The Book of Arcade Games.

for RISC OS computers

Chris Dewhurst

The Book of Arcade Games

for RISC OS Computers

DRAG'NDROP
www.dragdrop.co.uk

The Book of Arcade Games
for RISC OS Computers

ISBN 978-1-78926-845-4
First published 2018 by
Independent Publishing Network (IPN)
https://bookisbn.org.uk

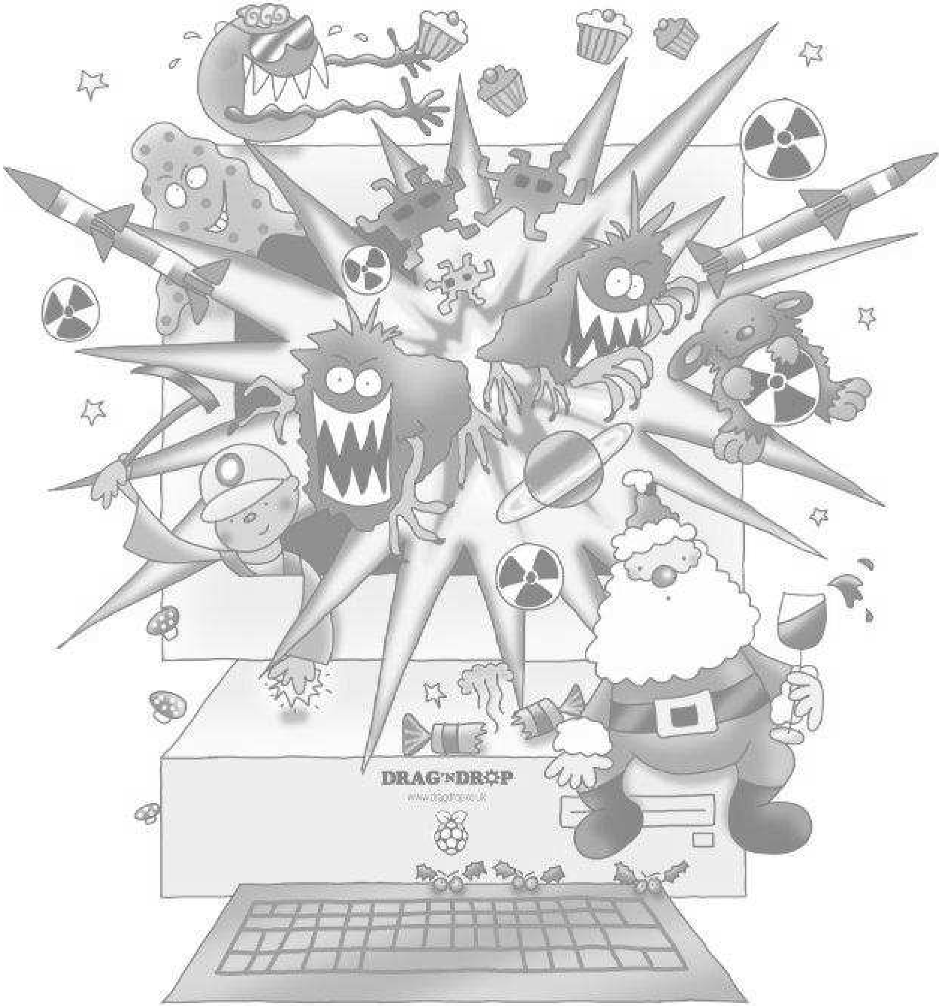
The author is indebted to the writers of the many fine applications developed for RISC OS computers, without which this book would have been very difficult to produce.

Produced on RISC OS computers using EasiWriter, Draw, PostScript3 printer driver, and the OKI-C321 colour laserjet printer.

Printed and bound in Great Britain by
Drag 'N Drop,
Burrsville, Essex UK

www.dragdrop.co.uk   dragdrop@dragdrop.co.uk

Typeset in 10/12 Plato.


DRAG'N DROP

Contents

Introduction

This volume was born of the idea that it was about time someone published a book of type-in games for RISC OS computers. Of course, in an age where computer software is available for immediate download on the internet you might well ask, why isn't it just put on a website for instant consumption?

Those of us who grew up with 8-bit machines like the BBC Micro remember the hours spent keying in listings from books and magazines. We also recall how much the debugging (and sore fingers) taught us about how programs work. A lot of inspiration – and perspiration – resulted from the prospect of an enjoyable game at the end of all your typing!

So, the answer to that question is, if you like to learn about computer coding by inputting programs and are looking for inspiration to write your own games then this book is for you.

The games have all appeared in *Drag 'N Drop*, the quarterly PDF magazine for RISC OS on the Raspberry Pi. They are written in BBC Basic V so, subject to memory limitations and speed, they are suitable for all machines, whatever version of RISC OS you have. BBC Basic V has come built in to every RISC OS machine ever released.

Each listing is entirely self contained: there are no sprites to create in Paint or data files to compile. There is a screen shot of the action, brief introduction and instructions on how to use the program and then the listing is presented in the 'System' font.

I hope you'll agree this is clearer than the Corpus (Courier) font employed in a lot of computer books where it's impossible to distinguish between 1 (numeral one) and l (lower case l).

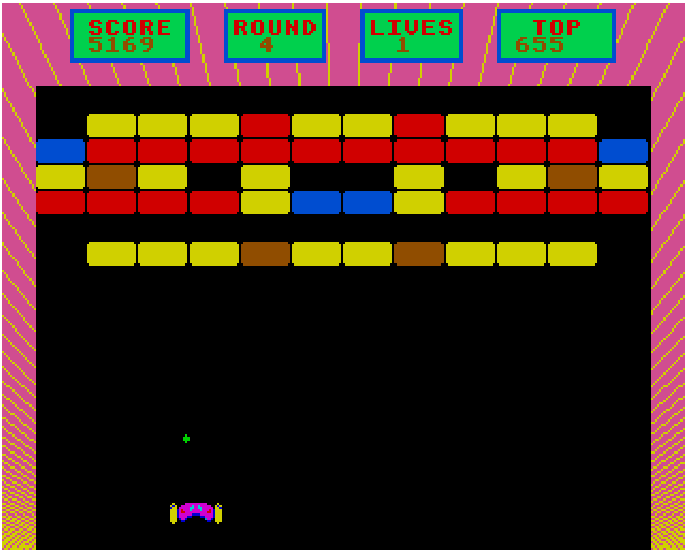
Typing in and running the programs

There are two recommended ways of entering the program listings.

Multitasking method: Load up Edit by opening the Apps folder on the left hand side of the icon bar and double click !Edit. From Edit's icon bar menu choose Create > BASIC. Then type in the program.

The benefits of this method are that you can type whilst getting on with other things such as checking your emails. You don't need to decide immediately where to save the program because you can press F3 (or click the menu button over the Edit window and choose Save F3 > ...) and drag the file icon to some desired location on your hard disc. The disadvantage of using Edit is that you have to type in line numbers.

PiBall

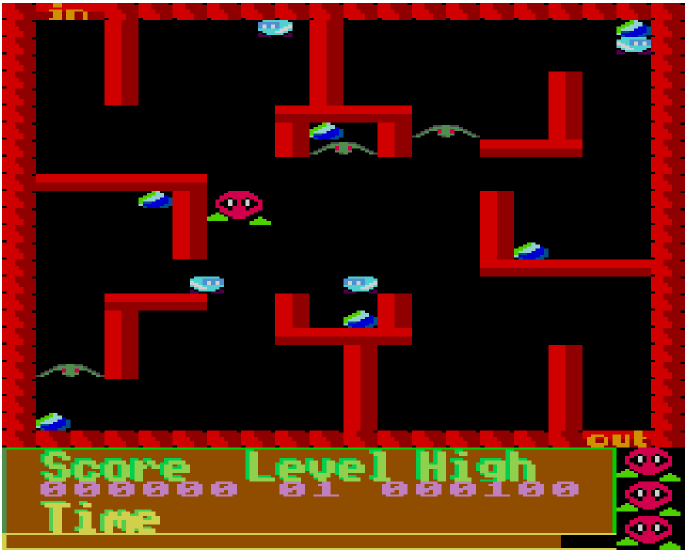


PiBall is a ‘Breakout’ clone and includes a screen designer (the PiBall construction set), multiple bats and balls, fast/slow ball, catch ball and break out of screen. Your bat movements are controlled by the mouse with any button firing the ball at the start or when it is caught.

The screen designer is easy to use. Click the mouse over the brick type you want then place it on the screen at the desired place by clicking the mouse. The screens are automatically saved as file *screens* in the currently selected directory and when you return to the game at a later date the *screens* will be loaded in for you.

```
   10REM *** PIBALL ***
   20ON ERROR GOTO 5720
   30MODE13:VDU23,0,10,32|
   40PROCset_system
   50PROCinitial_display
   60REPEAT
   70IF key>48 AND key<52 AND demo% THEN demo%=0:GOTO100
   80PROCtitles
   90IF demo% THEN scr%=RND(5):lives%=1:sc%=0:GOTO120
  100ON key-48 GOTO 110,3700,3690
  110scr%=1:sc%=0:lives%=6
  120REPEAT
  130GCOL 0:RECTANGLE FILL 0,0,1148,864
  140grey%()=0:a%()=0
  150PROCdraw_screen
  160PROCinit
  170MOUSE batx%,A,A
  180PROCsprite(1,batx%,48)
  190IF bricks%=0 THEN 220
  200K=INKEY(150)
  210GCOL0:RECTANGLE FILL 110,88,928,32
  220PROCsprite(2,ballx%,bally%)
  230*FX21
  240ax%=0:ay%=0:speed=INT(scr%*.3)+3:IF speed>6 speed=6
  250sx%=0:sy%=0
  260catch%=TRUE:TIME=0
  270capsule%=FALSE:cx%=0:cy%=0:ocx%=0:ocy%=0
  280breakout%=FALSE:breaktime%=0
  290B%=0
  300REPEAT
  310IF capsule% AND TIME>T% THEN SOUND &13,&380,200+capsule%*4,5:
PROCsprite(capsule%+2,cx%,cy%):capsule%=0
  320IF breakout% AND (batx%>=1056 OR TIME>breaktime%) THEN
  330PROCsprite(8,1148,0):breakout%=0:breaktime%=0
  340IF batx%>=1056 THEN bricks%=0
  350ENDIF
  360speed+=0.0005:IF speed>8 speed=8
  370PROCmove_bat
  380PROCupdate_ball
  390UNTIL bricks%<=0 OR lives%=0
  400IF lives%=0 THEN 420
  410scr%+=1:IF scr%>50 scr%=1
  420UNTIL lives%<=0 OR demo%
  430IF key<>-1 AND demo% THEN 450
  440PROCgame_over
  450UNTIL 0
  460END
  470
  480DEF PROCdraw_screen
  490COLOUR 128+col%(6):COLOUR col%(3)
  500VDU4,28,0,31,39,0:COLOUR col%(2):PRINTTAB(15,2);scr%;" "
  510PRINTTAB(23,2);lives%;" "TAB(30,2);hi%
  520PRINTTAB(5,2)"     "
  530COLOUR128+col%(6):COLOURcol%(2)
  540addr%=(scr%-1)*200
  550bricks%=0:hard%=FALSE
  560FOR lpy%=144 TO 816 STEP 48
  570FOR lpx%=0 TO 1056 STEP 96
  580A=screen%?addr%
  590hits%=((scr%-1) DIV 2)+2:IF hits%>6 hits%=6
```

Wallace Goes Walking

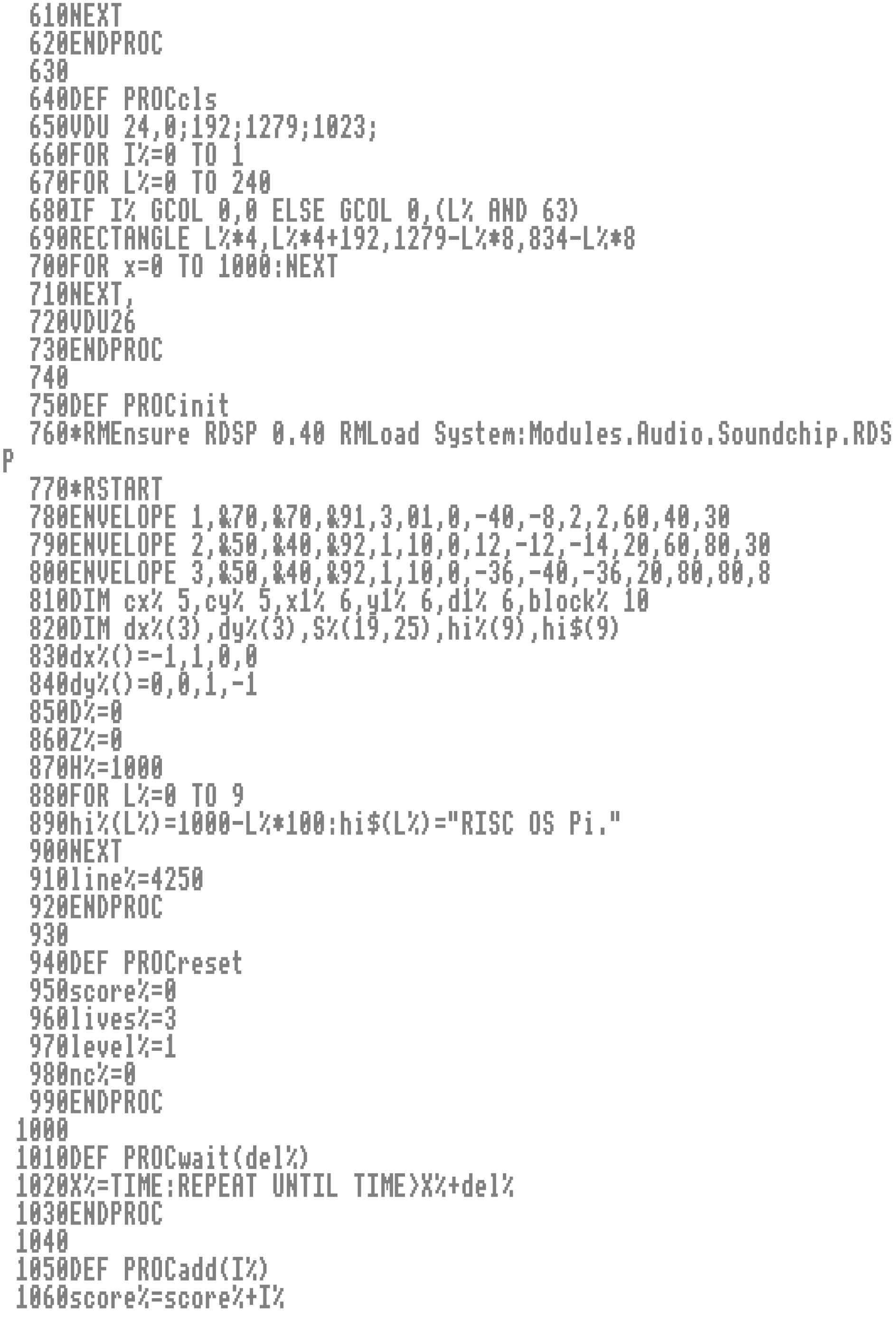


In this game (inspired by a BBC Micro game of a similar name) the evil Gaarts have stolen all the Easter eggs. Wallace must collect them all and make his way to the exit on the bottom right of the screen which will move him to a different and harder maze. Use Z and X for left and right, and K and M for up and down.

There are four robots and a bird that try to stop you. Contact with them is fatal and after every nine screens another bird appears up to a maximum of three. The game features colourful MODE 10 graphics, amusing sound effects and a high score table.

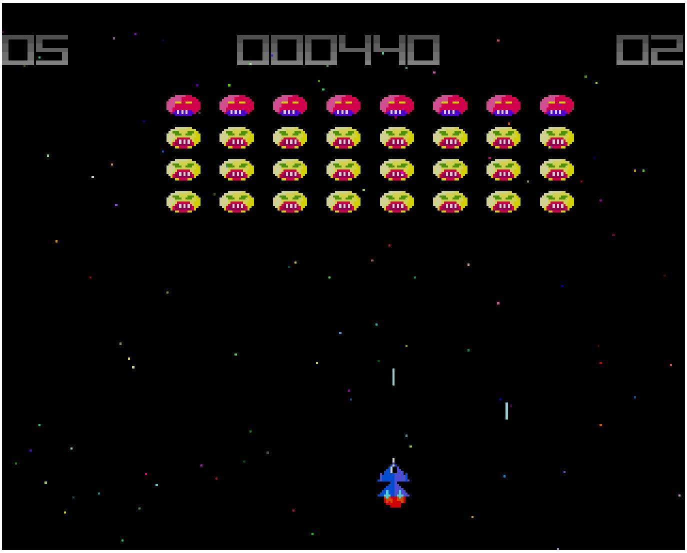
```
 10REM *** WALLACE GOES WALKING ***
 20MODE 10:VDU23,0,10,32|
 30ON ERROR GOTO4690
 40TINT 2,0
 50PROCinit
 60PROCsprites
 70REPEAT
 80PROCreset
 90PROCtitle
100REPEAT
110REPEAT
120PROCscreen
130REPEAT
```

```
140PROCman
150PROCwait(10)
160UNTIL done% OR dead%
170IF dead% PROChit
180UNTIL lives%=0 OR done%
190IF done% THEN level%=level%+1
200UNTIL lives%=0
210*FX21,0
220PROCgameover
230UNTIL 0
240END
250DEF PROCgameover
260PROCbig("GAME",0,10,42,40,40,16)
270PROCbig("OVER",0,15,42,40,40,17)
280PROCwait(100)
290PROCcls
300IF score%<hi%(9) THEN ENDPROC
310VDU23;10,1;0;0;0;
320PROCbig("Please enter your name!",0,0,8,16,7,60)
330J%=-1:REPEAT J%=J%+1:UNTIL hi%(J%)<=score%
340FOR I%=1 TO J% STEP -1
350hi%(I%+1)=hi%(I%):hi$(I%+1)=hi$(I%)
360NEXT
370hi%(J%)=score%:hi$(J%)=""
380PROChiscores
390VDU 31,7,J%*2+3
400SYS "OS_ReadLine",block%,10,32,128
410hi$(J%)=$block%
420VDU23;10,32|
430PROCcls
440ENDPROC
450
460DEF PROCtitle
470PROCreset
480PROClives
490PROCbig("Wallace Goes Walking",1,0,8,16,7,5)
500PROCpanel
510PROChiscores
520PROCbig("Press any key to play!",0,23,8,16,7,8)
530REPEAT UNTIL GET
540ENDPROC
550
560DEF PROChiscores
570FOR L%=0 TO 9
580COLOUR 44
590PRINT TAB(0,L%*2+3)LEFT$("000000",6-LENSTR$hi%(L%));hi%(L%);
600COLOUR 48:PRINT "-";:COLOUR 38:PRINT hi$(L%)
```



```
  610NEXT
  620ENDPROC
  630
  640DEF PROCcls
  650VDU 24,0;192;1279;1023;
  660FOR I%=0 TO 1
  670FOR L%=0 TO 240
  680IF I% GCOL 0,0 ELSE GCOL 0,(L% AND 63)
  690RECTANGLE L%*4,L%*4+192,1279-L%*8,834-L%*8
  700FOR x=0 TO 1000:NEXT
  710NEXT,
  720VDU26
  730ENDPROC
  740
  750DEF PROCinit
  760*RMEnsure RDSP 0.40 RMLoad System:Modules.Audio.Soundchip.RDS
P
  770*RSTART
  780ENVELOPE 1,&70,&70,&91,3,01,0,-40,-8,2,2,60,40,30
  790ENVELOPE 2,&50,&40,&92,1,10,0,12,-12,-14,20,60,80,30
  800ENVELOPE 3,&50,&40,&92,1,10,0,-36,-40,-36,20,80,80,8
  810DIM cx% 5,cy% 5,x1% 6,y1% 6,d1% 6,block% 10
  820DIM dx%(3),dy%(3),S%(19,25),hi%(9),hi$(9)
  830dx%()=-1,1,0,0
  840dy%()=0,0,1,-1
  850D%=0
  860Z%=0
  870H%=1000
  880FOR L%=0 TO 9
  890hi%(L%)=1000-L%*100:hi$(L%)="RISC OS Pi."
  900NEXT
  910line%=4250
  920ENDPROC
  930
  940DEF PROCreset
  950score%=0
  960lives%=3
  970level%=1
  980nc%=0
  990ENDPROC
 1000
 1010DEF PROCwait(del%)
 1020X%=TIME:REPEAT UNTIL TIME>X%+del%
 1030ENDPROC
 1040
 1050DEF PROCadd(I%)
 1060score%=score%+I%
```

Attack of the Raspberry Macaroons



You thought you'd seen them all off – Raspberry As and Bs, the Twos and Threes, not to mention the Zeros . . . but now there's a new wave.

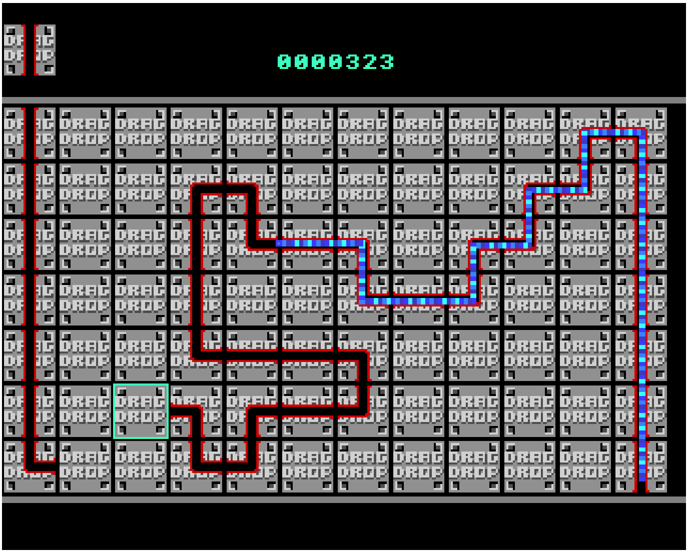
Big (they run Windows), yellow and pink – they're lethal and they're the Raspberry Macaroons. Can you save the earth from a deadly invasion?

Find out in this type-in, classic Space Invaders clone. It features fast colourful graphics, scrolling stars, a high score table and exciting sound effects.

```
 10REM *** ATTACK OF THE RASPBERRY MACAROONS ***
 20MODE13 :REMMODE640,480,8
 30ON ERROR REPORT:PRINT " at ";ERL:END
 40PROCinit:REPEAT
 50PROCtitle:REPEAT UNTIL INKEY-99
 60score%=0:lives%=5:level%=1
 70PROCsetgame
 80OFF
 90REPEAT C%=0
100REPEAT
110PROCmovestars
120PROCmovealiens
130IF C%=0 THEN oaa%=aa%:aa%=aa% EOR 1
```

```
  140PROCplayer:PROCbullets:PROCdelay(1):
  150C%=(C%+1)AND3
  160UNTIL dead% OR killed%=na%+1
  170IF dead% PROCexplode:PROClives(-1)
  180IF killed%=na%+1 level%=level%+1:PROCsetgame
  190PROCdelay(200)
  200IF lives% PROCrestart
  210UNTIL lives%=0
  220SOUND&13,&2FF,&C8E,200:PROCfont(230,512,"Game Over",60,&FF22)
:PROCdelay(500)
  230PROChiscore
  240UNTIL 0
  250END
  260
  270DEF PROCmovestars
  280PROCdrawstars
  290FOR I%=1 TO 100
  300sy%(I%)-=4
  310IF sy%(I%)<0 THEN sy%(I%)=980
  320NEXT
  330PROCdrawstars
  340ENDPROC
  350DEF PROCdrawstars
  360LOCAL I%
  370FOR I%=1 TO 100
  380GCOL 3,I%
  390POINT sx%(I%),sy%(I%)
  400NEXT
  410GCOL 0
  420ENDPROC
  430DEF PROCinit
  440*RMEnsure RDSP 0.23 RMLoad System:Modules.Audio.Soundchip.RDS
P
  450*RSTART
  460ENVELOPE 1,&80,&41,&92,7,57,0,-1,0,25,28,104,109,104
  470ENVELOPE 2,&41,&F2,&59,3,254,227,-52,-31,-22,151,126,95,129
  480ENVELOPE 3,&C3,&C4,&C1,47,24,80,-40,-85,-35,5,200,20,2
  490ENVELOPE 4,&80,&89,&2F,10,0,8,0,-1,9,1,0,0,60
  500nb%=6:na%=31
  510DIM hi$(7),hi%(7),sc% 15,col% 3,H% &20000,n$(9),ax%(na%),ay%(
na%), bx%(nb%),by%(nb%),ba%(nb%),ex%(7),ey%(7), sy%(100), sx%(100)
  520n$()="ACOMA","DBN","ACIGMO","ACGIOM","AGICO","CAGIOM","CAMOIG
","ACM","AGIOMGICA","OCAGI"
  530FOR I%=1 TO 7:hi$(I%)="Drag 'N Drop":hi%(I%)=1400-I%*200:NEXT
  540!col%=&F0703:!sc%=120:sc%!4=120
  550!H%=&20000:H%!8=16:o$="OS_SpriteOp"
  560SYS o$,256+9,H%
```

Floozy



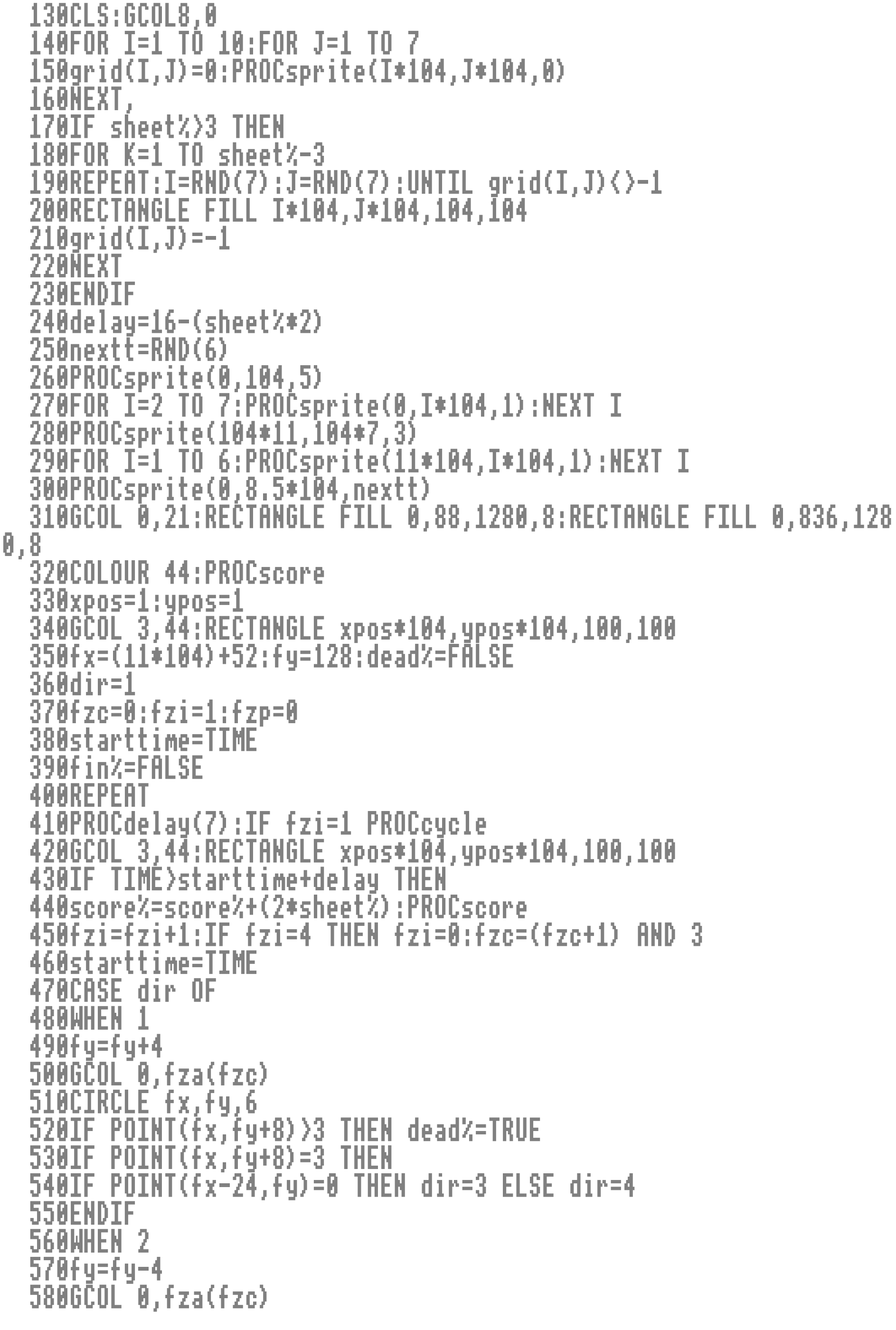
The local water board has claimed to invent a revolutionary new pipe to replace existing ones but unfortunately the Tetraflurine Carbonadium making up the pipes has disintegrated and you have been employed to put all the pipes back before the houses are flooded!

Move the cursor around the screen with the Z, X, / and @ keys and press Return to lay a pipe section from the display in the top left corner.

If you fail to connect the pipe before the water reaches you then you lose the game!

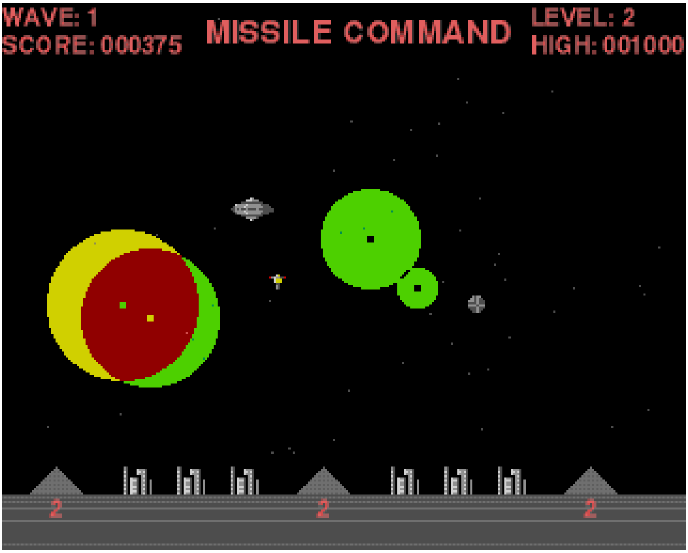
The game features animated water, six types of pipe section and and a high score table.

```
 10REM *** FLOOZY ***
 20MODE 13:OFF
 30ON ERROR REPORT:PRINT" at ";ERL:END
 40PROCinit
 50PROCmc
 60REPEAT
 70PROCscores
 80PROCgame
 90UNTIL FALSE
100:
110DEF PROCgame
120sheet%=1:score%=0
```



```
  130CLS:GCOL8,0
  140FOR I=1 TO 10:FOR J=1 TO 7
  150grid(I,J)=0:PROCsprite(I*104,J*104,0)
  160NEXT,
  170IF sheet%>3 THEN
  180FOR K=1 TO sheet%-3
  190REPEAT:I=RND(7):J=RND(7):UNTIL grid(I,J)<>-1
  200RECTANGLE FILL I*104,J*104,104,104
  210grid(I,J)=-1
  220NEXT
  230ENDIF
  240delay=16-(sheet%*2)
  250nextt=RND(6)
  260PROCsprite(0,104,5)
  270FOR I=2 TO 7:PROCsprite(0,I*104,1):NEXT I
  280PROCsprite(104*11,104*7,3)
  290FOR I=1 TO 6:PROCsprite(11*104,I*104,1):NEXT I
  300PROCsprite(0,8.5*104,nextt)
  310GCOL 0,21:RECTANGLE FILL 0,88,1280,8:RECTANGLE FILL 0,836,128
0,8
  320COLOUR 44:PROCscore
  330xpos=1:ypos=1
  340GCOL 3,44:RECTANGLE xpos*104,ypos*104,100,100
  350fx=(11*104)+52:fy=128:dead%=FALSE
  360dir=1
  370fzc=0:fzi=1:fzp=0
  380starttime=TIME
  390fin%=FALSE
  400REPEAT
  410PROCdelay(7):IF fzi=1 PROCcycle
  420GCOL 3,44:RECTANGLE xpos*104,ypos*104,100,100
  430IF TIME>starttime+delay THEN
  440score%=score%+(2*sheet%):PROCscore
  450fzi=fzi+1:IF fzi=4 THEN fzi=0:fzc=(fzc+1) AND 3
  460starttime=TIME
  470CASE dir OF
  480WHEN 1
  490fy=fy+4
  500GCOL 0,fza(fzc)
  510CIRCLE fx,fy,6
  520IF POINT(fx,fy+8)>3 THEN dead%=TRUE
  530IF POINT(fx,fy+8)=3 THEN
  540IF POINT(fx-24,fy)=0 THEN dir=3 ELSE dir=4
  550ENDIF
  560WHEN 2
  570fy=fy-4
  580GCOL 0,fza(fzc)
```

Missile Command

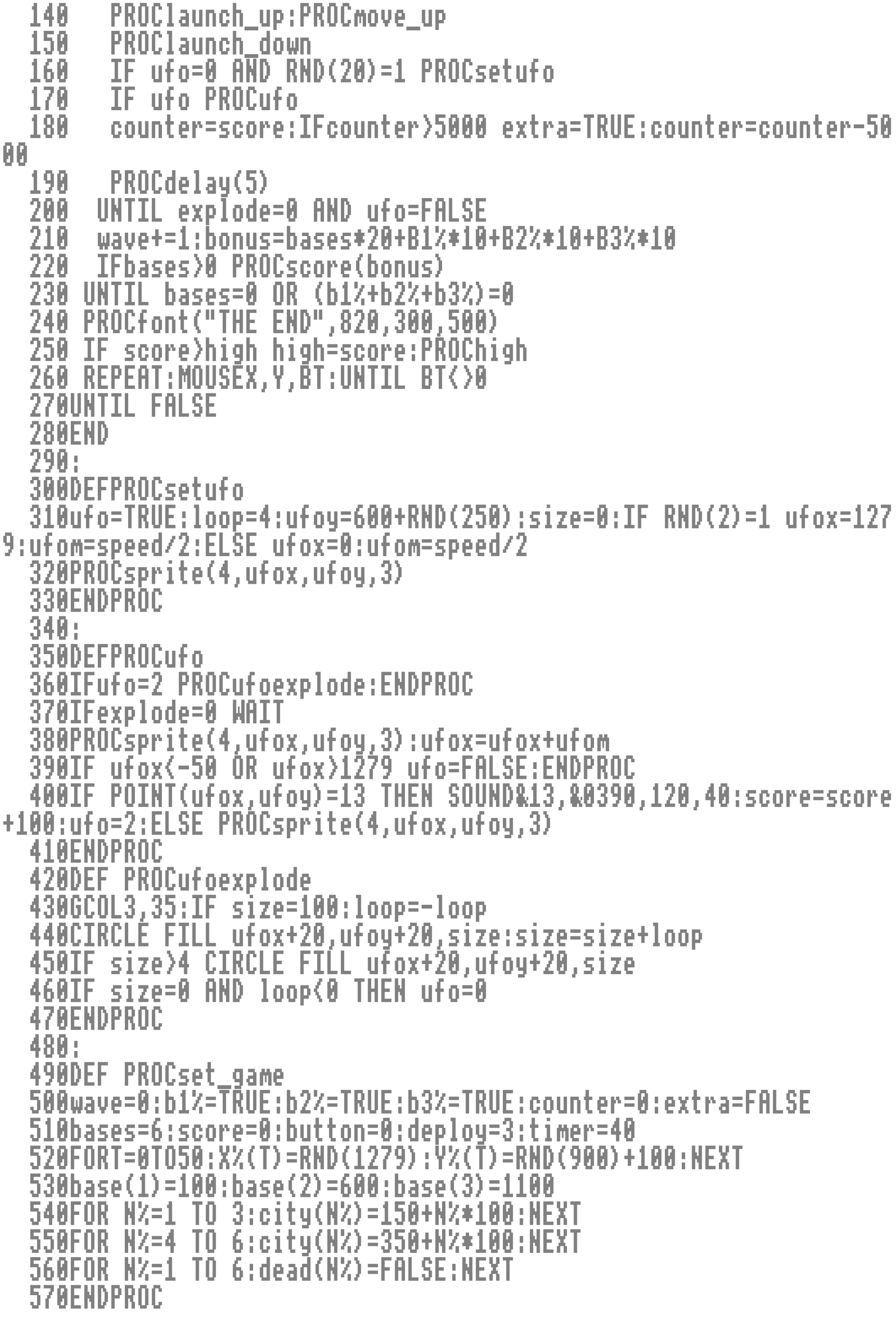


The earth is once again under invasion from outer space and your mission is to defend the cities against successive waves of bombs being dropped from the heavens.

Use the mouse to select your difficulty level then click the adjust, menu or select button to launch missiles from bases on the left, middle or right of the screen.

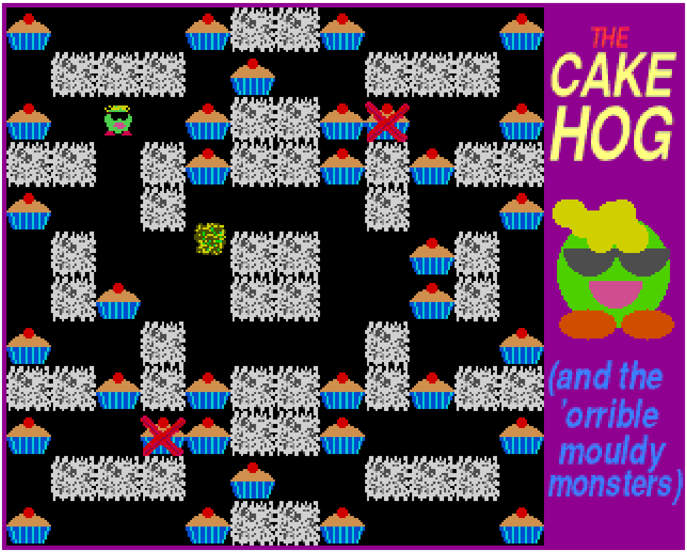
You advance to the next wave when all the enemy bombs have detonated. If all of your cities or missile bases have been destroyed the game ends.

```
 10REM *** MISSILE COMMAND ***
 20ONERROR GOTO 4000
 30MODE13:TINT2,0
 40OFF
 50PROCinit
 60REPEAT
 70 PROCset_game
 80 PROCdraw_screen
 90 PROCupdateinfo
100 PROCgetlevel
110 REPEAT
120  PROCset_wave:PROCupdateinfo
130  REPEAT
```



```
  140   PROClaunch_up:PROCmove_up
  150   PROClaunch_down
  160   IF ufo=0 AND RND(20)=1 PROCsetufo
  170   IF ufo PROCufo
  180   counter=score:IFcounter>5000 extra=TRUE:counter=counter-50
00
  190   PROCdelay(5)
  200  UNTIL explode=0 AND ufo=FALSE
  210  wave+=1:bonus=bases*20+B1%*10+B2%*10+B3%*10
  220  IFbases>0 PROCscore(bonus)
  230 UNTIL bases=0 OR (b1%+b2%+b3%)=0
  240 PROCfont("THE END",820,300,500)
  250 IF score>high high=score:PROChigh
  260 REPEAT:MOUSEX,Y,BT:UNTIL BT<>0
  270UNTIL FALSE
  280END
  290:
  300DEFPROCsetufo
  310ufo=TRUE:loop=4:ufoy=600+RND(250):size=0:IF RND(2)=1 ufox=127
9:ufom=speed/2:ELSE ufox=0:ufom=speed/2
  320PROCsprite(4,ufox,ufoy,3)
  330ENDPROC
  340:
  350DEFPROCufo
  360IFufo=2 PROCufoexplode:ENDPROC
  370IFexplode=0 WAIT
  380PROCsprite(4,ufox,ufoy,3):ufox=ufox+ufom
  390IF ufox<-50 OR ufox>1279 ufo=FALSE:ENDPROC
  400IF POINT(ufox,ufoy)=13 THEN SOUND&13,&0390,120,40:score=score
+100:ufo=2:ELSE PROCsprite(4,ufox,ufoy,3)
  410ENDPROC
  420DEF PROCufoexplode
  430GCOL3,35:IF size=100:loop=-loop
  440CIRCLE FILL ufox+20,ufoy+20,size:size=size+loop
  450IF size>4 CIRCLE FILL ufox+20,ufoy+20,size
  460IF size=0 AND loop<0 THEN ufo=0
  470ENDPROC
  480:
  490DEF PROCset_game
  500wave=0:b1%=TRUE:b2%=TRUE:b3%=TRUE:counter=0:extra=FALSE
  510bases=6:score=0:button=0:deploy=3:timer=40
  520FORT=0TO50:X%(T)=RND(1279):Y%(T)=RND(900)+100:NEXT
  530base(1)=100:base(2)=600:base(3)=1100
  540FOR N%=1 TO 3:city(N%)=150+N%*100:NEXT
  550FOR N%=4 TO 6:city(N%)=350+N%*100:NEXT
  560FOR N%=1 TO 6:dead(N%)=FALSE:NEXT
  570ENDPROC
```

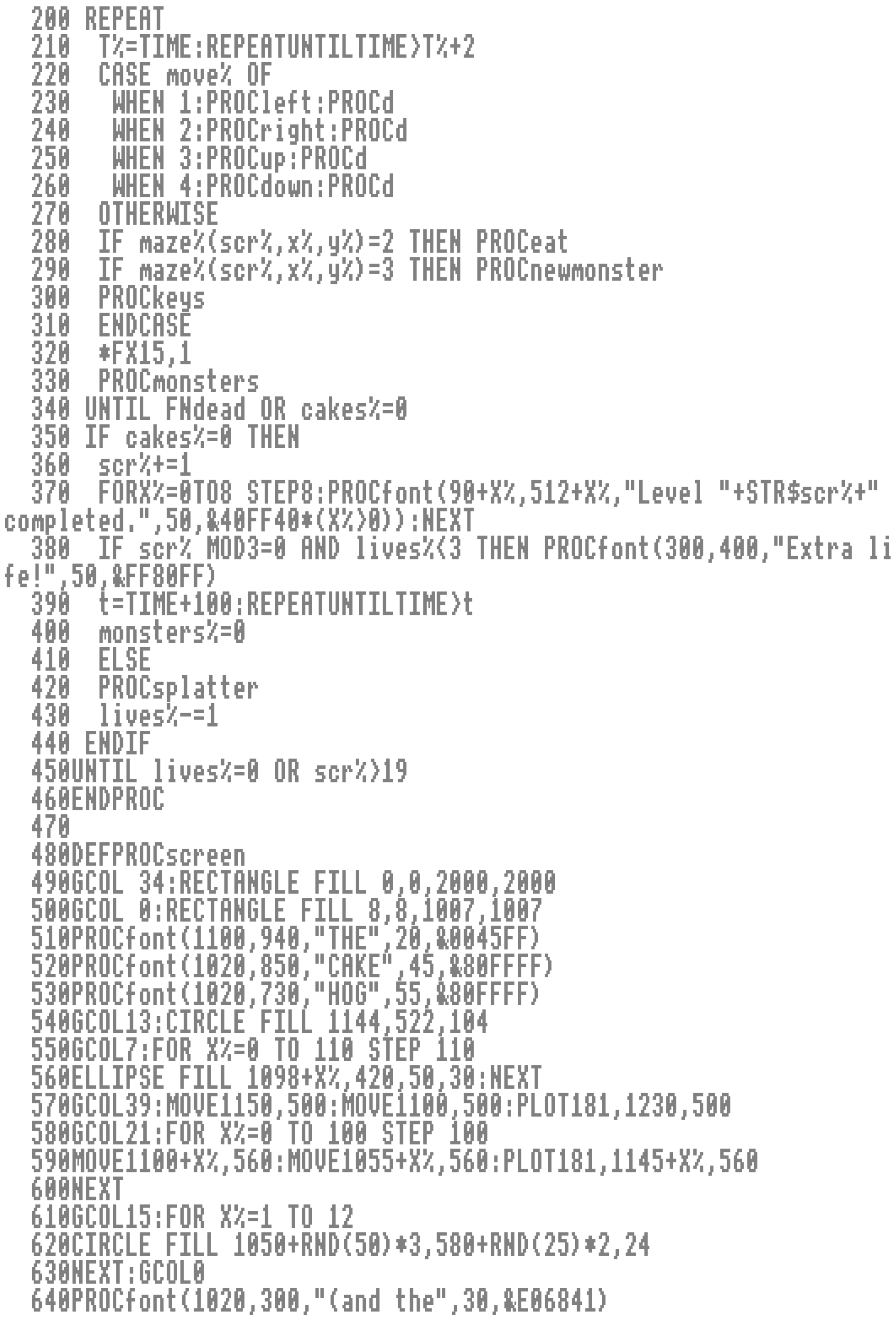
Cake Hog



It's just after midnight and you can't resist a snack of cream cakes but lurking inside the fridge are globs of horrible green mould. Can you eat all of the cakes before being captured by the evil stuff?

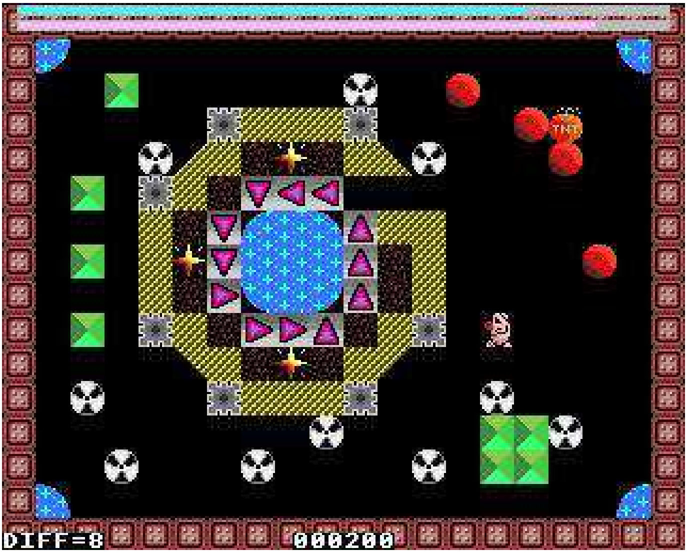
Use Z and X for left and right and K and M for up and down. Don't eat the wrong cakes which cause the mould to divide into more pieces!

```
 10REM *** CAKE HOG ***
 30MODE13:OFF
 40ON ERROR GOTO3260
 50TINT 2,0
 60PROCsprites
 70PROCinit
 80REPEAT
 90 PROCscreen
100 PROCgame
110 IF lives%=0 PROCgameover ELSE PROCwin
120UNTIL0
130END
140
150DEFPROCgame
160lives%=3
170REPEAT
180 PROCscreen
190 move%=0
```



```
  200 REPEAT
  210  T%=TIME:REPEATUNTILTIME>T%+2
  220  CASE move% OF
  230   WHEN 1:PROCleft:PROCd
  240   WHEN 2:PROCright:PROCd
  250   WHEN 3:PROCup:PROCd
  260   WHEN 4:PROCdown:PROCd
  270  OTHERWISE
  280  IF maze%(scr%,x%,y%)=2 THEN PROCeat
  290  IF maze%(scr%,x%,y%)=3 THEN PROCnewmonster
  300  PROCkeys
  310  ENDCASE
  320  *FX15,1
  330  PROCmonsters
  340 UNTIL FNdead OR cakes%=0
  350 IF cakes%=0 THEN
  360  scr%+=1
  370  FORX%=0TO8 STEP8:PROCfont(90+X%,512+X%,"Level "+STR$scr%+"
completed.",50,&40FF40*(X%>0)):NEXT
  380  IF scr% MOD3=0 AND lives%<3 THEN PROCfont(300,400,"Extra li
fe!",50,&FF80FF)
  390  t=TIME+100:REPEATUNTILTIME>t
  400  monsters%=0
  410  ELSE
  420  PROCsplatter
  430  lives%-=1
  440 ENDIF
  450UNTIL lives%=0 OR scr%>19
  460ENDPROC
  470
  480DEFPROCscreen
  490GCOL 34:RECTANGLE FILL 0,0,2000,2000
  500GCOL 0:RECTANGLE FILL 8,8,1007,1007
  510PROCfont(1100,940,"THE",20,&0045FF)
  520PROCfont(1020,850,"CAKE",45,&80FFFF)
  530PROCfont(1020,730,"HOG",55,&80FFFF)
  540GCOL13:CIRCLE FILL 1144,522,104
  550GCOL7:FOR X%=0 TO 110 STEP 110
  560ELLIPSE FILL 1098+X%,420,50,30:NEXT
  570GCOL39:MOVE1150,500:MOVE1100,500:PLOT181,1230,500
  580GCOL21:FOR X%=0 TO 100 STEP 100
  590MOVE1100+X%,560:MOVE1055+X%,560:PLOT181,1145+X%,560
  600NEXT
  610GCOL15:FOR X%=1 TO 12
  620CIRCLE FILL 1050+RND(50)*3,580+RND(25)*2,24
  630NEXT:GCOL0
  640PROCfont(1020,300,"(and the",30,&E06841)
```

Winglebith



Move Winglebith around the screen disposing of radioactive cannisters by pushing them into holes and avoiding monsters.

To complete each screen you must also collect the earth and crystals, defuse the bomb (surround it with boulders) and kill all the monsters (enclose with cannisters). And hurry there isn't much time!

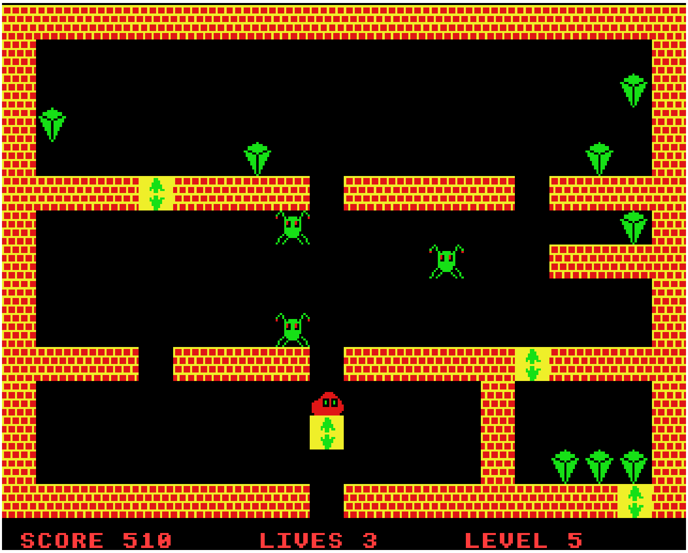
Move around the screen with Z for left, X for right, K for up and M for down.

Winglebith is a conversion of the BBC micro game called System Wadgebury by Andrew Cook.

```
 10REM *** WINGLEBITH ***
 20MODE 13
 30OFF
 40ON ERROR GOTO 5230
 50TEMPO 655:BEATS 32
 60PROCvariables
 70PROCsprites
 80REPEAT
 90PROCtitle
100REPEAT
110PROCgame
120IF death% PROCover ELSE PROCnext
```

```
130i%=1:REPEAT i%+=1
140UNTIL death% OR N%=4
150UNTIL 0
160END
170:
180DEF PROCover
190PROCfont("WINGLEBITH IS DEAD!",35,160,575)
200SYS "Sound_QInit"
210SOUND &11,&1F0,&E067,92
220PROCwait(200)
230PROChiscore
240ENDPROC
250:
260DEF PROCnext
270PROCbonus
280N%+=1
290IF N%=4 PROChiscore
300ENDPROC
310:
320DEF PROCgame
330death%=0
340PROCscreen(N%)
350wgx%=wcx%*64:wgy%=959-wcy%*64
360PROCsprite(wif%,wgx%,wgy%)
370 REPEATUNTILBEAT=0
380REPEAT
390PROCtune
400PROCkeys
410IF moves% PROCmoving
420PROCwait(2)
430PROCunder
440cnt%+=1:IF cnt%>mct% PROCmonsters
450GCOL 0,42
460LINE tme% DIV 8,1004,tme% DIV 8,1012
470LINE ener% DIV 8,976,ener% DIV 8,984
480IF bmx% tme%-=4
490ener%-=2
500IF tme% DIV8<32 OR ener% DIV8<32 death%=TRUE
510UNTIL death% OR (!count%=0 AND bmx%=0)
520ENDPROC
530:
540DEFPROCkeys
550IF INKEY-98 PROCleft
560IF INKEY-67 PROCright
570IF INKEY-102 PROCdown
580IF INKEY-71 PROCup
590IF INKEY-56 REPEATUNTILINKEY-52
```

Sid Slug



Sid Slug is in a spot of bother, can you help him collect all the diamonds and escape from the underground maze whilst avoiding the aliens, heaps of rubble and other pitfalls?

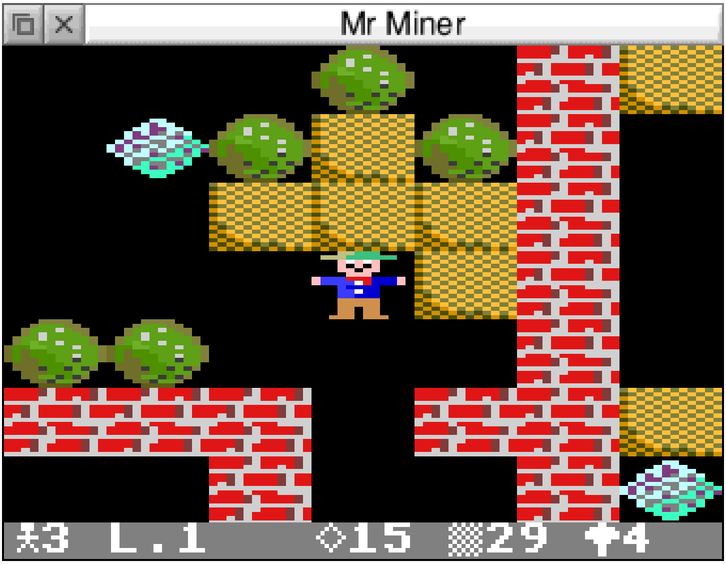
Use Z and X for left and right. If you are on a lift you can press / and @ to go up and down. Additionally, Shift and Return will ‘stretch’ Sid in the appropriate direction enabling you to short-cut distances.

When you have collected all the diamonds make your way to the exit gate, assuming of course you haven't blocked off your way back. You will then proceed to the next level.

The game ends when you have completed all levels or lost all three lives.

```
   10REM *** SID SLUG ***
   20ENVELOPE 1,&30,&C5,&59,98,105,105,-80,-16,-68,90,80,140,35
   30ENVELOPE 2,&30,&C5,&59,111,105,100,-40,-46,-68,85,110,120,35
   40ENVELOPE 3,&5B,&F0,8,0,0,0,0,0,0,1,25,15,5
   50MODE13
   60DIM AL(4,3),scale 12
   70ON ERROR GOTO 6280
   80PROCsprites:TINT2,0
   90OFF
  100REPEAT
  110LE=1:sco=0:lives=3
  120REPEAT CLS
  130RESTORE (3880+LE*160-160)
  140dias=0:rubble=0:ALI=1:CLS:FOR B=0 TO 14
  150READ A$
  160FOR A=0 TO 19:B$=MID$(A$,A+1,1)
  170IF B$="A" THEN PROCsprite(10,A*64,956-B*64):AL(ALI,1)=A*64:AL
(ALI,2)=956-B*64:AL(ALI,3)=1:ALI=ALI+1
  180IF B$="R" GCOL G%?4:MOVE A*64+28,1016-B*64:DRAW A*64+32,1016-
B*64
  190IF B$="O" PROCsprite(1,A*64,956-B*64)
  200IF B$="L" PROCsprite(2,A*64,956-B*64)
  210IF B$="D" PROCsprite(5,A*64,956-B*64):dias=dias+1
  220NEXT:NEXT:ALI=ALI-1
  230A=64:B=124:MOV=0:MT=TIME:MO=TIME:ALMT=TIME:PROCscore
  240PROCsprite(3,A,B)
  250REPEAT
  260IF INKEY-67 AND TIME>MO+8 PROCright
  270IF INKEY-98 AND TIME>MO+8 PROCleft
  280IF INKEY-80 PROCup
  290IF INKEY-105 PROCdown
  300IF INKEY-74 AND rubble=0 PROCstretchright
  310IF INKEY-1 AND rubble=0 PROCstretchleft
  320IF dias=0 PROCsprite(14,64,124):dias=-1
  330IF dias=-1 AND ((B=124 AND (A=64 OR A=96)) OR (A=64 AND B=188
)) THEN sco=sco+100:LE=LE+1:MOV=5
  340IF POINT(A+32,B-4)=0 AND POINT(A,B-4)=0 AND POINT(A+62,B-4)=0
 AND MOV=0 THEN PROCfall
  350IF FNpoint(A+32,B-4,2) AND FNpoint(A+30,B-4,2) AND MOV=0 THEN
 sco=sco+10:dias=dias-1:PROCscore:PROCfall
  360IF TIME>ALMT+10 AND ALI>0 THEN PROCmovealien
  370IF MOV=3 THEN lives=lives-1:PROCdie
  380IF MOV=2 AND TIME>MT+5 THEN PROCleft2
  390IF MOV=1 AND TIME>MT+5 THEN PROCright2
  400UNTIL MOV=3 OR MOV=5
  410UNTIL lives=0:*FX15,0
  420A=GET
  430UNTIL 0
  440DEFPROCright
  450IF MOV<>0 THEN ENDPROC
  460IF FNpoint(A+92,B+60,4) THEN rubble=1
  470IF FNpoint(A+92,B,1) OR FNpoint(A+92,B+60,1) OR FNpoint(A+92,
B,3) OR FNpoint(A+92,B+60,3) OR FNpoint(A+92,B,5) OR FNpoint(A+92,
B+60,5) THEN ENDPROC
  480IF FNpoint(A+94,B,2) AND FNpoint(A+94,B+60,2) THEN GCOL 0:REC
TANGLE FILL A+64,B,62,60:sco=sco+10:dias=dias-1:SOUND&12,&3F0,200,
5:PROCscore
  490exit=0:IF rubble=2 THEN
```

Desktop Miner



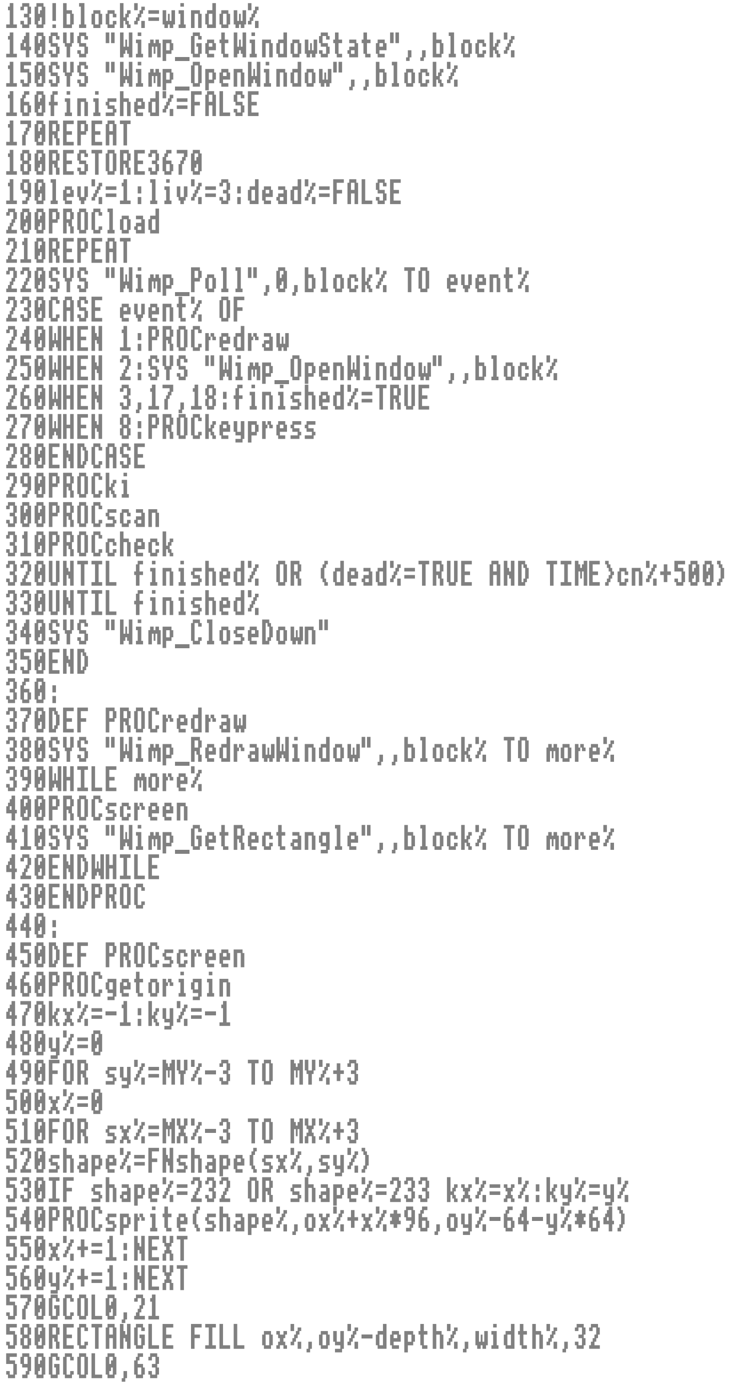
The object of this game, which runs in a window on the desktop, is to collect all of the jewels, dig out all of the earth, and kill the poisonous mushrooms.

Rocks always fall if they are unsupported and if they land on a gem or another rock they will roll off left or right if possible. Rocks must be dropped onto mushrooms to squash them, you cannot push rocks into mushrooms.

You must journey through five caves before you can leave the system.

Use Z and X for left and right and P and L for up and down. To quit just click on the window's close icon.

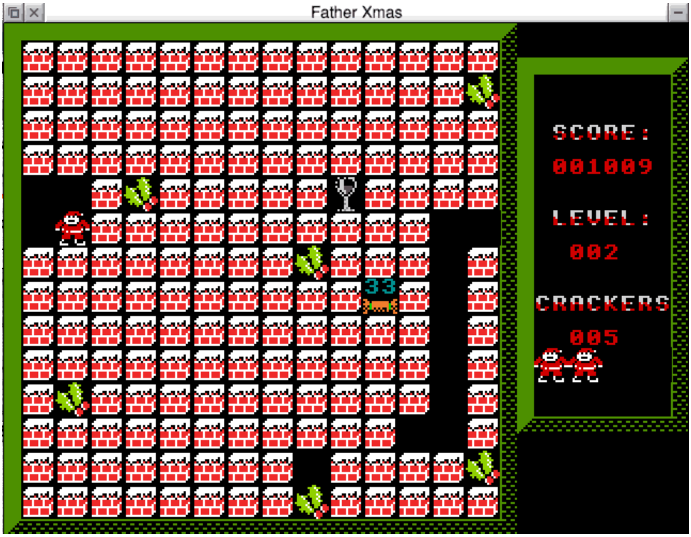
```
 10REM *** MR MINER ***
 20REM Converted from the Electron User game
 30game$="Mr Miner"
 40ON ERROR REPORT:PRINT" at ";ERL:END
 50PROCinit
 60ON ERROR PROCerror
 70SYS "Wimp_Initialise",500,&4B534154,"Mr Miner"
 80VDU 23,224,14,14,4,31,4,10,17,0
 90VDU 23,225,8,20,34,65,34,20,8,0
100VDU 23,226,85,170,85,170,85,170,85,170
110VDU 23,227,24,126,255,255,60,24,24,24
120PROCwindow
```



```
130!block%=window%
140SYS "Wimp_GetWindowState",,block%
150SYS "Wimp_OpenWindow",,block%
160finished%=FALSE
170REPEAT
180RESTORE3670
190lev%=1:liv%=3:dead%=FALSE
200PROCload
210REPEAT
220SYS "Wimp_Poll",0,block% TO event%
230CASE event% OF
240WHEN 1:PROCredraw
250WHEN 2:SYS "Wimp_OpenWindow",,block%
260WHEN 3,17,18:finished%=TRUE
270WHEN 8:PROCkeypress
280ENDCASE
290PROCki
300PROCscan
310PROCcheck
320UNTIL finished% OR (dead%=TRUE AND TIME>cn%+500)
330UNTIL finished%
340SYS "Wimp_CloseDown"
350END
360:
370DEF PROCredraw
380SYS "Wimp_RedrawWindow",,block% TO more%
390WHILE more%
400PROCscreen
410SYS "Wimp_GetRectangle",,block% TO more%
420ENDWHILE
430ENDPROC
440:
450DEF PROCscreen
460PROCgetorigin
470kx%=-1:ky%=-1
480y%=0
490FOR sy%=MY%-3 TO MY%+3
500x%=0
510FOR sx%=MX%-3 TO MX%+3
520shape%=FNshape(sx%,sy%)
530IF shape%=232 OR shape%=233 kx%=x%:ky%=y%
540PROCsprite(shape%,ox%+x%*96,oy%-64-y%*64)
550x%+=1:NEXT
560y%+=1:NEXT
570GCOL0,21
580RECTANGLE FILL ox%,oy%-depth%,width%,32
590GCOL0,63
```

```
 600PROCbig(CHR$224+STR$liv%,ox%,oy%-depth%+32)
 610PROCbig("L."+STR$lev%,ox%+3*32,oy%-depth%+32)
 620PROCbig(CHR$225+STR$maxdi%,ox%+9*32,oy%-depth%+32)
 630PROCbig(CHR$226+STR$maxea%,ox%+13*32,oy%-depth%+32)
 640PROCbig(CHR$227+STR$maxsq%,ox%+17*32,oy%-depth%+32)
 650ENDPROC
 660:
 670DEF PROCgetorigin
 680!block%=window%:SYS "Wimp_GetWindowState",,block%
 690ox%=block%!4
 700oy%=block%!16
 710ENDPROC
 720:
 730DEF PROCleft
 740L%=FNshape(MX%-1,MY%)
 750IF L%=224 OR L%=231 ENDPROC
 760IF L%=225 PROCrleft:IF no% ENDPROC
 770IF L%=226 PROCdia
 780IF L%=228 PROCearth
 790PROCshape(MX%,MY%,229)
 800MX%-=1
 810IF FNshape(MX%,MY%)>231 PROCsafes
 820PROCshape(MX%,MY%,230)
 830PROCscreen
 840ENDPROC
 850:
 860DEF PROCrleft
 870no%=FALSE
 880IF FNshape(MX%-2,MY%)<>229 no%=TRUE:ENDPROC
 890PROCshape(MX%-2,MY%,225)
 900PROCshape(MX%-1,MY%,229)
 910PROCscreen
 920IF L%=227 PROCdead
 930ENDPROC
 940:
 950DEF PROCright
 960L%=FNshape(MX%+1,MY%)
 970IF L%=224 OR L%=231 ENDPROC
 980IF L%=225 PROCrright:IF no% ENDPROC
 990IF L%=226 PROCdia
1000IF L%=227 dead%=TRUE
1010IF L%=228 PROCearth
1020PROCshape(MX%,MY%,229)
1030MX%+=1
1040IF FNshape(MX%,MY%)>231 PROCsafes
1050PROCshape(MX%,MY%,230)
1060PROCscreen
```

Father Christmas's Crackers

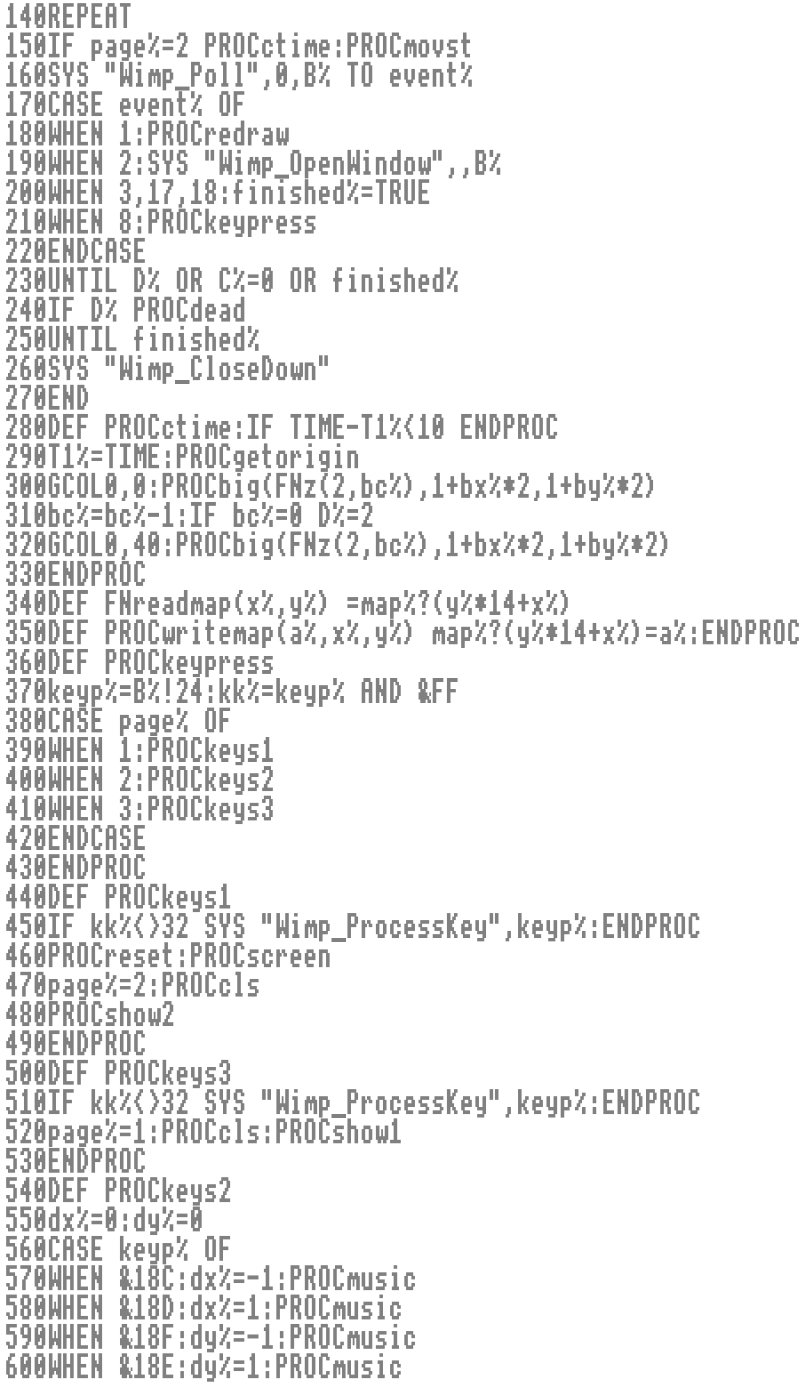


This game runs in a window on the desktop and sees Father Christmas out delivering presents when he discovers the mischevious gnomes have rigged the crackers with explosives! Can you defuse them all in time?

Move around with the cursor keys. Press shift in conjunction with the keys to move left or right the row of blocks which Father Xmas is on except when there is a cracker on the row. If you fail to defuse the cracker before its timer runs out you lose a life. The game ends when you've lost all your lives.

Beware the deadly holly and mutant stockings which roam the screens on later levels. But drink the brandy for bonus points.

```
  10REM *** FATHER CHRISTMAS'S CRACKERS ***
  20game$="Father Xmas"
  30ON ERROR REPORT:PRINT" at ";ERL:END
  40PROCinit
  50ON ERROR PROCerror
  60SYS "Wimp_Initialise",500,&4B534154,game$
  70PROCwindow
  80!B%=window%
  90SYS "Wimp_GetWindowState",,B%
 100SYS "Wimp_OpenWindow",,B%
 110PROCreset:page%=1:PROCupdate
 120finished%=FALSE
 130REPEAT
```



```
140REPEAT
150IF page%=2 PROCctime:PROCmovst
160SYS "Wimp_Poll",0,B% TO event%
170CASE event% OF
180WHEN 1:PROCredraw
190WHEN 2:SYS "Wimp_OpenWindow",,B%
200WHEN 3,17,18:finished%=TRUE
210WHEN 8:PROCkeypress
220ENDCASE
230UNTIL D% OR C%=0 OR finished%
240IF D% PROCdead
250UNTIL finished%
260SYS "Wimp_CloseDown"
270END
280DEF PROCctime:IF TIME-T1%<10 ENDPROC
290T1%=TIME:PROCgetorigin
300GCOL0,0:PROCbig(FNz(2,bc%),1+bx%*2,1+by%*2)
310bc%=bc%-1:IF bc%=0 D%=2
320GCOL0,40:PROCbig(FNz(2,bc%),1+bx%*2,1+by%*2)
330ENDPROC
340DEF FNreadmap(x%,y%) =map%?(y%*14+x%)
350DEF PROCwritemap(a%,x%,y%) map%?(y%*14+x%)=a%:ENDPROC
360DEF PROCkeypress
370keyp%=B%!24:kk%=keyp% AND &FF
380CASE page% OF
390WHEN 1:PROCkeys1
400WHEN 2:PROCkeys2
410WHEN 3:PROCkeys3
420ENDCASE
430ENDPROC
440DEF PROCkeys1
450IF kk%<>32 SYS "Wimp_ProcessKey",keyp%:ENDPROC
460PROCreset:PROCscreen
470page%=2:PROCcls
480PROCshow2
490ENDPROC
500DEF PROCkeys3
510IF kk%<>32 SYS "Wimp_ProcessKey",keyp%:ENDPROC
520page%=1:PROCcls:PROCshow1
530ENDPROC
540DEF PROCkeys2
550dx%=0:dy%=0
560CASE keyp% OF
570WHEN &18C:dx%=-1:PROCmusic
580WHEN &18D:dx%=1:PROCmusic
590WHEN &18F:dy%=-1:PROCmusic
600WHEN &18E:dy%=1:PROCmusic
```

Appendix 1: Upscaling

Apart from the two desktop games, the programs in this book run in Mode 10 or 13 which are available on RISC OS machines up to the Raspberry Pi (assuming the Anymode module has installed on the Pi). Newer machines are capable of displaying screens with a *miminum* resolution of 800 × 600 pixels. This means it is necessary to upscale the graphics which incidentally adds greater colour depth. This article, written by Tony Bartram – and which appeared in the Summer 2018 edition (9i4) of *Drag 'N Drop* – explains the general principle.

Upscaling and interpolation are used by various digital systems to handle legacy low resolution content. For example, HD (High Definition) televisions use upscaling to improve image resolution. Many old recordings used 576i source material.

The 576, incidentally, refers to the number of lines traditionally used on your television screen to display the picture and the 'i' refers to 'interlaced'. There are other standards in use around the world and you can read more on the internet if you are interested.

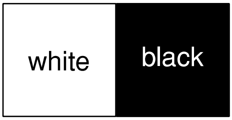
This technique can also be employed to improve the graphics used in 'type-in' games such as those published in *Drag 'N Drop* where space for data is limited.

Interpolation and Estimation

There are a number of popular scaling algorithms, including:

- Nearest-neighbour interpolation
- Bilinear interpolation
- Bicubic interpolation

Interpolation means 'filling in the missing information' or, put another way, create an intermediate unknown value between known values. Let's say we had a very simple (and very small) image consisting of just two pixels, white and black:


white
black