Protect Yer Booty is a tower defense game. The main aim of this game is for players to protect their treasure from other pirates who will be coming to steal it! There will be 5 lanes in the game, each with a cannon for the player to use. The player will have to top up the cannons by collecting cannonballs from a box and running to place it into a cannon, which will then fire the cannonball at the on-coming pirates. Anytime a pirate manages to reach the cannon before dying, it will cause the player to lose health. However, if the cannonball is able to hit the pirate before it arrives, the pirate will be eliminated. The game will continue to run until the player runs out of health.

**What does the game do?**

The game follows a tower defence game where

**What are the rules it follows? Can you write down those rules?**

The

**What information do you want to store for each user?**

The

**What can the user do? What does the program do for each user action?**

The

**When does the game end?**

The

This is my first week’s game progress diary! I haven’t actually started coding the game, I have only started to draw out what are the different interactable items and actions that will occur within the game.

For now, I am in the process of doing up my own assets (Just because I like to draw pixel art) So I’m getting excited, and will start actually coding everything on Saturday!

By the end of this week, I hope to be able to code out the location of every single interactable item into my game, and coding to let arrow keys to control the movement of the playable character! If I have time, I try attempting to also code how the player is unable to pass a certain position!

I will also code out the functions for the bombs and the repair kit to create a circle svg or an image svg that the character can hold.

To share more as to what I am envisioning for the game, it is similar to a tower defence game which utilises gameplay moments from plants versus zombies and mechanics from overcooked.

To simply it, the player will be able to pick up items from boxes to put into the cannon. Each cannon can only hold one cannonball at a time, and will only fire after two second intervals of putting a cannonball in. Every time an enemy touches the cannon, it will lose it’s ability to function, and will only work again when the player uses the repair kit to fix it. However, if the player fails to fix the cannon before another enemy touches it, it will result in game over! To accommodate with players accidentally picking up something that they did not need, there will be a trashcan for players to throw away the item that they are holding.

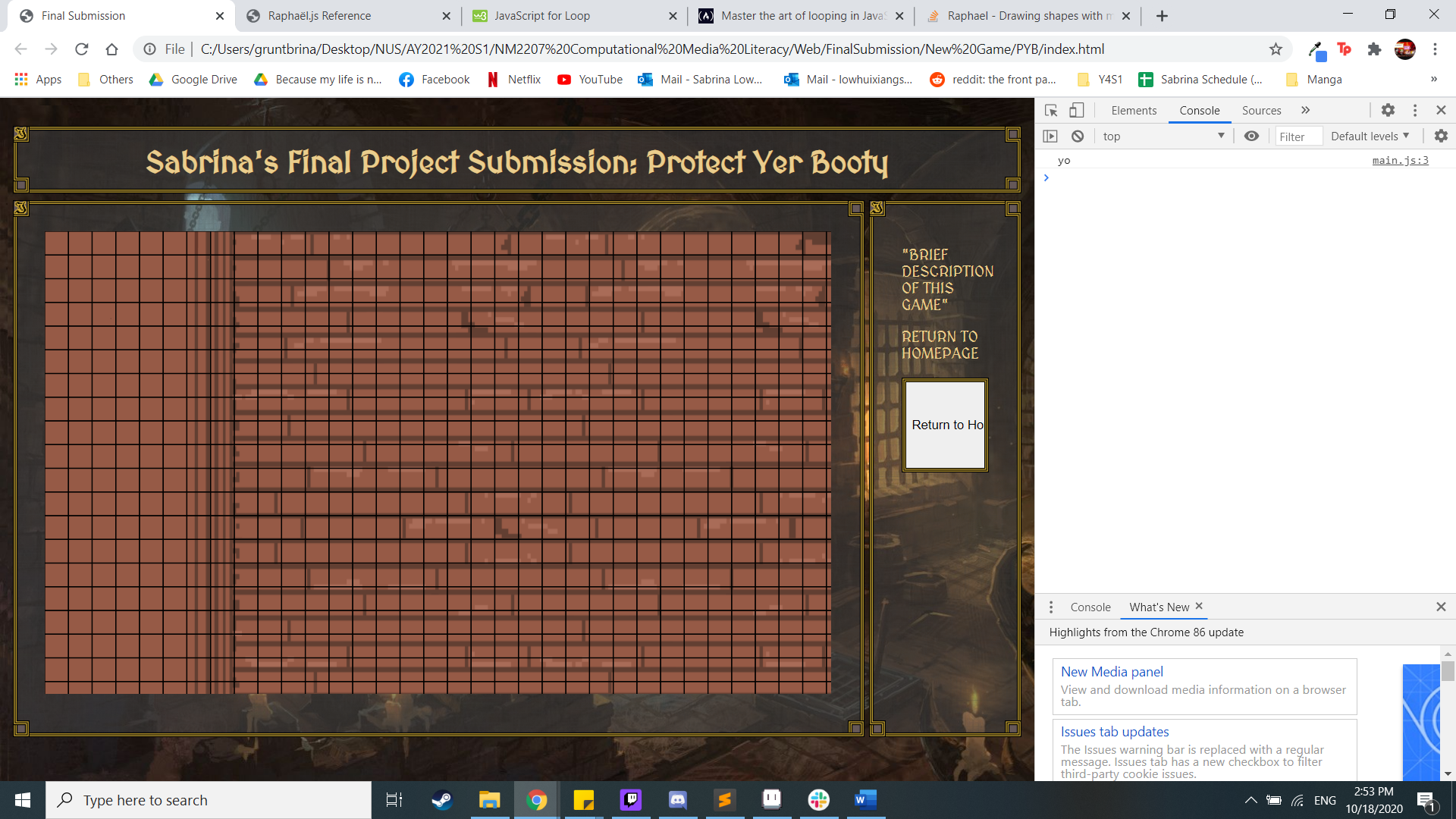
For the rules of the game, there are at least 6 main interactable objects that has it’s own set of rules. Firstly, we have the player. The player is the main star of the entire game. They need to be able to move and interact with objects. The things that it would need to interact with would be the cannonball box, repair kit box, rubbish bin, and cannons. For the cannonball box, it would need to be able to give the players a cannonball whenever they interact with it. Whereas for the repair kit box, it would need to give the player a repair kit whenever they interact with it. Cannonballs are only interactable with cannons and the rubbish bin. When a cannonball interacts with the cannons, it will load up the cannon and fire, while removing the cannonball from the player’s hand. Whereas if it interacts with the rubbish bin, it will be removed from the player’s hand. Whereas for the repair kit, it would only be able to interact with a destroyed cannon and the rubbish bin. There is another iteration of cannonballs called the fired cannonballs which will interact with enemies. Enemies are randomly spawning objects that can spawn at 5 different y coordinates. Enemies can interact with fired cannonballs and cannons. When a fired cannonball hits the enemies, it will eliminate the enemy. However, if no fired cannonball eliminates the enemies and it collides with a cannon, it will destroy the enemy and the cannon. If the enemy collides with a destroyed cannon, it will then cause the game to end. The repair kit will be in play when there is a destroyed cannon, as the player will be able to use the repair kit to repair the destroyed cannon. Those are the main rules for my current game. I will also be planning to add a point system, where any fire cannonball that hit the enemy will result in a +1 in score for the player.

For me, the challenge in coding this game would be finding out how to pin an image onto the player whenever they pick up an item. Another challenge would be deciding how collision works, so that when fireballs are fired, the enemy can be eliminated. As of now, I am hoping to figure out how to accomplish the above challenges for my game. As I progress, I believe that I will be faced with more issues in which I will share in my future diary progress!

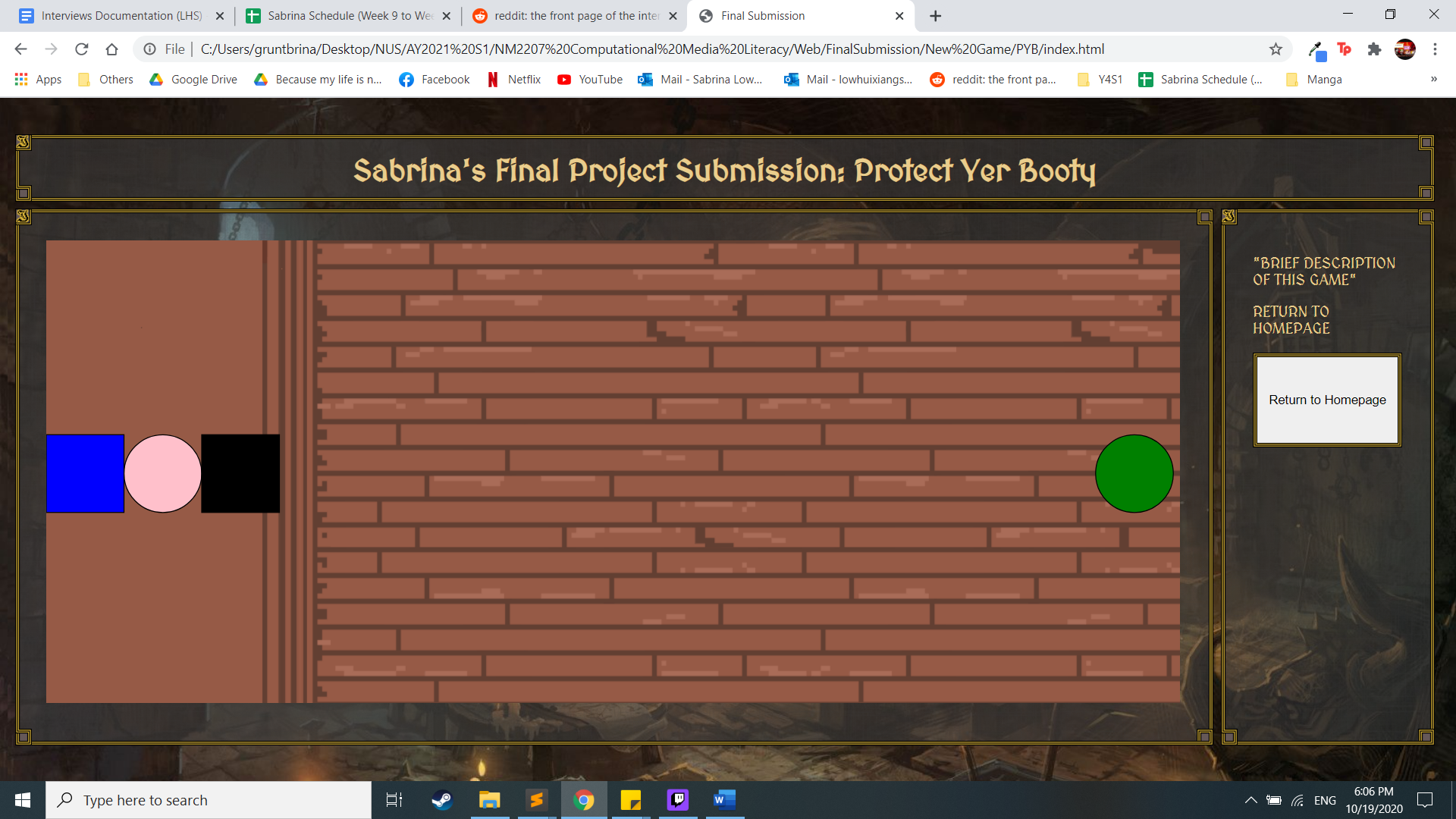
Welcome to my week 9’s diary entry! I have to say, as I write this, I have come to a realisation that even if you know exactly what you want your program to do and what type of codes you could use, it’s totally different when you’re implementing it. This said, I have decided to give myself even more time to focus on coding out my program instead of my current half a day model.

Anyways, moving along to what I have done this week, and how I am planning to progress! For my current progress, it is really very minimal! I have my background along with placeholder position of the objects in my game. And clicking on the black square will result in a cannonball being shot out! I know that originally, I wanted to be able to control my character’s movements by the first round, but I met with some difficulty. Originally, I assumed that I had to use the switch-case, but that was not the case (Ooo, pun?). I realised that I could simply utilise and if-else function, but I realised it too late, so I’ll only be implementing it the next time round! Instead, I decided to tackle the cannonball shooting situation first.

To share an in-depth understanding of how my first attempt went: I basically added comments to partition my code into the different objects. This was done so I know where to look for the different objects in the code for my game. After doing so, I created a grid using a for loop to help me determine where I could locate my objects.



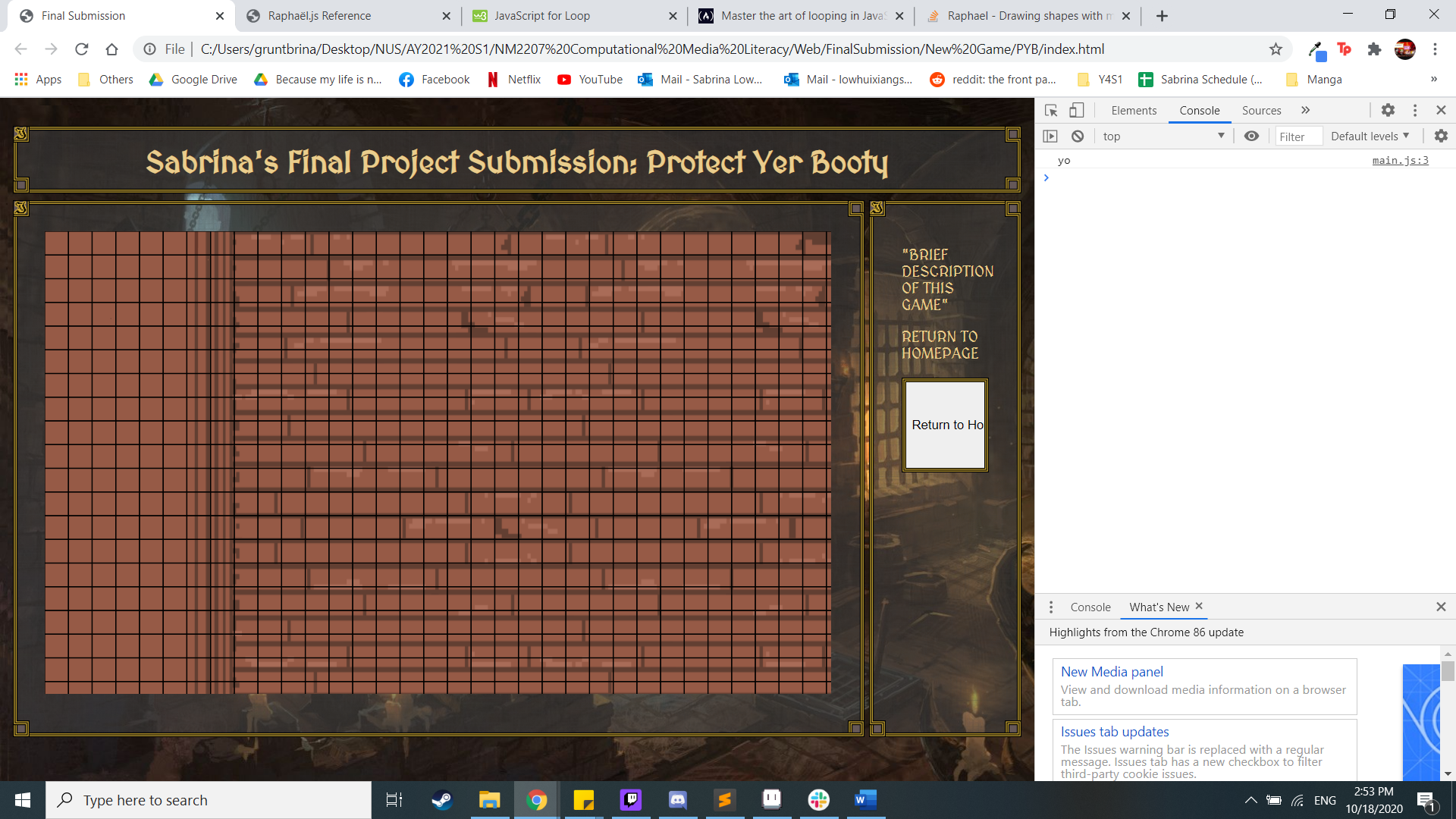
From there, I managed to obtain both the size and the position of my objects.

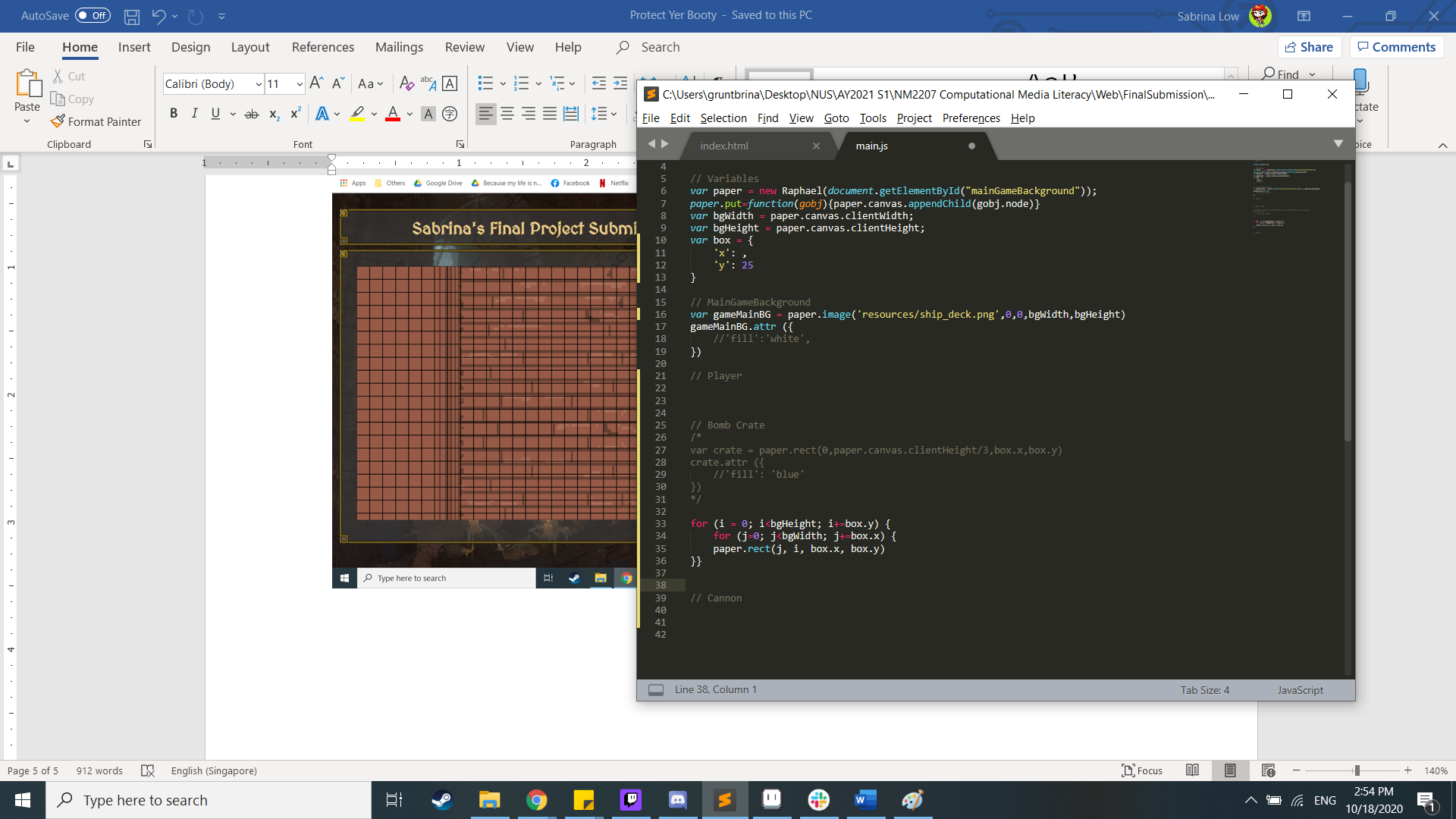


Afterwards, I struggled with creating the code for the cannonball! The ball refused to move (Or it just moved real awkwardly). This caused me to go into a frenzy. However, after panicking heavily and eventually calming down, I realised that it just stemmed from my lack of understanding on how setInterval would really work! Thus, I went back to look at all the different lectures, and went online to view videos on how setInterval would work. To be honest, none of it really helped me. So, I decided to just try and decipher to code through staring and making of sense as to what is happening when I do certain things. This led me to realise that a reason why my code was not working was because I failed to realise that the creation of the bomb and the movement of the bomb should not be in the same line, because it would result in a conflict. Even though this was how I came about to using the if-else statement to help me separate both the creation of the bomb and the movement of the bomb (With a global state variable to help me with the 2 variation), I realised that my headache with this portion has not come to an end. On top of having to do the collision with enemies, I did not realise earlier that I would figure out how to make sure that when I click on the cannon another time after the ball has been fired, it would not speed up the original cannonball, but create a new cannonball instead. I’m planning to use loops for this, to help separate the cannonballs. (We’ll see how it goes)

Moving on, I am planning to work on player movement, separating cannonballs, and potentially collision between cannonballs and enemy. However, player movement would actually be the last thing on the list, as I am planning to ensure that my cannon is workable when I click on it. If I have more time, I will also work on creating enemy movements, which would actually work similarly to how the cannonball works. Furthermore, I will also consider how to create enemies at random intervals. I’ll stick to working on a single lane first, as once I get the mechanics for one lane, I can easily replicate it using loops. Therefore, for my first few iterations, I am hoping to get things done for the middle lane.

Piling onto the challenges that I mentioned in my previous diary entry, I foresee challenges in my implementation of loops as I have to be honest, I’m not that familiar with it yet. But I think that my ability of being able to look through and slowly absorb what the code does is really helpful, so I hope that as much as I think it would be a challenge, I can still work on it quick enough to accomplish more! 😊 I would also say that I may face challenges in my implementation of player movement, but I realise that it would most likely work similarly to the movement of the cannonball. Looking forward to coding this week! :D





It’s finally week 10! Sadly, I did not have as much time as I originally wanted to use for this module in the previous week because of the crazy workload of a submission that I had to do. Therefore, the only improvements I made to the game was allowing the character to walk.

I did spend more time on it, trying to work on the cannonball shooting mechanism, but it doesn’t seem to be working out. Originally, I wanted to make it such that when I click on the black square, the cannon will continuously shoot out cannons, while the others that were already moving, were still moving. However, after adjusting the code a few times, I still could not really figure out how it would work (Even with loops). The issue lies with me not being able to determine how exactly would I use arrays and loops in this scenario. As I coded my code, I tried to word out what would happen at every step, however, I realised that I had not really understood how to add to an array properly. When I have more time, I need to understand what I need to do to add on something as a new element to the array instead of modifying the original existing member in the array. Also, I need to determine how my loop would be working. Whenever I use the loop, I know that I am always repeating the actions inside it. This has made me wonder if I even need to use loops. But without loops, I’m unsure how I could reach out to the different arrays such as “array[0], array[1], etc.”. Anyway, this has been my existing headache and problematic issue. So for now, I have just made it such that the cannon can only be shot again when the existing cannonball disappears. I will revisit this when I get the other features correct. But this is an even more basic version of the other version that I wanted. But at least it works!

I have also just added movement for the enemy, allowing it to move from right to left on the screen. There was not much problem in implementing it was similar to the cannonball looping. The next issue is the spawning on enemies. I will be looking online for assistance when I get the time during my next coding session! Anyway, I also finally made it so that my character moves with the arrow keys. It used simple if-else functions!

Looking at my progress, I think I will need to focus on doing the collision between objects for my next round and get it right. I will also try to make it such that enemies will spawn repeatedly. I am unsure as to how to do the enemy spawning repeatedly, but I will be trying my best using what I already know (E.G: setTimeOut, setInterval, etc.)

Once I am able to at least get the collision and enemy spawning right, it can form a basic shooting game at least. To be honest, looking at this situation, I think that my knowledge on how arrays work is definitely not there… Especially since I somehow think that arrays and loops are always paired up. But it may not be the case. Nevertheless, I will try to also refer back to the notes and see what I’m misunderstanding (Especially because next week is the final quiz – whoops).

This is my week 11 update! It’ll be kind of short as this week I was extremely busy, so I am grateful that I am given time to code during the class itself. Moving along to what I have accomplished, I have managed to get collision between the enemy and the cannon to work. This meant that whenever the enemy manages to reach the cannon, the enemy would disappear. Apart from this, I have decided to tackle my issue on the spawning on enemy by trying to do multiple enemies this week. Although it is not yet done (Which is why I probably won’t be updating the latest version), but at least I am slowly trying to get it to work.

The progress that I had accomplished in week 11 would be that I had managed to get collision to work for my cannon and enemy. The next step of collision would be getting the cannonball and enemy to collide. That would bring me to what I hope to accomplish in the coming week, given that I would have more time for this module as most of my busy and rushed submissions were last week.

For the upcoming week, I will begin by creating multiple objects of the same type (E.G: Enemies) and coding them individually to see how they will move. The next step will be for me to identify the similarities of these code, and formulate a loop out it. I realised that as I have a problem with immediately creating a loop, especially when setInterval is involved, I splitting the different objects to know and understand how it would work. A current idea that I have is to ensure that I have 3 enemy objects (As a start), each with a different timing of coming out. Thus, I will see how this plan will work out. Also, in order to catch up, I will also be implementing the collision with cannon to see that work as well. If I am able to do all these, I will then start creating a score system, where users can see their score. This way, when the enemy collides with the cannon twice, it will result in game over, but every time a cannon hits the enemy, it will add a score.

I’m not too sure if what I plan to accomplish by this week may seem very ambitious, but I believe that I will be able to achieve it as long as I do not meet with any other unforeseen issues! Fingers crossed though. Also, this is how I make up for the past 2 weeks! :D

Once again, the time has come for me to upload another Dairy update! At first, I thought I could do a lot, but in reality, my other modules still sucked my time away… Anyway, I’m still happy as I managed to accomplish the collision with the cannon, bomb, and enemy! As this meant that my game is now functional where players can actually play as there is some form of interactions involved.

In order to achieve the collision system between the bomb and enemy, I took out the bomb from the function, allowing for it to be a global variable that can be acted upon by the other objects in the code. Through doing so, I then created if functions within the enemy movement function to help me check the conditions as to whether there were any colliding using the distance between the two objects.

Moving on from here, it’ll be a full-out rush to the final project! I’m just going to place my wish list of items that I hope to accomplish by the end of the Friday next week!

#1: Add a score and health system to establish game-over criteria

In order to implement the above health system, I will just add two new variables to store the score and health, with a full health starting from 3 hearts. Every time an enemy is able to interact with the cannon, it will result in a loss of a heart, whereas if a bomb interacts with the enemy, it will add 1 to the score. This should be simple enough to implement as it only utilises knowledge from the earlier portion of what we had learnt.

#2: Add multiple enemies to increase difficulty in the game

Now, this would require me to work a little harder. As I am still intending to pursue using setInterval and setTimeout to assist me with this. In order to provide an illusion of the enemy being spawned at random times, I will be using setTimeout to help with creating the enemy at a random time so that they will spawn after the player presses start. Hopefully I am able to use my existing knowledge to help me achieve this. As I feel that it would just work if I do it this way!

#3: Add assets to the game

I have attempted to add assets, but I realised that there might be some issues in doing so, especially since there may be changes required to be made with the collision as it seems that image svg added would convert it into a square instead. If I am unable to make it work with the newly added assets, I will skip this.

#4: Add feature of having to take bombs from the bomb box to add to the cannon in order for it to shoot

In order to increase the interactions with what the player character does, I will be including this ability for player to press a button on their keyboard in order to obtain the bomb and transfer it into the bomb. This way, it increases the difficulty of the game, and allows me to test my abilities in coding. Especially since I would hope to be able to code my own games even after this module for future practice.

#5: Replicate multiple lane systems

For this, I will try using loops to help me replicate my already existing lane. I am not sure how this works yet, as I did not clearly separate what would be included in a lane. Therefore, I will only work on it once I have accomplished the other portions above.

The wish list above is added according to level of importance and what I am going to work on first. I would probably only be able to begin next week, however, as I would no longer have any other modules that I would need to complete, I can fully focus on this module. Since I believe with time, I can solve any solutions, even if I had to modify from my original plan. I think that I have enough knowledge on how codes interact with each other, that given time, I will be able to slowly puzzle my way through and create a code that will work to what I want.

I really appreciate trying to navigate my way using existing knowledge to help me navigate and create interactions that I did not expect.

1. Final write-up (500-1000 words)
2. The idea is to "show off what you know" and how you can be creative with the material you've learned - both in class, and anything that you have learned on your own (for example, maybe you want to use video in some way).
3. Make clear what you actually had to do and explain your thinking processes. You can and should build on your weekly diary entries to write this. For example, you can add more detail to each of the diary entries in ways that explain the code and your thinking process.
4. You may have spent a lot of time self-learning with other material, such as YouTube tutorials, during this module, and we would want to reward you for that. Make sure you write about how you used your creativity and any outside learning that you applied.

<h1> **Final Project Write-up** </h1>

<h2> **Overview:** </h2>

<p> Given my interest in game development, I wanted to be able to obtain some knowledge in coding. Therefore, hearing that I would be able to develop a game for this module interested me enough to take it up. As I want to experience creating a game from scratch, I planned on developing this game called “Protect Yer booty” which is basically a tower defence game. Through this, I felt like I could utilise my knowledge that I had learnt from this module to the max.</p>

<h2> **Detailed Description:** </h2>

<p>“Protect Yer Booty” is a tower defense game with only one level of difficulty.

<br>

The main aim of this game is for players to protect their treasure from other pirates who will be coming to steal it!

<br>

There are 5 lanes in the game, each with a cannon for the players to use. The player will have to top up the cannons by collecting cannonballs from the cannon crate and run back to the specific cannon in which they would want to fire it at any in-coming pirates. Anytime a pirate manages to reach the cannon before dying, it will cause the player to lose health. However, if the cannonball is able to hit the pirate before it arrives, the pirate will be eliminated. The game will continue to run until the player runs out of health. </p>

<h2> **Required Elements:** </h2>

<p>The following were how I implemented the required components into my gameplay.

Grid layout: HTML Elements and Raphael Paper

Arrays and Loops: Loops used to help obtain the grid alignment, and mass produce the different images, while array was used to store similar objects (Bombs and enemies) with different individual attributes

SVG Graphics: Start and restart button, gameboard, and individual game pieces

User interaction: Users will be required to insert bombs into the target lane to shoot it at incoming enemies

JavaScript Timer Animation: Used to animate the bombs and enemies</p>

<h2> **Computational Logic & Skills:** </h2>

<p> As it was an original game, I wanted to ensure that I had all the details noted down properly for easier execution. Therefore, I sketched how I envisioned the game to look, along with noting the different actions that my users could take. This way, when I look at it again, I am able to identify whether I had not completed certain interactions.

<br>

<Image>

<br>

Following that, I easily created the background of the game with Raphael, using loops to help me create line grids to identify how I should size my SVG images. I also created objects to help me store the values as to where potential SVG images would be located, allowing for me to easily change the object’s location, and also to rectify any errors.

<br>

Once I had all the pieces I required on the board, I started with the animation for the bombs. Initially, I wanted to code it such that the bombs would continuously shoot out regardless of whether there was already a bomb that has been shot out. There were two methods that I could have gone about doing this, the first being that I could make it such that the bombs are created whenever users interacted with the cannon. This way, the SVG of the bomb is only created when users interacted with the cannon. The second way was to store an existing SVG of the bomb behind the cannon, allowing it to function as a global variable. When I tried the first method, I realised that there were too many issues that would continue to appear such as being unable to use the bomb’s information to identify its possible against the enemy. Another issue was that I had to figure out how to create individual functions for the different cannons every time a new one appears. Therefore, I decided to circumvent around that issue by going about the second route, in which my SVG elements are already present, but hidden. On top of this, as each bomb required its own function to call it to move, as if they were in the same function, it will affect the movements of the other bombs, I decided to make it one bomb per lane as it would also increase the difficulty level of the game overall.

<br>

The next issue that I had faced came about when I was creating the enemies. The collision went well because it was a simple same y, different x collision scenario. Therefore, I only needed to calculate the x positions of the objects. But the trouble came in crafting a code that would allow me to keep the enemies respawning whenever they were eliminated. I tried searching online to see how I could keep creating enemies to attack. However, I could not find any information that suited my situation as many of them used CSS instead. Instead of following online, I wanted to use what I had learnt from this module to solve the situation. Therefore, I worked it out myself and circumvented around the situation by making sure that whenever an enemy has been eliminated, they will continuously come back, by changing their ‘x’ coordinate to bring them back to their originally location. This would give an illusion that there are multiple enemies.

<br>

Most of the gameplay was almost done, I had easily set the condition for the player to accept bombs from the crate by using a numBomb as a state variable to check if the player has a bomb or not. Using if-else to help facilitate the placing of bombs into the cannons, where if players had a bomb and the lane did not have any bombs flying, they can use it. The next issue was to have separate lanes. This required me to create even more functions to facilitate it. As for each moving object to animate properly and independently from each other, they would need a separate timer function. Thus, I went online to check out if there was such a thing as a function array in which the function could be stored and called from within an array, for example: “function[i] ()”. A google search quickly revealed that there was no such thing. Thus, I decided to create separate functions to facilitate the different movements. Despite creating multiple functions, I tried to create a sub-function to help reduce the amount of code, and also to keep all the potential bug-solving issues to one area. </p>

<h2> **Reflections:** </h2>

<p> As I navigated my way through this module, what I really wanted was to use the existing knowledge that I had gathered to help push my boundaries as to what I can do. Therefore, I tried my best to try and solve problems that I had encountered using the pre-existing knowledge that I had gathered. As I also enjoy the process of game development (Every single aspect of it – Game design, Art, etc.), I took the time to draw the game assets myself to help keep me going during moments when I was mind-blocked.

<br>

Throughout the developmental process, I did face numerous setbacks that resulted in me being unable to continue with that particular aspect of code. However, I really enjoyed breaking down the situation to see which part of the code worked and which didn’t, allowing me to identify and solve the puzzle myself. Being someone who enjoys solving puzzles myself, and not looking at walkthroughs until the end, resulted in me trying my best to solve all the situations that I faced without going online unless I could not figure it out or to gain some hints. </p>

<h2> **Future Work:** </h2>

<p> The possible improvements to be made in the game would be to configure the current screen sizing modifications, so that it would not adjust the images according to the current window size of the computer, and would instead, fit perfectly despite the monitor size that one has. As doing some short playtesting with friends had allowed me to find out that there was difference in visuals on both our screens.

<br>

An example of the difference between the two:

<br>

<Input my visual>

<br>

<Input Janice’s visual>

<br>

I would also like to have added sound effects, more visual animations to facilitate user interaction with more in-game feedback, and more features that I had originally intended to add.

<br>

For my current code, I would test out my theory on whether I could consolidate all the functions into one by creating an array which looks like: “var Array = []; Array[i] = function(i);”. This is in hopes of being able to reduce the amount of code that is written. </p>

<h2> **Conclusion:** </h2>

<p> From this module, I feel like I learnt a lot of applicable skills. Furthermore, it also allowed me to learn a little more about myself. From attempting to do coding, I realised that I can be a logical and systematic person. Especially as I am able to navigate and understand the different ways in which codes can be written or were written. This allows me to be able to grasp the basics faster, and allows me to try and attempt to apply it. Also, I will definitely continue putting these skills to use, because I do enjoy the process of coding as it is basically puzzle-solving, which is something I really enjoy doing. </p>