This is the second part of the blog series “Numbers in Game Development” if you haven’t seen the previous one, I highly recommend to do so, links are below

Link

Part 2

Maths functions, Whaaaat?

So, in the last part we decided to make a zombie game where the zombie spawn rate increases exponentially with respect to the amount of zombies you killed. So what does the word “exponentially” mean? You might have seen equations like   
[https://latex.codecogs.com/gif.latex?2^4](https://latex.codecogs.com/gif.latex?2%5e4)  
in school, and you would know that its answer is  
[https://latex.codecogs.com/gif.latex?2^5&space;=&space;2&space;\times&space;2&space;\times&space;2&space;\times&space;2&space;=&space;16](https://latex.codecogs.com/gif.latex?2%5e5&space;=&space;2&space;\times&space;2&space;\times&space;2&space;\times&space;2&space;=&space;16)  
lets see what happens if we change this equation a bit by, replace 4 with x and make that equal to y  
[https://latex.codecogs.com/gif.latex?y&space;=&space;2^x](https://latex.codecogs.com/gif.latex?y&space;=&space;2%5ex)  
This is known as exponential function, what does it mean then? it simply means that, if you want the answer to become 16 like before then we have to plug in x = 4,  
{Graph}  
so if you draw a line on x = 4, it will intersect the exponential curve on y = 16, as you see in the graph. Similar to x=4, if you plug any value for x and you will receive a value of y. Try it your self   
<https://www.desmos.com/calculator/dvf0nw7qbh>  
drag the slider for x and click on the intersection you will get the value for y for x you have selected

I am growing Exponentially :D

By trying the graph yourself, you might have noticed that as the value of x increase, not only y increase, but the rate of which y increases also increases. If you don’t understand this sentence, compare the exponential graph with linear graph  
<https://www.desmos.com/calculator/tabho1ys4v>  
y = 2x  
In linear graph, the rate by which y increases with respect to x remains constant. In the case of given equation, if x is increased by 1, y will increase by 2, no mater what the previous value of x was. But in case of exponential function. If previous value of x was 1 and you increase it by 1, y will increase by 2, whereas if the previous value of x was 2 and you it increase by 1, y will increase by 4. This is what exponential function is.

Rise my zombies

As I told you before, looking at just the maths will confuse anyone, but when we see the maths working in a game, we shall understand how these maths function work. So lets see how we will implement this maths in the game. First I will show you how the game works with a constant frequency of zombie spawning, then we will use linear growth function to understand how the variables in the game engine and maths variable links. And then we will implement our exponential growth function. Piece of cake, isn’t it?

The undead variables

Lets just see what are the variables we will use

* killCount (c) : this is the number of zombies we killed from the start
* spawnFrequency (f) : this shows how many zombies will spawn in the map in 1 second, this is what we want to increase exponentially with respect to zombieKillCount
* spawnIncreaseRate (r): this is the rate by which zombieSpawnFrequency increases
* initialSpawnFrequency (fi) : this is the zombieSpawnFrequency in the start of the game
* finalSpawnFrequency (ff) : this is the maximum value for zombieSpawnFrequency, we are keeping this because we don’t want our player to get destroyed by crazy amounts of zombies LOL
* killCountThreshold (cf) : this is the threshold amount of kills after which frequency becomes constant.

Enough with the boring stuff

Yeh, I know this has been boring enough, lets just see how the game works right now, with constant spawnFrequency.  
{game video 1}  
I know the frequency is constant, but just to visualize it, lets plot a graph of killCount vs spawnFrequency, to have a better understanding of the situation  
{game graph 1}  
As you can see, for any value of killCount, spawnFrequency stays the same. We don’t want that. So how do we want it?

Growing linear

What we want to do is, increase zombie spawnFrequency as the killCount increase, lets try to make a rough graph for it, then then we’ll see it in desmos  
{drawn graph 1)  
In the graph, we can see that when zombieKillCount is 0, the spawnFrequency is fi, that means starts in the start of the game when the player doesn’t have any kills, spawnFrequency will start at fi. We can see that from fi to ff the graph looks like a line, so we can plug in equation for line, which is  
y = mx + c  
Lets plug it into desmos and see how it looks  
{desmos link}  
we were dealing with the variables we defined earlier, but what are these new variables, y, m, x, c. By simply comparing the graph we drew and the graph in desmos we can see that, Y represents the Y-axsis, which is f, x represents the X-axis which is c. m represents the rate at which y increases as the x increase, this also known as gradient in mathematics. So for use m = spawnIncreaseRate (r). And c is called y-intercept in mathematics, which means the point on y axis where the line cuts or the value for y when x is zero. So c = fi. So writing the equation in terms of variables we are using for the game we get

f = rc + fi

looking back at our graph line just goes from fi to ff after that the frequency stays at ff so

if 0 <= c <= cf

f = rc + fi

if c > cf

f = ff

Coding It Linearly

We have finally calculated the equation for spawnFrequency but we can’t simply plug these formulae in the code. First we’ll have to decide what we want to adjust manually and use those to calculate rest of the formuales.

The varables we’ll need to enter manually are initalSpawnFrequency (fi), finalSpawnFreqency (ff) and spawnIncreaseRate (r). We will get killCount (c) from the game itself, as the player kill enemies killCount will increase. And as for f, we calculated the formulae in the previous section. Only variable he are left to calculate is cf. To calculate cf, what we will have to do is plug the point (cf, ff) into the equation (x) and solve for cf.

{solving}

Watching it work (Linear)

So this is the equation for we got for cf, now lets code these equation and see the result for ourselves

{code screenshot}

{video of graph visualizing}

As you can see, the we can literally visualize the graph that we calculated by maths into the game itself. This is the beauty of mathematics.

In the next part we will make it so the spawnFrequency grows exponentially instead of linearly, Thanks for reading my post, follow me to stay tuned for upcoming parts, and don’t forget to press the like button. I’ll see you guys later.

Squad fam out.