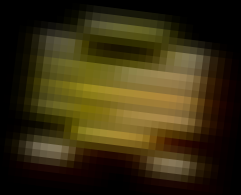


TenLander

A C=64 game written in 2020 in just ten lines of code and in less than 800 characters

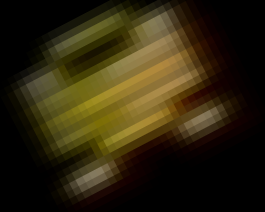
100% pure BASIC V2

Rosario De Chiara
rosario@dechiara.eu
www.dechiara.eu

TenLander:
Background history

You feel your brain power reducing any minute because of the cosmic rays, you just have the computational power of ten lines of code to understand what is happening:
your ship is not strong enough to protect your body; there is so little fuel to handle the landing, luckily you can see a shiny landing platform in the dark…

It is matter of seconds, ground is running fast toward you…



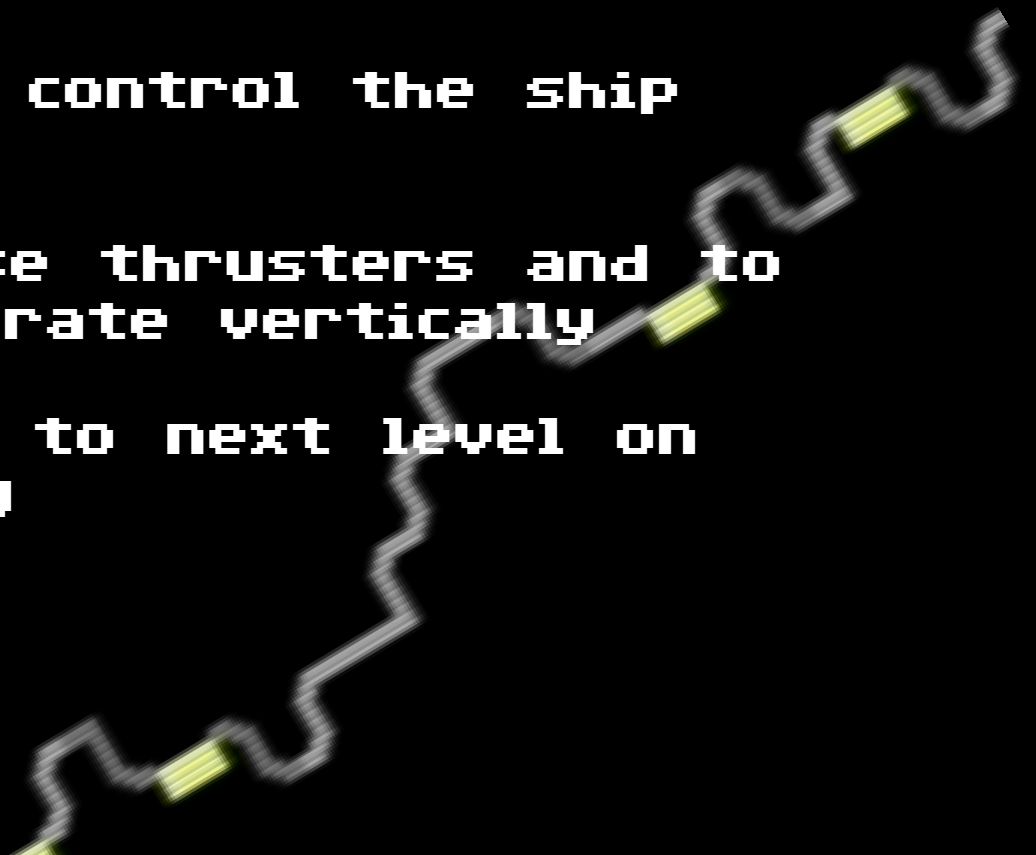
TenLander:
Controls

Use joystick in port 2 to control the ship

Left and right to control the ship without using fuel

Up and down to use thrusters and to decellerate/accellerate vertically

Fire button to go to next level on succesful landing



TenLander: Code Description

Setup and constants

```
0v=53248:fOz=0to21:rEj:pO832+z,j:nEz:pO2040,13:g=0.1:q=255:pOv+32,0:pOv+33,0
```

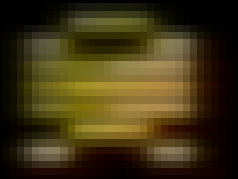
Level initialization

```
1pOv+21,3:pOv+39,7:pOv+40,1:w=0:h=0:x=150:y=80:f=20:pOv+16,0:pOv+39,9:pOv+28,1
2pOv+38,8:?"{clear}{down*9}":d$(0)="{gray}{sh asterisk}":d$(1)="{yellow}{cm
i}":d$(2)="{gray}K{up}{left}U":d$(3)="{dark gray}I{down}{left}J"
3d$(4)="{gray}{cm x}{up}{left}U":d$(5)="{dark gray}{cm
s}{down}{left}J":d$(6)=d$(1):fOi=1to40:?d$(rN(1)*6);:nE
```

Game loop

```
4j=q-pE(56320):u=(jaN2)/2-(jaN1):c=(jaN8)/8-(jaN4)/4:pO54296,0:pOv+37,aB(u*f)
5y=y+w:w=w+g+u*.5*-(f>0):x=x+h:h=h+c/2:b=-(x>q):ifutHf=f+(f>0):pO54296,-(f>0)*9
6pOv+16,b:pOv,(x-q*b)*(1+(x<0))+(q+x)*(1+(x=>0)):pOv+1,y:sx=int((x-16)/8)
7sy=int((y-45)/8):l=pE(1024+sx+sy*40):ifl=98aNw<1.5tH?"landed":wA56320,16,16:gO1
8pO781,0:sY59903:?"{home}";cH(30+2*(w>1.5));"v:";int(w*100);cH(30+2*(f<5));"f:";f
9on-(l=32)gO4:pOv+33,2:pO2040,10:eN:dA40,,,131,,,175,,,191,,,175,,,60,,,195,,,65
```

TenLander:
Level initialization



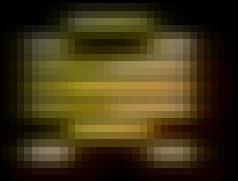
```
print "{clear}{down*9}" :
d$(0) = "{gray}{sh asterisk}" :
d$(1) = "{yellow}{cm i}" :
d$(2) = "{gray}K{up}{left}U" :
d$(3) = "{dark gray}I{down}{left}J" :
d$(4) = "{gray}{cm x}{up}{left}U" :
d$(5) = "{dark gray}{cm s}{down}{left}J" :
d$(6) = d$(1) :
for i = 1 to 40 : print d$(rnd (1) * 6); : next
```

This is the piece of code in charge of drawing the planet landscape. The d$ array contains the pieces of the mountain profile. The for loop just draw 40 of this pieces.

In each piece there is also embedded the shading color.

Please note how the probability of generating a landing platform d$(1) is doubled by assigning d$(1) to d$(6)

TenLander:
Game loop 1



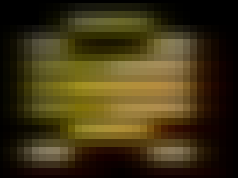
```
j = q - peek (56320) :
u = (j and 2) / 2 - (j and 1) :
c = (j and 8) / 8 - (j and 4) / 4

y = y + w :
w = w + g + u * .5 * - (f > 0) :
x = x + h :
h = h + c / 2 :
```

This is where the joystick position is read and the increments for the horizontal and vertical axis are assigned

Please note how the vertical velocity (w) is updated by applying the gravity g and the current acceleration from the engine u, only if there is still fuel.

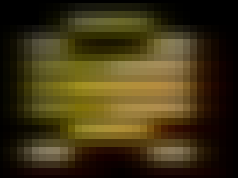
TenLander:
Game loop 2



```
b = - (x > q) : if u then f = f + (f > 0)
poke v + 16,b :
poke v,
  (x - q * b) * (1 + (x < 0)) + (q + x) * (1 + (x = > 0)) :
poke v + 1,y :
```

This piece of code contains the logic to update sprite position. It is worth noting how it takes care of the x>255 condition by applying all the modifications needed to pass the correct x value to the poke. At the core of the logic there is the variable b which also used to enable the bit in v+16

TenLander: Game loop 3



```
sx = int ((x - 16) / 8)
sy = int ((y - 45) / 8) :
l = peek (1024 + sx + sy * 40) :
if l = 98 and w < 1.5 then print "landed" : wait
56320,16,16 : goto 1
on - (l = 32) goto 4 : poke v + 33,2 : poke 2040,10 : end
```

This is the complex logic that understands what is happening under the legs of the ship. The two variables sx and sy are the screen coordinates of the character under the ship and in l we read it right from the screen.

If l is 98 means we have happily landed but only if velocity (the variable v) is low enough: if so we wait for the trigger to go to the next level

The on..goto is used as a poor man solution for if..then..else if l=32 then we are still flying, if not then we just crash the sprite and finish the game

TenLander:
Constants and variables

b	True when sprite x coord 255
c	Horizontal joystick axis increment
f	Fuel
g	Gravity acceleration
h	Horizontal velocity
i	Index for a for-loop
j	Keeps what is read from data
l	Contains the character under the ship
q	Just the 255 constant
sx	Screen coordinates of the sprite
sy	
u	Vertical joystick axis increment
v	Just the 53248 constant
w	Vertical velocity
x	Sprite coordinates
y	
z	Index for a for-loop

The end

Rosario De Chiara
rosario@dechiara.eu
www.dechiara.eu