



## Sinclair QL Retro-Computing



## Sinclair QL Retro-Computing







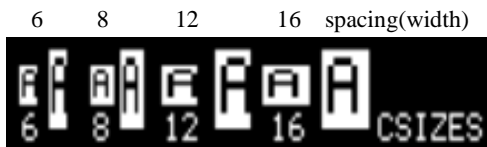
## Introduction

Home Computers in the 1980's created Characters from Bitmaps. The most common were 8x8 bit matrixes covering the ASCII code sets. The first two bytes can denote the Start/End ASCII codes the next two width and height. A function was then needed to write/chop a bitmap(font) into a bitmap(display) and presented at x,y coordinates. There might also be Colour, Background and XOR coding aspects included.

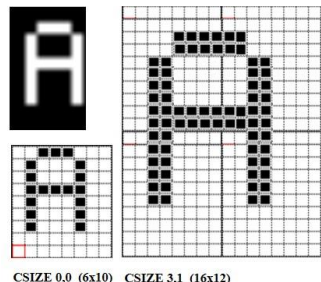
## Sinclair QL Character Fonts

The QL Font display driver has default Bitmap Patterns for ASCII Codes 32 (Space) to 127, the common character set (alphanumeric, maths signs, brackets etc.) the second is for Codes 128 to 191 called the extended character set. The Format is the first two Bytes hold the Lowest Code number followed by Total number of Fonts. Then 9 Bytes for each Font to give a 9x8 matrix used by the display app to scale in two widths and two heights of 10 or 20 rows with a single or double blank row above the Bitmap(Display).

The spacing between characters is entirely flexible, so two additional spacings are added to create Characters sizes (CSIZE) in the range of:-



and with 10 or 20 rows(height).



When creating the different CSIZES not all of the encoded Bits are displayed. This can lead to some interesting and at times frustrating outcomes when using Modified Fonts for say Retro Gaming.

## QBITS Font Editor Concept

The aspirations for a QBITS Font Editor began in the eighties. The Program was never finished for release, other events taking up ever more of my time. The concept was to Load alternative QL Font sets into memory, display them to screen as a Chart. A selected character would then be shown in a Bitmap display with the option to change the bit patten and save back to memory. After a number of Fonts had been edited in this way the whole Font Set could be saved as a new Font file for use with other Programs.

## QBITS QLFont Edit2 \_fnt Files

By using alternative Font Tables, the Character Patterns can be extended to include codes 0 to 31 and 192 to 255. The QBITS QLFont Editor2 creates Four Groups, QLFontA\_fnt (0/128), QLFont1\_fnt (32/96), QLFont2\_fnt (128/64) and QLFontB\_fnt (128,128).

The screen at start-up displays the default QL Character Fonts. These are the QLFontA & QLFontB fnt files loaded from the default storage device which needs to be set before running the program (see Program line 1007/8 Dev\$=???).

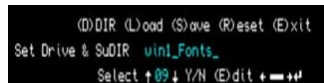
## QBITS QLFont Editor2 Navigation

Navigate using the Cursor keys and Select the Highlighted Character with Spacebar. This actions the Bitmap Grid, again navigate with cursor keys to highlight a Grid Cell and toggle the binary bit between 0 and 1 with Spacebar. Select 'N' to return without changing the existing Bitmap, 'Y' to write the changed bit pattern into memory.

## QBITS (D)DIR (L)oad (S)ave (R)eset (E)xit

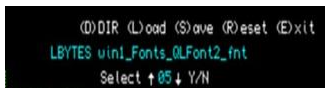
Access by pressing the bracketed Character. Information is displayed relative to the action requested. (D) Selects the drive and SubDIRectories if any are present.

Drive & SubDIR's can be entered directly into the Program on DATA Lines 1011 to 1013 or when run use the (E)dit ie. to change 'drv1\_' to 'win1\_Fonts'



For Load / Save : first Select Storage Device with Up/Down Cursor keys and Y/N

For **LOAD** a search is made of available '\_fnt' front files. A 'File Not found' will be returned if none are found. Use the Up/Down cursors to scroll through and make your choice then Y/N to load or abort. A program check is made to select the relevant Character Set Memory address, before the File is loaded and Fonts displayed to screen.



To **SAVE** use Up/Down cursors to Select from (1-2-3-4). the current Font filename for that Group is displayed. Included is a Line Editor to Rename the Font file. When ready to save a check is made and if device unavailable a 'DEVICE ERROR' is given. An 'Overwrite Y/N' is given if the file is detected as already existing. If good to go or answer Y the file is saved with 'Saving...' being displayed before returning to character display.

Press 'R' for **Reset** which prompt with 'Y/N', 'N' aborts and 'Y' reloads Default Fonts.



Pressing 'E' for **Exit** will again prompt with Y/N, 'N' returns to program, 'Y' will close opened channels and free Memory before halting the Program. If desired an LRUN can be added to Line 1089 start a Boot or Menu Program.

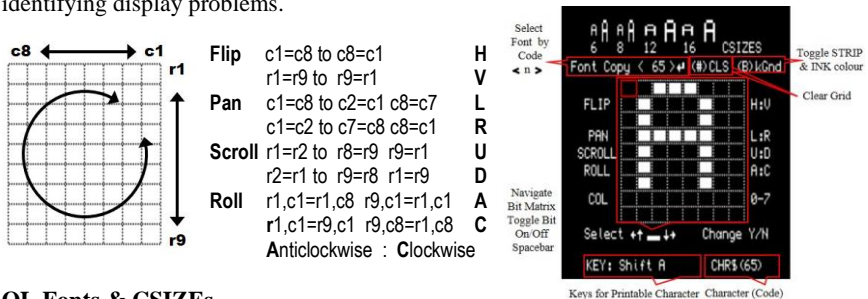
## QBITS QLFont Edit2

The heart of the Editor is the Bitmap and what changes you can make to the character Fonts. It may be changes to reflect Bold, Italics, or your own stylised alphanumerical Fonts or maybe futuristic, or to depict an ancient language such as Cuneiform or Runes. For Retro Games transforming Fonts into game pieces might be desirable.

The **Bitmap** shows bits set to 0's & 1's for chosen Font. Returning from the Bitmap the Character under review is displayed in its different **CSIZES** with **KEY**: showing the key(s) needed for printing the **CHR\$(n)** Character code 'n'.

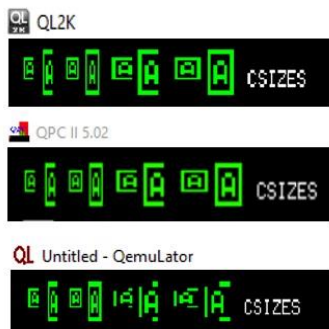
Where slight pattern changes might be desired, the ability to Copy an existing Font Character into the Font Grid could be useful. Use Keys '<' >' to select and Enter to action. For new designs (#) CLS clears the Bitmap setting all to '0'. (B)kGnd toggles STRIP & INK colours between Black(0) and White(7). For further actions Flip (H)orizontally, or (V)ertically, Pan (L)eft or (R)ight, Scroll (Up) & (D)own and Rotate (A)nti-clockwise or (C)lockwise by 90°. To change Font INK colour use (0-7). Use cursor keys to navigate the Bitmap and toggle On/Off Bite INK colour with Spacebar.

Showing the Fonts in their various CSIZES with different colours can be helpful to identifying display problems.



## QL Fonts & CSIZES

The bit Pattern 'A' is surrounded with a box. Not all CSIZES show the whole of the box and some QL O/S display different to expected.



As mentioned, the Font app for the QL does not retrospectively produce all of the bit pattern held by a character's Bitmap across the range of CSIZES.

## QBITS QL Font Installation

**CHAR\_USE** is used set and reset one or both character QL Font sets. First memory from the common heap (RAM) area is allocated for loading a Character set. The number of bytes rounded off to even values for LBYTES to work properly, for example:

```
FBBase1 = ALCHP(876)   LBYTES FLP1_FONT1,FBBase1  
FBBase2 = ALCHP(588)   LBYTES FLP1_FONT2,FBBase2
```

**CHAR\_USE**#ch,FBBase1,FBBase2 assigns Font Sets to a specified **channel**.

With the Fonts Bitmaps assigned to memory a character pattern can be Read and Overwritten with the PEEK and POKE commands. Each is identified by an offset from the assigned Base address set by **ALCHP**. The first 2 Bytes hold the Lowest code & Total number of Font Characters, then followed by the number of Bitmap 8x9 matrixes each forming multiples of 9 Bytes.

When the program is finished **RECHP**(FBBase?) releases memory and reset to the default character set with **CHAR\_USE**#ch,0,0. If **RESPR**(876+588) is used to acquire Memory space the loaded Fonts remain available until a power down or a System Reset.

## ARCADE CLASSIC

### PIXEL Based Font Designs

Pixel-based Characters were popular in the 1980s, when the digital era was just beginning. One typeface stood above the rest the “Atari” font set of 1984 (or known as “Namco” font in Japan). Based on the western ACSII character set it lasted until larger resolutions and 3D gaming became the norm, it stands out as a shining example of how video game designers used an 8-by-8 Grid to communicate to players.



The 8x8 bitmap matrix was also used in Arcade Games from the 1970s through to the 1990s. Their chunky and primitive visuals revealed artistry and ingenuity to make full-fledged characters out of just a few pixels. These Fonts were designed by hand on graph paper and coded into the game pixel by pixel. The result was a mix of some good ideas and some really bad ones.

### Note: Retro Game Control

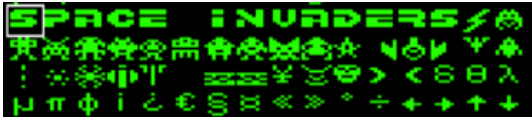
One other useful Character keyword is **CHAR\_INC**#ch, x\_inc, y\_inc which sets a Character width and height beyond those set by **CSIZE**. Where the subject Character Font(s) are moved across the screen, an independent height can be set for **PAN**.

**PAN** #ch, distance, 3 (the action which PAN's the whole of the cursor line left or right).

## QBITS QLFont Edit2 SE

The Special Edition of FontEdit2 is an opportunity to explore Font designs used in classical ARCADE Games. My choice Space Invaders, Dino Run and Giro Rescue. Each uses redefined Fonts of the ASCII extended range 128 to 191. Load relative \_fnt file into the Editor and then press 'G'. The game should now be displayed in WINDOW#3.

### Space Invaders



This simplified version has four levels. Move the Defender left or Right using Cursor Keys and Spacebar fires the laser. You have four Lives so watch out for enemy bombs.



### Dino Run



Spacebar Jumps Dion over the approaching Cacti or you lose. Displayed are High Score and current Accumulated Points. This Prog uses CHAR\_INC to PAN the screen.



### Giro Rescue



The aim is to rescue all the Life pods scattered through space, But watch out for space Debry they can weaken your shields and you have a limited fuel to reach high levels.



**Note:** In recognition of their published free programs, I wish to give thanks to Dilwyn Jones DINO - 2022 and Henry Wrighton GIRO - Aug 1889 and for borrowing their code.

## QBITS Program Code

To maintain compatibility across the various QL Platforms has its drawbacks. The original QL came with 128K of Ram 32K used for the screen, and more for such as the SuperBASIC Interpreter. QBITS\_QLFont\_Edit2 should load and run on a BBQL but requires TK2 keywords CHAR\_USE & CHAR\_INC to be present. It is recommended to fit some extended memory as the use of Arrays can overload available memory.

For higher speed QL platforms as you scan through the Font Charts the relative Bitmap is displayed, for the BBQL this aspect can be disabled and the Bitmap shown only when a Font is selected.



## QBITS Editor Code

1000 REMark **QBITS\_FontEdit2\_SE** (QBITS Font Editor2 SE QBITS 2022)

1002 WMON:gx=0:gy=0:CHAR\_USE 0,0 :REMark gx gy Offset High Res Screens

1004 DIM Fnt\$(9,8),Tmp\$(9,8) :REMark Font Grid Arrays

1005 DIM File\$(50,20):fm%=50 :REMark Font Files Array fm max Files

**Note:** Change value for fm% as required but may require additional memory.

1007 REMark Set drv\$(n)<>Drives/SubDIRectories & Dev\$ as Default

1008 DIM drv\$(16,16):Dev\$='win1\_'

1009 dm%=16:dn%=5 :REMark dm% device max dn% device number

1010 RESTORE 1011:FOR d=1 TO dm%:READ str\$:drv\$(d)=str\$

1011 DATA 'mdv1\_',mdv2\_',flp1\_',flp2\_',win1\_',win2\_',win3\_',win4\_'

1012 DATA 'win1\_Fonts\_',drv1\_',drv1\_',drv1\_'

1013 DATA 'drv1\_',drv1\_',drv1\_',drv1\_'

**Note:** Change DATA entries of Drive and or SubDIR as required to reflect the connected devices.

1015 FBaseA=ALCHP(1164) : FBase1=ALCHP(876) :REMark Font RAM Allocation

1016 FBaseB=ALCHP(1164) : FBase2=ALCHP(588)

1018 WHEN ERRor

1019 eck=1:CONTINUE

1020 END WHEN

1022 Init\_Screen:ARCADE:FontMain

1024 DEFine PROCedure Init\_Screen

1025 WINDOW#0,512,32,gx,gy+224:PAPER#0,0 :CSIZE#0,0,0:BORDER#0,1,3:CLS#0

1026 WINDOW#1,512,224,gx,gy :PAPER#1,0 :CSIZE#1,0,0:BORDER#1,1,3:CLS#1

1027 WINDOW#2,512,224,gx,gy :PAPER#2,0 :CSIZE#2,0,0:BORDER#2,1,3:CLS#2

1028 OPEN#3,scr\_:WINDOW#3,276,178,gx+18,gy+42:CSIZE#3,3,0:INK#3,4

1029 OPEN#4,scr\_:WINDOW#4,180,146,gx+314,gy+56:CSIZE#4,3,0:INK#4,4

1030 CSIZE#2,2,1:OVER#2,1

1031 INK#2,2:FOR i=0 TO 1:CORSOR#2,6+i,6:PRINT#2,'QBITS QLFont Editor2'

1032 INK#2,6:FOR i=0 TO 1:CORSOR#2,8+i,4:PRINT#2,'QBITS QLFont Editor2'

1033 CSIZE#2,0,0:OVER#2,0:INK#1,7

1034 CURSOR#1,280,14:PRINT#1,'(D)IR (L)oad (S)ave (R)eset (E)xit'

1035 CURSOR#1,4,28:PRINT#1,'Grp Select ← ↑ ↓ → ':BLOCK#1,12,3,86,32,7

1036 FOR i=1 TO 5:READ fy,ch\$:CURSOR 6,fy:PRINT ch\$

1037 DATA 42,'1',64,'2',130,'3',152,'&',174,'4'

1038 END DEFine





```

1040 DEFine PROCEDURE FontMain
1041 cx=0:cy=2:x=1:y=1:FontReset 0:chk=0:eck=0
1042 REPEAT Main_lp
1043   cn=cx+(cy)*16:BLOCK 280,10,226,28,0:BLOCK 200,12,300,40,0:INK 7
1044   FontChar cx,cy,7:KeyPad:FontPeek:FontSize
1045   K=CODE(INKEY$(-1))
1046   FontChar cx,cy,0 :IF K<>32:FontGrid
1047   SElect ON K
1048     =192:cx=cx-1:IF cx< 0:cx=15:cy=cy-1:IF cy< 0:cy= 0:cx=0
1049     =200:cx=cx+1:IF cx>15:cx= 0:cy=cy+1:IF cy>15:cy=15:cx=15
1050     =208:cy=cy-1:IF cy< 1:cy= 0
1051     =216:cy=cy+1:IF cy>15:cy=15
1052     = 32:FontPeek:FontMod :REMark Modify Char
1053     =100,68:FontDIR :REMark (D)IR Set Drive/SubDIR
1054     =103,71:FontGame :REMark (G)ames
1055     =108,76:FontLoad :REMark (L)oad
1056     =115,83:FontSave :REMark (S)ave
1057     =114,82:FontReset 1 :REMark (R)eset
1058     =101,69:FontExit :REMark (E)xit
1059   END SElect
1060 END REPEAT Main_lp
1061 END DEFine

```

```

1063 DEFine PROCEDURE FontReset(cK)
1064 IF cK=1:CURSOR#1,412,28:PRINT#1,'Y/N':PAUSE:IF KEYROW(5)<>64:RETurn
1065 LBYTES Dev$&QLFontA_fnt',FBaseA:cn1$=QLFontA_fnt':cn2$=QLFont1_fnt'
1066 LBYTES Dev$&QLFontB_fnt',FBaseB:cn4$=QLFontB_fnt':cn3$=QLFont2_fnt'
1067 POKE FBaseA,0,127:POKE FBaseB,127,127
1068 POKE FBase1,32,96:POKE FBase2,127,64
1069 FontSets:FontGrid:cn=32:col=7
1070 END DEFine

```

```

1072 DEFine PROCEDURE FontSets
1073 CHAR_USE#3,FBaseA,FBaseB:CHAR_USE#4,FBaseA,FBaseB
1074 fx=2:fy=1:CSIZE#3,3,0:INK#3,4:CLS#3:CLS#4:INK#1,7
1075 FOR c=0 TO 255
1076   CURSOR#3,fx,fy:PRINT#3,CHR$(c)
1077   fx=fx+17:IF fx>260:fx=2:fy=fy+11
1078 END FOR c
1079 END DEFine

```

```

1081 DEFine PROCEDURE FontChar(cx,cy,ci)
1082 INK#3,4:CURSOR#3,2+cx*17,1+cy*11:PRINT#3,CHR$(cn)
1083 BLOCK#3,20,1,cx*17,cy*11,ci:BLOCK#3,20,1,cx*17,12+cy*11,ci
1084 BLOCK#3,1,12,cx*17,cy*11,ci:BLOCK#3,1,12,19+cx*17,cy*11,ci
1085 END DEFine

```

**Note:** Highlights a Character within the Font Charts

```

1087 DEFine PROCEDURE FontExit
1088 CURSOR#1,460,28:PRINT#1,'Y/N':PAUSE:IF KEYROW(5)<>64:RETurn
1089 CLOSE#4:CLOSE#3:CLS#1:RECHP FBase1:RECHP FBase2:STOP
1090 END DEFine

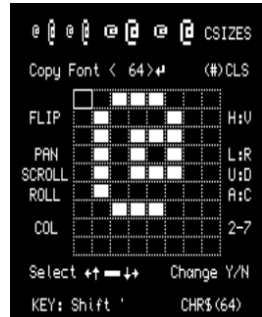
```

**Note:** Exit closes open channels and releases heap memory. STOP can be replaced with an LRUN command to return to another program such as LRUN flip1\_Boot or win1\_Progs\_Menu etc.



# 1100 DEFine PROCEDURE FontMod

```
1101 CURSOR#1,306, 82:PRINT#1,'Font Copy < > (#)CLS (B)kGnd'
1102 CURSOR#1,338,190:PRINT#1,'Select ←↑ ↓→ Change Y/N'
1103 BLOCK 2,4,408,84,7:BLOCK 12,3,376,194,7
1104 RESTORE 1097:FOR i=1 TO 10:READ a,b,c$:CURSOR a,b:PRINT c$
1105 DATA 318,106,'FLIP',322,126,'PAN',312,137,'SCROLL',318,148,'ROLL'
1106 DATA 470,106,'H:V',470,126,'L:R',470,137,'U:D',470,148,'A:C'
1107 DATA 322,166,'COL',470,166,'2-7'
1108 FontChar cx,cy,7:FontSize:KeyPad:rm=9:cs=8:co=cn
1109 REPEAT Chg_lp
1110 CURSOR 374,82:PRINT FILL$(' ',3-LEN(cn))&cn
1111 FontBit x,y,7:K=CODE(INKEY$(-1)):FontBit x,y,248
```



```
1112 SElect ON K
1113 =192:x=x-1:IF x<1:x=1
1114 =200:x=x+1:IF x>8:x=8
1115 =208:y=y-1:IF y<1:y=1
1116 =216:y=y+1:IF y>9:y=9
1117 =48 TO 55:col=K-48:FOR r=0 TO 8:FontDraw :REMark Colour change
1118 = 32:BitSwap :REMark Bit Swap 0<>1
1119 = 60,44:cn=cn-1:IF cn<0:cn=0 :REMark <, Lower cn
1120 = 62,46:cn=cn+1:IF cn>255:cn=255 :REMark >, higher cn
1121 = 10:FontPeek:FontDraw :REMark Change Font Pattern
1122 = 35:FontNew :REMark (#) Reset Grid
1123 = 66,98:FontBkGnd :REMark (B)gnd Colour
1124 =104,72:FontFlip 9,-1,0,1 :REMark (H)orizontal Flip
1125 =118,86:FontFlip 0,1,10,-1 :REMark (V)ertical Flip
1126 =108,76:FontSlid 0,8,1,0,1 :REMark (L)eft PAN Grid
1127 =114,82:FontSlid 0,1,8,1,0 :REMark (R)ight PAN Grid
1128 =117,85:FontSlid 1,9,1,1 :REMark (U)p SCROLL Grid
1129 =100,86:FontSlid 1,1,9,-1 :REMark (D)n SCROLL Grid
1130 = 97,65:FontRoll 1 :REMark (A) Roll Anti-Clockwise
1131 = 99,67:FontRoll 2 :REMark (C) Roll Clockwise
1132 =110,78:EXIT Chg_lp :REMark (N)o Change Return
1133 =121,89:cn=co:FontPoke:EXIT Chg_lp :REMark (Y)es Change Font
1134 END SElect
1135 END REPEAT Chg_lp
1136 cn=co:CLS#4:FontGrid
1137 END DEFine
```

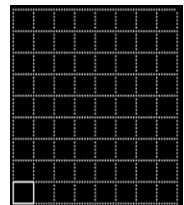
# 1139 DEFine PROCEDURE FontGrid

```
1140 BLOCK#4,112,90,46,40,bcol
1141 FOR i=0 TO 8:BLOCK#4,1,91,40+i*14,40,248
1142 FOR i=0 TO 7:BLOCK#4,112,1,40,40+i*10,248
1143 END DEFine
```

# 1145 DEFine PROCEDURE FontBit(x,y,ci)

```
1146 BLOCK#4,14,1,26+x*14,30+y*10,ci:BLOCK#4,14,1,26+x*14,40+y*10,ci
1147 BLOCK#4,1,10,26+x*14,30+y*10,ci:BLOCK#4,1,10,40+x*14,30+y*10,ci
1148 END DEFine
```

Note: Highlights a Bit Square within the Font BitMap



# 1150 DEFine PROCEDURE FontBkGnd

```
1151 IF bcol=0:bcol=7:col=0:ELSE bcol=0:col=7
1152 FontGrid:FOR r=0 TO 8:FontDraw
1153 END DEFine
```

Note: Toggle Black/White STRIP & White/Black INK

```

1155 DEFine PROCEDURE FontNew
1156 FOR r=0 TO 8:Font$(r+1)='00000000':FontDraw
1157 END DEFine

```

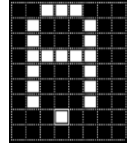
**Note:** Set all Matrix Bits to Zeros

```

1159 DEFine PROCEDURE FontDraw
1160 FOR c=0 TO 7
1161 IF Font$(r+1,c+1)='1':pcol=col:ELSE pcol=0
1162 BLOCK#4,11,7,42+c*14,42+r*10,pcol
1163 END FOR c
1164 END DEFine

```

**Note:** Print Pixel colour



```

1166 DEFine PROCEDURE BitSwap
1167 IF Font$(y,x)='0':Font$(y,x)='1':ELSE Font$(y,x)='0'
1168 IF Font$(y,x)='0':BLOCK#4,11,7,42+(x-1)*14,42+(y-1)*10,bcol
1169 IF Font$(y,x)='1':BLOCK#4,11,7,42+(x-1)*14,42+(y-1)*10,col
1170 END DEFine

```

**Note:** STRIP

**Note:** INK

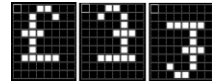


```

1172 DEFine PROCEDURE FontFlip(cf,cz,rf,rz)
1173 FOR r=1 TO 9:Tmp$(r)=Font$(r)
1174 FOR r=1 TO 9:FOR c=1 TO 8:Font$(r,c)=Tmp$(rf+r*rz,cf+c*cz):END FOR c
1175 FOR r=0 TO 8:FontDraw
1176 END DEFine

```

**Note:** Flip Horizontal or Vertical



```

1178 DEFine PROCEDURE FontSlid(md,a,b,d,e)
1179 FOR r=1 TO 9:Tmp$(r)=Font$(r)
1180 FOR r=1 TO 9
1181 IF md=0:Font$(r,a)=Tmp$(r,b):FOR c=1 TO 8:Font$(r,c+d)=Tmp$(r,c+e):END FOR c
1182 IF md=1:Font$(a)=Tmp$(b) :FOR c=1 TO 8:Font$(r,c)=Tmp$(r+d,c) :END FOR c
1183 END FOR r
1184 FOR r=0 TO 8:FontDraw
1185 END DEFine

```

**Note:** Slide Left Right



**Note:** Slide Up Down

```

1187 DEFine PROCEDURE FontRoll(rd)
1188 FOR r=1 TO 9:Tmp$(r)=Font$(r)
1189 FOR r=1 TO 8
1190 IF rd=1:rx=r :ry=8:FOR c=1 TO 8:Font$(ry,rx)=Tmp$(r,c):ry=ry-1:END FOR c
1191 IF rd=2:rx=9-r:ry=1:FOR c=1 TO 8:Font$(ry,rx)=Tmp$(r,c):ry=ry+1:END FOR c
1192 END FOR r
1193 FOR r=0 TO 8:FontDraw
1194 END DEFine

```

**Note:** Rotate 90° Anticlockwise



**Note:** Rotate 90° Clockwise



```

1196 DEFine PROCEDURE FontSize
1197 CURSOR 452,60:PRINT 'CSIZES':RESTORE 1184:INK#4,col
1198 CURSOR 324,70:PRINT '6 8 12 16'
1199 FOR i=1 TO 8:READ a,b,c,d:SIZE#4,a,b:CURSOR#4,c,d:PRINT#4,CHR$(cn)
1200 DATA 0,0,8,2,0,1,18,0,1,0,30,2,1,140,0,2,0,54,2,2,1,70,0,3,0,90,2,3,1,108,0
1201 END DEFine

```



```

1203 DEFine PROCEDURE FontPeek
1204 IF cn<128:addr=FBaseA+2+cn*9
1205 IF cn>127:addr=FBaseB+2+(cn-127)*9
1206 FOR r=0 TO 8:Font$(r+1)=BIN$(PEEK(addr+r),8):FontDraw
1207 END DEFine

```

**Note:** Read Font Bit Pattern from RAM

1209 **DEFine PROCEDURE FontPoke**

**Note: Write New Font Bit Pattern to RAM**

```
1210 IF cn<128:addr=FBaseA+2+cn*9:ELSE addr=FBaseB+2+(cn-127)*9
1211 FOR r=0 TO 8:POKE(addr+r),BIN(Fnt$(r+1))
1212 END DEFine
```

```
@DIR Load Save Reset Exit
LBYTES vint_Fonts_0LFont2_int
Select +05 Y/N
```

1250 **DEFine PROCEDURE FontLoad**

```
1251 chk=0:eck=0:Lchk=0:sf%=1:ft%=0:FOR i=1 TO fm%:File$(i)="
1252 INK 7:CURSOR 300,46:PRINT 'Select ↑ ↓ Y/N':INK 5:SelDrv 1:FntList 1
1253 SetFont 1,'LBYTES':IF Lchk=0:RETurn
1254 BLOCK 280,10,226,28,0:CURSOR 310,28:PRINT#1,'Loading...':PAUSE 30
1255 OPEN _IN#9,drv$(dn%)&File$(sf%)
1256 sg%=CODE(INKEY$(#9)):cg%=CODE(INKEY$(#9))
1257 CLOSE#9
1258 IF sg%< 31 AND cg%> 96:addr1=FBaseA:cy= 0:cn1$=File$(sf%):cn2$=cn1$:cn1=0
1259 IF sg%>=31 AND sg%<127:addr1=FBase1:cy= 2:cn2$=File$(sf%):addr2=FBaseA:cn1=96
1260 IF sg%>126 AND cg%< 65:addr1=FBase2:cy= 8:cn3$=File$(sf%):addr2=FBaseB:cn1=64
1261 IF sg%>126 AND cg%> 64:addr1=FBaseB:cy=12:cn4$=File$(sf%):cn3$=cn4$:cn1=0
1262 LBYTES drv$(dn%)&File$(sf%),addr1:cx=0:IF cn1=96:os=sg%:ELSE os=sg%-127
1263 IF cn1=96 OR cn1=64
1264 FOR a=1 TO cn1
1265 FOR b=0 TO 8:POKE addr2+2+(a+os)*9+b,PEEK(addr1+2+a*9+b)
1266 END FOR a
1267 END IF
1268 FontSets:eck=0:INK 7:FontGrid
1269 END DEFine
```

```
@DIR Load Save Reset Exit
Loading...
```

1271 **DEFine PROCEDURE FontSave**

```
1272 chk=0:eck=0:Lchk=0:sf%=1:ft%=$4
1273 File$(1)=cn1$:File$(2)=cn2$:File$(3)=cn3$:File$(4)=cn4$
1274 INK 7:CURSOR 300,42:PRINT 'Select ↑ ↓ Y/N (E)dit ← → ←':
1275 BLOCK 12,3,454,46,7:BLOCK 2,4,480,44,7:INK 5:SelDrv 2
1276 SetFont 2,'SBYTES':IF Lchk=0:RETurn
1277 FntList 2:BLOCK 200,10,300,42,0 :CURSOR 312,42:INK 7
1278 IF eck=1:PRINT#1,'DEVICE ERROR...':PAUSE 50:RETurn
1279 IF chk=1:PRINT#1,'Overwrite Y/N' :PAUSE:IF KEYROW(5)<>64:RETurn
1280 INK 5:DELETE drv$(dn%)&File$(sf%) :CURSOR 226,28:CLS 4
1281 CURSOR 310,28:PRINT#1,'Saving...':CURSOR 300,42:CLS 4:PAUSE 30
1282 IF sf%=1:addr1=FBaseA:lgth=1154:cn1=127
1283 IF sf%=2:addr1=FBase1:lgth=876 :cn1= 96:addr2=FBaseA:os=31
1284 IF sf%=3:addr1=FBase2:lgth=588 :cn1= 64:addr2=FBaseB:os= 0
1285 IF sf%=4:addr1=FBaseB:lgth=1154:cn1=127
1286 IF cn1=96 OR cn1=64
1287 FOR a=1 TO cn1
1288 FOR b=0 TO 8:POKE addr1+2+a*9+b,PEEK(addr2+2+(a+os)*9+b)
1289 END FOR a
1290 END IF
1291 SBYTES drv$(dn%)&File$(sf%),addr1,lgth
1292 END DEFine
```

```
@DIR Load Save Reset Exit
SBYTES /ip1_0LFontH_int
Select +01 Y/N (E)dit ← → ←
```

```
@DIR Load Save Reset Exit
Checking...
DEVICE ERROR...
```

```
@DIR Load Save Reset Exit
Checking...
Overwrite Y/N
```

```
@DIR Load Save Reset Exit
Saving...
```

1294 **DEFine PROCEDURE FontDIR**

```
1295 CURSOR 226,28:PRINT 'Set Drive & SuDIR'
1296 INK 7:CURSOR 300,42:PRINT 'Select % 1 2 Y/N (E)dit % 1 2%'
1297 BLOCK 12,3,454,46,7:BLOCK 2,4,480,44,7:INK 5:SelDrv 3
1298 END DEFine
```

```
@DIR Load Save Reset Exit
Set Drive & SuDIR vint_Fonts_
Select +02 Y/N (E)dit ← → ←
```

```

1300 DEFine PROCEDURE SelDrv(act%)
1301 REPeat drv_ip
1302 CURSOR 342,28:PRINT drv$(dn%):CLS 4
1303 CURSOR 351,42:PRINT FILL$(0',2-LEN(dn%))&dn%
1304 K=CODE(INKEY$(-1))
1305 SElect ON K
1306 =208:dn%=dn%-1:IF dn%<1:dn%=dm%
1307 =216:dn%=dn%+1:IF dn%>dm%:dn%=1
1308 =101,69:IF act%=3:EditName 3,342,16,drv$(dn%)
1309 =121,89,110,78:EXIT drv_ip
1310 END SElect
1311 END REPeat drv_ip
1312 END DEFine

```

```

DIR Load Save Reset Exit
fp1_
Select ↑ 88 ↓ Y/N

```

Yes or No Selects Device

```

1314 DEFine PROCEDURE SelfFont(act%,Act%)
1315 BLOCK 200,10,292,28,0:str$=Act&drv$(dn%)
1316 IF ft%<1:CURSOR 310,28:PRINT 'No Files Found...':PAUSE 30:RETURN
1317 CURSOR 226,28:PRINT FILL$( ' ',23-LEN(str%))&str$
1318 REPeat File_ip
1319 CURSOR 364,28:PRINT File$(sf%):CLS 4
1320 CURSOR 351,42:PRINT FILL$(0',2-LEN(sf%))&sf%
1321 K=CODE(INKEY$(-1))
1322 SElect ON K
1323 =208:sf%=sf%-1:IF sf%<1:sf%=ft%
1324 =216:sf%=sf%+1:IF sf%>ft%:sf%=1
1325 =101,69:IF act%=2:EditName 2,364,16,File$(sf%)
1326 =110,78:Lchk=0:EXIT File_ip
1327 =121,89:Lchk=1:EXIT File_ip
1328 END SElect
1329 END REPeat File_ip
1330 END DEFine

```

```

DIR Load Save Reset Exit
No Files Found...
Select ↑ 88 ↓ Y/N

```

No abort action

Yes Load / Save Font File

```

1332 DEFine PROCEDURE FntList(act%)
1333 BLOCK 280,10,226,28,0:CURSOR 310,28:PRINT 'Checking...'
1334 PAUSE 20:DELETE drv$(dn%)&'FList'
1335 OPEN_NEW#9,drv$(dn%)&'FList':DIR#9,drv$(dn%):CLOSE#9
1336 OPEN_IN#9, drv$(dn%)&'FList':dl%=LEN(drv$(dn%))
1337 REPeat dir_ip
1338 IF act%=1
1339 IF EOF(#9) OR sf%>fm%:sf%=sf%-1:ft%=sf%:CLOSE#9:EXIT dir_ip
1340 INPUT#9,F$:F$=F$(dl%-4 TO):fl%=LEN(F$)
1341 IF fl%<=20 AND '_fnt' INSTR F$>0:File$(sf%)=F$:sf%=sf%+1
1342 END IF
1343 IF act%=2
1344 IF EOF(#9):CLOSE#9:chk=0:EXIT dir_ip
1345 INPUT#9,F$:F$=F$(dl%-4 TO)
1346 IF F$=File$(sf%):CLOSE#9:chk=1:EXIT dir_ip
1347 END IF
1348 END REPeat dir_ip
1349 END DEFine

```

```

DIR Load Save Reset Exit
Checking...
Select ↑ 88 ↓ Y/N

```

Note: Checking \_nft' Files to LOAD

Note: Checking If SAVE File Exists

**Note:** Before Loading to FBase1 or FBase2 memory address GrpChk checks Lowest Character held in first byte of file. If less than 127 file LBYTES to address FBase1 if greater to FBase2 memory address.

**Note** The Filename Editor restricts characters to those used for filenames. Position the underline with left/Right cursors to a character then **Add** a new or **Delete** the existing character. Adding a Character expands the string to the right, Deleting shrinks the string from the right. The QL Delete uses CTRL Right Cursor for character above underline to make thing easier I have added the **Spacebar** as a **Delete** key. To Delete character to left of underline CTRL Left Backspace still applies.

```

1351 DEFine PROCEDURE EditName(act%,x,sm%,str$)
1352 IF act%=2:sl%=(' _fnt' INSTR str$)-1:str$=str$(1 TO sl%)
1353 temp$=str$:sl%=LEN(str$):cp%=1
1354 REPEAT Ed_lp
1355   Ln_Prn:Ln_Cur:k$=INKEY$(#0,-1):K=CODE(k$)
1356   SElect ON K
1357     = 10:EXIT Ed_lp
1358     = 48 TO 57, 65 TO 90,95, 97 TO 122:Add_chr
1359     =194 :IF cp%>1:cp%=cp%-1:Del_chr
1360     =202,32:Del_chr
1361     =192 :IF cp%>1:cp%=cp%-1
1362     =200 :IF cp%<sl%+1:cp%=cp%+1
1363   END SElect
1364 END REPEAT Ed_lp
1365 IF sl%=0:str$=temp$
1366 IF act%=2:str$=str$&_fnt'
1367 IF act%=3 AND str$(sl%)<>'_':IF sl%<sm%:str$=str$&'_'ELSE str$(sl%)='_'
1368 IF act%=3 AND str$(5)<>'_':str$(5)='_'
1369 END DEFine

```



Delete Character to left of cursor  
Delete Character above cursor

**Note:** Returns with original str\$  
**Note:** Checks for absence of ' \_fnt' Suffix

```

1371 DEFine PROCEDURE Ln_Prn
1372 IF LEN(str$)>sm%:str$=str$(1 TO sm%):cp%=sm%
1373 INK 5:CUSOR x,28:PRINT str$:CLS 4
1374 END DEFine

```

**Note:** Prints Str\$ clears to end of Cursor Line

```

1376 DEFine PROCEDURE Ln_Cur
1377 BLOCK 6,1,x+cp%*6-6,37,2
1378 END DEFine

```

**Note:** Character Highlight Cursor Bar

```

1380 DEFine PROCEDURE Add_chr
1381 IF cp% =1 AND sl%=0 :str$=str$&k$
1382 IF cp%>=1 AND cp%<sl%:str$=str$(1 TO cp%-1)&k$&str$(cp% TO sl%)
1383 IF cp%>=1 AND cp%=sl%:str$=str$(1 TO cp%-1)&k$&str$(cp%)
1384 IF cp%> 1 AND cp%>sl%:str$=str$&k$
1385 IF cp%=sm%:str$(cp%)=k$
1386 IF sl%<sm%:sl%=sl%+1:ELSE sl%=sm%
1387 IF cp%<sm%:cp%=cp%+1:ELSE cp%=sm%
1388 END DEFine

```

**Note:** Checks for Adding Character to String

```

1390 DEFine PROCEDURE Del_chr
1391 IF cp%=sl%:str$=str$(1 TO sl%-1):sl%=sl%-1
1392 IF cp%>=1 AND cp%<sl%:str$=str$(1 TO cp%-1)&str$(cp%+1 TO sl%):sl%=sl%-1
1393 IF cp%=sm%:str$=str$(1 TO sm%-1):cp%=cp%-1:sl%=sm%-1
1394 IF cp%=1 AND sl%=1:str$="" :sl%=0
1395 END DEFine

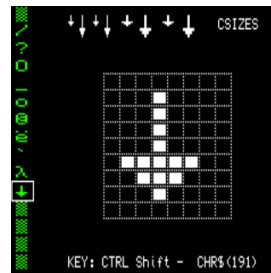
```

**Note:** Checks for Deleting Character from String

#### 1400 DEFine PROCEDURE KeyPad

Note: Identify Key(s) for Printable Characters

```
1401 SElect ON cn=32 TO 93 :Key$=CHR$(cn)
1402 SElect ON cn=32 :Key$='Spacebar'
1403 SElect ON cn=33,34 :Key$='Shift '&(cn-32)
1404 SElect ON cn=36,37 :Key$='Shift '&(cn-32)
1405 SElect ON cn=38 :Key$='Shift 7'
1406 SElect ON cn=40 :Key$='Shift 9'
1407 SElect ON cn=41 :Key$='Shift 0'
1408 SElect ON cn=42 :Key$='Shift 8'
1409 SElect ON cn=43 :Key$='Shift ='
1410 SElect ON cn=58 :Key$='Shift ;'
1411 SElect ON cn=60 :Key$='Shift ,'
1412 SElect ON cn=62 :Key$='Shift .'
1413 SElect ON cn=63 :Key$='Shift /'
1414 SElect ON cn=64 :Key$='Shift '"
1415 SElect ON cn=65 TO 90 :Key$='Shift '&CHR$(cn)
1416 SElect ON cn=94 :Key$='Shift 6'
1417 SElect ON cn=95 :Key$='Shift -'
1418 SElect ON cn=96 :Key$='Shift 3'
1419 SElect ON cn=97 TO 122 :Key$=CHR$(cn-32)
1420 SElect ON cn=123 :Key$='Shift ['
1421 SElect ON cn=124 :Key$='Shift \'
1422 SElect ON cn=125 :Key$='Shift ]'
1423 SElect ON cn=126 :Key$='Shift #'
1424 SElect ON cn=127 :Key$='Shift Esc'
1425 SElect ON cn=128 :Key$='CTRL Esc'
1426 SElect ON cn=129 :Key$='CTRL Shift 1'
1427 SElect ON cn=130 :Key$='CTRL Shift '"
1428 SElect ON cn=131 TO 133 :Key$='CTRL Shift '&CHR$(cn-80)
1429 SElect ON cn=134 :Key$='CTRL Shift 7'
1430 SElect ON cn=135 :Key$='CTRL '"
1431 SElect ON cn=136 :Key$='CTRL Shift 9'
1432 SElect ON cn=137 :Key$='CTRL Shift 0'
1433 SElect ON cn=138 :Key$='CTRL Shift 8'
1434 SElect ON cn=139 :Key$='CTRL Shift ='
1435 SElect ON cn=140 TO 153 :Key$='CTRL '&CHR$(cn-96)
1436 SElect ON cn=154 :Key$='CTRL Shift ;'
1437 SElect ON cn=155 :Key$='CTRL ;'
1438 SElect ON cn=156 :Key$='CTRL Shift ,'
1439 SElect ON cn=157 :Key$='CTRL ='
1440 SElect ON cn=158 :Key$='CTRL Shift .'
1441 SElect ON cn=159 :Key$='CTRL Shift /'
1442 SElect ON cn=160 :Key$='CTRL Shift 2'
1443 SElect ON cn=161 TO 186 :Key$='CTRL Shift '&CHR$(cn-96)
1444 SElect ON cn=187 :Key$='CTRL ['
1445 SElect ON cn=188 :Key$='CTRL \'
1446 SElect ON cn=189 :Key$='CTRL ]'
1447 SElect ON cn=190 :Key$='CTRL Shift 6'
1448 SElect ON cn=191 :Key$='CTRL Shift -'
1449 SElect ON cn=0 TO 31,192 TO 255 :Key$='
1450 CURSOR 320,208:PRINT 'KEY: '&Key$;FILL$(' ',14-LEN(Key$));CHR$(';cn: ')
1451 END DEFine
```





## 1453 REMark Demo: RETRO ARCADE GAMES - QBITS 2022

### 1455 DEFine PROCEDURE ARCADE

1456 INK#0,6:CURSOR#0,212,4::PRINT#0,"ARCADE GAMES":INK#0,5:CURSOR#0,54,16

1457 PRINT#0,(L)oad Font File "\_fnt" for Aliens : Dino : Giro then Press (G)ame'

1458 END DEFine

### 1460 DEFine PROCEDURE FontGame

1461 IF cn3\$=="Aliens\_fnt":CLS#3:Space\_Invader

1462 IF cn3\$=="Dino\_fnt":CLS#3:Dino\_Run

1463 IF cn3\$=="Giro\_fnt":CLS#3:Giro\_Rescue

1464 END DEFine

## 1500 REMark ARCADE - Space Invaders simplified version for QL by QBITS 2022

### 1502 DEFine PROCEDURE Space\_Invader

1503 pd=2 :REMark Game Speed pd=pause delay set (1 to 5)

1504 CSIZE#3,1,0:INK#3,6:b=0:pd=2

1505 FOR a=128 TO 142:b=b+1:CURSOR#3,b\*8,8:PRINT#3,CHR\$(a) **Note: Font Grp 3 Codes128 to 191**

1506 CURSOR 127,200:PRINT"% ½":BLOCK 12,3,137,204,7

1507 InitLevel:DrawAliens 1:!=5:INK#3,7

1508 REPEAT Invader\_lp

1509 IF !%>4 OR at%>0

1510 CSIZE#3,2,0:CURSOR#3,54,80:PRINT#3,'New Game Y/N'

1511 REPEAT ans

1512 IF KEYROW(5)=64:z%=0:!=1:at%=27:tnum=0:num=0:CLS#3,3:EXIT ans

1513 IF KEYROW(7)=64:EXIT Invader\_lp

1514 END REPEAT ans

1515 END IF

1516 DrawAliens !%:INK#3,7:CURSOR#3,74,80:PRINT#3,'LEVEL ':!%

1517 FOR z=6 TO 0 STEP -1:CURSOR#3,124,96:PRINT#3,z:PAUSE 25

1518 BLOCK#3,120,30,72,80,i=8:n=5:s!%=0:ay=3:Invaders

1519 END REPEAT Invader\_lp

1520 CLS#3:FontSets

1521 END DEFine



### 1523 DEFine PROCEDURE InitLevel

**Note: 4 Levels of Play**

1524 DIM GL%(4,3),GA%(3,9):RESTORE 1526

1525 FOR a=1 TO 4:READ GL%(a,1),GL%(a,2),GL%(a,3)

1526 DATA 143,144,145,146,147,148,149,150,151,152,153,154

1527 END DEFine

### 1529 DEFine PROCEDURE DrawAliens(!%)

**Note: Display Aliens by Level**

1530 BLOCK#3,260,10,6,150,0:at%=27:GScore 0

1531 FOR a=1 TO 3:FOR b=1 TO 9:GA%(a,b)=GL%(!%,a):END FOR b:END FOR a

1532 INK#3,5:FOR i=1 TO 9:CURSOR#3,i\*24,28:PRINT#3,CHR\$(GA%(1,i))

1533 INK#3,4:FOR i=1 TO 9:CURSOR#3,i\*24,40:PRINT#3,CHR\$(GA%(2,i))

1534 INK#3,3:FOR i=1 TO 9:CURSOR#3,i\*24,52:PRINT#3,CHR\$(GA%(3,i))

1535 END DEFine



### 1537 DEFine PROCEDURE S Lives

1538 CSIZE#3,1,0:INK#3,7

1539 CURSOR#3,148,8:PRINT#3,FILL\$(CHR\$(159),4-s!%)&'

1540 CSIZE#3,3,0:s!%=s!%+1

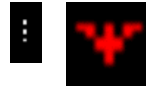
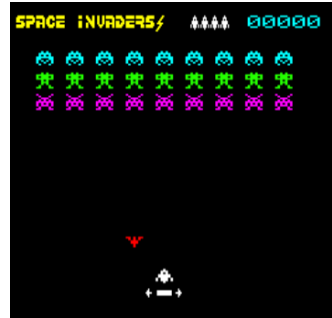
1541 END DEFine



```

1543 DEFine PROCEDURE Invaders
1544 sl%=0:n=5:at%=27:SLives
1545 REPEAT lp
1546 i=i+1:IF i>7:i=1:an=n+RND(-1 TO 1)
1547 DShip 7,n:x=an*24:y=60+i*12
1548 IF at%=0:lf%=l%+1:RETurn
1549 IF at%>0 AND sl%>4:l%=5:RETurn
1550 IF KEYROW(1)=64:FShip:ay=3:GScore l%*50
1551 IF KEYROW(1)= 2:DShip 0,n:n=n-1:IF n<1:n=1
1552 IF KEYROW(1)=16:DShip 0,n:n=n+1:IF n>9:n=9
1553 INK#3,2:CUSOR#3,x,y:PRINT#3,CHR$(158):PAUSE pd
1554 INK#3,2:CUSOR#3,x,y:PRINT#3,' ':PAUSE pd
1555 IF an=n AND KEYROW(1)=64
1556 BEEP 2000,1,255,200,4,2,0,0,0
1557 INK#3,7:CUSOR#3,x,140:PRINT#3,CHR$(160):PAUSE pd
1558 CLS#3,3:FExplode x,y:GScore 200:i=8
1559 END IF
1560 IF an=n AND i=7
1561 BEEP 3000,20,80,80,-8,15,15,15,i=8:SLives
1562 CURSOR#3,x,150:PRINT#3,CHR$(162):PAUSE pd*5
1563 END IF
1564 END REPEAT lp
1565 END DEFine

```



Note: Falling Bombs Chr\$(158)

```

1567 DEFine PROCEDURE GScore(num)
1568 CURSOR#3,192,8:CSIZE#3,2,0:INK#3,5:tnum=tnum+num
1569 PRINT#3,FILL$( '0',5-LEN(tnum))&tnum:CSIZE#3,3,0
1570 END DEFine

```



```

1572 DEFine PROCEDURE DShip(col,n)
1573 INK#3,col:CUSOR#3,n*24,150:PRINT#3,CHR$(159)
1574 END DEFine

```



```

1576 DEFine PROCEDURE FShip
1577 BEEP 2000,1,255,200,4,2,0,0,0

```

```

1578 FOR i=1 TO 5
1579 OVER#3,1:CUSOR#3,n*24,150-i*18:PRINT#3,CHR$(160):PAUSE pd
1580 OVER#3,0:CUSOR#3,n*24,150-i*18:PRINT#3,' '
1581 END FOR i

```

Note: Fire at Enemy



```

1582 IF ay=3 AND GA%(ay,n)=32:ay=2
1583 IF ay=2 AND GA%(ay,n)=32:ay=1
1584 IF ay=1 AND GA%(ay,n)=32:ay=3:RETurn
1585 FExplode n*24,16+ay*12:GA%(ay,n)=32:at%=at%-1
1586 END DEFine

```

Note: Check Array Entry

Note: Explode Alien and Delete



```

1588 DEFine PROCEDURE FExplode(x,y)
1589 FOR i=1 TO 6
1590 BEEP 3000,20,60,80,-8,15,15,15
1591 INK#3,2:CUSOR#3,x,y:PRINT#3,CHR$(160):PAUSE pd
1592 INK#3,6:CUSOR#3,x,y:PRINT#3,CHR$(161):PAUSE pd
1593 END FOR i
1594 CURSOR#3,x,y:PRINT#3,' '
1595 END DEFine

```





# 1602 **DEFine PROCedure Dino\_Run**

```

1603 dino_colour%      = 7
1604 cactus_colour%    = 4
1605 ground_colour%    = 2
1606 cloud_colour%     = 255
1607 background_col    = 0
1608 slowdown_steps    = 50000 :REMark 1/n Game Delay [?Processor Speed]
1609 pany%              = 90 :REMark Top Left of Main PAN Block
1610 byPerCent          = 1 :REMark Percentage change of Slowdown delays
1611 every%             = 1000 :REMark every this much of score
1612 sound_on%          = 1 :REMark 0=sound off 1=sound on
1613 demo_mode%         = 0 :REMark auto-jumping
1614 hiscore=score      = 0
1615 :
1616 DIM backg$(2,34) :REMark Cactus Height 0-2 Locations 0-34
1617 :
1618 REPEAT Dino_program
1619 PAPER#3;background_col%:CLS#3:REMark Title & Score bar
1620 OVER#3,1:STRIP#3,80:INK#3,7:CSIZE#3,1,1:BLOCK#3,276,22,0,0,80
1621 FOR i=0 TO 1:CURSOR#3,2+i,2:PRINT#3,' DINO Run'
1622 CURSOR#3,120,2:PRINT#3,'HI '&FILL$('0',5-LEN(hiscore))&hiscore;
1623 OVER#3,0:STRIP#3,0
1624 :
1625 REMark Set Ground level at pany%+50 [ie. (3*18)-4]
1626 RANDOMISE:CSIZE#3,1,0:INK#3,ground_colour%
1627 CURSOR#3,0,pany%+50:FOR a=1 TO 34:PRINT#3,CHR$(RAND(168 TO 178));
1628 :
1629 REMark Insert a couple of Random Cactii
1630 REMark 0=cactus 3 (top) 1=cactus 2 (middle) 2=cactus 1 (bottom)
1631 CHAR_INC#3,8,18:FOR y=0 TO 2:backg$(y)=FILL$(' ',34)
1632 backg$(2,RND(18 TO 22))=CHR$(132):backg$(2,RND(32 TO 34))=CHR$(132)
1633 OVER#3,1:CSIZE#3,1,1:INK#3,cactus_colour%
1634 FOR a=0 TO 2:CURSOR#3,0,pany%+(18*a):PRINT#3,backg$(a);
1635 OVER#3,0:CSIZE#3,1,0
1636 :

```

```

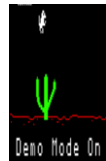
1637 REMark Place Random Cloud
1638 STRIP#3,background_col:INK#3,ground_colour%:cloudy%=48
1639 m=RND(1 TO 5):REMark avoids Turbo error
1640 SElect ON rn
1641   =1 : cloud_colour% = 7
1642   =2 : cloud_colour% = 56
1643   =3 : cloud_colour% = 63
1644   =4 : cloud_colour% = 248
1645   =5 : cloud_colour% = 255
1646 END SElect
1647 CSIZE#3,1,0:INK#3,cloud_colour%:CURSOR#3,8*RND(10 TO 30),cloudy%
1648 IF RND(1 TO 2)=1
1649   PRINT#3,CHR$(181)&CHR$(182);
1650 ELSE
1651   PRINT#3,CHR$(183)&CHR$(184);
1652 END IF
1653 CSIZE#3,1,1:INK#3,7
1654 cloud_gap%=RND(10 TO 20):cloud_wide=0:cloud_run%=0
1655 slowdown=slowdown_steps :REMark disable [Set to 0 for BBQL]
1656 IF demo_mode%=1 :REMark Demo Mode ON (auto Jumping)
1657   CSIZE#3,1,1:INK#3,7:CURSOR#3,6,158:PRINT#3,'Demo Mode On':CLS#3,4
1658 END IF
1659 :
1660 cactus_gap% =RND(10 TO 20) :REMark columns blank before a cactus
1661 cactus_wide =0 :REMark 1/2/3
1662 cactus_high =0 :REMark 1/2/3
1663 cactus_run% =0 :REMark Cactus width (1/2/3) drawn
1664 dinobasey% =pany%+36 :REMark Dinosaur 'ground level'
1665 dinoy% =dinobasey% :REMark Dino moves up from here
1666 score =0 :REMark Score is one point per step
1667 cloudy% =RND(36 TO 48) :REMark Location above cactus
1668 counter% =0 :REMark Dino Animation Frame Counter
1669 ticker =0 :REMark Speed Control Delay
1670 jumping% =0 :REMark <>0 Dino jumping (+ve=up -ve=down)
1671 demo_mode% =0 :REMark Demo mode OFF=0 ON=1
1672 :
1673 CURSOR#3,6,158:PRINT#3,'(D)emo Mode (P)ause (E)xit Game'
1674 :
1675 REpeat Dino_GAME
1676 INK#3,dino_colour%:OVER#3,-1 :REMark Show Dino
1677 CURSOR#3,32,dinoy%:PRINT#3,CHR$(128+(counter% MOD 4));
1678 REMark Speed control - ignore (slowdown-1) passes of the loop
1679 IF slowdown>0 :REMark slowdow=0 to disable
1680   REpeat wait
1681     ticker=ticker+1
1682     IF ticker<slowdown:NEXT wait
1683     ticker=0:EXIT wait
1684   END REpeat wait
1685 END IF#

```

```

1686 key=CODE(INKEY$)
1687 SElect ON key
1688 =69,101 :REMark (E)xit
1689 OVER#3,0 : CURSOR#3,6,158 : CSIZE#3,1,1 : EXIT Dino_GAME
1690 =10,32 :REMark Jump
1691 IF jumping%=0 AND demo_mode%=0
1692     jumping%= -1:dinoy%=dinobasey%:dy%=18
1693     IF sound_on%:BEEP 100,10
1694 END IF
1695 =80,112 :REMark (P)ause
1696 OVER#3,0:CURSOR#3,6,158:CLS#3,3:PRINT#3,'PAUSED':CLS#3,4
1697 k$=INKEY$(-1):CURSOR#3,6,158:CLS#3,3
1698 IF k$=CHR$(27):EXIT GAME
1699 IF demo_mode%=1:PRINT#3,'Demo Mode On':CLS#3,4:ELSE CLS#3,3
1700 OVER#3,-1:CURDIS
1701 =68,100 :REMark (D)emo_mode (auto-jumping)
1702 demo_mode%=NOT demo_mode%
1703 OVER#3,0:CURSOR#3,6,158:CSIZE#3,1,1:INK#3,7
1704 IF demo_mode%=1:PRINT#3,'Demo Mode On':CLS#3,4:ELSE CLS#3,3
1705 END SElect
1706 INK#3,dino_colour% :REMark - Erase Dino before PAN'ing -
1707 OVER#3,-1:CURSOR#3,32,dinoy% :PRINT#3,CHR$(128+(counter% MOD 4));:OVER#3,0
1708 y%=(dinobasey%-dinoy%) DIV 18 :REMark - Detect Dino Striking Anything -
1709 IF y%<3
1710     IF backg$(2-y%,5)<>' ' :REMark Dino has Struck Something
1711     IF sound_on% : BEEP 5000,0,50,50,1,0,0
1712     m = RND(1 TO 3):CURSOR#3,6,158
1713     SElect ON m
1714     =1 : PRINT#3,'Ouch!';
1715     =2 : PRINT#3,'Oops!';
1716     =3 : PRINT#3,'Oh dear!';
1717     END SElect
1718     OVER#3,-1:CLS#3,4:INK#3,dino_colour%
1719     CURSOR#3,32,dinoy%:PRINT#3,CHR$(128+(counter% MOD 4));
1720     FOR a=1 TO 10:BLOCK#3,10,18,32,dinoy%,7:PAUSE 5
1721     OVER#3,0:PAUSE 30:CURSOR#3,24,158:EXIT Dino_GAME
1722 END IF
1723 END IF
1724 IF jumping%<>0 :REMark - move Dino to next position -
1725 IF jumping%=-1 :REMark downward
1726     dinoy%=dinobasey%-72*(SIN(RAD(dy%)))
1727     dy%=dy%+18:IF dy%=90:jumping%=1 :REMark change direction
1728 ELSE
1729     dinoy%=dinobasey%-72*(SIN(RAD(dy%))) :REMark upward
1730     dy%=dy%-18:IF dy%=0:jumping%=0:dinoy%=dinobasey%
1731 END IF
1732 ELSE
1733     IF demo_mode%=1 :REMark Demo-Mode Check for Cactus ahead
1734     IF backg$(2,10)<>' ' :REMark Start to Auto-Jump
1735     jumping%=-1:dinoy%=dinobasey%:dy%=18
1736     IF sound_on% : BEEP 100,10
1737     END IF
1738 END IF
1739 END IF

```



```

1740 REMark - PAN Cactus display leftward - height 4x18 pixels -4(overlap) -
1741 INK#3,cactus_colour%:CURSOR#3,0,pany%
1742 CHAR_INC#3,8,68:PAN#3,-8,3:CHAR_INC#3,8,18
1743 FOR y=2 TO 0 STEP -1:backg$(y)=backg$(y,2 TO 34)&' '
1744 REMark add any extra objects to right of background array
1745 REMark decrement the count until next cactus(es)
1746 IF cactus_gap%>0
1747     cactus_gap%=cactus_gap%-1
1748     IF cactus_gap%=0
1749         cactus_run%=1 :REMark how far are we across a cactus?
1750         cactus_wide=RND(1 TO 3) :REMark cactus width 1-3
1751         cactus_high=RND(1 TO 3) :REMark cactus height 1-3
1752     END IF
1753 END IF
1754 IF cactus_run%>0 :REMark Add a cactus to the right
1755     SElect ON cactus_high
1756     =1:REMark cactus 1 high
1757         base_char =131+(cactus_wide=2)+(3*(cactus_wide=3))
1758         backg$(2,34)=CHR$(base_char+cactus_run%)
1759     =2:REMark cactus 2 medium
1760         base_char =137+(cactus_wide=2)+(3*(cactus_wide=3))
1761         backg$(1,34)= CHR$(base_char+cactus_run%)
1762         backg$(2,34)= CHR$(base_char+6+cactus_run%)
1763     =3:REMark cactus 3 low
1764         base_char =149+(cactus_wide=2)+(3*(cactus_wide=3))
1765         backg$(0,34)=CHR$(base_char+cactus_run%)
1766         backg$(1,34)=CHR$(base_char+6+cactus_run%)
1767         backg$(2,34)=CHR$(base_char+12+cactus_run%)
1768     END SElect
1769     cactus_run%=cactus_run% + 1
1770     IF cactus_run%>cactus_wide
1771         cactus_gap%=RND(10 TO 20) :REMark start a new gap between cacti
1772         cactus_run%=0:cactus_high=0:cactus_wide=0 :REMark RESet
1773     END IF
1774 END IF
1775 REMark - Move ground : Update Cacti / display ground -
1776 CSIZE#3,1,0:INK#3,ground_colour%:OVER#3,1
1777 CURSOR#3,260,pany%+50:PRINT#3,CHR$(RND(168 TO 178));
1778 CSIZE#3,1,1:INK#3,cactus_colour%
1779 REMark - Then show Cactus part of array -
1780 FOR y=2 TO 0 STEP -1
1781     CURSOR#3,260,pany%+(18*y):PRINT#3,backg$(y,34);
1782 END FOR y
1783 OVER#3,0
1784 REMark - Handle the score IF >99999 wrap around
1785 STRIP#3,80:score=score+1:IF score>99999:score=0
1786 INK#3,7:CURSOR#3,226,2:PRINT#3,FILL$( '0',5-LEN(score))&score
1787 STRIP#3,0
1788 IF every%>0
1789     REMark Speed up slightly as score gets higher (if set to do so)
1790     IF (score MOD every%)=0
1791         slowdown=slowdown-(slowdown*byPerCent/100)
1792     END IF
1793 END IF

```

```

1794 REMark - Increment dino frame counter -
1795 counter%=counter%+1:IF counter% >4:counter%=0
1796 REMark - Handle Clouds once every 4 frames -
1797 IF (counter% MOD 4)=0
1798     CURSOR#3,0,cloudy%:CHAR_INC#3,8,36:PAN#3,-8,3:CHAR_INC#3,8,18
1799     cloud_gap%=cloud_gap%-1
1800     IF cloud_gap%=0 :REMark Time for new Cloud
1801         cloud_run% =1 :REMark how far across
1802         cloud_wide =RND(2 TO 3) :REMark cloud width 1-3
1803         cloud_high =RND(0 TO 1) :REMark cloud height 0-1
1804         cloudyy% =cloudy% + RND(0 TO 26-(9*cloud_high))
1805         rn=RND(1 TO 5) :REMark Cloud - Random Colour/Stipple
1806         SElect ON rn
1807             =1:cloud_col%=7
1808             =2:cloud_col%=56
1809             =3:cloud_col%=63
1810             =4:cloud_col%=248
1811             =5:cloud_col%=255
1812         END SElect
1813         SElect ON cloud_wide
1814             =1:IF RND(1 TO 10)=10 :REMark 191 is Sun
1815                 cloud_base_char=191:cloud_high=0:cloud_col%=6
1816             ELSE
1817                 cloud_base_char=179+(RND>.5):REMark small cloud
1818             END IF
1819             =2:cloud_base_char=180+(2*(RND>.5))
1820             =3:cloud_base_char=184+(3*(RND>.5))
1821         END SElect
1822     END IF
1823     IF cloud_run%>0 :REMark add a cloud/sun to the right
1824     CSIZE#3,1,cloudy_high:INK#3,cloud_col%:CURSOR#3,32*8,cloudyy%
1825     PRINT#3,CHR$(cloud_base_char+cloud_run%);:INK#3,7
1826     cloud_run%=cloud_run%+1
1827     IF cloud_run%>cloud_wide
1828         cloud_gap%=RND(10 TO 20):cloud_run%=0:cloud_wide=0
1829     END IF
1830 END IF
1831 END IF
1832 END REPEAT Dino_GAME
1833 CURSOR#3,6,158:PRINT#3,'GAME OVER < Another Game Y/N? >'
1834 REPEAT Ans_ip
1835     IF KEYROW(7)=64:EXIT Dino_program
1836     IF KEYROW(5)=64:EXIT Ans_ip
1837 END REPEAT Ans_ip
1838 IF score>hiscore:hiscore=score
1839 END REPEAT Dino_program
1840 CSIZE#3,0,0:CLS#3:FontSets
1841 END DEFINE

```





1850 REMark **ARCADE - Giro\_Rescue** based on version by Henry Wrighton Aug 1889

```

1852 DEFine PROCEDURE Giro_Rescue
1853 Init_Giro
1854 REPeat Giro
1855   set_level           :REMark New Game Setup
1856   REPeat Rescue
1857     tcap=tcap+1:cap=tcap:fu=90:sh=90:Giroscore sc
1858     INK#3,4:set_junk:INK#3,7:set_pod:main
1859     IF fin=1 OR tcap>15:EXIT Rescue :ELSE fin=0
1860     REPeat Bonus
1861       fuel 2:sc=sc+(2*(tcap-9)):Giroscore sc:beeps 1
1862       PAUSE 5:IF fin:EXIT Bonus
1863     END REPeat Bonus
1864   END REPeat Rescue
1865   OVER#3,0:fin=0:i=0:CUSOR#3,32,80:PRINT#3,'Another Game y/n'
1866   REPeat Key_lp
1867     i=(i+1) MOD 7:INK#3,i:AT#3,4,6:PAUSE 5:PRINT#3,'GAME OVER'
1868     IF KEYROW(5)=64 THEN EXIT Key_lp
1869     IF KEYROW(7)=64 THEN EXIT Giro
1870   END REPeat Key_lp
1871 END REPeat Giro
1872 CLS#3:CSIZE#2,0,0:FontSets
1873 END DEFine

```

Note: Load Giro-fnt Press (G)ame to activate



```

1875 DEFine PROCEDURE Giroscore(sc)
1876 CSIZE#2,1,1:CUSOR#2,200,42:PRINT#2,FILL$(0',5-LEN(sc))&sc Note: sc score
1877 CUSOR#2,282,42:PRINT#2,tcap-9:CSIZE#2,0,0 Note: tcap levels
1878 END DEFine

```

```

1880 DEFine PROCEDURE set_level
1881 DIM al(36,19):BLOCK#3,276,140,0,22,0:INK#3,7
1882 sh=90:sc=0:tcap=9:pod$='OEZ':pm=1:junk$='*^' Note: pod$ CHR$(139 to 143)
1883 FOR i=1 TO 80:INK RND(1 TO 5):POINT#3,RND(5 TO 110),RND(12 TO 84) pm <1 - 5>
1884 INK#3,4:FOR K=1 TO 12:set_junk
1885 END DEFine

```



Note: junk\$ CHR\$(144 to 148)

```

1887 DEFine PROCEDURE set_pod
1888 FOR i=1 TO tcap
1889   REPeat pp
1890     ay=RND(3 TO 14):ax=RND(2 TO 20)
1891     IF ax<>11 AND ay<>8 AND al(ax,ay)=0:EXIT pp
1892   END REPeat pp
1893   al(ax,ay)=1:AT#3,ay,ax:PRINT#3,"":beeps 5
1894 END FOR i
1895 END DEFine

```

Note: Life pod CHR\$(149)



```

1897 DEFine PROCEDURE set_junk
1898 REPeat junk
1899   x=RND(2 TO 20):y=RND(3 TO 14):IF x<>10 AND y<>8:EXIT junk
1900 END REPeat junk
1901 al(x,y)=2:AT#3,y,x:PRINT#3,junk$(RND(1 TO 5)):beeps 6
1902 END DEFine

```

```

1904 DEFine PROCEDURE main
1905 yy=0:xx=0:x=120:y=80:ix=2:iy=1:fu=90:fin=0:pm=1
1906 OVER#3,0:BLOCK#3,90,4,52,170,5:BLOCK#3,90,4,180,170,5
1907 INK#3,7:CUSOR#3,x,y:PRINT#3,pod$(pm):OVER#3,-1
1908 REPEAT loop
1909   get_keys:PAUSE 5
1910   IF x+xx< 0 OR x+xx>260:xx=-xx :beeps 2
1911   IF y+yy<20 OR y+yy>150:yy=-yy :beeps 2
1912   IF al((x)/12,(y)/10)=1:pod_clear :beeps 3
1913   IF al((x)/12,(y)/10)=2:shield 1 :beeps 4
1914   CUSOR#3,x,y:PRINT#3,pod$(pm):IF fin :PAUSE 30:EXIT loop
1915   x=x+xx:y=y+yy:pm=(pm MOD 5)+1:CUSOR#3,x,y:PRINT#3,pod$(pm)
1916 END REPEAT loop
1917 END DEFine

```

```

1919 DEFine PROCEDURE get_keys
1920 K=KEYROW(1):tx=xx:ty=yy
1921 SElect ON K
1922   =128:yy=yy+iy
1923   =144:yy=yy+iy:xx=xx+ix
1924   =16 :xx=xx+ix
1925   =20 :yy=yy-iy:xx=xx+ix
1926   =4 :yy=yy-iy
1927   =6 :yy=yy-iy:xx=xx-ix
1928   =2 :xx=xx-ix
1929   =130:yy=yy+iy:xx=xx-ix
1930 END SElect
1931 SElect ON K=128,144,16,20,4,6,2,130:fuel 1
1932 IF xx>11 OR xx<-11 THEN xx=tx
1933 IF yy> 8 OR yy< -8 THEN yy=ty
1934 END DEFine

```



```

1936 DEFine PROCEDURE pod_clear
1937 al(x/12,y/10)=0:AT#3,y/10,x/12:PRINT#3,'':sc=sc+10
1938 Giroscore sc,l:cap=cap-1:IF cap=0:fin=2
1939 END DEFine

```

Note CHR\$(149)



```

1941 DEFine PROCEDURE fuel(f)
1942 fu=fu-f:BLOCK#2,f,4,196+fu,211,0:IF fu<=0:fin=1
1943 END DEFine

```



```

1945 DEFine PROCEDURE shield(s)
1946 sh=sh-s:BLOCK#2,s,4, 70+sh,211,0:IF sh<=0:fin=1
1947 END DEFine

```



```

1949 DEFine PROCEDURE beeps(b)
1950 SElect ON b
1951   =1:BEEP 100,0
1952   =2:BEEP 3000,0,200,2,3
1953   =3:BEEP 500,0
1954   =4:BEEP 1000,10,20,1,1
1955   =5:BEEP 1000,0,150,100,1
1956   =6:BEEP 1000,100
1957 END SElect
1958 END DEFine

```

1960 **DEFine PROCEDURE Init\_Giro**

1961 CLS#3:FOR i=1 TO 100:INK#3,(RND(1 TO 5)):POINT#3,RND(5 TO 110),RND(10 TO 85)

1962 **Star\_Ship 40,50**:PAUSE 20:CSIZE#3,2,1:OVER#3,1

1963 INK#3,6:FOR i=0 TO 1:CURLSOR#3, 1+i, 1:PRINT#3,'€•f,...†‡‰%\$'

**Note:** CHR\$(128 to

138)

1964 INK#3,2:FOR i=0 TO 2:CURLSOR#3,90+i,22:PRINT#3,'RED ALERT'

1965 CSIZE#3,2,0:OVER#3,0:INK#3,5

1966 **RED\_Alert 40,50**:PAUSE 20:BEEP:CSIZE#3,0,0:INK#3,5

1967 CURSOR#3,58,132:PRINT#3,'Press Any Key to begin RESCUE'

1968 CURSOR#3,36,144:PRINT#3,'Use ◀▶ keys to Guide your Giro Pod'

1969 CURSOR#3,86,154:PRINT#3,'to pick up Survivors'

1970 PAUSE:FOR i=1 TO 9:PAUSE 3:BLOCK#3,276,i\*16,0,90-i\*8,0

1971 OVER#3,1:INK#3,7:RESTORE 1975

1972 FOR a=1 TO 4

1973 READ x,y,str\$:FOR b=0 TO 1:CURLSOR#3,x+b,y:PRINT#3,str\$:END FOR b

1974 END FOR a

1975 DATA 146,9,'Score:',228,9,'Level:',2,166,'Shields:',148,166,'Fuel:'

1976 OVER#3,0:CURLSOR#3,0,0:CSIZE#3,2,0

1977 **END DEFine**

1979 **DEFine PROCEDURE Star\_Ship(sx,sy)**

1980 FILL#3,1:INK#3,7:CIRCLE#3,sx+4,sy-3,8:FILL#3,0

1981 FILL#3,1:OVER#3,1:LINE#3,sx+40,sy+8

1982 **RESTORE 1988**:FOR i=1 TO 12:**READ x,y**:LINE#3 TO sx+x,sy+y

1983 FILL#3,0:OVER#3,0:INK#3,0

1984 LINE#3,sx+12,sy TO sx+26,sy+4 TO sx+28,sy+6 TO sx+40,sy+8

1985 LINE#3,sx+44,sy+4 TO sx+32,sy+1 TO sx+30,sy+1.2

1986 CIRCLE#3,sx+4,sy-3,8:ARC#3,sx-4,sy-2 TO sx+6,sy-4,PI/2

1987 FOR i=1 TO 6:CIRCLE#3,sx+12.5+i\*2,sy-4.2+i\*.6,1

1988 DATA 30,10,20,8,18,6,0,2,10,0,26,4,28,6,40,8,44,4,40,0,12,-8,10,0

1989 **END DEFine**

**Note:** The Intro Screen to Giro Rescue



1991 **DEFine PROCEDURE RED\_Alert(sx,sy)**

1992 **RESTORE 2000**:BEEP 0,10,100,5,-5,10,10,15

1993 FOR i=1 TO 8

1994 INK#3,RND(2 TO 5):CURSOR#3,70,42:PRINT#3,'Abandon Ship'

1995 **READ x,y**:INK#3,248:LINE#3,sx+12.5+i\*2,sy-5+i\*.6 TO sx+x,sy+y

1996 FILL#3,1:INK#3,7:CIRCLE#3,sx+x,sy+y,1:FILL#3,0:PAUSE 5

1997 **READ x,y**:INK#3,248:LINE#3,sx+6+i\*2,sy+5+i\*.6 TO sx+x,sy+y

1998 FILL#3,1:INK#3,7:CIRCLE#3,sx+x,sy+y,1:FILL#3,0:PAUSE 9

1999 END FOR i

2000 DATA 4,-18,-24,10,10,-22,-18,18,18,-23,-2,15,28,-23,8,18

2001 DATA 34,-18,15,15,46,-21,20,18,55,-18,24,19,55,-5,28,15

2002 **END DEFine**

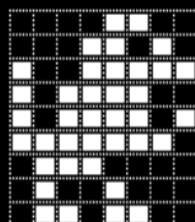
# QBITS QFont Editor2°

(D)IR (L)oad (S)ave (R)eset (E)xit

Grp Select ↑↓ ←→



6 8 12 16 CSIZES



KEY: CTRL Esc CHR\$(128)

"ARCADE GAMES"

(L)oad Font File "\_fnt" for Aliens : Dino : Giro then Press (G)ame

# QBITS QFont Editor2°

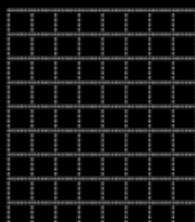
(D)IR (L)oad (S)ave (R)eset (E)xit

Grp Select ↑↓ ←→

1 DINO Run HI 00000 00004



6 8 12 16 CSIZES



(D)emo Mode (P)ause (E)xit Game

KEY: CTRL Esc CHR\$(128)

"ARCADE GAMES"

(L)oad Font File "\_fnt" for Aliens : Dino : Giro then Press (G)ame

