```
65 REM  **** HI-RES DEMO ****
70 PRINT CHR$(147): REM CLEAR SCREEN
80 POKE 53280,0 : REM COLOUR BORDER BLACK
90 :
100 REM **** COLOUR SCREEN MEMORY AREA ****
110 FOR I=1024 TO 2023: POKE I,4: NEXT I
120 :
130 REM **** SET BIT MAP POINTER ****
140 BASE =8192: POKE 53272. PEEK(53272) OR 8
150 :
160 REM **** CLEAR BIT MAP MODE ****
170 FOR I=BASE TO BASE + 7999:POKE I,0:NEXT I
180 :
190 REM **** SET BIT MAP MODE ****
200 POKE 53265, PEEK(53265) OR 32
210 :
220 REM **** DRAW STRAIGHT LINE ****
230 X1=20: X2=190: Y1=15: Y2=180
240 GOSUB 800: REM PLOT LINE
250 :
300 REM **** DRAW CIRCLE ****
310 CX=150: CY=100: RA=60
320 GOSUB 900: REM PLOT CIRCLE
330 :
370 **** ANOTHER CIRCLE ****
380 CX=100: CY=60: RA=20
390 GOSUB 900: PLOT CIRCLE
400 :
410 REM **** DRAW TRIANGLE ****
420 XA=200:XB=250:XC=300:YA=50:YB=100:YC=80
430 GOSUB 600: REM PLOT TRIANGLE
440 :
450 GOTO 450: REM END OF MAIN PROGRAM
460 :
470 :
600 REM **** PLOT TRIANGLE SUBROUTINE ****
610 :
620 X1=XA: X2=XB: Y1=YA: Y2=YB
630 GOSUB 800: REM PLOT LINE
640 X1=XB: X2=XC: Y1=YB: Y2=YC
650 GOSUB 800: REM PLOT LINE
660 X1=XC: X2=XA: Y1=YC: Y2=YA
670 GOSUB 800: REM PLOT LINE
680 RETURN
690 :
800 REM ***** PLOT LINE SUBROUTINE *****
810 S=1
820 IF X2<X1 THEN S=-1
830 FOR X=X1 TO X2 STEP S
840 Y=(Y2-Y1)*(X-X1)/(X2-X1)+Y1
850 GOSUB 1000: REM PLOT POINT
860 NEXT X
870 RETURN
880 :
900 REM **** PLOT CIRCLE SUBROUTINE ****
910 :
920 FOR ANGLE = 0 TO 2*PI STEP .04
930 X=INT (RA*COS(ANGLE)+CX)
940 Y=INT (CY-RA*SIN(ANGLE))
950 GOSUB 1000: REM PLOT POINT
960 NEXT ANGLE
970 RETURN
980 :
1000 REM **** PLOT POINT SUBROUTINE ****
1010 :
1020 IF X>319 OR X<0 OR Y>199 OR Y<0 THEN
GOTO 1070
1030 HB=INT(X/8): VB=INT(Y/8)
1040 R0=VB*320+HB*8: A= Y AND 7: B=X AND 7
1050 BY=BASE+R0+A
1060 POKE BY, PEEK(BY)OR(2^(7-B))
1070 RETURN
```

Be sure to SAVE this program before running it, as an incorrect POKE instruction can cause the machine to 'hang-up' or come to an unexpected halt. When you wish to return to the normal low resolution screen after running this program, press the Run/Stop and Restore keys simultaneously.

Subhunter Program

An important part of the subhunter game we are designing is the routine that updates the score. There are many ways of allocating scores during a game of this type; our scoring system will be based on the following rules.

1) The depth and speed of the sub are important factors. A fast-moving sub at depth is more difficult to hit than a slow sub near the surface. The score allocated to any sub will take account of this.

2) If the sub is hit, its value is added to the player's score, but if the sub reaches the edge of the screen unharmed, its value is subtracted from the player's score. No negative scores will be allowed.

Later in the project, we will deal with the routine that randomly selects the speed and depth of a sub, but all we need to know for now is that the sub's depth is stored in the variable Y3, and its speed is stored in DX. The sub's value can be calculated on this basis. To ensure that only whole number sub values are calculated, the INT function is used as follows:

Sub value = INT(Y3+DX*30)

We will store the player's current score in a variable SC. All that remains to be done is either to add or subtract the sub value from SC according to whether the sub was hit or it escaped. The UPDATE SCORE subroutine is used by two program sections:

1) Where the sub's position is tested to see if it has reached the edge of the screen and
2) during the HIT routine.

The flag DS can be set during these two parts of the program to indicate which part is using the UPDATE SCORE subroutine. Setting DS = 1 in the HIT routine and DS = −1 in the EDGE OF SCREEN routine, the score can be increased or decreased by the sub value as follows:

SC = SC + INT(Y3+DX*30)*DS

After making sure that the score has not dropped below zero, the new value of SC can be PRINTed to the top line of the screen. Add these lines to your program and SAVE it.

```
5500 REM **** UPDATE SCORE ****
5510 SC=SC+INT(Y3+DX*30)*DS
5520 IFSC<0THEN SC=0
5530 PRINT CHR$(19);CHR$(144);" SCORE ";
SC;CHR$(157);" "
5540 RETURN
```

In the next section we will look at the creation of the sprites that will be used for the ship, sub, depth charges, and explosion.