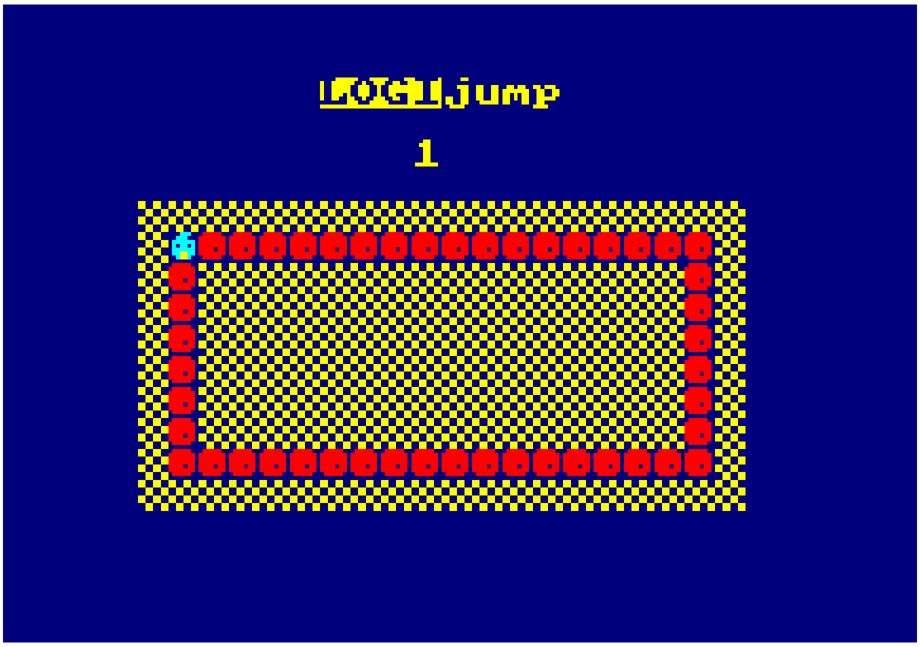
LogiJump

A puzzle game for the Amstrad CPC.

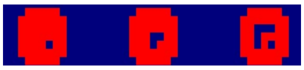
DESCRIPTION

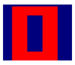


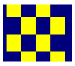
Logi needs to jump. Jump over the tiles in order to remove them. Sometimes you need to walk once over them, sometimes twice, sometimes three times. If you can remove each tile and land on a secure platform finally, then you have won. If you get off the path or don't land on a secure platform finally, then you lose. But you can try the level again. There is a total of 6 levels.

HOW TO PLAY

Use wasd to move. Only walk on the tiles (red) until they disapear. Be carefull when moving, the keys might react quicker than expected. Use space to advance to the next level or to restart the current level. You also may use wasd, but that might cause you falling somewhere if you push the button too long.


Tiles that need one, two or three "jumps".


Save tile / platform.


Death zone.

EMULATOR USAGE

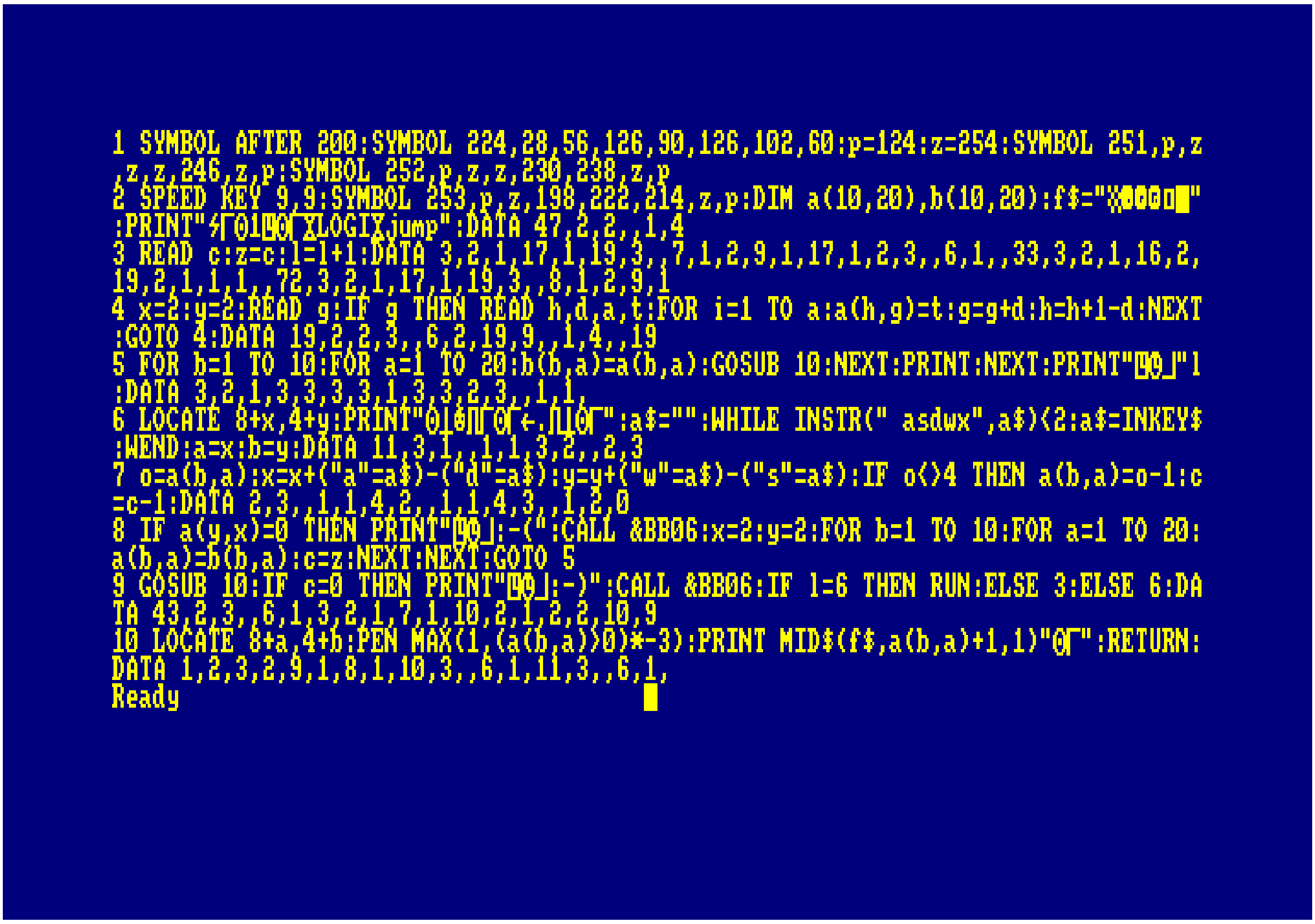
Disk: Run the emulator of your choice. Insert the disk. (Show the directoy with cat.) Load the program (load "logijump"). Run.

BAS-File: Just drag and drop the file over the JavaCPC emulator (or another one, that allows that).

VARIABLES AND DATA STRUCTURES

Variable	Usage
f$	defines shape (character) of tiles
l	level
x,y	coordinates of player (protagonist)
c	count; number of tile walks saved
a,b	old coordinates of player (protagonist)
g,h	start coordinate of tile row
d	direction (0..vertically / down; 1..horizontally / right)
a	number of tiles
t	type of tile [0..space, 4..secure place, 1..requires one walk, 2..requires two, etc.]
b(,)	Backup of a(,)
z	Backup of c
p and z	Just used to shorten code (1st and 2nd line only)

CODE



```
1 SYMBOL AFTER 200:SYMBOL 224,28,56,126,90,126,102,60:p=124:z=254:SYMBOL 251,p,z
,z,z,246,z,p:SYMBOL 252,p,z,z,230,238,z,p
2 SPEED KEY 9,9:SYMBOL 253,p,z,198,222,214,z,p:DIM a(10,20),b(10,20):f$="▒●●●○█"
:PRINT"ϟ┌01⊔0┌XLOGIXjump":DATA 47,2,2,,1,4
3 READ c:z=c:l=l+1:DATA 3,2,1,17,1,19,3,,7,1,2,9,1,17,1,2,3,,6,1,,33,3,2,1,16,2,
19,2,1,1,1,,72,3,2,1,17,1,19,3,,8,1,2,9,1
4 x=2:y=2:READ g:IF g THEN READ h,d,a,t:FOR i=1 TO a:a(h,g)=t:g=g+d:h=h+1-d:NEXT
:GOTO 4:DATA 19,2,2,3,,6,2,19,9,,1,4,,19
5 FOR b=1 TO 10:FOR a=1 TO 20:b(b,a)=a(b,a):GOSUB 10:NEXT:PRINT:NEXT:PRINT"⊔09┘"l
:DATA 3,2,1,3,3,3,3,1,3,3,2,3,,1,1,
6 LOCATE 8+x,4+y:PRINT"0┴8∏┌0┌←.∏┘0┌":a$="":WHILE INSTR(" asdwx",a$)<2:a$=INKEY$
:WEND:a=x:b=y:DATA 11,3,1,,1,1,3,2,,2,3
7 o=a(b,a):x=x+("a"=a$)-("d"=a$):y=y+("w"=a$)-("s"=a$):IF o<>4 THEN a(b,a)=o-1:c
=c-1:DATA 2,3,,1,1,4,2,,1,1,4,3,,1,2,0
8 IF a(y,x)=0 THEN PRINT"⊔09┘:-(":CALL &BB06:x=2:y=2:FOR b=1 TO 10:FOR a=1 TO 20:
a(b,a)=b(b,a):c=z:NEXT:NEXT:GOTO 5
9 GOSUB 10:IF c=0 THEN PRINT"⊔09┘:-)":CALL &BB06:IF l=6 THEN RUN:ELSE 3:ELSE 6:DA
TA 43,2,3,,6,1,3,2,1,7,1,10,2,1,2,2,10,9
10 LOCATE 8+a,4+b:PEN MAX(1,(a(b,a)>0)*-3):PRINT MID$(f$,a(b,a)+1,1)"0┌":RETURN:
DATA 1,2,3,2,9,1,8,1,10,3,,6,1,11,3,,6,1,
Ready
```

CODE DESCRIPTION

1: define symbols for field and tiles

2: set key speed; define another symbol; create field and backup field; define all fields; print title

3+4: Read all data into the field; increase level by one

5: Print the field and create backup; show the current level

6: print main character and wait for key input (wasd)

7: calculate new position; if field is not a save platform: decrease its value & decrease left tiles value
8: if you landed in nowhere you gotta restart the level
9: print the current position; if all tiles cleared you either go on with next level or you won and game restarts
10 routine to print current field at position b,a according to field a(,)

LINE LENGTHS

As you can see in the code above each line with more than 120 characters (=1 and a half lines; =80+40) contains PRINT statements that can be shortened and/or the space after the line number is not necessary as well.