ASTEROID LANDER

Double danger and difficulty confront you in Asteroid Lander as your ship hurtles through the darkest depths of space.

Your treacherous mission is to safely reach a scientific outpost located on a large asteroid. The scientists at the station have set up an invisible force-shield to protect their base from space bandits.

The sky is cluttered with small asteroids and it is on one of these that the interplanetary terrorist is based. Can you get your ship, with little fuel left, beneath the scientist's shield before the plasma cannon's lethal laser beam blasts you and your craft to smithereens?

It's a difficult game to play and one requiring quick finger movement. Level One is hard to handle until you have mastered the controls. Level Three is the most taxing and provides you with maximum frustration. A word of advice — try to keep the lander upright as much as possible to counter gravity pulls.

The best strategy is to find a path as far as possible from the gunman and then hedge hop to the pad beneath the force field.

If you're feeling adventurous manoeuvre so that the gunman blasts a path through the asteroids for you. It's not advisable at Level One and can be dangerous at other levels.

The listing is very compact because all non-essential spaces have been omitted to fit the program into the BBC Model A Microcomputer.

So man the controls and happy landings.

HOW IT RUNS

20 — 40	Prints instructions 'land on the red pad' and 'Choose difficulty 1, 2, or 3'.
50	The VDU 26 command cancels the effect of the previous VDU 28 command to allow full use of of the screen.
60	Ensures that the difficluty level input is either 1, 2, or 3.
70 — 80	Define nine characters. The first character is used to represent the small asteroids and the other eight represent the different rotations of the lander.
90	Selects mode time for playing and clears screen.
100 — 140	Draw a frame about the playing area. There are three gauges at the top of the screen to assist the player. From left to right these are vertical velocity, horizontal velocity and fuel.
150 — 210	Draw the frames of the above by the For loop.
230	Prints the F into the the fuel gauge for identification.
240	First draws a line across the screen just below the gauges to complete the frame about the playing area, then paints in the base of the landscape.
250	Paints in the mountains.
260	Places a varying number of small asteroids on the screen depending on the difficulty chosen.
270	Draws in the landing pad.
280	Initialises most of the variables not yet defined.
290 — 370	The section which is executed continuously during the game achieving the motion on the screen.
290 — 310	Blank and update the veritcal velocity gauge.
320	Moves the lander to its new position. It has been specially designed to give a completely flicker free image.
330 — 340	Pass control to the appropriate procedure if a valid input is made.
350	Blanks out the plasma bolt, then updates its position and tests whether it has left the playing area or hit anything. If the test is positive then procedure E at line 550 is executed.
360	Plots the plasma bolt then provides for the effect of gravity on the lander and calculates the lander's new position.
370	Tests four points about the lander to see whether the lander has hit anything. It also tests whether the lander has left the playing area or whether it has been struck by a plasma bolt. If any of these conditions is true then the Repeat Until loop (lines 290 to 370) is executed.
390	Causes the lander to disintigrate if it has not landed lightly on the pad.
400 — 420	Clear the keyboard buffer then print the message "Press escape to step" for four seconds before they return control to line 10.
430 — 500	Defines the procedure T whose function is to turn the lander respecitively clockwise or anticlockwise if either the eight key or the nine key is pressed.
430 — 440	Select the character which represents the new orientation of the lander and update the variables DX and DY which direct any thrust appropriately. The rest of the procedure sets AX, AY, BX, and BY so that DX and DY will be altered correctly the next time the procedure is executed.
510 — 540	Define the procedure t which provides a burst of thrust when the nine key is pressed by altering the lander's horizontal and vertical velocities depending on DX and DY.
510	Emits a tone and leaves the procedure if there is no fuel left.
520 — 530	Deduct the correct amount of fuel and alter the landers vertical and horizontal velocity, then they update the horizontal velocity gauge.
540	Updates the fuel gauge.
560	Defines procedure which blots out anything the plasma bolt has hit, resets the bolt's position to the gun man's asteroid, and aims the bolt at the landers current position.

PROGRAM LISTING

```
10MODE7:CLEAR
20VDU28,5,24,35,0:PRINT:PRINT:PRINTCHR$(129);" Land on
   the red pad":PRINT:PRINT"Turn the lander clockwise
   with":PRINT:PRINT"the 8 key, anti-clockwise
   with":PRINT:PRINT"the 0 key."
30PRINT:PRINT"Press the 9 key for thrust
40PRINT:PRINT:PRINTCHR$(136);"Choose difficulty 1,2,or3"
50VDU26:d$=GET$
60d=ASCd$-48:IFd<10Rd>3THENGOTO50
70VUD23,224,0,24,60,60,60,24,0,0,23,225,24,60,60,60,24,
   60,90,90,23,226,0,56,120,127,124,28,18,16,23,227,0,3,
   116,255,3,0,23,228,16,18,28,124,127,120,48,0,23,229,90,90,
   60,24,60,60,60,24
80VDU23,230,8,73,56,60,254,30,28,0,23,23,1,0,192,46,225,
   46,192,0,23,232,0,28,30,254,60,56,72,8
90MODE/
100MOVE31,15
110DRAW31,1015
120DRAW1247,1015
130DRAW1247,15
140DRAW31,15
150FORI=58TO826STEP384
160MOVEI,1000
170PLOT1,280,0
180PLOT1,0,-44
190PLOT1,-280,0
200PLOT1,0,44
210NEXTI
220VDU19,2,1,0,0,0
230PRINTTAB(13,1);"F"
240MOVE31,940:DRAW1247,940:X=1200:MOVE35,20:MOVE
   1234,20:PLOT85,X,RND(100)+20:PLOT85,X+RND(50),
   20:UNTILX<135
260=RND(1000)+100:GY=RND(200:VDU5:MOVEGX,GY:
   PRINTCHR$(224):FORI=1TO30*d:MOVERND(1000)
   +140,RND(500)+200:PRINTCHR$(224):NEXTI
270MOVERND(100)+140,20:GCOL0,2:PLOT1,0,100:PLOT1
   ,20,0:PLOT1,-40,0
280F=180:VX=0:VY=0:UX=RND(3):UY=RND(3):AX=-d:
   BX=d:AY=-d:BY=-d:P=225:p=P:LX=RND(1000)+140:
   SX=GX:SY=GY:LY=900:lx=LX:ly=LY:DX=0:DY=d*2:
   m=10
290REPEATVDU4:PRINTTAB(1,1);"      ";TAB(1,1);
300IFVY>=0THENPRITN"   ";ELSEPRINT"v";
310PRINT;ABS(INT(VY)):VDU5
320GCOL0,2:MOVElx,ly:PRINTCHR$(p):GCOL0,1:MOVELX,
   LY:PRINTCHR$(P):GCOL2,1:MOVElx,ly:PRINTCHR$(p):
   p=P:lx=LX:ly=LY:A$=INKEY$(0):*FX15,0
330IFA$="8"ORA$="0"THENPROCT
340IFA$="9"THENPROCT
350GCOL0,0:PLOT69,SX,SY:SY:SX=SX+UX:SY=SY+UY:
   IFPOINT(SX,SY)<>0ORSX>1200RSX<1000RSY>9000
   RSY<200THENPROCe
360GCOL0,3:PLOT69,SX,SY:VY=VY-d/4:LX=LX+VX:LY
   +VY
370UNTILPOINT(LX+32,LY+5)=30RPOINT(LX+15,LY-32)>
   10RPOINT(LX-5,LY-16)=30RPOINT(LX+32,LY+5)=
   30RPOINT(LX+15,LY-32)>10R((LX+32-SX)  2+(LY-16
   -SY)  2)1000RLX<400RLY>940
380MOVE1x,ly
390IFPOINT(LX+32,LY-32)<>20RABS(VX)+ABS(VY)>
   5THENSOUND0,-15,4,20:FORI=1TO100:GCOL0,2:PRINT
   CHR$(224);:VDU8:GCOL0.0:PRINTCHR$(P);:VDU8:NEXTI
400*FX15,0
410VDU4:PRINT"PRESS ESCAPE TO STOP":TIME=0:REPEAT
   :UNTITIME>400:GOTO10
420END
430DEF  PROCT:IFA$="0"THENP=P+1:DX=DX+AX:DY=
   DY+AY:IFP=233THENP=225
440IFA$="8"THENP=P-1:DX=DX+BX:DY=DY+BY:IFP
   THENP=232
450IFDY>=0THENAD=-d:BX=d ELSEIFDY<0THENAX=d
   BX=-d
460IFDX=d*2THENBX=d-  ELSEIFDX=-d*2THENAX=d
470IFDX>=0THENAY=d:BY=-d
480IFDX<0THENBY=d:AY=-d
490IFDY=d*2THENAY=-d ELSEIFDY=-d*2THENBY=d
500ENDPROC
510DEF PROCt:IFF=0THENSOUND1,-5255,1:ENDPROC
520F=F-d:VX=VX+DX:VY=VY+DY:VDU4:PRINTTAB(7,1);
   "      ";TAB(7,1);:IFVX<0THENPRINT"<";ELSEPRINT">";
530PRINT;ABS(INT(VX))
540PRINTTAB(14,1);"      ";TAB(14,1)'F:VDU5ENDPROC
550DEF PROCe:MOVESX-32,SY+16:PRINTCHR$(224):SX=
   GX:SY=GY:m=m+d:sq=SQR((LX-GX)  2+(LY-GY)  2)
   /m:UX=(LX+32-GX)/sq:UY=(LY-16-GY)/sq
560IFm>50THENm=50
570SOUND0,-10,5,5:ENDPROC
```

HINTS ON CONVERSION

The program uses quite a few commands unique to the BBC Micro. VDU 5 and VDU 4 respectively join and separate the graphics and text cursors. When these cursors are joined the move command enables a character to be printed with its upper left most corner at any point on a 1280 by 1024 grid. If this is not possible on your machine then PRINT TAB(X,Y) may be used with X and Y scaled to your machines display; eg if your display is 40 by 25 then X will be INT(LX/32) and Y will be INT(LY/41).

POINT(X,Y) returns the logical colour of the pixel at (X,Y). It should be possible to simulate POINT with PEEK (the scalling mentioned above will also apply).

ASC(d$) is identical to CODE(d$).

VDU 28 defines a text window.

VDU 23 redefines the ASCII character whose code is the number following the 23. The new character is an 8 by 8 grid whose rth row is a representation in binary of the rth number following the code. Thus the syntax is VDU 23, code, row 1, row 2, row 3, up to row 8.

PLOT 69,X,Y prints a point at location X,Y.

PLOT 1,x,y draws a line between locations (X,Y) and (X+x, Y+y) where (X,Y) is the present position of the graphics cursor.

PLOT 85,x,y fills a triangle with vertices (x,y) and the last two places visited by the graphics cursor.

*FX 15,0 clears the keyboard and sound buffers.

SOUND c, v, f, d causes a sound of duration d to be emitted by channel c (white noise if c=0 and f=4) while f and v determine frequency and volume respectively.

PROC and ENDPROC can be replaced by a GOSUB to the first line of the procedure, and a RETURN respectively.

REPEAT and UNTIL can be replaced by a single GOTO. Here is an example

```
10 REPEAT
20 ------
30 UNTIL condition is true
```

This can be replaced by (30 IF NOT(condition is true) THEN GOTO 10) TRUE and FALSE always return -1 and 0 respectively.

MODE is used to switch between the various graphics modes of the BBC Micro. In mode 5 there are 4 logical colours which can be thought of as paint pots numbered from 0 to 3. Unless VDU 19 is used to change the colour of paint in a pot then these colours are black, red, yellow and white. GCOL 0,p selects the colour to be used from pot p. GCOL 2,p selects the colour from pot P where P=p added with the colour already on the screen. If you do not have an approximation of the GCOL command then you will have to abandon the special function of line 320. Delete line 220 and replace line 320 with:-

```
320 PRINT
    TAB(lx,ly);"  ":PRINT
    TAB(LX,LY):CHR$(P):
    p=P:lx=LX:ly=LY:A$=
    INKEY$(0):*FX 15,0
```

LX , LY , lx , and ly should of course be scaled appropriately.

SOFTWARE

Variables Used

D$ = The level of difficulty required by the player.

D, is a numerical variable to which the level of difficulty is passed; it is used to modify, gravity, thrust, number of small asteroids, and the rate of increase of the speed of the plasma bolts.

I = For loop variable used in three loops in the program. It is defined at lines 150, 260, 390.

X = Draw the mountains at the bottom of the playing area.

GX and GY = the horizontal and vertical positions of the gunmans asteroid.

F, defined at line 280, = the fuel remaining.

VX and VY = The horizontal and vertical velocities of the lander respectively.

AX BX AY and BY = The numbers to be added to DX and DY when the lander is turned. The X and Y in the variables name determines which A or B is added to which D, and A is for an anticlockwise turn, whilst B is for a clockwise turn.

DX and DY together fix the direction of any burst of thrust by their addition to VX and VY respectively.

P = The character code of the current image of the lander.

P, defined in line 280 = The previous value of P.

LX and LY respectively are the horizontal and vertical positions of the lander. They are also defined at line 280.

lx and ly = The previous values of LX and LY.

SX and SY = The horizontal and vertical positions of the plasma bolt.

UX an UY = The horizontal and vertical velocities of the plasma bolt.

m = Modify the speed of the plasma bolt.

A$ = Input a character from the keyboard and if it is an 8 or a 0 then the lander turns whilst if it is a 9 then a burst of thrust is imparted to the lander.

sq = Aim the plasma bolt at the lander and ensure that its speed is correct.