POKEU(1)-23, 102: POKEU(1)-22, 93: GOTO500
230 U(1)=U(1)-1: T(1)=T(1)+1
240 IFT(1)=20THENU=U(1): GOSUB2100: T(1)=0: GOSUB2000: U(1)=T3: DI(1)=T2
250 POKEU(1)-1, 60: POKEU(1), 149: POKEU(1)+1, 62: POKEU(1)+2, 102
260 POKEU(1)-22, 93: POKEU(1)-21, 102
500 IFDC=1THENC1=C1+22: GOTO520
510 IFPEEK(197)=35THENDC=C1: C1=7836+A
520 IFDC=1THENPOKEC1, 81: POKEC1-22, 102: IFC1>8164THENPOKEC1, 102: DC=0
530 IFU(1)=C1THENU=U(1): GOSUB3000: GOSUB3600
540 IFU(2)=C1THENU=U(2): GOSUB3000: GOSUB3600
600 IFMR=1THENM=M-22: GOTO620
610 GOSUB4000
```

```

620 IFM=1THENPOKEM,42:POKEM+22,102:IFM<7811THENPOKEM,32:MR=0
630 IFM=7792+AORM=7791+AORM=7793+ATHENU=M:GOSUB3000:U=M+88:GOSUB5000
780 IFD(2)=1THEN830
790 U(2)=U(2)+1:T(2)=T(2)+1
800 IFT(2)=20THENU=U(2):GOSUB2100:T(2)=0:GOSUB2000:U(2)=T3:DI(2)=T2
810 POKEU(2)-2,102:POKEU(2)-1,60:POKEU(2),149:POKEU(2)+1,62
820 POKEU(2)-23,102:POKEU(2)-22,93:GOTO999
830 U(2)=U(2)-1:T(2)=T(2)+1
840 IFT(2)=20THENU=U(2):GOSUB2100:T(2)=0:GOSUB2100:U(2)=T3:DI(2)=T2
850 POKEU(2)-1,60:POKEU(2),149:POKEU(2)+1,62:POKEU(2)+2,102
860 POKEU(2)-22,93:POKEU(2)-21,102
870 IFMS=1THENM1=M1-22:GOTO890
880 GOSUB4100
890 IFMS=1THENPOKEM1,42:POKEM1+22,102:IFM1<7811THENPOKEM1,32:MS=0
900 IFM1=7792+AORM1=7791+AORM1=7793+ATHENU=M1+88:GOSUB7000
999 GOTO160
1000 POKEV,15:FORJ=221TO244:POKES2,J
1003 FORJ1=1TO5:NEXTJ1,J:POKES2,0:POKEV,0
1005 IFC0THENA=0
1010 IFAD17THENA=17
1020 PRINT"#####TAB(A)" | "
1030 PRINT"#####TAB(A)" | "
1040 PRINT"#####TAB(A)" | "
1050 RETURN
2000 T2=INT(RND(1)*2)+1:IFT2=1THEN2030
2020 T3=7857+(INT(RND(1)*13)+1)*22:RETURN
2030 T3=7876+(INT(RND(1)*13)+1)*22:RETURN
2100 POKEU-2,102:POKEU-1,102:POKEU,102:POKEU+1,102:POKEU+2,102
2120 POKEU-23,102:POKEU-22,102:POKEU-21,102:POKEU-44,102:RETURN
3000 POKEU-1,77:POKEU,93:POKEU+1,70:POKEU-22,93
3510 POKE36879,47:POKES,220
3520 FORJ=15TO0STEP-.05:POKEV,J:NEXT
3530 POKES,0:POKE36879,27:RETURN
3600 POKEV,15:POKES,254
3610 POKEU-1,60:POKEU,149:POKEU+1,62
3620 POKEU-23,102:POKEU-22,93:POKEU-21,102:POKEU-44,102
3630 IFU>8164THENGOSUB3000:GOSUB2100:GOSUB2000:GOTO3650
3640 U=U+22:GOTO3600
3650 IFU(1)=C1THENU(1)=T3:DI(1)=T2:DC=0:C1=0:T(1)=0:SS=SS+1:PRINT"#####SS:RET
URN
3660 IFU(2)=C1THENU(2)=T3:DI(2)=T2:DC=0:C1=0:T(2)=0:SS=SS+1:PRINT"#####SS
4000 R1=INT(RND(1)*9)+1:IFR1>GTHENMR=1:M=U(1)
4020 RETURN
4100 R1=INT(RND(1)*9)+1:IFR1>GTHENMS=1:M1=U(2)
4120 RETURN
5000 POKEV,15:POKES,254
5010 PRINT"#####OS#:PRINT"#####OS#:PRINT"#####OS#
5020 POKEU-1,95:POKEU,160:POKEU+1,105:POKEU-22,160:POKEU-44,93
5030 POKEU-23,102:POKEU-21,102:POKEU-66,102
5040 IFU>8164THENGOSUB3000:GOTO5060
5050 U=U+22:GOTO5020
5060 GOSUB2100
5070 A=0:DS=DS+1:PRINT"#####SHIPS"DS:GOSUB1000:MR=0:M=0:RETURN
6000 POKEV,15:POKES3,220:FORJ1=1TO5:NEXT:POKES3,0:POKEV,0:RETURN
7000 POKEV,15:POKES,254
7010 PRINT"#####OS#:PRINT"#####OS#:PRINT"#####OS#
7020 POKEU-1,95:POKEU,160:POKEU+1,105:POKEU-22,160:POKEU-44,93
7030 POKEU-23,102:POKEU-21,102:POKEU-66,102
7040 IFU>8164THENGOSUB3000:GOTO7060
7050 U=U+22:GOTO7020
7060 GOSUB2100
7070 A=0:DS=DS+1:PRINT"#####SHIPS"DS:GOSUB1000:MS=0:M1=0:RETURN

```

READY.

REVERSI

In this program (much like the game Othello), written by Mike O'Neill, you play against the computer which can put up a pretty mean fight. The program uses simple but very effective algorithm where each square is allocated a "priority" from one to sixty. The computer looks at each square on the board, in order of priority, until one is found where a move is possible. The program checks that you're not cheating, and will know if a move is not possible.

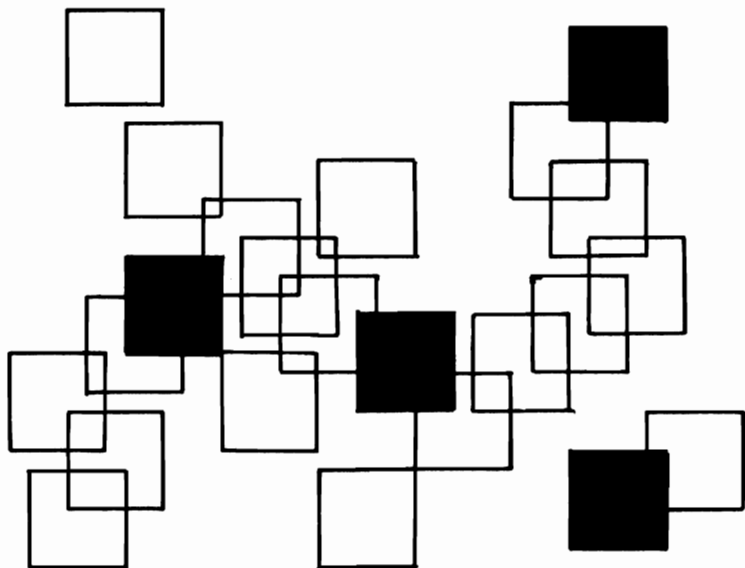
If the computer has already moved, but you can't move, it will move twice. If you have just moved, and there are no moves for it, you will get another go. This will only happen rarely. If neither player can move, the game is over. At the end of the game the final score is shown and you have the option of starting a new game, or terminating the program.

```
1 S2=36875:VV=36878:POKE36879,89:PRINT"J"
10 DIMB%(8,8),AX(8),AV(8),AM%(60),AY%(60)
11 FORI=1TO60:READAX%(I),AV%(I):NEXT
12 DATA1,1,1,8,8,8,8,1,3,1,6,1,8,3,8,6,6,8,3,8,1,6,1,3,3,3,6,3,6,6,3,6
13 DATA4,1,5,1,8,4,8,5,5,8,4,8,1,5,1,4,4,3,5,3,6,4,6,5,5,6,4,6,3,5,3,4
14 DATA4,2,5,2,7,4,7,5,5,7,4,7,2,5,2,4,3,2,6,2,7,3,7,6,6,7,3,7,2,6,2,3
15 DATA7,1,8,2,8,7,7,8,2,8,1,7,1,2,2,1,7,2,7,7,2,7,2,2
17 FORI=1TO8:READAX(I),AV(I):NEXT
18 DATA1,-1,1,0,1,1,0,1,-1,1,-1,0,-1,-1,0,-1
20 FORI=1TO8:FORJ=1TO8:B%(I,J)=0:NEXTJ,I
21 B%(4,4)=1:B%(5,5)=1:B%(4,5)=2:B%(5,4)=2:SS=0
25 GOSUB8000:GOSUB1000
98 PRINT"DO YOU WANT TO":INPUT"OO FIRST(Y OR N)":A$:IFA$="N"THEN200
99 GOSUB9999:GOTO110
100 GOSUB5000
110 GOSUB2000
111 P=7770+X#2+Y#44:IFB%(X,Y)C0THENGOSUB6500:GOTO110
112 POKEP+30720,0:POKEP,81:SX=X:SY=Y:IS=1:TK=1:GOSUB3000
130 IFV=0THENPOKEP,32:GOSUB6500:GOTO110
140 B%(SX,SY)=2:GOSUB9500
200 GOSUB9999:DDX=0:FORJ=1TO60
220 SX=AX%(J):SY=AV%(J):I1=0:TK=0:IS=2:GOSUB3000:IFV=1THEN300
250 NEXT:GOSUB9000:GOTO100
300 P=7770+SX#2+SY#44:POKE30720+P,1
305 POKES2,225:FORI1=1TO10
310 POKEVV,15:POKEP,32:FORI2=1TO50:NEXT
```

```

315 POKEV,0:POKEP,81:FORI2=1TO50:NEXT
320 NEXT
340 TK=1:IS=2:GOSUB3000:B%(SX,SY)=1:GOSUB9500:GOTO100
1000 POKE38674,1:POKE7954,81
1010 POKE38676,0:POKE7956,81
1020 POKE38718,0:POKE7998,81
1030 POKE38720,1:POKE8000,81
1200 RETURN
2000 GOSUB9999
2005 DD%=0
2010 INPUT"MCOLUMN":X
2020 INPUT" ROW":Y
2030 IFX<1ORX>8ORY<1ORY>8THENGOSUB6500:GOTO2000
3000 V=1:IFB%(SX,SY)<>0THENV=0:RETURN
3010 T0=0
3020 FORI=1TO8
3030 TI=-1:TX=SX:TY=SY
3040 TX=TX+RX(I):TY=TY+RY(I):TI=TI+1:IFTX<1ORTX>8ORTY<1ORTY>8THEN3200
3050 IFB%(TX,TY)=0THEN3200
3060 IFB%(TX,TY)=1STHEN3040
3070 T0=T0+TI
3080 IFTX=0ORTI=0THEN3200
3090 TX=TX-RX(I):TY=TY-RY(I)
3100 IFTX=8XANDTY=8VTHEN3200
3110 B%(TX,TY)=3-B%(TX,TY)
3120 I1=7770+TX#2+TY#44
3130 IFIS=1THENPOKE30720+I1,0:GOSUB7000
3140 IFIS=2THENPOKE30720+I1,1:GOSUB7500

```



60 • GAMES FOR YOUR VIC20

```

3100 GOTO3090
3200 NEXT
3210 IFT0=0THENV=0
3220 RETURN
5000 FORSX=1TO8:FORSY=1TO8
5010 TK=0:IS=1:GOSUB3000
5020 IFV<>0THENRETURN
5030 NEXTSY,SX
5040 IFDD%=1THEN10000
5045 DD%=1:PRINT"  YOU CAN'T MOVE":PRINT"      I'LL GO AGAIN  "
5050 FORI=1TO3000:NEXT:GOTO200
6500 POKES2,195:FORI1=1TO10
6510 PRINT"ILLEGAL MOVE"
6520 POKEV,15:FORI2=1TO50:NEXT:PRINT" ]"      ]":POKEV,0:FORI2=1TO50:NEXTI
2,I1
6550 RETURN
7000 POKES2,0:POKEV,15:FORI2=195TO240STEP0.4:POKES2,I2:NEXT:POKEV,0:RETURN
7500 POKES2,0:POKEV,15:FORI2=240TO195STEP-0.4:POKES2,I2:NEXT:POKEV,0:RETURN
8000 PRINT"XXXXXXXX 1 2 3 4 5 6 7 8"
8010 PRINT"
      | | | | | | | | | |
8020 A$=" | | | | | | | | | |"
8030 B$=" | | | | | | | | | |"
8040 FORI=1TO7:PRINTI;A$:PRINTB$:NEXT
8050 PRINT" 8 ";A$:PRINT"
      | | | | | | | | | |"
8200 RETURN
9000 IFDD%=1THEN10000
9010 PRINT"  I CAN'T MOVE":PRINT" YOU HAVE ANOTHER GO":FORI=1TO1000:NEXT:DD%=
1:RETURN
9500 SS=SS+1:IFSS<60THENRETURN
9650 I1=0:I2=0
9660 FORI=1TO8:FORJ=1TO8
9670 IFB%(I,J)=1THENI1=I1+1
9680 IFB%(I,J)=2THENI2=I2+1
9690 NEXTJ,I
9695 GOSUB9999
9700 IFI1>I2THEN9750
9710 IFI2>I1THEN9800
9730 PRINT"  IT'S A DRAW !!":GOTO9900
9750 PRINT"  I WIN  ";I1:"TO";I2:GOTO9900
9800 PRINT"  YOU WIN  ";I2:"TO";I1:GOTO9900
9900 INPUT"  ANOTHER GAME(Y/N)  [ ]";A$:IFASC(A$)=78THENPRINT"]":POKE36879,27:
END
9905 IFASC(A$)=89THENRESTORE:CLR:GOTO1
9910 GOTO9900
9995 PRINT"]":END
9999 FORI=7680TO7745:POKEI,32:NEXT:RETURN
10000 PRINT"  NEITHER OF US CAN MOVE      END OF GAME      ":FORI=1TO3000:NEXT:GOTO
9650

```

READY.

APOLLO II

In this highly effective graphics game, you are the commander of the lander module VIC-20-CBM. You must guide your lander onto the landing pad, which moves from game to game. If you crash onto the surface that particular game is over, though it will automatically restart. If you are successful you will be given another landing to attempt, only this time it will be more difficult. This process will continue until you finally fail to land. You move left with the "Z" key, and right with the "X".

READY.

```
0 I=160:POKE650,255
1 PRINT"#####"
2 PRINT"#####"
3 B=6:I=I-10
4 A=INT(RND(1)*19+1):PRINTTAB(A):"I"
5 PRINT"|"
6 I=I-10
15 GETA$:IFA$="Z" THENB=B-1
16 IFA$="X" THENB=B+1
17 IFA$="|" THENI=10
18 IFA$="|" THENI=50
20 PRINTTAB(B):" |"
30 FORF=1 TO I+20:NEXT
32 POKE36878,15
33 POKE36876,INT(RND(1)*10+230)
35 PRINT"|"
36 K=K+1:IFK=19 THENA0
37 GOTO15
40 IFB=A THENA2
41 GOTO1000
42 PRINTTAB(B):" |"
43 PRINT"#####GOOD LANDING":J=J+1:A=0:B=0:K=0
44 POKE36876,0:F0R=1T01000:NEXT:GOTO1
1000 POKE36876,0
1010 POKE36877,220:F0R=15T00 STEP-1:POKE36878,L
1020 F0R=1T0100:NEXTM
1030 NEXTL
1040 POKE36877,0:POKE36878,0
1050 PRINT"#####GOODBYE ASTRONAUT!!"
1060 F0R=1T02000:NEXT:RUN
```

READY.

SOUND EFFECTS

Turn up the volume on your TV and brace yourself. Ken Staples' program is about to launch an assault on your ears. This program bears a number of similarities to one of the programs in your user manual. Compare them and see how the program writer has improved and changed the program.

```
1 REM##SOUND EFFECTS
2 REM##KEN STAPLES
10 S1=36874:S2=36875:S3=36876:S4=36877:V=36878:MC#="#####"
15 ER$="
20 PRINT"##### SOUND EFFECTS ###"
30 PRINT"####SPC(6)"EXPLOSION":FORW=1T01000:NEXT
40 POKEV,15:POKES4,255:FORI=15T008STEP-.01:POKEV,I: NEXT:POKES4,0
100 PRINTMC#ER$:PRINTMC#SPC(4)"WHISTLING BOMB":FORW=1T01000:NEXT
110 POKEV,5:FORI=230T0128STEP-1:FORJ=1T020:NEXT:POKES3,I: NEXT:POKES3,0:POKES4,25
5
120 FORI=15T008STEP-.05:POKEV,I: NEXT:POKES4,0
170 POKEV,15:FORI=1T015:FORJ=200T0220+I#2:POKES3,J: NEXTJ,I:POKES3,0:POKEV,0
200 PRINTMC#ER$:PRINTMC#SPC(4)"MUSICAL NOTES":FORW=1T01000:NEXT
210 POKEV,15:FORI=250T0200STEP-2:POKES3,I:FORJ=1T0100:NEXTJ,I:FORJ=205T0250STEP2
220 POKES3,I:FORJ=1T0100:NEXTJ,I:POKES3,0:POKEV,0
400 PRINTMC#ER$:PRINTMC#SPC(6)"BEDLAM":FORW=1T01000:NEXT
410 POKEV,15:FORI=1T0100:POKES3,INT(RND(1)*128)+128:FORJ=1T010:NEXTJ,I:POKES3,0:
POKEV,0
500 PRINTMC#ER$:PRINTMC#SPC(6)"RED ALERT":FORW=1T01000:NEXT
510 POKEV,15:FORI=1T010:FORJ=100T0235STEP2:POKES3,J:FORJ1=1T010:NEXTJ1,J:POKES3,
0
520 FORJ1=1T0100:NEXTJ1,I:POKEV,0
700 PRINTMC#ER$:PRINTMC#SPC(6)"LASER BEAM":FORW=1T01000:NEXT
710 POKEV,15:FORI=1T030:FORJ=250T0240STEP-1:POKES3,J: NEXTJ:FORJ=240T0250:POKES3,
J
720 NEXTJ:POKES3,0: NEXT:POKEV,0
900 PRINTMC#ER$:PRINTMC#SPC(6)"POLICE":FORW=1T01000:NEXT
910 POKEV,15:FORI=1T010:POKES2,235:FORJ=1T0300:NEXT:POKES2,0:POKES1,235:
920 FORJ=1T0300:NEXT:POKES1,0: NEXTI:POKEV,0
1000 PRINTMC#ER$:PRINTMC#SPC(6)"TELEPHONE":FORW=1T01000:NEXT
1010 POKEV,15:FORI=1T05:FORJ=1T050:POKES3,228:FORJ1=1T08:NEXT:POKES3,0:NEXT
1020 FORJ1=1T01500:NEXTJ,I:POKEV,0
1100 PRINTMC#ER$:PRINTMC#SPC(5)"ENGAGED TONE":FORW=1T01000:NEXT
1110 POKEV,5:FORI=1T015:POKES3,200:FORJ=1T0500:NEXT:POKES3,0:FORJ=1T0500:NEXTJ,I:
POKEV,0
1200 PRINTMC#ER$:PRINTMC#SPC(9)"BIRDS":FORW=1T01000:NEXT
1205 POKEV,15:FORI=1T040:FORJ=254T0240+INT(RND(1)*10)STEP-1:POKES3,J: NEXT:POKES3,
0
1210 FORJ=0T0INT(RND(1)*100)+120:NEXTJ,I:POKEV,0
1300 PRINTMC#ER$:PRINTMC#SPC(5)"QUARTERMASS":FORW=1T01000:NEXT
1310 FORI=0T015:POKEV,I:FORJ=220T0240:POKES1,J:FORJ1=1T010:NEXTJ1,J,I:POKES4,240
```

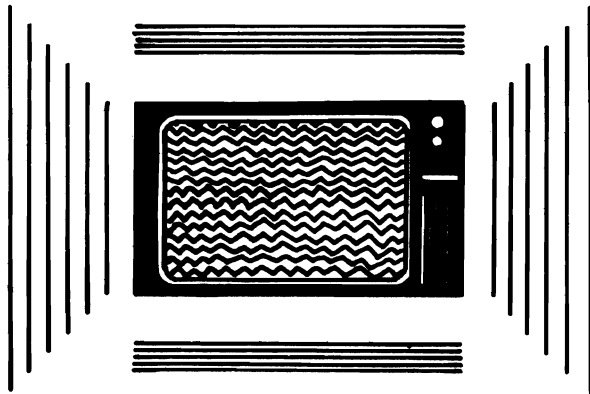


```

1320 FORI=1T0200:NEXT:FORI=254T0200STEP-2:POKES1,I:NEXT:FORI=15T00STEP-.05:POKEV
,I:NEXT
1330 POKES4,0:POKES1,0
1400 PRINTM#ER#:PRINTM#SPC(5)"OCEAN WAVES":FORM=1T01000:NEXT
1410 POKES4,135:FORI=1T010:D=INT(RND(1)*90)+30:FORJ=0T010:POKEV,J:FORJ1=1T0D:NEX
TJ1,J
1415 FORJ=15T00STEP-1:POKEV,J:FORJ1=1T0D:NEXTJ1,J,I:POKES4,0:POKEV,0
1500 PRINTM#ER#:PRINTM#" VANISHING UFO":FORM=1T01000:NEXT
1510 POKEV,15:FORI=130T0254:POKES3,I:FORJ=1T040:NEXTJ,I:POKEV,0:POKES3,0
1600 PRINTM#ER#:PRINTM#SPC(5)"UFO LANDING":FORM=1T01000:NEXT
1605 POKEV,15:FORI=1T020:FORJ=220-1T0160-1STEP-4:POKES3,J:NEXTJ
1610 FORJ=160-1T0220-1STEP4:POKES3,J:NEXTJ,I:POKEV,0:POKES3,0
1700 PRINTM#ER#:PRINTM#SPC(5)"UFO SHOOTING":FORM=1T01000:NEXT
1710 POKEV,15:FORI=1T015:FORJ=200T0220+I#2:POKES3,J:NEXTJ,I:POKES3,0:POKEV,0
1800 PRINTM#ER#:PRINTM#SPC(5)"WOLF WHISTLE":FORM=1T01000:NEXT
1810 POKEV,15:FORI=150T0240:POKES3,I:NEXT:POKES3,0:FORJ1=1T0200:NEXT
1820 FORI=205T0120STEP-.3:POKES3,I:NEXT:POKES3,0:POKEV,0
1900 PRINTM#ER#:PRINTM#SPC(6)"DANGER!!!":FORM=1T01000:NEXT
1910 POKEV,15:FORI=1T030:POKES1,220:FORJ=1T0100:NEXT:POKES1,0:FORJ=1T0100:NEXTJ,
I
1920 POKEV,0
2000 PRINTM#ER#:PRINTM#SPC(0)"BLIPS":FORM=1T01000:NEXT
2005 FORJ=1T020:POKEV,15:POKES3,220:FORI=1T05:NEXT:POKES3,0:FORI=1T0500:NEXT:POK
ES3,200
2010 FORI=1T05:NEXT:POKES3,0:FORI=1T0500:NEXTI,J:POKEV,0
3000 PRINTM#ER#:PRINTM#SPC(3):PRINT"LEAVING HARBOUR":FORM=1T01000:NEXT
3010 FORI=1T03:POKEV,15:POKES1,210:POKES4,254:FORJ=1T01500:NEXT
3020 POKES1,0:POKES4,0:POKEV,0:FORJ=1T0500:NEXTJ,I
4000 PRINTM#ER#:PRINTM#SPC(5)"INTO ACTION":FORM=1T01000:NEXT
4010 FORT=1T02:POKEV,15:FORI=140T0220STEP5:POKES3,I:POKES4,254:FORJ=1T05:NEXTJ,
I
4020 POKEV,0:POKES4,0:POKES3,0:FORM=1T0100:NEXTM,T:FORM=1T02000:NEXT:A=A+1:IFA=3
THEN5000
4030 GOTO4010
5000 PRINTM#ER#:PRINTM#SPC(5)"# THE END #":PRINTSPC(5);:FORI=1T011:PRINT"";:N
EXT:PRINT
5010 FORI=1T04000:NEXT:RUN

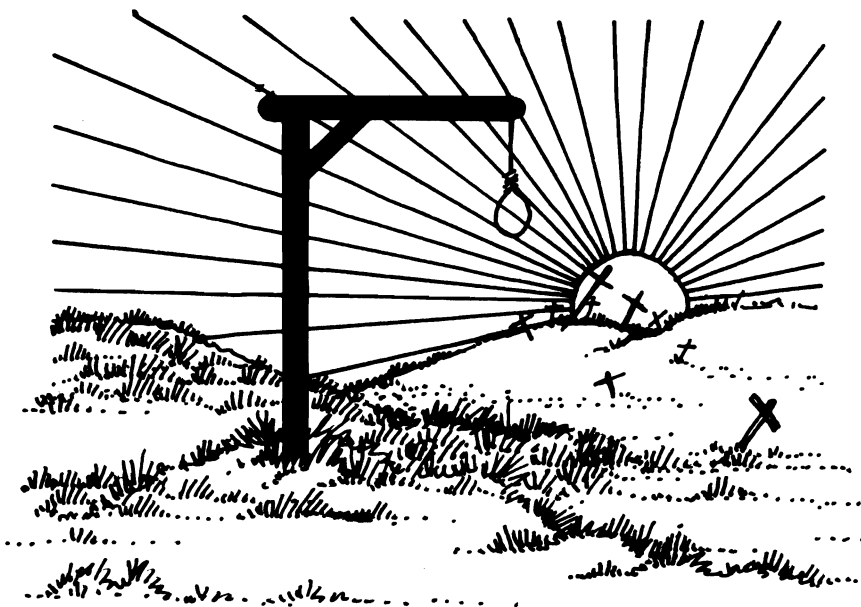
```

READY.



HANGMAN

This is a graphic and musical version of the popular word game. The letters of the alphabet are displayed on the left of the screen, and once you've selected a letter, it is shown in reverse mode, to avoid choosing it again. You position an arrow under the letters, moving it to the one you want, using the "K" (left) and "L" (right) keys. You hit the space bar when the arrow is under the required letter. The hanged man is drawn on the right of the screen.



If you fail to guess the word, a box is removed from underneath the man and he drops to the ground, stretching his neck considerably. A mournful excerpt from The Death March is played and the hidden word is revealed. An excerpt from the Hallelujah Chorus will play if you manage to guess the word before your time is up. Hangman was written by Mike O'Neill.

```

1 00SUB5000:POKE36879,27
10 DIMZ$(53):R=RND(-TI):S1=36874:S2=36875:V=36878
11 FORI=1TO53:READZ$(I):NEXT
12 DATATHO,SIX,BASIC,VIC,DOG,BOX,TOP,APPLE,RHYTHM,HYMN,SKV,SYRINGE,KEY,WRIST,TYR
ANT
13 DATALIGHT,HVENA,CYANIDE,ADD,DYE,FLV,BUY,GIN,MING,ARM,VASE,VET,BISCUIT,LIMP,BL
IOHT
14 DATAFRAME,FRIGHT,RING,BLOOD,SHOW,HANO,HAWK,KING,CLOCK,COMB,SLY,STICK,BLOCK,CR
AR,FLACK
15 DATAMOOSE,VELCRO,HELL,WHISKER,SHIRT,HOSE,SADDLE,GLASS
100 PRINT"?: I=INT(RND(1)*53)+1:0=0
110 A$=Z$(I)
120 L=LEN(A$):IN=0
130 FORI=1TOL:POKEI+7770,ASC(MID$(A$,I,1))-64:POKE38512+I,0:POKE7792+I,46:NEXT
200 FORI=1TO9:POKE38606+I,0:POKE78566+I,I:NEXT
210 FORI=10TO18:POKE38743+I,0:POKE8023+I,I:NEXT
220 FORI=19TO26:POKE38800+I,0:POKE8080+I,I:NEXT
300 P=7989:POKE38720+P,0:POKEP,30
310 GETA$:IFA$="K"THENGOTO350
320 IFA$="L"THENGOTO400
330 IFA$=" "THENGOTO450
340 GOTO310
350 POKEP,32:P=P-1:IFP=8054THENP=7997
360 IFP=8120THENP=8063
370 IFP=7988THENP=8128
380 POKEP+38720,0:POKEP,30:GOTO310
400 POKEP,32:P=P+1:IFP=7998THENP=8055
410 IFP=8064THENP=8121
420 IFP=8129THENP=7989
430 POKEP+38720,0:POKEP,30:GOTO310
450 IFPEEK(P-22)>128THEN310
460 POKEP-22,PEEK(P-22)+128
470 V=0:FORI=1TOL
480 IFPEEK(7770+I)=PEEK(P-22)-128THENV=I:IN=IN+1:POKE38490+I,0
490 NEXT
500 IFV=0THEN600
510 IFIN=LTHEN7000
520 GOTO310
600 O=0+1:IFO=9THEN900
610 ONO0TO620,640,660,680,700,720,740,760
620 PRINT"#####";
630 PRINT"#####";
631 REMPRINT"#####";
640 PRINT"#####";
660 PRINT"#####";
680 PRINT"#####";
700 PRINT"#####";
720 PRINT"#####";
740 PRINT"#####";
760 PRINT"#####";
900 FORI=1TO3:PRINT"#####";
910 FORK=0TO44STEP22:FORJ=9TO0STEP-1:FORI=1TO3

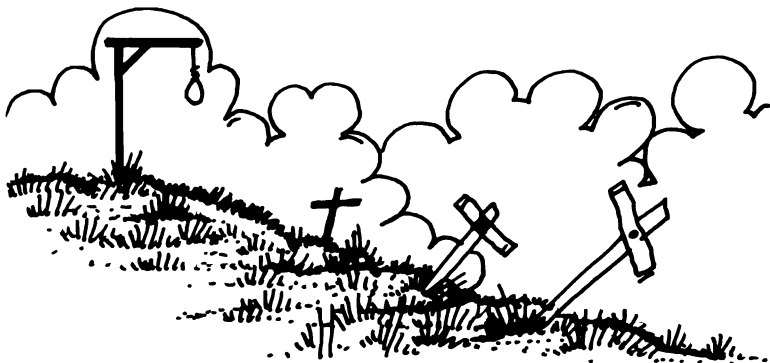
```

66 • GAMES FOR YOUR VIC20

```

920 M=7872+I+J*22+K:POKEM,PEEK(M-22):POKEM+38720,PEEK(M+38690)
930 NEXT I,J,K:GOTO9000
3000 GETA$:IFA$<" THEN3000
3010 INPUT"DO YOU WANT ANOTHER GAME (Y/N)";A$
3020 IFA$="Y" THEN3010
3030 IFA$="N" THEN100
3040 IFA$="N" THENPRINT"GAME":END
3050 GOTO3010
5000 PRINT"THE HONANMAN":PRINT"BY MIKE O'NEILL"
5010 PRINT"DO YOU HIT ANY KEY TO START"
5020 GETA$:IFA$=" " THEN5020
5030 RETURN
7000 POKES1,0:GOSUB8000:GOSUB8000
7010 POKES1,225:POKES2,225:FORJ=1TO2
7020 POKEV,15:FORI=1TO800:NEXT:POKEV,0:FORI=1TO30:NEXTI
7030 POKEV,15:FORI=1TO50:NEXT:POKEV,0:FORI=1TO30:NEXTI
7040 POKEV,15:POKES1,228:POKES2,228:FORI=1TO200:NEXT:POKEV,0:FORI=1TO100:NEXTI
7050 POKEV,15:POKES1,225:POKES2,225:FORI=1TO100:NEXT:POKEV,0:FORI=1TO50:NEXTI
7060 FORI=1TO200:NEXTI,J
7100 POKEV,15:FORI=1TO200:NEXT:POKEV,0:FORI=1TO100:NEXT
7110 POKES1,228:POKES2,228:POKEV,15:FORI=1TO200:NEXT:POKEV,0:FORI=1TO100:NEXT
7120 POKEV,15:POKES1,232:POKES2,232:FORI=1TO200:NEXT:POKEV,0:FORI=1TO100:NEXT
7130 POKEV,15:POKES1,231:POKES2,231:FORI=1TO400:NEXT:POKEV,0:FORI=1TO100:NEXT
7140 POKEV,15:POKES1,232:POKES2,232:FORI=1TO500:NEXT:POKEV,0:FORI=1TO100:NEXT
7200 GOTO3000
8000 POKEV,0:POKES1,232:POKES2,232:POKEV,15:FORI=1TO500:NEXT:POKEV,0:FORI=1TO200
:NEXT
8005 POKES1,225:POKES2,225
8010 POKEV,15:FORI=1TO200:NEXT:POKEV,0:FORI=1TO50:NEXT:POKES1,228:POKES2,228:POK
EV,15
8020 FORI=1TO200:NEXT:POKEV,0:FORI=1TO50:NEXT:POKES1,225:POKES2,225
8030 POKEV,15:FORI=1TO200:NEXT:POKEV,0:FORI=1TO500:NEXT
8100 RETURN
9000 POKEV,0:POKES1,195:POKES2,225:POKEV,15:FORI=1TO1000:NEXT
9010 POKEV,0:FORI=1TO50:NEXT:POKES1,163:POKEV,15:FORI=1TO1000:NEXT
9020 POKEV,0:FORI=1TO50:NEXT:POKES1,0:POKEV,15:FORI=1TO100:NEXT:POKEV,0:FORI=1TO
50
9030 NEXT:POKES1,195:POKEV,15:FORI=1TO1000:NEXT:POKEV,0:FORI=1TO100:NEXT
9040 POKES1,163:POKES2,229:POKEV,15:FORI=1TO1000:NEXT:POKEV,0
9050 POKES1,0:POKES2,228:POKEV,15:FORI=1TO100:NEXT:POKEV,0:FORI=1TO100:NEXT
9060 POKES1,195:POKES2,228:POKEV,15:FORI=1TO600:NEXT:POKEV,0:FORI=1TO50:NEXT
9070 POKES1,0:POKES2,225:POKEV,15:FORI=1TO500:NEXT:POKEV,0:FORI=1TO100:NEXT
9080 POKES1,163:POKEV,15:FORI=1TO700:NEXT:POKEV,0:FORI=1TO50:NEXT
9090 POKES1,0:POKEV,15:FORI=1TO200:NEXT:POKEV,0:FORI=1TO50:NEXT
9100 POKES1,195:POKEV,15:FORI=1TO1000:NEXT:POKEV,0
9110 FORI=1TO1:POKE38490+I,0:NEXT
9200 GOTO3000

```



SIMON

This is a game like the toy of the same name. The computer selects a sequence of colors, with accompanying sounds, and you must repeat the sequence exactly. There are four different colors, shown at the top, bottom, left and right of the screen. The sequence starts with one randomly selected color and each time the sequence is repeated another color is added. The maximum length of the sequence is from five to 20. To make the game easier to use, more than one key is used for each color.

Keys used: Top color—any number
 Bottom color—space bar
 Left color—any character
 Right color—any function key

```

10 PRINT"SIMON":PRINT"BY MIKE O'NEILL"
20 PRINT"KEYS:"
21 PRINT"  "
22 PRINT"  ANY  "
23 PRINT"  NUMBER  "
24 PRINT"  "
25 PRINT"  ANY  |  ANY  "
26 PRINT"  CHARACTER | FUNCTION  "
27 PRINT"  "
28 PRINT"  |  SPACE  |  "
29 PRINT"  |  BAR  |  "
30 PRINT"  "
35 PRINT"ENTER MAXIMUM SEQUENCE"
36 INPUT"  LENGTH (5-20)":M:IFM<5ORM>20THEN36
40 PRINT" HIT ANY KEY TO START"
50 GETA$:IFA$=""THEN50
60 POKE36879,25
70 S=36876:V=36878:POKEV,0:POKES,0:DIMA(20)
80 R=RND(-TI)
100 PRINT"  "
110 FORI=1TO6:PRINT"  "
120 PRINT"  "
130 FORI=1TO6:PRINT"  "
140 PRINT"  "
150 PRINT":FORI=1TO6:PRINT"  "
160 PRINT"  "
170 FORI=1TO1000:NEXT
200 N=0:Z=500
  
```

```

210 N=N+1
220 A(N)=INT(RND(1)*4+1)
230 FORL=1TON
240 ONA(L)GOSUB1000,2000,3000,4000
250 NEXT
300 FORL=1TON
310 C=0
320 GETA#:IFA#=""THEN320
330 B=ACC(A#)
340 IFB>47ANDB<C58THENC=1:REM NUMBER
350 IFB>64ANDB<91THENC=2:REM CHARACTER
360 IFB>132ANDB<141THENC=3:REM FUNCTION
370 IFB=32THENC=4:REM SPACE
380 IFC=0THEN320
400 IFA(L)C>THENGOTO6000
410 ONCGOSUB1000,2000,3000,4000
420 NEXT
430 IFN=M2THEN5000
440 FORI=1TO2000:NEXT
500 GOTO210
1000 POKES,195:POKEV,15:PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
1010 FORI=:TOZ:NEXT:POKEV,0
1020 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
1030 RETURN
2000 POKES,215:POKEV,15
2010 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
2020 FORI=:TOZ:NEXT:POKEV,0
2030 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
2040 RETURN
3000 POKES,225:POKEV,15
3010 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
3020 FORI=:TOZ:NEXT:POKEV,0
3030 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
3040 RETURN
4000 POKES,235:POKEV,15
4010 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
4020 FORI=:TOZ:NEXT:POKEV,0
4030 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
4040 RETURN
5000 Z=5:FORL=1TO10:GOSUB1000:GOSUB2000:GOSUB4000:GOSUB3000:NEXT:GOTO7010
6000 POKES,135:POKEV,15:ONCGOTO6100,6200,6300,6400
6100 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT:GOTO7000
6200 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT:GOTO7000
6300 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT:GOTO7000
6400 PRINT"#####";FORI=1TO6:PRINT"#####";NEXT
7000 FORI=1TO2000:NEXT
7010 GETA#:IFA#C>""THEN7010
7020 POKEV,0:INPUT"DO ANOTHER GAME (Y/N):A#
7030 IFA#="Y"THEN100
7040 IFA#="N"THENPOKE36079,27:END
7050 GOTO7020

```

**RED YELLOW
BLUE GREEN**

BLACKJACK

Prepare to lose your shirt in this ruthless Blackjack game by Ken Staples. The computer plays the bank and you always play first. This means the wily VIC only has to equal your

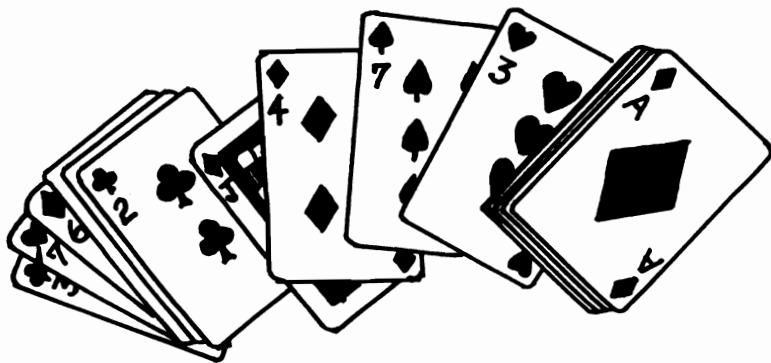


hand and it wins your stake. The winner of each round is the player with the highest total, with the following exceptions:

BLACKJACK—this hand consists of a ten-card and an ace. It beats all other hands

FIVE CARD TRICK—Five cards with a total of 21 or less beats all other hands except for blackjack

If the player scores over 21 (goes bust), then the bank automatically wins.



```

1 REM#PONTOON
2 REM#KEN STAPLES
5 Q=25:P=25
6 DIMD(5),S(5),H(5),C(5)
10 PRINT"J":POKE36879,29:M#="00000000":O#=""
100 L=7661:C=38401:GOSUB5000:A1=E
110 INPUT"00000000BID £":A:B=FRE(0):B=A
120 IFA>5THENPRINTM#"MAX. BID £5":FORM=1T04000:NEXT:PRINTM#O#:GOTO110
200 L=7685:C=38405:X=1:GOSUB5000:X=0:A2=E:GOSUB2000:IFFGOT0600
300 L=7689:C=38409:GOSUB5000:A3=E:GOSUB2000:IFFGOT0600
400 L=7693:C=38413:GOSUB5000:A4=E:GOSUB2000:IFFGOT0600
500 L=7697:C=38417:GOSUB5000:A5=E:GOSUB2000
600 A1=0:A2=0:A3=0:A4=0:A5=0:M#="0000000000000000"
610 L=7857:C=38577:GOSUB5000:A1=E:GOSUB3000
700 L=7861:C=38581:X=1:GOSUB5000:X=0:A2=E:GOSUB3000:IFFGOT01500
800 L=7865:C=38585:GOSUB5000:A3=E:GOSUB3000:IFFGOT01500
900 L=7869:C=38589:GOSUB5000:A4=E:GOSUB3000:IFFGOT01500
1000 L=7873:C=38593:GOSUB5000:A5=E:GOSUB3000
1500 IFN=1THENM=0:P=P+A:Q=Q-A
1510 IFQ=1THENM=0:P=P-A:Q=Q+A
1512 IFQ<0THENFORM=1T01000:NEXT:PRINT"CONGRATULATIONS!. YOU HAVE BROKEN THE BANK!":END
1515 IFQ<0THENFORM=1T01000:NEXT:PRINT"SORRY! YOU ARE BROKE!":END
1520 PRINT"BANK £":PRINT"PLAYER £"P
1530 A=0:A1=0:A2=0:A3=0:A4=0:A5=0:I=0:J=0:K=0:M=0:PT=0:FC=0:FORM=1T05000:NEXT:GO T010
2000 F=0:PRINTM#O#:IFA3<0THEN2020
2010 IFA1=1ANDR2=1ORR2=1ANDR1=10THENPRINTM#"PONTOON":F=1:PT=1:RETURN
2020 T=A1+A2+A3+A4+A5:PRINTM#"POINTS":T;
2025 IF(A1=1ORR2=1ORR3=1ORR4=1)ANDT=10C21THENPRINT"OR"'+10
2030 FORM=1T04000:NEXT:PRINT:IFT>21THENPRINTM#O#:PRINTM#"SORRY! BUST!":F=1:RETURN

```



```

2036 IFAS>0ANDTC=21THENPRINT#0$:PRINT#5"5 CARD TRICK":FC=1:RETURN
2100 PRINTM$*1 THIST.2 BUV.3 STICK:INPUT"WHICH";R:IFR>3THENPRINTM$*M$0$:GOTO2040
2060 PRINT#0$:PRINT"#####":ONRGOTO2110,2120
2080 IFA1=1ORR2=1ORR3=1ORR4=1ORR5=1THENT=T+10:IFT>21THENT=T-10
2090 IFT<15THENPRINTM$*MCANNOT STICK UNDER 15":FORW=1TO4000:NEXT:GOTO2040
2100 PRINTM$*STUCK ON"TC"STAKE £"A":F=1:RETURN
2110 IFT<11THENPRINTM$*MCANNOT TWIST UNDER 11":FORW=1TO4000:NEXT:GOTO2040
2115 RETURN
2120 PRINT#0$:PRINTM$*STAKE £"A";INPUT" BID £";O:IFOC=3THENA=A+O:RETURN
2140 PRINT#0$:PRINTM$*ONLY UP TO FIRST BID":FORW=1TO4000:NEXT:GOTO2120
3000 U=A1+A2+A3+A4+A5:PRINT#0$
3010 IFA1=1ORR2=1ORR1=1ANDR2=10THENPRINTM$*PONTOON.BANK WINS!":F=1:O=1:RETURN
3012 PRINTM$*POINTS"U:
3015 IF(A1=1ORR2=1ORR3=1ORR4=1)ANDU+10<=21THENPRINT"OR"U+10:IFU+10>=21THEN3090
3020 FORW=1TO4000:NEXT:IFPT=1ORU>21ANDTC=21THENPRINTM$*YOU WIN!":F=1:N=1:RETURN
3025 IFT>21ANDR2>0ORR3>0ANDUC=21THENPRINTM$*BANK WINS!":F=1:O=1:RETURN
3070 IFFC=0ANDUC=21ANDU>=21THENPRINTM$*BANK WINS!":F=1:O=1:RETURN
3080 RETURN
3090 PRINT"###BANK WINS!":F=1:O=1:RETURN
5000 POKEL,79:POKEC,0:POKEL+1,99:POKEC+1,0:POKEL+2,80:POKEC+2,0
5010 POKEL+22,101:POKEC+22,0:POKEL+24,103:POKEC+24,0
5020 POKEL+44,101:POKEC+44,0:POKEL+46,103:POKEC+46,0
5030 POKEL+66,101:POKEC+66,0:POKEL+68,103:POKEC+68,0
5040 POKEL+88,76:POKEC+88,0:POKEL+89,100:POKEC+89,0:POKEL+90,122:POKEC+90,0
6000 S=INT(RND(1)*4)+1:IFS=1THENS=63:H=0:I=I+1:GOSUB7000:GOSUB8000:S(I)=D:RETURN
6010 IFS=2THENS=63:H=2:J=J+1:GOSUB7000:GOSUB9000:H(I)=D:RETURN
6030 IFS=3THENS=63:H=0:K=K+1:GOSUB7000:GOSUB10000:C(K)=D:RETURN
6040 IFS=4THENS=90:H=2:M=M+1:GOSUB7000:GOSUB11000:D(M)=D:RETURN
7000 D=INT(RND(1)*13)+1:IFD=1THENE=1
7020 IFD=2THENE=2:D=50
7030 IFD=3THENE=3:D=51
7040 IFD=4THENE=4:D=52
7050 IFD=5THENE=5:D=53
7060 IFD=6THENE=6:D=54
7070 IFD=7THENE=7:D=55
7080 IFD=8THENE=8:D=56
7090 IFD=9THENE=9:D=57
7100 IFD=10THENE=10:D=0
7110 IFD=11THENE=10:D=10
7120 IFD=12THENE=10:D=17
7130 IFD=13THENE=10:D=11
7140 IFR1=1ANDX=1ANDE=1THEN7000
7170 RETURN
8000 FORW=1TO1:IFD<S(W)THEN8030
8020 F=1
8030 NEXT:IFFTHENF=0:GOSUB7000:GOTO0000
8040 GOSUB12000:RETURN
9000 FORW=1TOJ:IFD<H(W)THEN9030
9020 F=1
9030 NEXT:IFFTHENF=0:GOSUB7000:GOTO9000
9040 GOSUB12000:RETURN
10000 FORW=1TOK:IFD<C(W)THEN10030
10020 F=1
10030 NEXT:IFFTHENF=0:GOSUB7000:GOTO10000
10040 GOSUB12000:RETURN
11000 FORW=1TOM:IFD<D(W)THEN11030
11020 F=1
11030 NEXT:IFFTHENF=0:GOSUB7000:GOTO11000
11040 GOSUB12000:RETURN
12000 POKEL+23,S:POKEC+23,H
12010 IFD=0THENPOKEL+45,49:POKEC+45,0:POKEL+46,48:POKEC+46,0:RETURN
12020 POKEL+45,D:POKEC+45,0
12030 RETURN

```

READY.

SKI RUN

In this fast action game from Kevin Smith, you control a skier who you must guide through 20 gates, at any one of 20 skill levels, ranging from one (beginner) to four (professional).

"Z" will move the skier left, "M" will move her right. The game restarts automatically after a crash.

```
5 PRINT"J":POKE650,255
10 PRINT" SKI RUN"
15 B=7691
20 PRINT"NZ=LEFT,M=RIGHT"
25 PRINT"MSKI THROUGH THE GATES"
30 PRINT"MSPEED:M1=PRO"
40 PRINT"4=BEGINNER"
45 POKE198,3
50 INPUT:A=A#15
60 PRINT"J"
100 REM ROUTINE
110 PRINT" I"
120 POKEB,25
130 C=C+1:IFC=24 THEN200
135 POKEB,96
140 GETA$
150 IFA$="Z" THENB=B-1
160 IFA$="M" THENB=B+1
165 POKEB,25
170 FORT=1TOR:NEXT
180 GOTO100
200 C=0:D=INT(RND(1)#17)
205 E=E+1:IFE=21 THENPRINT" GAME OVER":FORC=1TO2000:NEXT:RUN
210 PRINTTAB(D)" I" I"
220 GOTO100
```

READY.



DIAMOND GLORY

This maze game is a type of "solitaire" which can take anything from seconds to hours to play. You control between one and five circular pieces; there are an equal number of diamond pieces which do not move during the game. The object is to move your pieces so that each covers a different diamond. This may seem easy, but wait till you've played it with four or five pieces.

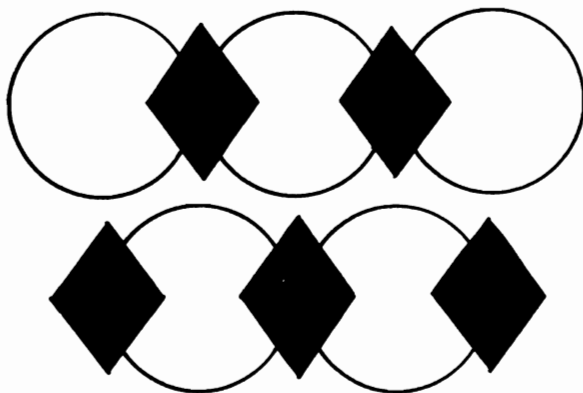
"A"—up

"S"—down

"K"—left

"L"—right

Space—to stop



74 • GAMES FOR YOUR VIC20

```

10 R=RND(-TI):S=36876:V=36878:DIMP(5),C(5)
20 PRINT"3000 DIAMOND GLORY":PRINT"    BY MIKE O'NEILL":POKEV,8
22 PRINT"8 KEYS":"PRINT"8 A=UP":PRINT" S=DOWN":PRINT" K=LEFT":PRINT" L=RIGHT"
24 PRINT" SPACE=STOP"
30 INPUT" ENTER LEVEL (1-5)";R:IFR>5ORR<1THEN30
40 IR=800-R*100:PRINT
50 INPUT" HOW MANY (1-5)";NN:NN=INT(NN):IFNN>5ORNN<1THEN50
60 POKE36879,57:PRINT"#"
100 PRINT"
101 PRINT"
102 PRINT"
103 PRINT"
104 PRINT"
105 PRINT"
107 PRINT"
108 PRINT"
109 PRINT"
110 PRINT"
111 PRINT"
112 PRINT"
113 PRINT"
114 PRINT"
115 PRINT"
116 PRINT"
117 PRINT"
118 PRINT"
119 PRINT"
120 PRINT"
121 PRINT"
190 FORN=1TONN
200 P(N)=INT(RND(1)*418+7724):IFPEEK(P(N))<>32THEN200
210 POKE(N)+38720,1:POKES,200:FORJ=1TO10
211 POKEV,0:POKE(P(N),32:FORI=1TO50:NEXT:POKE(P(N),81:POKEV,15:FORI=1TO50:NEXTI,J:
POKEV,0
212 C(N)=32
220 NEXT
222 FORN=1TONN
224 CC=INT(RND(1)*418+7724):IFPEEK(CC)<>32THEN224
226 POKECC+38720,2:POKES,200:FORJ=1TO10
228 POKEV,0:POKECC,32:FORI=1TO50:NEXT:POKECC,90:POKEV,15:FORI=1TO50:NEXTI,J:POKE
V,0
230 NEXT
300 RS=""
310 GETB$:IFB$=""THEN330
320 RS=B$:GOTO310
330 IFR$=""ANDID=8THEN485
335 IFR$=""THEN480
340 IFR$="A"THEND=-22
350 IFR$="S"THEND=22
360 IFR$="K"THEND=-1
370 IFR$="L"THEND=1
380 IFR$="" THEND=0:GOTO485
480 POKEV,15:FORI=1TO100:NEXT:POKEV,0
485 FORN=1TONN
410 CH=PEEK(P(N)+D):IFCH=32THEN420
412 IFCH<>90THEN430
414 GOTO2900
420 POKE(P(N),C(N):P(N)=P(N)+D:C(N)=PEEK(P(N)):CL(N)=PEEK(P(N)+38720)
430 POKE(P(N)+38720,CL(N):POKE(P(N),81
450 NEXT
460 FORI=1TOIR:NEXT
500 GOTO3000
2000 J=0:FORI=1TONN
2010 IFC(I)<>90ANDI<>0THENJ=1
2020 NEXT
2030 IFJ=1THEN420
2040 POKE(P(N),32:P(N)=P(N)+D
2100 POKES,0:FORI=1TO100STEP3:POKES,150+I:POKEV,15:FORK=1TO5:NEXT
2110 FORJ=1TONN:POKE(P(J),32:NEXT:POKEV,0:FORK=1TO5:NEXT
2120 FORJ=1TONN:POKE(P(J),81:NEXTJ,I
3000 INPUT"3000 ANOTHER GAME (Y/N)";R$:IFR$="N"THENPOKE36879,27:END
3010 IFR$="Y"THEN30
3020 GOTO3000

```

ZOMBIES

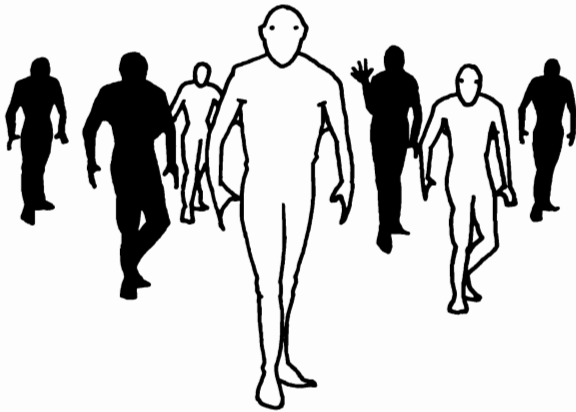
Mike O'Neill places you smack dab in the middle of an island, surrounded by zombies and potholes. The object of the game is to lure the zombies down the potholes, while avoiding being eaten by a zombie or falling down a pothole yourself.

You can move horizontally or vertically. You keep moving in the same direction until a new direction is selected; or if you wish you can stand still. The zombies always move toward you, regardless of what may be in the way.

At the start of the game you have four lives and there are 20 zombies to destroy. You get one point for each zombie that falls down a pothole. Each time you clear the island of zombies you get ten bonus points and the island is redrawn, with two additional zombies. This goes on to a maximum of 30 zombies. If you fall down a pothole, or you are eaten by a zombie, you lose a life and the island is redrawn with 20 new zombies.

Your controls:

"A"—up
"S"—down
"K"—left
"L"—right
"S"—stop



```

0 GOTO9000
1 DIMZ%(30,1)
2 I0=7746: S2=36875: 84=36877: V=36878: IS=0: POKE36879,94: IL=20
3 IN=4
10 POKEV,0: POKES2,0: GOSUB1000: GOSUB2000: GOSUB3000: GOSUB4000: GOSUB5000
20 IW=100: IC=0
100 FORI=1TOZZ
120 IFZ%(I,0)=0THEN300
130 IC=IC+1: IFIC>CZTHENIC=0
135 IFIC=0THENGOSUB500
140 ZX=-1: ZY=-22
150 IFIX>ZX(I,0)THENZX=1
160 IFIY>ZY(I,1)THENZY=22
163 IFIX=ZX(I,0)THENZX=0
165 IFIY=ZY(I,1)THENZY=0
170 T=I0+ZX(I,0)+ZX(I,1)*22+ZX+ZY: T1=PEEK(T)
190 IFT1=32THEN625
200 IFT1=81THEN600
210 IFT1=26THEN300
220 IFT1=87THEN310
225 FORTT=1TOIW: NEXT
250 POKE(I0+ZX(I,0)+ZX(I,1)*22),32
251 POKES2,250-IZ#2: POKEV,15
252 FORTI=1TO50: NEXT
260 POKEV,0: POKET,26: POKET+30720,0
270 Z%(I,0)=ZX(I,0)+ZX: Z%(I,1)=Z%(I,1)+ZY/22
300 NEXT
305 GOTO100
310 T=Z%(I,0)+Z%(I,1)*22+I0
320 POKET,32
321 POKEV,15
322 FORG=100TO250: POKES2,G: NEXT
323 POKEV,0: Z%(I,0)=0: IZ=IZ-1
350 CZ=IZ/6+1: IFIZ<3THENCZ=0
360 IS=IS+1: IW=IW-3
370 PRINT"SCORE": IS
380 IFIZ=0THEN8000
390 GOTO300
500 GOSUB5000
520 IFA#=""THEN650
530 IFA#="K"THEN610
540 IFA#="L"THEN620
550 IFA#="A"THEN630
560 IFA#="S"THEN640
565 IFA#=" "THEN645
600 GOTO650

```


78 • GAMES FOR YOUR VIC20

```

6810 FORI=2T030STEP2
6820 FORQ=1T010:NEXT
6830 POKET,26:POKEIP,81:POKES2,220+I
6840 FORQ=1T010:NEXT
6850 POKET,32:POKEIP,26:POKES2,200+I
6100 NEXT
6400 GOTO7300
6500 T1=26:IC=0
6600 GOTO7650
7000 T1=87:IC=1
7050 POKEIP,32
7100 POKEV,15
7200 FORI=2T030STEP2
7210 POKET,81:POKET+30720,0:POKES2,180+I
7220 FORQ=1T010:NEXT
7230 POKET,T1:POKET+30720,IC:POKES2,200+I
7240 FORQ=1T010:NEXT
7250 NEXT
7300 IL=20:IN=IN-1
7400 IFIN=0THEN9500
7500 GOTO10
8000 FORI=1T010:POKEV,15
8001 FORQ=1T010
8002 POKES2,140+G+I#10
8003 NEXT
8004 PRINT"      BONUS"
8005 FORQ=1T010
8006 POKES2,130+G+I#10
8007 NEXT
8010 PRINT"      "
8015 NEXT:IS=IS+10:POKEV,0
8020 IL=IL+2:IFIL>30THENIL=30
8500 GOTO10
9000 PRINT"ZOMBIES":PRINT"BY MIKE O'NEILL"
9005 PRINT"KEYS :":PRINT" A=UP":PRINT" S=DOWN":PRINT" K=LEFT":PRINT" L=RIGHT"
"
9006 PRINT" SPACE=STOP"
9007 PRINT" HIT ANY KEY TO START"
9100 GETA$:IFA$=""THEN9100
9200 GOTO1
9500 GETA$:IFA$<">"THEN9500
9501 POKEV,0:INPUT"ANOTHER GAME (Y/N) ";A$:IFA$="N"THENPRINT"
9502 IFA$="N"THENPRINT"POKE36879,27:END
9505 IFA$<">"THEN9501
9600 CLR:GOTO1

```

READY.



MICRO MUSIC

This music program can be used in two ways:

- To repeat a simple set of sounds, such as drumbeats, to give a rhythmic backing to other music
- To produce complete tunes

The music is entered in steps. Each step consists of the notes stored in the low, middle, high and noise generators. The volume and duration of the note is also stored. The notes are in the range 128 to 255, as shown in your manual. The volume is in the range zero to 15. The duration has a base value of one, so half notes are entered as .5, thirds as .333 and quarters as .25. When the program is run, you enter the six factors listed. The tune will start playing when you enter "—1" for the "low" note at any stage.

When "Tempo?" appears, first try it with a value of 100. When the music is underway, pressing F5 will allow the tempo to be altered to suit the tune. The value should be in the one to 1,000 range. Pressing F7 allows any stage number to be re-entered.



82 • GAMES FOR YOUR VIC20

```

1 GOSUB1000:GOSUB2000
2 DIMA$(150),B$(150):V=36878
3 S=36875:POKE36879,120:C=38720
4 I%=1:POKEV,0:POKES,0
10 FORI=1TO150:R%(I)=B$(I)=0:NEXT
15 D1%=1:D2%=-1
100 PRINT"J";B$,C$:PRINTS1,S2
110 PRINT"███ █"
120 FORI=1TO16
130 PRINT" █ █"
140 NEXT
150 PRINT" █"
160 R%(I)=7972:B$(I)=7981
170 POKER%(I)+C,2:POKER%(I),81
180 POKEB$(I)+C,5:POKEB$(I),90
190 FORI=1TO2000:NEXT
200 I%=I%+1
201 POKES,0:POKEV,15:FORI=1TO200:POKES,160+I%/2+I:NEXT
210 GETA$:IFA$=""THEN300
220 IFA$="M"THEND2%=-22
230 IFA$="N"THEND2%=22
240 IFA$=","THEND2%=-1
250 IFA$="/"THEND2%=1
260 IFA$="A"THEND1%=-22
270 IFA$="S"THEND1%=22
280 IFA$="D"THEND1%=-1
290 IFA$="F"THEND1%=1
295 GOTO210
300 R%(I%)=R%(I%-1)+D1%
310 IFPEEK(R%(I%))>32THEN400
320 POKER%(I%)+C,2:POKER%(I%),81
350 B$(I%)=B$(I%-1)+D2%
360 IFPEEK(B$(I%))>32THEN500
370 POKEB$(I%)+C,5:POKEB$(I%),90
371 POKES,0:POKEV,15:FORI=1TO200:POKES,160+I%/2-I:NEXT
380 FORI=1TO500:NEXT
390 GOTO200
400 S2=S2+1
410 I%=I%-1:IFI%=0THEN6
420 POKER%(I%),32
421 POKES,250-I%/2:POKEV,15
430 FORI=1TO5:POKES,250+I-I%/2:NEXT
440 GOTO410
500 S1=S1+1
510 I%=I%-1:IFI%=0THEN6
520 POKEB$(I%),32
521 POKES,250-I%/2:POKEV,15
530 FORI=1TO5:POKES,250+I-I%/2:NEXT
540 GOTO510
1000 PRINT"JM RAT-TRAP"
1005 PRINT"MI BY MIKE O'NEILL"
1010 PRINT"MM██ KEYS"
1015 PRINT"MM██ PLAYER 1"
1017 PRINT"MI ████████ M*****"
1020 PRINT"MM██ A=UP F5=UP"
1025 PRINT"MI S=DOWN F7=DOWN"
1030 PRINT"MI D=LEFT =LEFT"
1035 PRINT"MI F=RIGHT /=RIGHT"
1040 PRINT"MM██ HIT ANY KEY TO START"
1045 GETA$:IFA$=""THEN1045
1050 RETURN
2000 PRINT"JM██ ENTER YOUR NAMES"
2005 PRINT"(MAX. 9 CHARACTERS)"
2010 INPUT"MM██PLAYER 1";B$
2020 IFLEN(B$)<10THEN2050
2030 PRINT"MM██MAXIMUM 9 CHARACTERS"
2035 INPUT"PLEASE RE-ENTER";B$:GOTO2020

```

```
2050 INPUT "PLAYER 2"; C$
2060 IF LEN(C$) < 10 THEN 2090
2070 PRINT "MAXIMUM 9 CHARACTERS"
2075 INPUT "PLEASE RE-ENTER"; C$: GOTO 2060
2090 RETURN
```

READY.

JUPITER RUN

In this program a never ending stream of interplanetary ships and debris hurtles toward you. Your task, of course, is to destroy these objects while avoiding a collision. In this game you only get one chance!

Your ship is armed with one phaser which can only fire one shot at a time. And to make matters worse it has a limited range and cannot reach the targets at the top of the screen. This game is a matter of endurance and survival as you battle against constant enemy onslaught and your own loss of energy. Aim carefully because each shot of the phaser costs ten units of energy and if your energy level descends to zero you are considered dead. If your shot is a hit then you gain ten energy units with which to continue the struggle.

There are five control keys available to you:

1. Tilts the ship at a 45 degree angle to the left.
2. Tilts the ship at a 45 degree angle to the right.
8. Moves the ship to the left.
9. Moves the ship to the right.
0. Fires your phaser.

Your score is displayed in the top left hand corner and your energy level in the top right. Press "Y" to start a new game.

```

10 REM JUPITER RUN
20 GOSUB1000
30 IFPEEK(653)THEN30
40 GOSUB500
80 K=PEEK(197)
90 IFK<59ANDFNT(X)>1THENX=X-1
100 IFK<4ANDFNT(X)<21THENX=X+1
110 IFK<0ANDS>1THENS=S-1
120 IFK<56ANDS<3THENS=S+1
140 IFPEEK(X)>32THEN2000
150 IFK<>0XORPEEK(X)=32THENPOKE0X,32:POKEY,S+26:0X=X
160 IFK<60THENGOSUB300
170 PRINT"###&"SC;TAB(10)"(E" " "
180 IFE<50THENPOKE36879,10
190 IFE>50THENPOKE36879,8
200 GOTO30
300 REM FIRING
310 N=230:E=E-5:SF=SF+1
320 M=M(S):D=D(S):Y=X
330 FORI=1TO6:IFY<XTHENPOKEY,32
340 Y=Y-D
350 N=N-5:POKES3,N
360 IFPEEK(Y)>=34THEN400
370 POKEY,M:POKEFNC(Y),2
375 B=FNT(Y):IFB=0ORB=22THEN390
380 NEXT
390 POKES3,0:POKEY,32:RETURN
400 REM HIT!
410 POKES4,0:POKES1,245:POKES2,225:POKES3,196:POKEY,39:POKEFNC(Y),1
420 SC=SC+10:E=E+10:H=H+1
470 FORI=1TO50:NEXT:POKEY,32
480 POKES1,0:POKES2,0:POKES3,0
490 RETURN
500 REM PLANT ASTEROID
510 FORI=1TOR
520 POKE646,FNR(7)+1
530 PRINT"###"TAB(FNR(20))CHR$(FNR(3)+34)
540 NEXT
550 PRINT"###":POKE218,158:
560 IFPEEK(X)>34THEN2000
570 POKEY,S+26:POKEFNC(C),1
580 RETURN
999 END
1000 REM READY
1010 PRINT"J":POKE52,28:POKE56,28:POKE55,0:POKE51,0
1020 FORI=27*8TO40*8+7
1030 READA:POKEI+7168,A:NEXT
1040 DATA136,68,42,17,160,64,32,16
1050 DATA16,16,146,146,254,130,130,0
1060 DATA17,34,84,136,5,2,4,8
1070 DATA0,0,0,16,8,4,0,0,0,0,0,0,8,16,32,0,0
1080 DATA0,0,0,0,0,0,0,0,0,0,16,16,16,0,0
1090 DATA24,112,60,254,126,124,63,0,0,0,48,124,62,24,0,0
1100 DATA0,0,56,124,82,252,120,48
1110 DATA0,0,219,146,210,82,219,0,0,0,179,170,179,170,171,0
1120 DATA0,68,16,56,16,68,0,0
1130 DATA126,64,64,120,64,64,126,0
1150 FORI=48*8TO57*8+7:POKE7168+I,PEEK(32768+I):NEXT
1200 POKE36879,8:POKE36878,15:POKE36869,255

```

86 • GAMES FOR YOUR VIC20

```

1210 DEF FNR(X)=INT(RND(1)*X)
1220 DEF FNT(X)=22*((X-7680)/22-INT((X-7680)/22))
1230 DEF FNC(X)=X+30720
1240 X=8174
1250 S=2
1260 S1=36874:S2=S1+1:S3=S2+1:S4=S3+1
1270 FORI=1TO3:READM(I),D(I):NEXT
1280 DATA30,23,33,22,31,21
1290 A=1
1300 E=100
1999 RETURN
2000 REM END
2010 POKES4+2,42
2020 POKES1,180
2030 POKEX,39:POKEFNC(X),1
2040 FORI=1TO1000:NEXT
2050 POKES1,0:POKE198,0
2070 WAIT198,1:POKE198,0
2080 POKE36869,242
2090 PRINT"YOU'RE DEAD!"
2100 PRINT"AND ONLY SCORED",SC
2110 PRINT"HOTS FIRED"TAB(11)SF
2120 PRINT"ITS"TAB(11)H
2130 IFH>0THENPRINT"% ITS"TAB(11)INT(H/SF*100)"%"
2150 PRINT"WANT TO DO BETTER?"
2160 GETA$:IFA$="Y"THENRUN
2170 IFA$="N"THENPRINT"POKE631,ASC("N"):POKE198,1:END
2180 GOTO2160

```

READY.

VOGON

Your mission in this program is to destroy a "helpless city". If your conscience is not pricked by this description of the target, launch the missiles from your craft, as it crosses the screen from left to right, by pressing the shift key. Holding down the shift key will keep the missile flying through the air instead of dropping immediately. This enables you to aim more accurately. As time progresses your craft gradually descends; for your mission to be successful all the buildings below must be destroyed before the craft reaches the ground.

The game is not difficult to master but it does display good graphics and the closing sequence has a particularly menacing, throbbing sound effect as the screen fills with replicas of your craft.

```
10 REM VOGON
20 GOSUB1000
30 GOSUB900
40 FOR MOVE=0 TO MAX STEP INC
50 S=PEEK(653)
60 IFSANDF=0THENGOSUB800
70 IFS=0ANDF=1THENF=2
100 ON F GOSUB400,500,600
150 IFFANDB=>8185THENGOSUB700
200 PRINT"MISS SCORE"SC;
210 IFF=0THENPRINTTAB(16)"FIRE!"
300 NEXT MOVE:GOTO30
400 REM UPDATE BOMB (F=1)
410 POKEB,32:B=B+1:P=PEEK(B)
420 IFP=30THEN700
430 POKEFNC(B),13:POKEB,27
440 Z=B-7680/22:POKES2,220+22*(Z-INT(Z))
450 V=V+A
490 RETURN
500 REM SLOWING (F=2)
510 V=V-D
520 POKEB,32
530 IFV<0THENF=3:RETURN
540 B=B+1:IFPEEK(B)=30THENSC=SC+5:GOTO700
```

88 • GAMES FOR YOUR VIC20

```

550 POKEFNC(B),5:POKEB,31
560 POKES2,200+VAND255
590 RETURN
600 REM DROP (F=3)
610 POKEB,32
620 B=B+22:IFPEEK(B)=30THENS0=SC+10:GOTO700
640 POKEFNC(B),5:POKEB,33
690 RETURN
700 REM EXPLOSION
710 POKES1,0:POKES2,0:POKES3,0:POKES4,201
720 POKEFNC(B),2:POKEB,34
730 FORI=1TO100:NEXT
740 POKES4,0
750 POKEB,32
780 F=0
790 RETURN
800 REM LAUNCH
810 POKES3,220:PRINT"■"TAB(16)" "
820 B=X+22
830 FORI=1TOFNR(3)
840 POKES3,PEEK(S3)-10
850 POKEB,32
860 B=B+22:POKEFNC(B),5:POKEB,31
870 NEXT
880 POKES3,0:F=1
890 RETURN
900 REM MOVE SHIP
910 POKEB,32
920 X=X+1
930 IFPEEK(X+1)=30THEN2000
940 POKEB,28
950 POKEFNC(X+1),13:POKEB+1,29
960 IFX=8184THEN3000
999 RETURN
1000 REM SET UP
1010 PRINT"■":POKE36879,8
1020 POKE36878,44
1030 POKE51,0:POKE52,28:POKE55,0:POKE56,28
1040 FORI=27*ST07+8*(27+7)
1050 READA:POKEI+7168,A:NEXT
1100 DATA32,32,40,250,250,40,32,32
1110 DATA0,3,10,43,191,191,42,10
1120 DATA0,192,160,232,254,254,168,160
1130 DATA126,126,66,66,126,126,66,66
1140 DATA48,48,60,63,63,60,48,48
1150 DATA0,0,0,0,0,0,0,0
1160 DATA0,0,126,126,60,60,24,24
1170 DATA0,68,16,56,16,0,68,0
1300 REM SHIP DATA
1310 A=1:D=2:REM ACCELERATION & DE...
1320 X=7701
1330 MAX=1:INC=1:REMMAX&INCREMENT MOVE
1350 REM CITY BIT
1360 CH=0:REM HEIGHT
1370 CW=10:REM WIDTH
1380 CC=8175:REM CENTRE
1400 FORI=0TOCW
1410 P1=CC-I:F2=CC+I
1420 FORJ=0TOCH-I
1430 POKEP1-22*J,30:POKEP2-22*J,30

```

```

1440 NEXT J, I
1500 DEF FNC(X)=X+30720
1510 DEF FNR(X)=INT(RND(1)*X)
1600 S1=36874:S2=S1+1:S3=S2+1:S4=S3+1
1700 POKE36869,255
1999 RETURN
2000 REM DEAD
2010 POKES1,200:POKES2,0:POKES3,0:POKES4,0
2020 POKEX,32:POKEX+1,32
2030 X=X+22:IFX=0105THEN2070
2040 POKEFNC(X),13:POKEX,28
2050 POKEFNC(X+1),13:POKEX+1,29
2060 GOTO2020
2070 FORI=1TO50
2080 FORJ=-1TO1
2090 POKE36864,PEEK(36864)+J:POKE36865,PEEK(36865)-J
2100 NEXT J, I
2110 POKES1,0
2120 PRINT"YOU FAILED!"
2130 FORI=1TO200:NEXT
2140 PRINT"HIT A KEY TO TRY AGAIN"
2150 POKE198,0:WAIT198,1:POKE198,0
2160 RUN
3000 REM INVASION!
3010 POKES1,135:POKES2,195:POKES3,136
3020 POKES4+1,111
3030 FORI=1TO500
3040 X=FNR(253)*2+7680
3050 POKEFNC(X),13:POKEX,28
3060 POKEFNC(X+1),13:POKEX+1,29
3070 NEXT
3080 PRINT"YOU SUCCEEDED!"
3090 PRINT"SCORING":PRINT"SC
3100 PRINT"WANT TO DESTROY"
3110 PRINT"ANOTHER HELPLESS CITY?"
3120 PRINT"HIT A KEY"
3130 POKE198,0:WAIT198,1:RUN

```

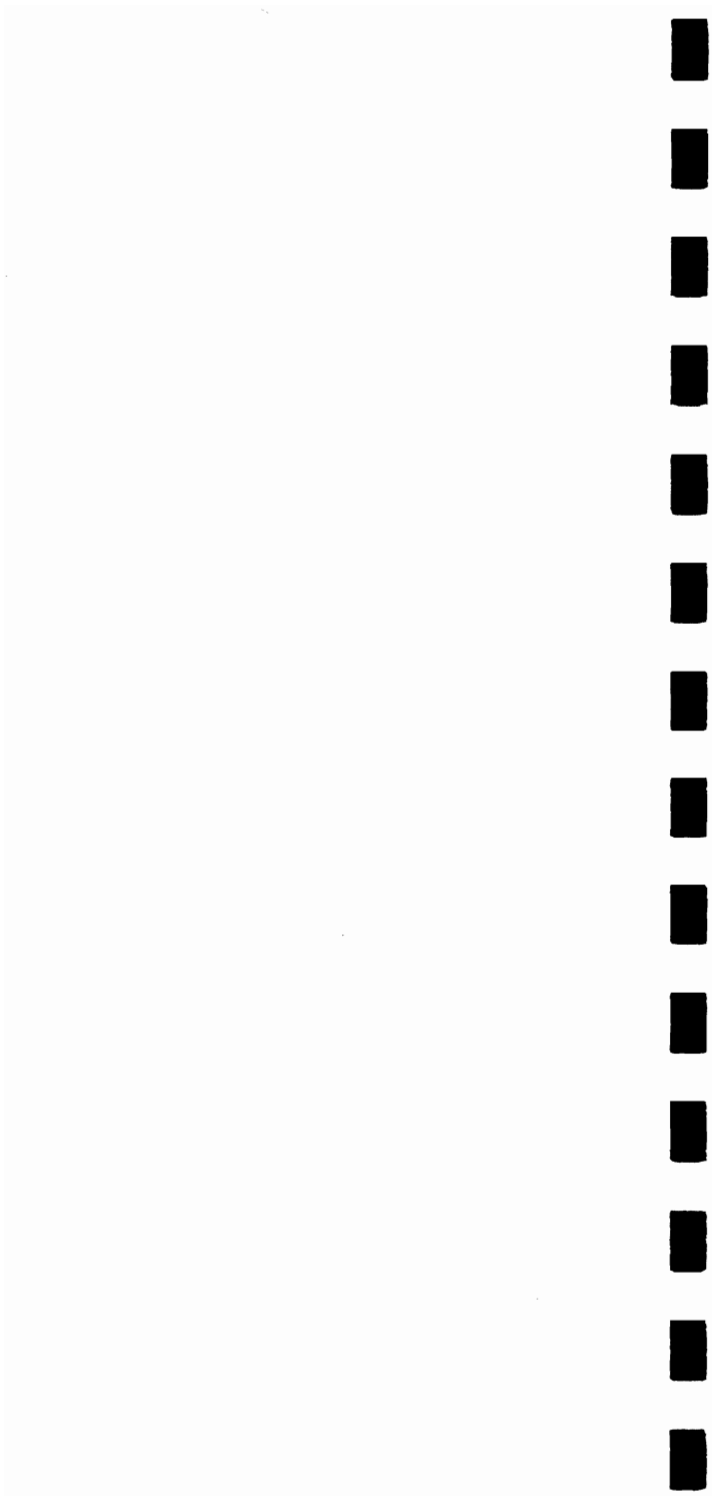
READY.





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 others. 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 toward 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 improve pub-

lished 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 categorize 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 (e.g. 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 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 formula 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 entering 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



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 mem-

orized 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 criticized 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 of most 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 recognize.

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 synchronizes 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 sensitized 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 organized 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 TS-1000 you are using is a micro-computer; medium sized computers are known as mini-computers; 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.

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 synthesizer.

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 (see *also* 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 (see also 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 the 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. Teletext messages can be transmitted by cable or over phone lines. Examples of this are the Prestel service in Britain and 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

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 address 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 recognizes 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.

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 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.

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. First, 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. Second, it is intended to give people who have no specialized 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 hobby 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 flow charts) 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, Rodney, Sybex).

I had lunch with Rodney 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.

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.

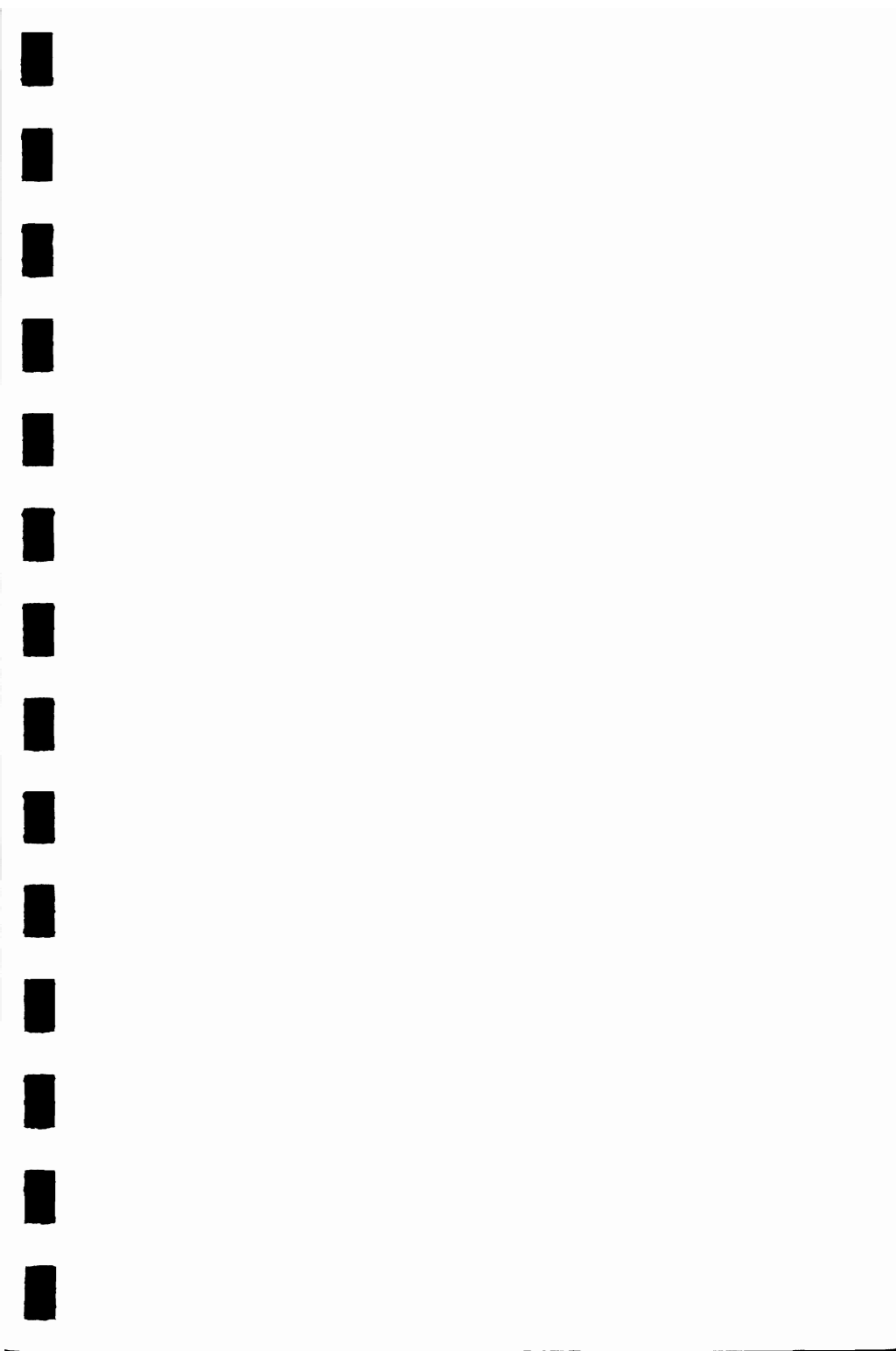
1001 Things to Do with Your Personal Computer (Sawush, Mark, Tab Books, Blue Ridge Summit, Pa.).

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.

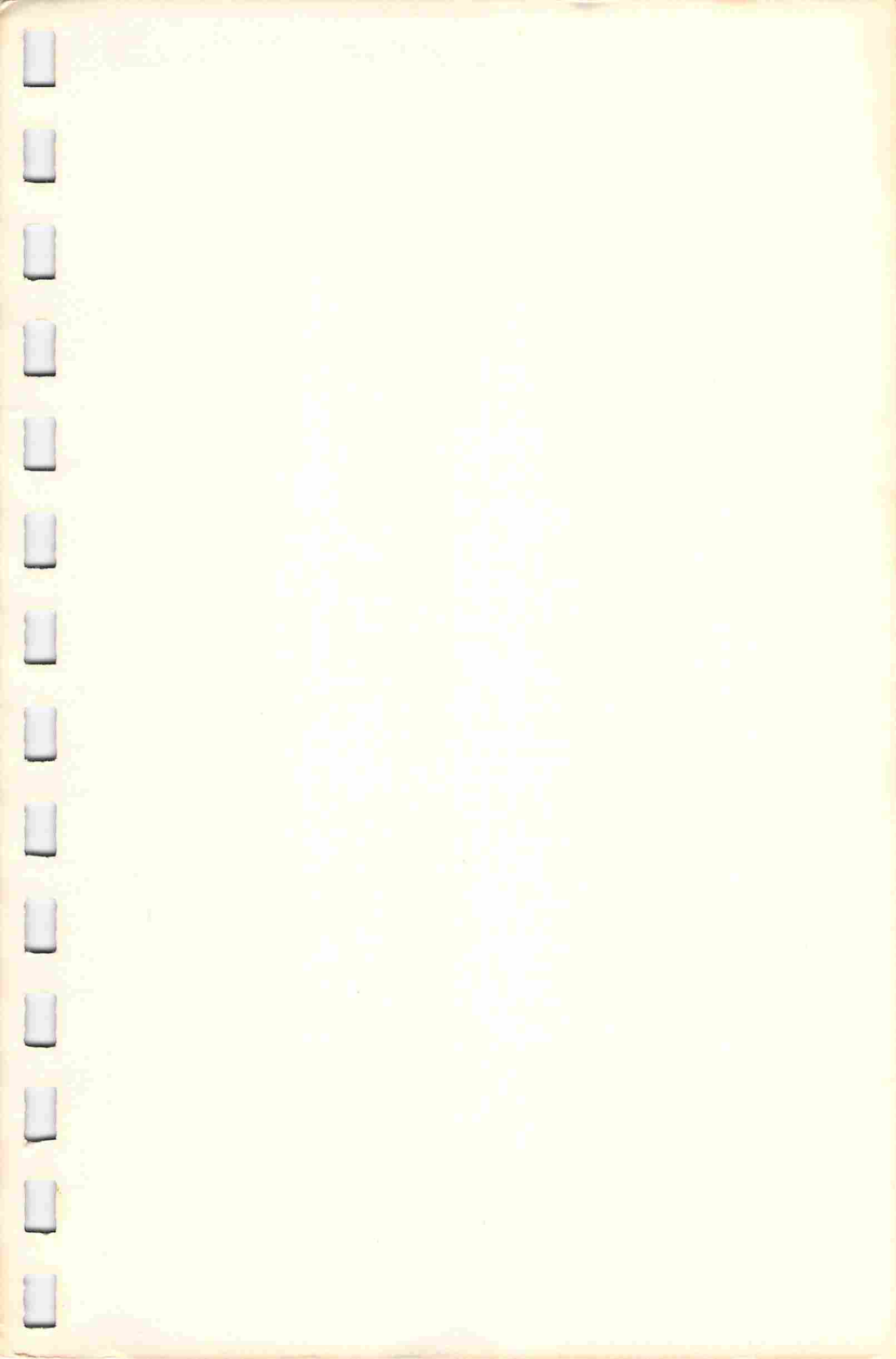
The World Computer Chess Championship (Hayes, Jean E., and Levy, David N. L., Edinburgh University Press, Edinburgh).

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 analyzed 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 realize how far the computer world has come in only a few years.







GAMES FOR YOUR VIC20

EXCITEMENT!... Destroy the interplanetary enemy in Jupiter Run. . . Avoid the obstacles and gobble up the numbers in Digit-Dodge. . . Hold the fate of the Federation starship in your hands in Tail Gunner. . . Turn your VIC20 into a futuristic musical instrument with Piano 2001. . . Let your imagination run wild with the multi-colored Sketchpad. . . and much, much more.

ENTERTAINMENT!... More than 25 fascinating, original programs, including arcade-style games, brain teasers, word games, programs, and puzzles written exclusively for THE DELL COMPUTER GAMES SERIES and guaranteed to give you endless hours of fun and entertainment.

CHALLENGE!... Improve your computer-game skills. Extend your programming expertise. Learn tricks and techniques to apply to your own programming as you follow instructions to enter each game into your VIC20. Once you've mastered the programs in their present form, you're the expert. Try your hand at improving them!

ORIGINAL GAME PROGRAMS * FUN FOR THE WHOLE FAMILY**

COMPLETE WITH A GLOSSARY OF COMPUTER TERMS, A SELECTIVE BIBLIOGRAPHY, AND HINTS ON HOW TO IMPROVE AND EXTEND THE GAME PROGRAMS THAT WILL ADD TO YOUR ALL-ROUND PROGRAMMING EXPERTISE.

A Dell Trade Paperback/Dell Publishing Co., Inc.



ISBN 0-440-52789-9