Teng! (Ten Line Basic Pong!)

Teng! Is a simple Pong-like Game. The game is designed for two players with joysticks.
The game is written for C-64
it’s category is PUR-80

Lines:

```
0b=53248:z=56320:a=2040:pOa+1,194:pOa+2,193:y=99:t=12315:?"{clear}{green}":pOb,y:pOb+2,y:k=1
```

Variables:

- b base to set sprites positions
- z is used to read joysticks positions
- a is the base for sprites pointers
- y is the abscissa of ball
- t is used like a memory base address to draws sprites
- k is one of the variables for score

Sprites pointers are initialized and some positions too.
The screen is cleared and the character color was set on green

```
1pOb+16,0:fOj=1to21:?:fOi=1to30:?".";:nE:nE:v=3:pOb+1,55:pOb+3,240:pOb+21,7
```

Playfield is filled of “.”, v is the “velocity” of the ball, other sprites positions was set

```
2k=k+(y>d):q=q+(y<60):fOi=0to191:pO12288+i,0:nE:pOa,i:x$="{17}{125}{125}{255}":?"{home}teng!",q,k
```

Variable k and q (the two scores variables) are recalculated based on y position. This line is the fist line of the end-game loop so y position will be out of screen before upper pad or after the bottom one.
The cycle is used to clears sprites, and the poke a,i is used to initialize the first prite pointer with 193 (i value after the cycle).
The variable x$ is initialized with 4 characters (curly brackets contais the ascii value of the character):

- {17} is cursor down
- {125} is shift + ’B’
- {255} is Commodore Key + ‘B’

Last operation of this line prints the game name and the player’s score

```
3fOi=0to3:p=aS(mI(x$,i+1,1))-1:pO12352+i*3,p:pO12373-i*3,p:nE:x=rN(ti)*100+30
```

Cycle to build ball sprite. Variable x (the x coordinate of the ball) is initialized with a random value.

```
4pOb+4,x:s=54296:y=140:pOb+5,y:fOi=0to5:pOt+i,255:pOt+101+i,255:nE:d=233:h=3
```

Initialization of ball’s coordinates and cycle to build pad’s sprites. At the end some initialization of usefuls variables.

```
5x=x+h:y=y+v:n=pE(z):l=pE(b)+sG((naN4)-(naN8)+4)*4:ifl>0aNl<256tHpOb,l
```

Increment x and y with v values (v can be positive or negative)
Test position of Joy 2 and calculate the shift value.
At the end there is a test to avoid illegals values for x-coordinates, if x is ok the x-coordinate will be assigned.

```
6n=pE(z+1):l=pE(b+2)+sG((naN4)-(naN8)+4)*4:ifl>0aNl<256tHpOb+2,l
```

Test position of Joy 2 and calculate the shift value.
At the end there is a test to avoid illegals values for x-coordinates, if x is ok the x-coordinate will be assigned.
Thanks to Felice Nardella for x-shift calculation

```
7on(aB(x<256aNx>24))gO8:h=-h:x=x+h+h:pOb+4,x:pOb+5,y:pOs,15:pOs,0:gO5
```

In this line the ball’s coordinates are checked for x-side bouncing effect. If there is a bounce the h variable will change its sign and a sound is emitted.

```
8j=x-pE(b+aB(y>d)*2):if(y>dory<65)aNj>-10aNj<28tHv=-v:y=y+v+v:pOs,15:pOs,0:gO5
```

In this line the ball’s coordinates are checked for y-side bouncing effect. If there is a bounce the v variable will change its sign and a sound is emitted.

```
9on(aB(y>dory<60))gO2:pOb+4,x:pOb+5,y::gO5
```

If ball’s y coordinate is out of playfield this game is over.
Thanks to Davide Fichera for the “On(...)” trick

That’s all folks!
Emanuele Bonin