COMMODORE 64 MIND STRETCHERS

gma Press

IAN CREASEY

TP11
C1
8562517

COMMODORE 64 MINDSTRETCHERS

Ian Creasey

SIGMA PRESS

Copyright © 1984 by Ian Creasey

All Rights Reserved.

No part of this book may be reproduced or transmitted by any means without the prior permission of the publisher. The only exceptions are for the purposes of review, or as provided for by the Copyright (Photocopying) Act or in order to enter the programs herein onto a computer for the sole use of the purchaser of this book.

ISBN 0 905104 74 9

Typesetting and Production by:

DESIGNED PUBLICATIONS LTD.
8-10 Trafford Road,
Alderley Edge,
Cheshire.

Published by:

SIGMA PRESS
5 Alton Road,
Wilmslow,
Cheshire.
U.K.

Distributors:

Europe, Africa:
JOHN WILEY & SONS LIMITED,
Baffins Lane, Chichester,
West Sussex, England.

Australia, New Zealand, South-East Asia:
Jacaranda-Wiley Ltd., Jacaranda Press,
JOHN WILEY & SONS INC.,
GPO. BOX 859, Brisbane,
Queensland 40001, Australia.

Printed and bound in Great Britain by
J. W. Arrowsmith Ltd., Bristol

NOTES ON LISTINGS

This book contains forty programs for you to use with your Commodore 64. Some are 'utility' programs, designed to help with various tasks; the majority of the others are games programs.

Each program is fully explained, with operating instructions, listing dissection, variable lists, and technique explanations. With most of them, there is also a section which explains how to tailor the program for your own use.

The listings in this book have been done on a plotter, which produces much more readable text than a dot matrix printer. A special conversion routine was used to make the graphics and Commodore hieroglyphic control codes more legible.

For instance, instead of a reversed S character which is normally used for the home key, these listings will show the word HOME in square brackets. The colours and cursor controls are similarly treated: LBL for light blue, etc. The graphics have been done the same way. For example, instead of an unidentifiable line, these listings will have SHIFT *, CBM T, and so on . All these messages are in square brackets to avoid confusion, so when you see a square bracket, don't type it in, just follow the instruction inside the bracket to produce the character.

A further readability measure is that instead of having long strings of the same character, the listing shows 2 SHIFT V, meaning a shifted V character twice; or 13 CR meaning 13 cursor rights. The number is inside the square brackets as well, so you'll understand what it means.

Spaces are also converted. The number of spaces required is often ambiguous, or indeed whether there are any at all, so to avoid this confusion, the listings use a new character. It looks like this: △ . So whenever you see this sign, it means press the space bar. If there is more than one space, the listing will say 5 △, meaning five spaces.

The programs were originally written without spaces between the keywords, but the converter routine inserts spaces between keywords to aid readability. When typing in, these spaces can be left out if desired. Because of this spacing, some lines will appear to be more than 80 characters long, especially if there are control codes in them. Just type these lines without the spacing. On the rare occasions when a line exceeds 80 characters anyway, you will have to use abbreviated keywords: for example, ? for PRINT, etc.

All REM statements may be left out.

So, on to the programs.

All the programs in this book were written by me, and I hope you enjoy them. But because of this, and because progammers develop certain styles, many of the games have points in common, which are tedious to repeat in every set of instructions (although in some cases, I have done this).

Firstly, many of the games can be used with a joystick. If so, the joystick should almost always be plugged into port 2, the one nearest the power socket. Also, when asked if you want another game, pressing the fire button means 'Y'.

Secondly, for many games there is a standard system of keyboard controls. The actual system used is always specified in the instructions, but it might be wise to get used to the following:

'T'=up,
'B'=down,
'F'=left,
'H'=right.

The way I play is to have the index finger of the left hand on 'T', the left thumb on 'F', the index finger of the right hand on 'B', and the middle finger of the right hand on 'H'. Of course, this is only a personal preference: the customization section will usually tell you how to change the controls if you wish.

Thirdly, when a program asks you to press one key or another, or to answer a yes or no question, it MEANS press one key or the other: any other keys will be ignored. It sounds silly to say this, but if nothing happens when you press a key, do make sure you have pressed the right key!

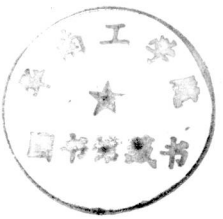
Some of the programs have various 'options' which you type in to create your own preferred game, or difficulty levels to help you improve. In these games, when asked whether you want another game, usually pressing 'Y' or the fire-button gives you the same game, and pressing 'O' or 'D' gets you back to the options/difficulty screen. Pressing 'N', of course, quits the program.

Some of the programs contain machine-code, for speed or convenience. All these programs have a built-in data checker to spot mistakes, but it is not infallible. It works by adding up all the numbers to see if they come to the right total, the checksum. If they don't you are told so, and you must check your typing. However, if you type two wrong numbers, they may cancel each other out. Better safe than sorry: SAVE the program on tape before RUNning it, and if it crashes, you can load it up again. Machine-code is not disassembled or explained in detail, but it is worthwhile to look at it yourself.

此为试读,需要完整PDF请访问: www.ertongbook.com

8562517

CONTENTS



ALIEN ATTACK

The earth has been invaded, says the radio. Yawn, not again. Are you tired of being a superhero? Well, that's tough. Get down to the landing bay immediately, and I hope you haven't had too many Pan Galatic Gargle Blasters.

You are moving along the top of the screen, changing direction when you hit the edge or when you press F7. The aliens come up from the bottom, and you fire at them using the space bar. Don't let them get too close, because if you pass above them when they're in range, they will zap you with their x-raser beam.

How it works:

Lines

10	Initialise whole program.	
20-50	Set up screen and variables.	
100-110	Check for you've pressed change direction key.	
120	Check for you've hit the edge of the screen.	
130	Move you.	
140-150	Check for 'gone too close to alien' so zapped.	
160-170	Check for fire bullet.	
180	Start a new bullet.	
190-360	Move bullet:	
	190	Make bullets move twice as fast as you; check for no current bullet.
	200	Check for empty space below.
	210-220	Check for hit side of an alien.
	230-300	Destroy alien.
	310-330	Do routine for hit side of an alien.
	340-350	Move bullet; if gone off bottom then stop it.
	360	Terminate loop.
360-380	Decide whether to print a new alien.	
390-400	Print the new alien.	
500-590	Zap effects.	
600	Scroll everything off screen.	
610-670	Print score, etc.	
680	Check for key pressed.	
690	Wait for a minute.	
700	Put computer in control.	

Variables:

OF	Difference between character and colour memory: 54272.
V	Position in SID chip of volume control: 54296.
P	Your position.
D	Your direction.
R	Chance of alien appearing.
S	Sound variable.
T	Direction of bullet.
B	Position of bullet.
A	Loop variable to make bullet move faster than you.
F	Flag to indicate that alien has appeared while you fired.
SC	Score.

Customisation:

The screen colour is set to black by the '0's in line 10. Your initial position is the top left corner with the value '1024' in line 50; if you change this, the new value must be between 1024 and 1063 at a point on the top line. The initial chance value of aliens appearing is the '.975' in the same program line. The change direction key is controlled by the 'F7's in lines 100 and 110. Your colour spacing and graphic are respectively the '5' and '83' in line 130, which make a green heart. The fire key is controlled by the space symbols in lines 160 and 170. The number of times the bullet moves faster than you do is set by the '2' in line 190. The colour of the aliens is set by the 'YEL' and 'CYN' controls in line 390.

```
3 REM
4 REM ********************
5 REM ***ALIEN ATTACK***BY IAN CREASEY**
*
6 REM ********************
7 REM
10 POKE 193,0:POKE 194,0
20 POKE 53280,0:POKE 53281,0:PRINT "[CLR
]"CHR$(142)CHR$(8)
30 OF=54272:V=OF+24:POKE V,5:POKE OF+5,0
:POKE OF+6,240:POKE OF,239:POKE OF+1,19
```

```
40 POKE OF+12,0:POKE OF+13,249:POKE OF+8
,17:POKE OF+19,0:POKE OF+20,249:POKE OF+
15,17
50 PRINT "[CLR][26 CD]":P=1024:D=1:R=.97
5
100 S=0:IF I$<>"[F7]" THEN GET I$
110 IF I$="[F7]" AND PEEK(194)=0 THEN D=
-D:POKE OF+4,33:S=1
120 IF P+D<1024 OR P+D>1063 THEN D=-D:PO
KE OF+4,33:S=1
130 POKE P,32:P=P+D:POKE P+OF,5:POKE P,8
3:IF S THEN POKE OF+4,32
140 IF PEEK(P+79)=233 OR PEEK(P+159)=233
 OR PEEK(P+239)=233 OR PEEK(P+319)=233 T
HEN 500
150 IF PEEK(P+399)=233 OR PEEK(P+480)=21
4 OR PEEK(P+560)=214 OR PEEK(P+640)=214
THEN 500
160 IF I$<>"▵" THEN GET I$
170 IF NOT (((I$="▵" AND PEEK(194)=0) OR
 PEEK(194)=1) AND B=0) THEN 190
180 T=40:B=P+T:POKE B+OF,1:POKE B,46:POK
E OF+11,17:FOR I=0 TO 9:NEXT:POKE OF+11,
16
190 FOR A=1 TO 2:IF B=0 THEN 360
200 IF PEEK(B+T)=32 THEN 340
210 IF PEEK(B+T)=233 AND T=40 THEN T=39:
GOTO 310
220 IF PEEK(B+T)=223 AND T=40 THEN T=41:
GOTO 310
230 POKE B,32:B=B+T:SC=SC+1:POKE OF+18,1
29
240 IF PEEK(B)>220 THEN B=B+(PEEK(B)-228
)/5
250 IF (PEEK(B) OR 128)>220 THEN B=B-40+
(PEEK(B)-100)/5
260 IF PEEK(B)=90 THEN B=B-40:GOTO 240
270 POKE B-1,77:POKE B,93:POKE B+1,78:PO
KE B+39,64:POKE B+40,42:POKE B+41,64
280 POKE OF+18,128:FOR I=1 TO 200:NEXT
290 POKE B-1,32:POKE B,32:POKE B+1,32:PO
KE B+39,32:POKE B+40,32:POKE B+41,32
300 B=0:GOTO 360
```

```
310 POKE OF+4,17:J=T-40:C1=PEEK(B+J):C2=
PEEK(B+T)
320 POKE B,93:POKE B+J+OF,1:POKE B+J,77.
5+J/2:POKE B+T+OF,1:POKE B+T,64
330 FOR I=0 TO 99:NEXT:POKE B,32:POKE B+
J,C1:POKE B+T,C2:POKE OF+4,16:GOTO 200
340 POKE B,32:B=B+T:IF B>1983 THEN B=0:G
OTO 360
350 POKE B+OF,1:POKE B,46
360 NEXT:IF F>B THEN 390
370 R=R-.0001:IF RND(1)<R THEN 100
380 IF B THEN F=1:GOTO 100
390 POKE OF+11,33:F=0:I$="[YEL][RVSON][S
HIFT £][CBM *][RVSOFF][CD][3 CL][3 SHIF
T Z]":IF RND(1)>.66 THEN I$="[CYN][RVSON
][SHIFT £][SHIFT V][CBM *][RVSOFF][CD][3
 CL][SHIFT £][SHIFT A][CBM *]"
400 PRINT TAB(INT(36*RND(1))+1)I$:POKE O
F+11,32:GOTO 100
500 POKE V,8:S=37:POKE OF+20,251:POKE OF
+15,S:POKE OF+18,129:POKE P,211
510 P=P+80:IF PEEK(P-1)<>233 THEN 510
520 P=P-40:S=S+1:POKE OF+15,S:FOR I=1 TO
 50:NEXT
530 IF PEEK(P)<>211 THEN POKE P+OF,4:POK
E P,160:GOTO 520
540 POKE P,32:P=P+40:POKE P+OF,5:POKE P,
211:S=S-1:POKE OF+15,S
550 FOR I=1 TO 50:NEXT:IF PEEK(P+39)<>23
3 THEN 540
560 POKE OF+15,16:POKE V,10
562 POKE P-1+OF,5:POKE P+1+OF,5:POKE P-4
1+OF,5:POKE P-40+OF,5:POKE P-39+OF,5
565 POKE P-1,64:POKE P+1,64:POKE P-41,77
:POKE P-40,93:POKE P-39,78:FOR I=0 TO 99
:NEXT
570 POKE P-1,32:POKE P+1,32:J=1:POKE P,3
2:POKE OF+18,128
580 POKE P-J*39+OF,5:POKE P-J*40+OF,5:PO
KE P-J*41+OF,5:FOR I=1 TO 20:NEXT
585 POKE P-J*39,46:POKE P-J*40,46:POKE P
-J*41,46:FOR I=1 TO 20:NEXT
590 POKE P-J*39,32:POKE P-J*40,32:POKE P
```

```
-J*41,32:J=J+1:IF P-J*41>1023 THEN 580
600 FOR I=1 TO 30:PRINT :FOR J=0 TO 99:N
EXT:NEXT
610 IF PEEK(194) THEN PRINT TAB(10)"[ORA
]I▵SCORED"SC"BUT▵THEN":PRINT TAB(12)"I'M
▵ONLY▵YOUR▵64":GOTO 640
620 PRINT TAB(16)"[ORA][RVSON]SCORE"SC
630 IF SC>PEEK(193) THEN POKE 193,SC:FOR
 I=0 TO 99:NEXT:PRINT TAB(10)"[CD][LRD][
RVSON]THIS▵IS▵THE▵HI▵SCORE":GOTO 650
640 FOR I=0 TO 99:NEXT:PRINT TAB(14)"[CD
][LRD][RVSON]HI▵SCORE:"PEEK(193)
650 FOR I=0 TO 99:NEXT:PRINT TAB(13)"[CD
][LGN][RVSON]TO▵PLAY▵AGAIN,":PRINT TAB(1
3)"[RVSON]PRESS▵ANY▵KEY."
660 FOR I=0 TO 99:NEXT:PRINT TAB(12)"[LB
L][CD][RVSON]AFTER▵ONE▵MINUTE":PRINT TAB
(12)"[RVSON]64▵PLAYS▵ITSELF."
670 FOR I=1 TO 8:PRINT :FOR J=0 TO 99:NE
XT:NEXT:POKE V,0:POKE 194,0:POKE 198,0:T
I$="000000"
680 IF PEEK(198) THEN RUN 20
690 IF TI$<"000100" THEN 680
700 POKE 194,1:RUN 20
```


此为试读,需要完整PDF请访问: www.ertongbook.com

AWARI

This is an ancient African game of logic, played with 14 pits and 36 beans, arranged like this:

Commodore 64's side

```
          3  3  3  3  3  3
64's home 0                 0  Your home
          3  3  3  3  3  3
```

Your side

The numbers represent the number of beans in each pit. A move is made by taking all the beans from a particular pit and sowing them one in each pit, working anticlockwise from the pit. If the last bean sown lands in your home pit, you get another go. If the last bean lands in an empty pit, and the pit opposite is not empty, then that bean and all the beans in the opposite pit are moved to your home pit. Your pits are numbered 1 to 6 from left to right, and the computer's are numbered 1 to 6 from right to left. For example, if for your first move you chose pit 4, then the board would look like this:

```
  3  3  3  3  3  3
0                   1
  3  3  3  0  4  4
```

As the last bean sown landed in your home pit, you get another go. If you chose pit 1, the board would then look like this :

```
  3  3  3  0  3  3
0                   5
  0  4  4  0  4  4
```

Because the last bean landed in an empty pit, and the opposite pit was not empty, it and all the beans in the opposite pit were moved to your home pit.

The game ends when all the pits on one side are empty. The winner is the one who has the most beans in his home pit.

How it works:

What the computer does is to execute a move from each of its own pits on a copy of the board, to see which is the 'best' move, in terms of how many beans it will get in its home pit. As there are only six pits to try, the method is fast, and it works quite well - it takes practice to beat.

Lines

10-30	Initialise whole program.
100-110	Set up board.
120-130	Get your move(s).
140-160	Get computer's move(s).
200-280	Game over. Another game?
1000-1100	Print board and check whether game over.
2000-2040	Ask for your move.
3000-3140	Calculate computer's move.
4000-4040	Execute move (move beans etc).

Variables:

G	Game over? flag.
M	Current position.
H	Current home.
N	Number of beans in current pit chosen as move.
B	Computer's best move.
X,Y,Z	Variables used in calculating computers move.
D	Difference between number of your beans and the computer's at end of game.
VS	Number of games the computer has won.
YS	Number of games you have won.
B()	Board.
S()	Auxiliary board used for calculating computer's move.
R()	Array of computer's equally good moves.
FNR(X)	Random number function, used to choose between two or more equally good moves.

Customisation:

The screen colour is set to white in line 30 by the '1's. The initial number of beans in each pit is controlled by the '3' in line 100. If you want to alter the game so that you DON'T get another go if the last bean lands in your home pit, remove lines 130 and 150. To remove the section where if the last bean lands in an empty pit, you get that bean and the beans in the opposite pit, delete line 4030 and lines 3040 to 3090, and change the following lines:

```
3030 M=J:GOSUB 4000
3100 Y=B(13)-B(6):IF Y>B THEN B=Y:S=0
```

```
3 REM
4 REM **********
5 REM ***AWARI***BY IAN CREASEY***
6 REM **********
7 REM
10 DIM B(13),S(13),R(5)
20 I=RND(-TI):DEF FN R(X)=INT(X*RND(1))
30 POKE 53280,1:POKE 53281,1:PRINT "[CLR
]"CHR$(142)CHR$(8)
100 FOR I=0 TO 13:B(I)=3:NEXT:B(6)=0:B(1
3)=0
110 GOSUB 1000
120 PRINT "[HOME][5 CD][13 ▵]":PRINT "[C
D][9 ▵]"
125 PRINT "[HOME][5 CD][LBL]YOUR▵MOVE?▵"
:;GOSUB 2000:IF G THEN 200
130 IF M=H THEN PRINT "[HOME][7 CD][LBL]
AGAIN?▵";:GOSUB 2000:IF G THEN 200
140 PRINT "[3 CD][LRD][22 ▵][22 CL]COMPU
TER'S▵MOVE▵IS▵";
145 GOSUB 3000:IF G THEN 200
150 IF M=H THEN PRINT "[HOME][17 CD][20
CR],";:GOSUB 3000:IF G THEN 200
160 GOTO 120
200 GOSUB 1000:PRINT "[HOME]"TAB(15)"[BL
K][RVSON]GAME▵OVER"
210 D=B(6)-B(13):IF D<0 THEN PRINT "[CD]
[LRD]COMPUTER▵WINS▵BY";-D"POINTS":US=US+
1
220 IF D>0 THEN PRINT "[CD][LBL]YOU▵WIN▵
BY"D"POINTS":YS=YS+1
230 IF D=0 THEN PRINT "[CD][CYN]IT▵WAS▵A
▵DRAW"
240 PRINT "[15 CD][ORA]YOU:"YS"COMPUTER:
"US
250 PRINT "[CD][BRN]ANOTHER▵GAME?";:POKE
 198,0:POKE 204,0
260 GET I$:IF I$<>"Y" AND I$<>"N" THEN 2
60
270 POKE 204,1:PRINT I$:IF I$="Y" THEN 1
00
280 END
1000 PRINT "[CLR]"
```

```
1010 PRINT "[HOME][9 CD][11 CR][GRN]";:F
OR I=12 TO 7 STEP -1:GOSUB 1100:PRINT "▵
";:NEXT:PRINT
1020 PRINT "[CD][8 CR][RED]";:I=13:GOSUB
 1100:PRINT "[19 CR][BLU]";:I=6:GOSUB 11
00:PRINT
1030 PRINT "[CD][11 CR][PUR]";:FOR I=0 T
O 5:GOSUB 1100:PRINT "▵";:NEXT:PRINT
1040 G=0:FOR I=0 TO 5:IF B(I)=0 THEN NEX
T:G=1
1050 FOR I=7 TO 12:IF B(I)=0 THEN NEXT:G
=1
1060 RETURN
1100 PRINT RIGHT$(STR$(B(I)),2);:RETURN
2000 POKE 198,0:POKE 204,0
2010 GET I$:M=VAL(I$):IF M<1 OR M>6 THEN
 2010
2020 M=M-1:IF B(M)=0 THEN 2010
2030 POKE 204,1:PRINT I$
2040 H=6:GOSUB 4000:GOTO 1010
3000 FOR I=0 TO 13:S(I)=B(I):NEXT
3010 B=-99:H=13
3020 FOR J=7 TO 12:IF B(J)=0 THEN 3130
3030 M=J:GOSUB 4000:Y=-99
3040 FOR I=0 TO 5:IF B(I)=0 THEN 3100
3050 X=0:Z=B(I)+I
3060 IF Z>5 THEN X=X+1
3070 IF Z>13 THEN X=X-1:Z=Z-14:GOTO 3060
3080 IF B(Z)=0 AND Z<>6 AND Z<>13 THEN X
=X+B(12-Z)
3090 IF X>Y THEN Y=X
3100 NEXT:Y=B(13)-B(6)-Y:IF Y>B THEN B=Y
:S=0
3110 IF Y=B THEN R(S)=J:S=S+1
3120 FOR I=0 TO 13:B(I)=S(I):NEXT
3130 NEXT:M=R(FN R(S))
3140 PRINT CHR$(42+M);:GOSUB 4000:GOTO 1
010
4000 N=B(M):B(M)=0
4010 FOR I=1 TO N:M=M+1:IF M>13 THEN M=0
4020 B(M)=B(M)+1:NEXT
4030 IF B(M)=1 AND M<>6 AND M<>13 THEN I
```

```
F B(12-M) THEN B(H)=B(H)+B(12-M)+1:B(M)=
0:B(12-M)=0
4040 RETURN
```

BARS AND BOXES

This game looks very simple but is in fact frustratingly addictive. You are presented with a coloured box and two coloured bars underneath it. The background colour around the box will be either black or white. One of the bars will be the same colour as the box; the other will be a different colour. If the background is black, and the left bar is the same colour as the box, press the 'F' key; if the right bar is the same colour as the box, press the 'H' key. If the background is white, press the opposite key to that which you would press for a black background. Although this sounds complicated, you'll soon get the hang of it. Each time you press the correct key, you get one point; if you press the wrong key, the game ends. As you score more and more points, the screen divides into halves, then quarters, just to confuse you. You have one minute to score as many points as possible.

How it works:

All the graphics are handled by strings. The background is made by having a string full of reversed spaces, and printing it in either black or white, then printing the box on top of it.

Lines

10-90	Initialise whole program.
100	Set up one game.
200-220	Decide background colours and position of box.
300	Print background.
310	Print box.
320	Look at background colour behind box.
325-340	Print bars.
350-400	Print score and time left.
410	Check to see whether run out of time.
420	Wait for key to be pressed.
430	Check to see whether you have pressed correct key.
440	Pressed wrong key - game over.
500-530	Another game?

Variables:

BA$	Bar.
B1$,B2$	Background strings.

此为试读,需要完整PDF请访问: www.ertongbook.com