Great!

As you can see, we play with the coordinates x and y to create this illusion of movement.

x = x+3

Adding x+1 makes the object move to the right.

Adding x+3 makes the object move to the right, but faster. It simply tells x to skip 3 pixels per frame instead of just 1.

That is just one of the techniques programmers use to change the position of objects in a game.

How fast the position changes, or the overall path of the movement can be thought of as functions. Yup, functions like the ones you learned in school… f(x) type of thing.

These functions receive an input, calculate something, and return a new value as result.

In our case we received the last position x of the object as an input, added 1 pixel to it, and returned the object’s new position x, which will be used to display the object in the next frame.

<gif>

You probably studied - or at least pretended to study - the ideas of linear functions, quadratic functions, trigonometric functions, and many others.

If you remember, a linear function is a function that describes a simple line:

<the function f(x) had a simple x plus or minus a number, no power, no exponents, just a linear function of a linear x>

f(x) = x + 2

How many games did you play where the player movement is constant or the enemies follow a simple line movement? …well, we just saw one in the last code we worked, didn’t we?

We also had quadratic functions, which described parabolas:

<in that case, f(x) has an x-squared, which was the main element for us to say a function was quadratic>

Again, how many games have bombs or projectiles following a parabola path? Or an angry bird performing a parabola-like movement to hit a pig?

You see… all these math functions can help programmers define the characteristics of a game. Things like position, movement, speed, or acceleration.

Here at the company we have some games that need to simulate objects falling. Like enemies falling from the sky, or a bomb falling from a plane before hitting the ground.

All these games need to simulate the characteristics of acceleration of gravity.

Your next task is to study the code that performs this simulation of gravity.

Run the code to see the object falling. You’ll see that gravity makes the object start very slow and then accelerate as it falls down.

<wait 5 secs>

Your task here is to basically invert gravity.

The final result should have an object starting at the bottom of the screen and then accelerating upwards until it hits the top of the screen.

Your circle will start with speed 0 and proceed to accelerate upwards.

<wait 5 secs>

Take your time, understand the functions, the formulas, and when you are done just type “exit” to leave the console.

See you soon…