OCR GCE A

COMPUTER SCIENCE PROJECT

H446-03

Name : Yousef Darwish

Candidate Number : <INSERT CANDIDATE NUMBER>

<Institution Name> : Salford City College

Title of Project : Ultimate Tower Defence

H446-03 – Project CONTENTS

Table of Contents

[A. Analysis 3](#_Toc452555018)

[B. Design 3](#_Toc452555019)

[Systems diagram 3](#_Toc452555020)

[C. Developing the coded solution (“The development story”) 3](#_Toc452555021)

[D. Evaluation 3](#_Toc452555022)

[Project Appendixes 4](#_Toc452555023)

# Analysis

## PROBLEM DESCRIPTION

The game I am going to make is a 2D strategy game based of current Balloon’s TDM and Tower Defense Simulators, where I have multiple waves of entities spawn per wave which each have some sort of animation and health system and have Tower(turrets in a way) which damage the entities in the wave and if any entity reaches the end of the map it will damage the users health and if user health reaches 0 it will displace a “Game Over” screen and details like a scoreboard, However if user defeats all the entities then it will display a “You Win” display.

In the game I will add multiple waves which will have high tier of entities which may be faster or may have more health once I have completed the basics of the game and made fundamental elements of game work, such as entities, movement, currency, timing, turrets/towers etc.

I think the problem I will encounter is that I will have to create the graphics for certain entities and objects and background for the game(making the game look aesthetic pleasing to the users and stockholders, as well I will have to create certain functions for the game that are not already in-build(such as movement of the entities which may be a problem as I would have to create movement for each entity which its own element of movement) but also I will have to make functions for other parts of the code such as creating function to shoot the arrows, function to let towers be upgradeable, make entities have a health value and more in coding language of my choosing.

## STAKEHOLDERS

My game is going to be a 2D strategy game, which will by appealing to anyone near 15-19 years old age gap as It will need some strategic thinking but also will be visually appealing to my current target audience. Therefore, my stakeholders will by my younger brother Younes and my friend Ali (as they are within my target audience but also will give my different types of thinking which will be beneficial to make my game). I will be creating most of the game, but I will turn to my stakeholders for further information such as what elements I should add that I have not thought of but also changing elements such as speed or colour or design as they might be beneficial as they have no biased opinion about the game.

### STAKEHOLDER 1: JOE wOOLLISCROFT

Joe represents a demographic that is familiar with all genres of games, including puzzles and platformers. As such, his input will be vital in my game as it progresses to a more complete state. His honesty and knowledge of these kinds of games will be an important asset for me and the overall quality of my game.

### STAKEHOLDER 2: STEPHEN MA

Stephen represents a demographic who is unfamiliar with these types of games, however he has some knowledge on the subject. As such, he will be key in giving me a less knowledgeable input in comparison to someone who has a lot more knowledge on this subject.

My stakeholders are valuable assets in the creation of my game, as they give me both contrasting thoughts about the systems I implement and how I implement them. Furthermore, other people may see faults in the design of my game that I, the creator, cannot see thus preventing the development of my game getting bottlenecked.

## JUSTIFICATION

I believe to fix and debug my code I will have to think ahead of time to produce inputs or data I can input so it will save me time as I will already have values inputted. As my game is 2d it allows to remove certain elements of reality which will not factor in my game such as gravity and 3 dimensions this fixes problem of creating gravity-based game but also fixes limitation on what my game is capable of. However, I can break down my problem into smaller chunks which will allow me to process each fragment easier and allows me to slowly fix the problem without it becoming too complicated as I have created multiple functions and classes and by creating chunks of problems it allows me to debug the code most efficiently without confusion or interference. My game contains a few obvious decision points which will allow me to add different types of looping methods to help make it clean code, when code readable and maintainable, which will allow for less errors and problems to debug or decompose as my code will be quite large having a readable code allows me to function more efficiently and debug a lot faster by knowing what each part of the code does but also how to find the part of solution in my code when receiving a debug error.

## Research and interview

I have researched different types of strategy games which show similarities to tower defence simulators which have key element such as some sort of different entities and towers or turrets which have a health and damaging system implemented in the game, a few examples I have found which have these is Bloon’s TD 6 and Kingdom Rush.

**Kingdom Rush**

* **Strategy:**

Kingdom Rush is a Strategy game, such as my game, a game where the goal is to strategically place and upgrade items and turrets to protect a gate/final destination which has a health-based system which will lose health if an entity crosses the gate/final destination. The game has set entities with different type of health, speed and tier which have a predestined destination which is the gate have to reach. These entities are displayed on a 2D map which use x-axis and y-axis to move to each location. The map is a static image meaning it does not move however entities such as towers are can be placed in certain areas of the map to defend the fixed path the “enemy” entities have to cross. The “enemy” entities move throughout the map in walking animation by using multiple images/frames which change causing the effect of it moving in a walking animation.

* **Movement:**

In Kingdom Rush we have multiple entities which all have different version/tier which have their own abilities such as speed, health, abilities etc. Each of these different entities have a different movement speed depending on the tier of entities they are which also additionally has different animation for each character entity. The movement are built by using pathfinding which finding a path from any coordinate in the game world to another coordinate causing the entities to move throughout the predefined points.

**Enemies and Entities:**

The entities the game uses have a predefined speed and health which change throughout the game as each wave gets harder the tier of entities upgrade changing the speed, health and abilities which cause the game to become harder with time causing to keep the user engaged with the game and not get bored. We also have different tier and types of turrets which defend the redefined path by damaging and kill the entities, different turrets have different type of damage it causes to the entities, some do AOE (area of effect) damage which damage multiple entities in a certain area/range of each other. Each turret may have different speed of attack, different range that they can damage the entities or may have different ability to slow or kill the entities.

* **Storyline:**

This game focuses slightly more on the storyline and engagement with user than other TD (Tower defence) games where it mainly is just spamming down towers and upgrading them. The game has more interactivity with user adding more of a storyline to the game to make it more appealing and interesting to new users, the storyline is based in a Kingdom which is being attacked by hostile entities which are being leaded by main antagonist, the final boss. The dark wizard Vez’nan. The game creates storyline which the user is protecting the kingdom by defending the different paths throughout the Kingdom to stop the hostile entities moving and invading throughout the Kingdom.



**Bloon’s TD 6**

* **Strategy:**

Bloon’s TD 6 is strategy game but which mostly focuses on the speed and the animation/different types of enemy tiers using balloons as entities which different tiers are just different colours and may be smaller or bigger which therefore focuses less on animation of enemy entities but focuses more on animation and different types of turrets to defend the path with different maps to choose from.

* **Movement:**

The games is a faster pace game as it not only has option to speed up the game but as well throughout the game the speed of the balloons increase speed a lot and game creates user to have a higher reaction time. The balloons travel on a predefined path, which is mapped in an x-axis and y-axis using coordinates to position the balloons and other entities, which can be placed with objects to help defend the path. Balloons TD 6 focuses a lot on action in the game as balloons come different variants which have more health and are extremely fast speed which make it difficult to keep up with creating a challenge and entertainment for the users.

* **Enemies and Entities:**

Bloon’s TD 6 has many entities such as different type of turrets to defend the path throughout the map but also has different types of enemy “balloons” entities which have many different variants such switch to lower tier when attacked, these entities get higher tier with faster speed as game progresses, the different entities are prioritised in this game as it keeps game interesting and less predictable but it has basic looks of the balloons which keep the system requirements lower which is ideal for users making it more accessible to the public with lower PC specs.

* **Animations and Displaying:**

The game lowers its systems requirements as the animations and displaying the entities are more basic compared to other TD games however it utilises this processing power back into displaying more entities. Additionally, even though the balloons do not have an animation other than utilities like freezing them and destroying them they have no animation, but the game has evened this out by adding more different types of turret entities and adding more animation for each different types of turret entities.

## Evaluation of research

* **Evaluation of Kingdom Rush:**

Kingdom Rush has an interesting story base which keeps the users interested with good animation and different types of tiers of enemies and turrets but can easily get boring if the map is not field with entities at it may feel repetitive and boring as it starts to feel very slow pace and doesn’t give the user any sort of action to exciting them.

* **Evaluation of Bloon’s TD 6:**

Bloon’s TD 6 is a very fast-paced game giving the option to speed up the map making the game require a faster reaction time but also keeping the user interested but it may become easy for some people if the levels are quite simple compared to other TD games, yet the element of fast pace keeps many users interested.

## Features AND SUCCESS CRITeria

|  |  |  |  |
| --- | --- | --- | --- |
|  | Justification | Limitation | Success Criteria |
| Entity Pathing | As I do not have a distinct Entity pathing code, I had to create a function which uses predetermined coordinates to path entity to its destination | As the function I have created used predetermined coordinates it will not allow for many varieties or change to entity pathing but also means that the entity pathing might seem repetitive as of its limitation | Each entity will follow a distinct path which does not seem too basic or “straight line”. |
| Animation:  Entities and Tower | To create animation keeps user interested and allows for more visually pleasing display | Creating extreme animation is quite time consuming and will now be able to do | Some sort of Animation which keep game less basic. |

## Hardware and software requirements

The hardware and software requirements for my game consist of **operating systems of 64-bit Windows 7 or later or OS X 10.11 or later** which will allow the game to be run without outdated operating systems which may prevent this. We require **processor of at least 1.5GHz or faster** and **memory of 4GB (4,096MB) RAM** with a **Free HDD space of a minimum 3GB** to allow for all code, recourses, images and more. As of **software requirements** we need device to have **python** downloaded and **pygame library** installed.

OS: 64-bit Windows 7 or later or OS X 10.11 or later

Processor: 1.5GHz or faster

Memory: 4GB (4,096MB) RAM

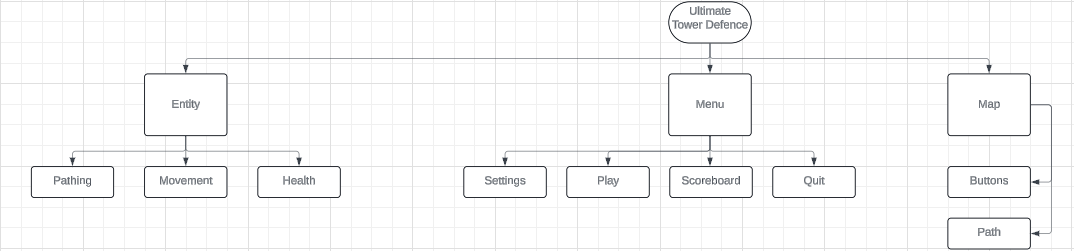
Free HDD space: 3GB

Hardware: keyboard and mouse

Software: Python, Pygame Library

# B. Design

## Structure diagram



## sUMMARY OF PROCESS

|  |  |  |
| --- | --- | --- |
| Main Branch: | Sub-Branch Component: | Component Explanation: |
| Entity | Pathing | In tower defense games, "pathing" refers to the predefined routes that enemies, often represented as balloons, take as they traverse the game map. These routes dictate the trajectory from the entry point to the exit, shaping the strategic placement of defensive towers. Understanding the pathing is essential for players to optimize their tower placements effectively, ensuring maximum coverage to intercept and eliminate enemies before they reach the exit. Balloons may follow varied paths, including straight routes or more intricate trajectories looping around obstacles, necessitating adaptable strategies to counter them. Mastery of pathing dynamics is crucial for success as players progress through levels, facing increasingly challenging waves of enemies. |
| Movement | In tower defense games, animation encompasses the dynamic visual representation of in-game elements like enemy movements, tower actions, and special effects, contributing to the immersive gameplay experience. These animations convey crucial information to players, such as the progression of enemies along their paths, the actions of defensive towers as they attack, and the impact of upgrades or abilities. By providing visual feedback on player actions, such as the destruction of enemies or obstacles, animations enhance gameplay engagement and communicate essential mechanics effectively. Overall, animation plays a vital role in enriching the gaming experience by bringing the game world to life and facilitating player interaction with its elements. |
| Health | In gaming, entity health, such as that of balloons, shows how much damage they can handle before getting destroyed. Balloons differ in strength based on size, color, and type. Players need to place defenses strategically to weaken balloons and stop them from advancing. Knowing balloon health is key to winning. |
| Menu | Play | Interactable button that allows the player to access the core game of the project. Players are presented with several options about the game’s settings when they interact with the play button. |
| Scoreboard | The Interactable button allows the player to access a scoreboard, including 10 names with the highest points achieved, and their associated points. |
| Settings | Allows the user to modify several aspects of the game:  Resolution – change the game’s screen size. Can also pick window or full screen.  Volume – Alter the volume of the game’s sound effects and music  Key binds – Can change what key does what (alter the inputs), which is a quality of life (QOL) improvement |
| Quit | A quit button is added to the menu screen, allowing the player to exit the program. |
| Map | Path | Different textures applied to the terrain (and player) that both make the game more visually appealing but can indicate where the player can and cannot go. |
| Button | Added buttons which allow for unique abilities and different options |

## Key variables and data structures

|  |  |  |
| --- | --- | --- |
| Identifiers: | Type: | Justification: |
| game | function | Holds the main program of the game. When set to True, the program and all its functionalities will run and be accessible to the player. When the program is closed, or the ‘Quit’ button is pressed on the menu screen, gameLoop will be set to False. |
| mainMenu | function | When True, the main menu will run. Here, the player may access several functionalities of the program. When interacting with the ‘Play’ button, main Menu is set to False and the game continues to the gameplay part of the program. |
| Player’s x and  y-coordinate | integer | The player’s x and y-coordinate are constantly worked out, allowing for collisions with the environment and a moving sprite mapped on the player. |
| Platform’s x and  y-coordinate | Integer | The platform’s x and y-coordinate are set. When the player lands on it, the program may calculate if they successfully landed or not. |
| score | integer | The player’s score is calculated and stored until the player loses all their lives, the player may input a name, this name being tied to their score and is stored within a text file. If the score is high enough, it will be viewable on the scoreboard. |
| time | float | Time is a constant throughout the game; however, it may be calculated as a real value for a more precise time modification. |
| Window width and height | integer | Important measurement as it adjust the height and width of window displaying game which could cut out important visuals . |

## Class diagrams

|  |  |
| --- | --- |
| Entities Class | Tower Class |
| Attributes:   * Image – spacecraft image * Width * Height * Initial x pos – where the player starts on the X-Axis * Initial y pos – where the player starts on the Y-Axis * Tier – difficulty of entity * Lives – entities total number of health * animation | Attributes:   * Tier – an array that stores all Tiers generated by upgrades |
| Methods:   * Get\_hit() - rotate image * Update() - update various things per clock tick (e.g., sound on or off) * died() - draw image onto screen * Animate\_on()- adjust if animation is on * Animate\_off() - adjust if animation is off | Methods:   * arrow() - generates the terrain by creating points on the screen * Animate\_on()- adjust if animation is on * Animate\_off() - adjust if animation is off * update() - update various things per clock tick (e.g., sound on or off) |

## development test data

|  |  |  |
| --- | --- | --- |
| Test | What is being tested? | Why am I testing it? |
| 1 | The player will input keys to access different areas of my game. These will be tested to ensure these work as intended. | A users initial experience to a game is particularly important and that is no different to my project. If a player were to launch it only to be stopped at the title screen, then that player will be lost instantly. A faulty menu also means players wont be able to access my game, thus this must be tested to ensure it works as expected. |
| 2 | Whether the Entities sprite moves as intended. | The sprite is a visual identifier to the Entities position. As such, it must update every clock tick to ensure that the player knows exactly where they are. On top of this, the sprite is acts to improve a players experience as it overall improves upon and refines the quality of a game |
| 4 | Whether the Entitles sprite collides with the Tower arrow or just phase through them. | If entities were to just phase through tower arrows there wouldn't be way to kill the entities to face or a win condition for the player to meet, eventually they would run out of health and lose the game indefinitely giving the game no purpose. This must be working as intended to prevent this. |
| 7 | The map and Entities should be rendered onto the game | Both the Map and Entities should be rendered as then they act as a visual guide to both the win condition and obstacles. If they were not rendered, the player would have no indication of where to go, thus end up most likely losing. |

## aLGORITHMS

## post development test data

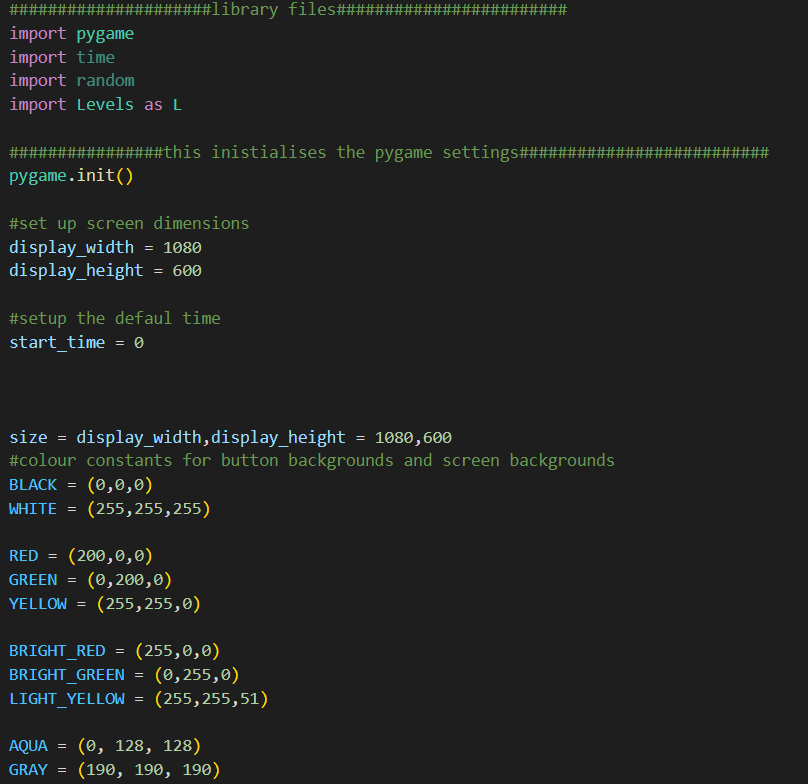
# Developing the coded solution (“The development story”)

## WEEK 1:

### GOAL:

My goal for the first week will be focusing on creating the initial layout of the code, creating the main menu and creating the most reusable variables. I will also spend time creating my entities and main menu display as I will design them myself. I will also need to set up and install python and pygame which are my coding language this game will be made in and the python library which will help create this project.

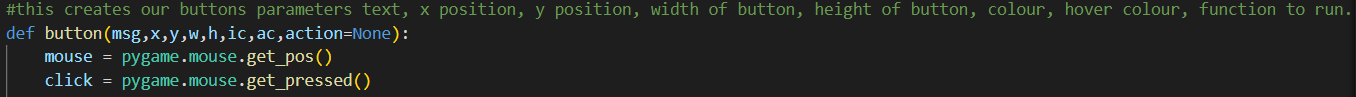
### PROGRESSION:



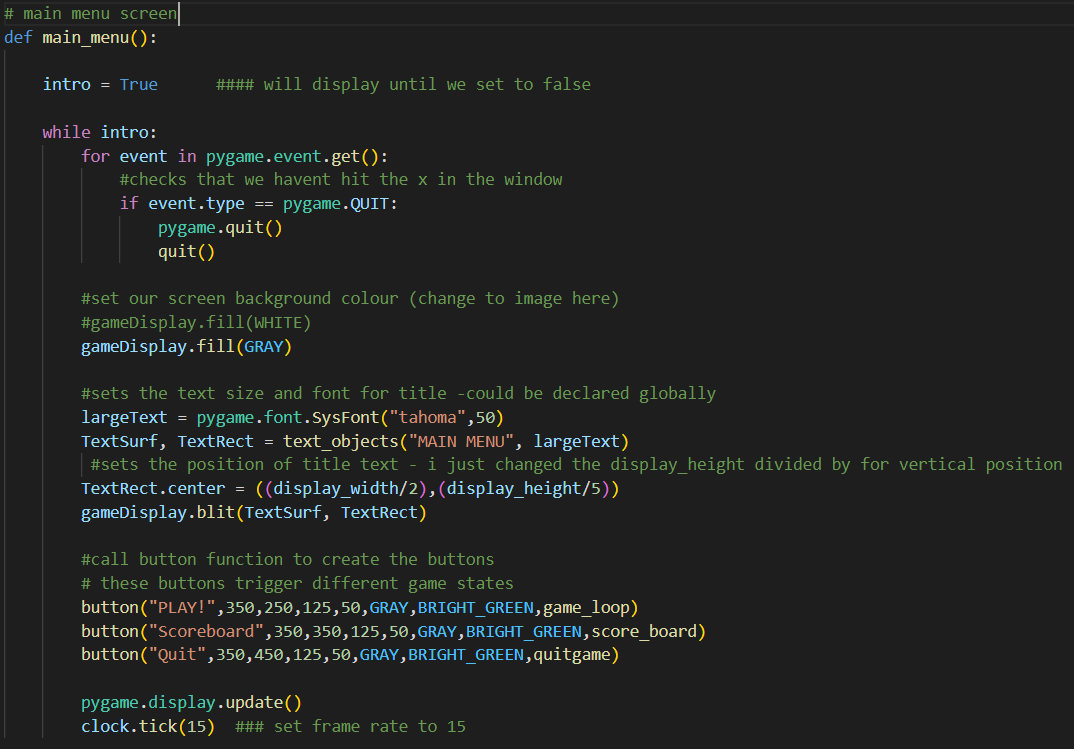
To begin with I will start by importing some python libraries which will be needed for this project such as pygame, time, random and importing my second file for the code as I will spit it up in 2 files to further make my code more coordinated and more readable. I require these libraries to as my game is based of time of when each wave spawn in the game and how much currency will be given as it is given per second alive. Pygame is a fundamental library which is needed to make this game possible as it will help with pregiven functions which can be used to make my game work functionally and allows for the game visuals to keep updating itself to do this as it is also allows us to change the width and height, colors and more of the games.



This line is set to display the window size of the game using the given width and height of pixels.



I have created this small function to create a button using thee width and height of the button, message colour and function however I will amend this function so it will use images rather than text as I believe it to be more visually pleasing.

This function is used to run the main menu and to display the different options for the main menu such as to start game, scoreboard or to quit, these buttons have their own size and colours and function that they are linked to. The main menu function is run at the beginning of the code to allow the user to choose what they will do.



This is the state of my game after the 1st week

### TESTING:

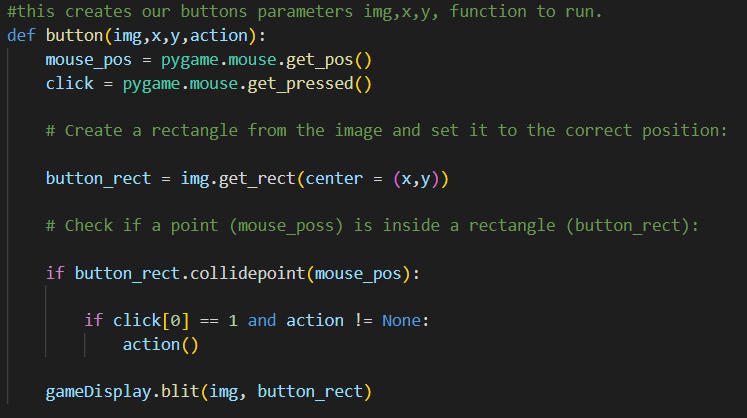
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No. | Description | Expected Outcome | Justification | Actual Outcome | Actions required |
| 1 | Test if pressing on menu will start the game | The game should start and pop up immediately | Gives users time and options between starting the game or closing the game | Successfully started the game | N/A |

## WEEK 2:

### GOAL:

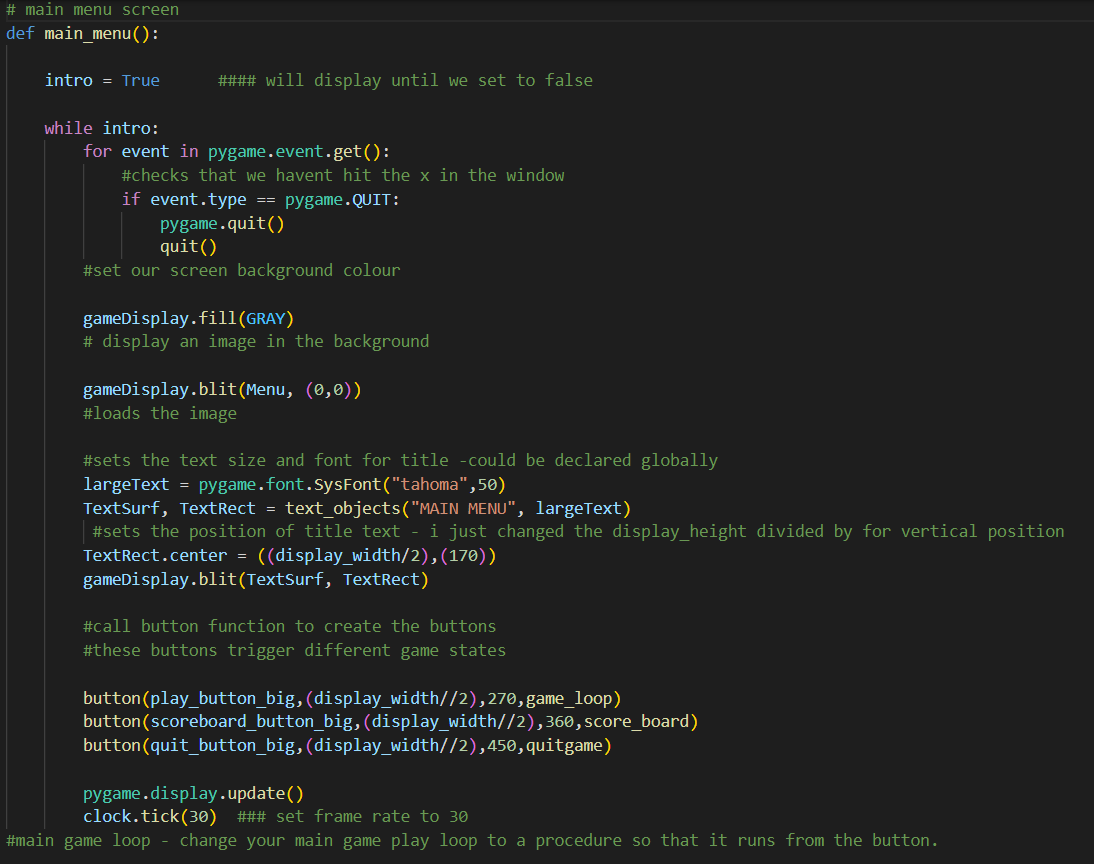
Week 2 will be focused on creating the visuals of the main menu and amending some of the functions.

PROGRESSION:





To create my main menu I wanted to redesign a more complex and visually pleasing main menu as I wanted to do this I wanted to use images I designed as the buttons so I had to recode this function to be suitable for images instead of text which made it more complex.



As i created these changes I updated my main menu function as well and added a font to make this less basic.



This is the state of my game after the 2nd week

### TESTING:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No. | Description | Expected Outcome | Justification | Actual Outcome | Actions required |
| 1 | Test the new button function worked | Buttons will respond to when clicking on the image | As the buttons are images it was important that updating such factor didn't break the code | Function worked as expected | N/A |
| 2 | Test all the buttons go to directed function | Buttons will direct the game to either start of the game, Scoreboard or to exit the game | The buttons functionality is vital as they are required to start the game or to quit it | All buttons worked as supposed to | N/A |
| 3 | Make sure all images are set up correctly | All images are right size and place | Without testing the buttons may have been misplaced inside another making them unable to see or click | All visuals worked | N/A |

## WEEK 3:

### GOAL:

I will be focusing on creating the frames for the animation for the entities and all other images as I wil need them when creating the class for the entities and their movements.

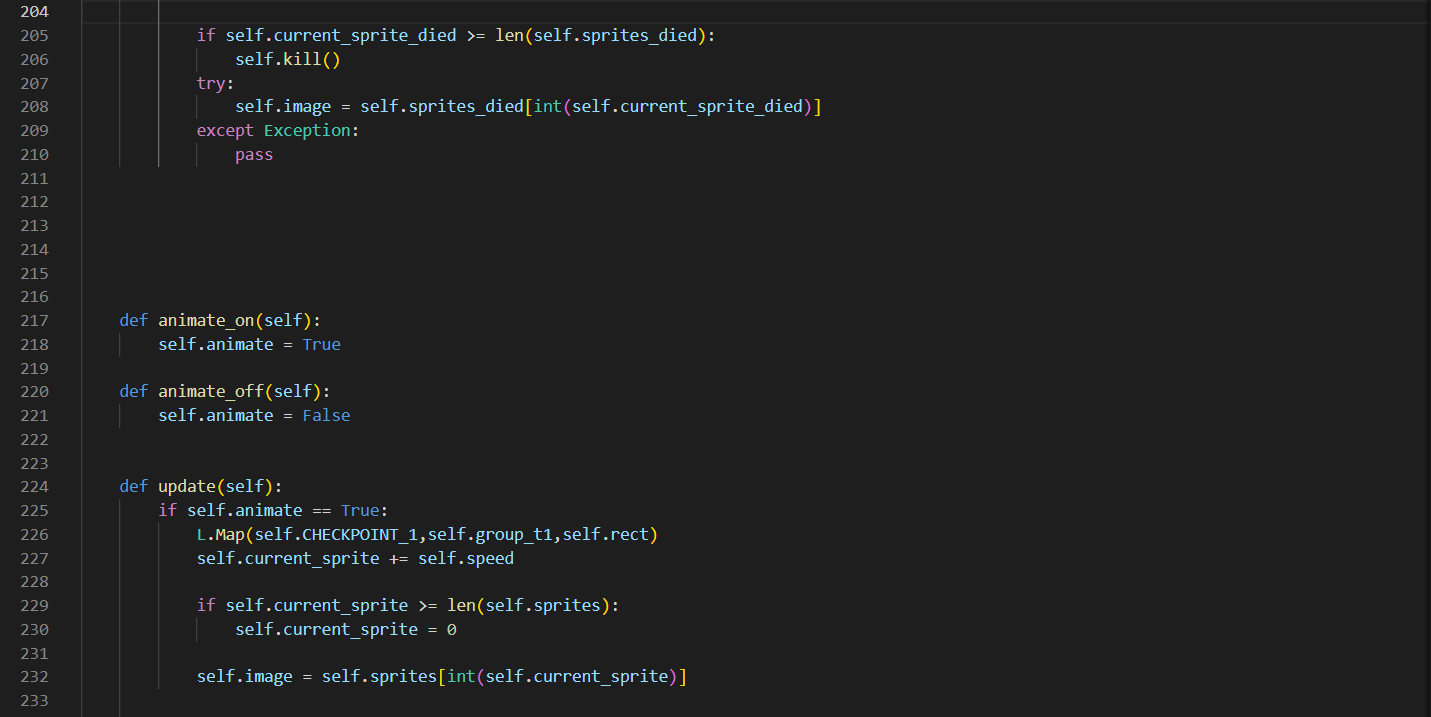
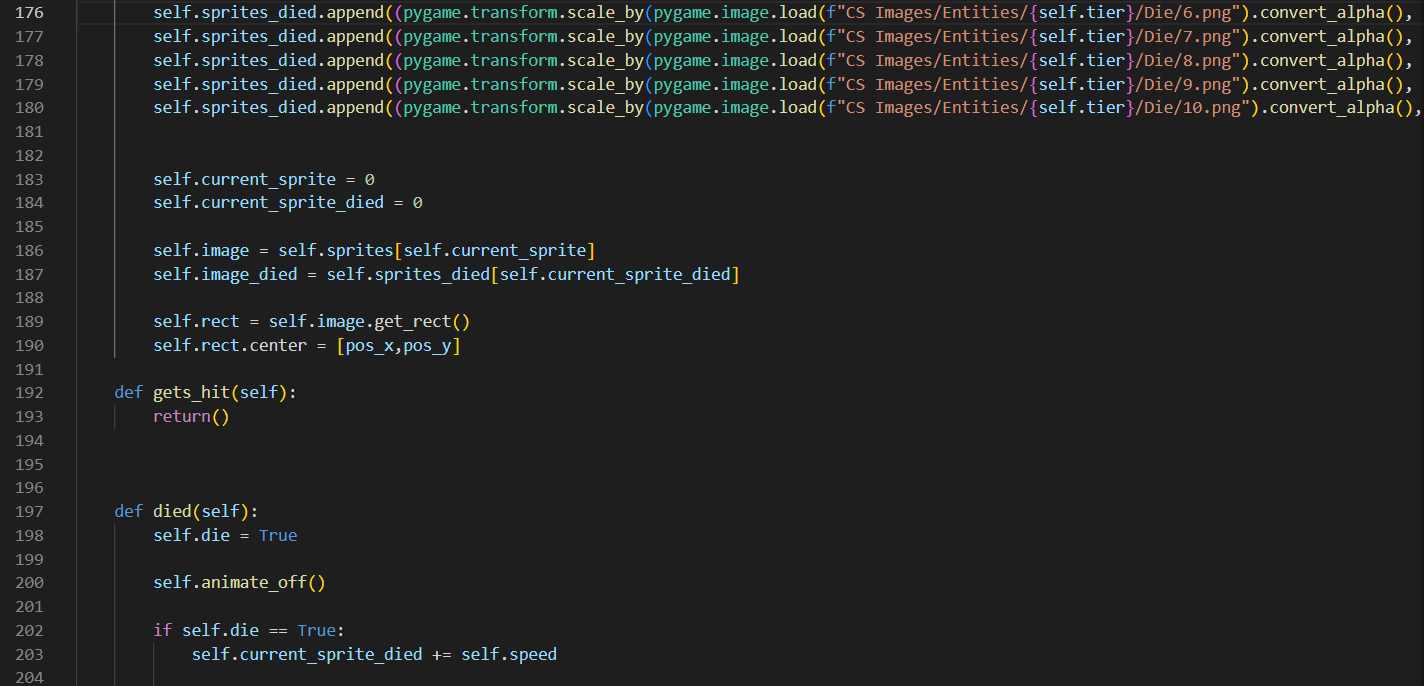
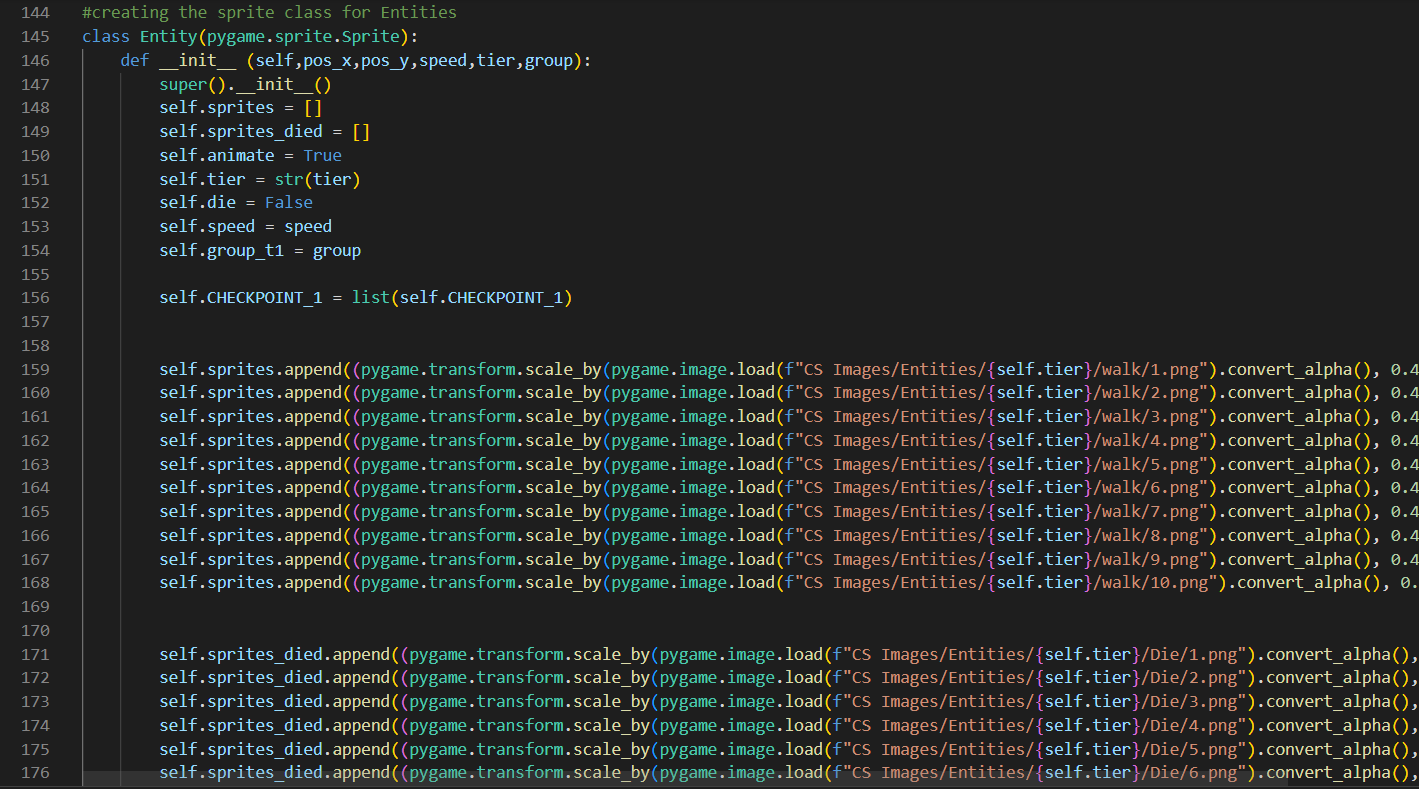
PROGRESSION:



I used a gif of these entities walking and started by cutting down each frame into a png also removing their white background to stop any extra background overlapping mine and creating and naming one correctly so each frame can be used to make animations.



In this code snippet, I have created a class Entities which will act as a blueprint for my Entities which will be moving around the map. On the first couple lines I have created base variables for this Class as I need it for animation and more as otherwise it will needed to be imputed each time. I created a list which will be needed as each image I add needs to be looped in a specific order which I will all insert into a list and displaying each image into the class.

The next couple functions in this class are for the animations of the movement of the Entities such as the death and moving functions which are vital to make these animations. I also made a class called update within which is main function which runs and controls that each frame is played in order.

### TESTING:

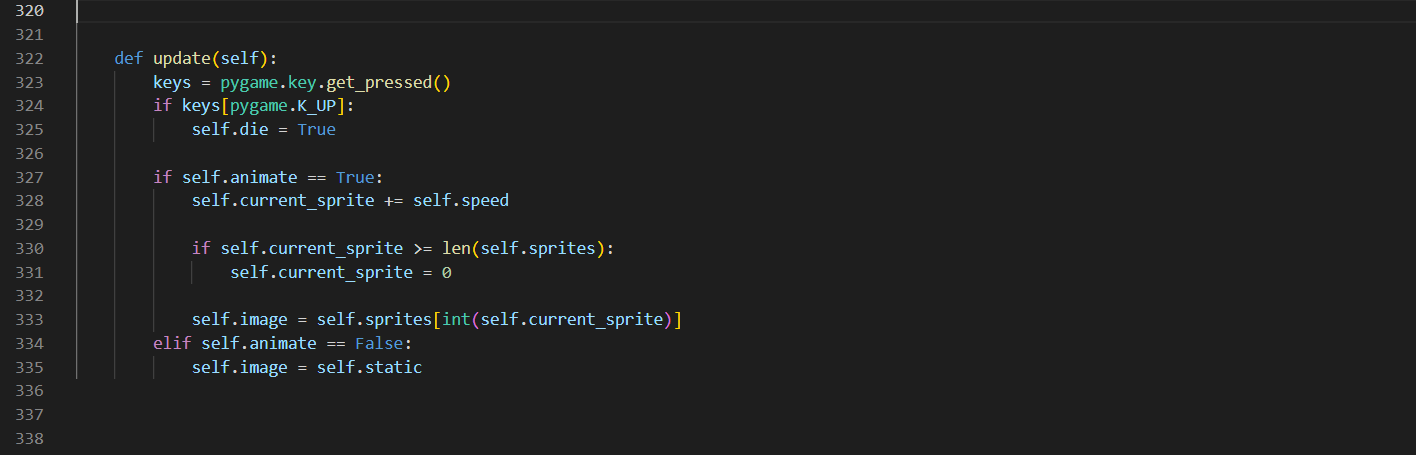
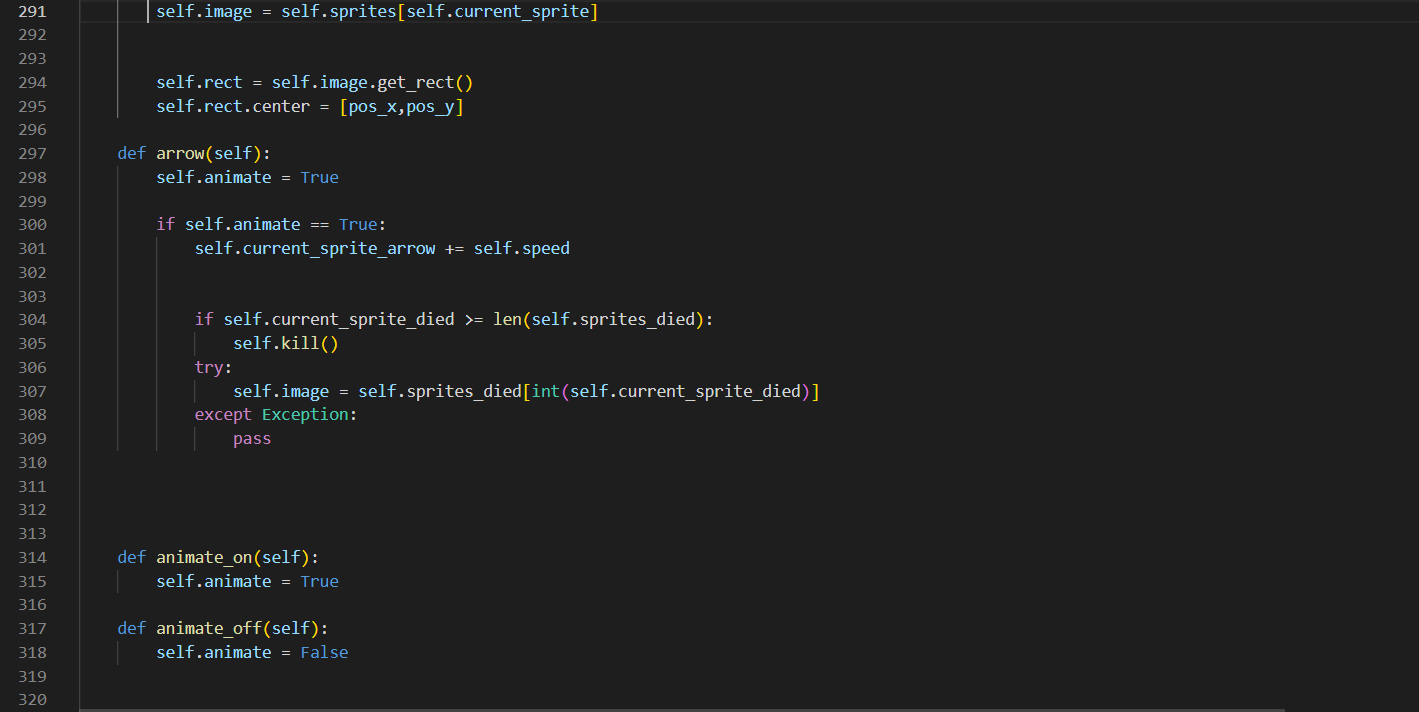
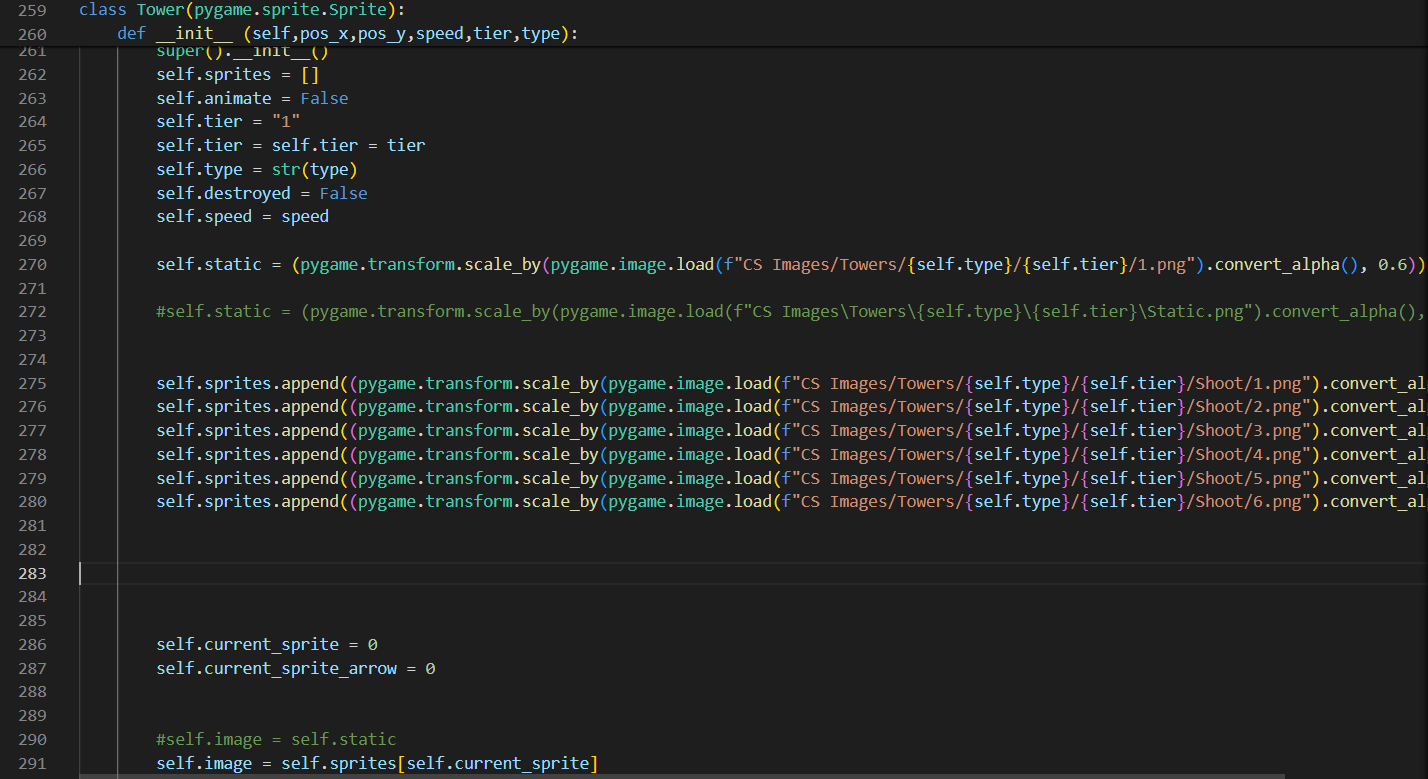
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No. | Description | Expected Outcome | Justification | Actual Outcome | Actions required |
| 1 | Check each frame is in order | Each frame will be in order creating a stable animation | Without the frames being in order the animation will not be as displayed movement | Frames did not change | Make sure to adjust the frames order |
| 2 | Fix frame order | Frames will be in correct order displayed | The frames did not display so fix is needed | Frames in order after fixing frames are being updated | N/A |
| 3 | Test Entity is displayed | Entities are displayed in correct size | If size is incorrect the Entities may be massive or small which will take up the window | Entity spawned in at correct size | N/A |

## WEEK 4:

### GOAL:

The goal of the 4th week is create Tower class and start making them be placed on the map

### PROGRESSION:



Similarly to how I created Entities the week prior, I first created a class for the Towers. A function is written that creates platforms on the map which will be interactive and allow for me to place my Towers, this Tower class is very similar to the Entitles class however it is animation does not require it to move around map but at platforms which are around the map interactive allowing to place or build a tower which will shoot arrows at entities, this will require us to make a function called arrows which will be in charge of colliding into the entities lowering their health and causing them to die which will active their died() function.

The towers are quite important to the program as they are cause of many functions run in code such as the platform(), died() , arrow() and a lot more which cause this games algorithm.



This is the state of the game after the 4th week

### TESTING:

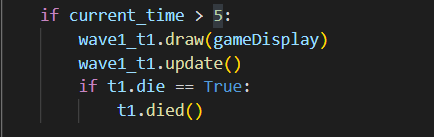
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No. | Description | Expected Outcome | Justification | Actual Outcome | Actions required |
| 1 | Verify that the platforms are drawn onto the screen | An image of platform on map which is interactive | Platforms are only slots towers can be placed on | Platforms successfully placed around the map | N/A |
| 2 | Platforms interact to place Tower | Towers are placed when clicking on platform | Platforms need to be able to build the Towers otherwise won't be any Towers | Towers were placed once clicked on platform | N/A |

## WEEK 5:

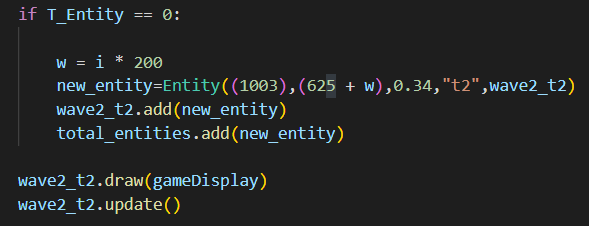
### GOAL:

This week I plan to add some tweaks to improve and refine the game. Things such as adding a fuel gauge or adding the players lives left on the screen. I also want to reset the player and terrain when the player either crashes or lands on the landing zone. A final tweak I want to make is to make the sides of the screen loop so the player will loop from the left of the screen to the right and vice versa.

### PROGRESSION:



This code is used to create a wave of entities and checks for their death which is important as we do not wish for multiple waves push at 1s as they will be too much for the user to kill at ones and create multiple issues.



To create entities waves I had make sure each entity is spread across correct gap so they don't overlap and cause visual unpleasantries.

In the development of "Ultimate Tower Defence," strategic spacing emerged as a critical technical consideration shaping the game's mechanics. Beyond mere aesthetics, it profoundly influences gameplay balance and player engagement. From the outset of development, we recognized the importance of implementing robust systems to manage wave distribution, tower placement, and tier upgrades.

Strategically spacing out waves required the implementation of sophisticated algorithms to ensure a dynamic and challenging experience. Balancing the frequency and intensity of waves while allowing players sufficient time to strategize and reinforce their defenses was a complex technical challenge. We leveraged various techniques, including wave scheduling algorithms and dynamic difficulty scaling, to achieve this delicate balance.

Furthermore, optimizing tower placement involved the development of precise grid-based systems to facilitate strategic positioning. We implemented algorithms to calculate optimal tower placement based on factors such as coverage area, attack range, and terrain obstacles. Additionally, we incorporated pathfinding algorithms to enable balloons to navigate the game map efficiently, ensuring that tower placement remained impactful.

The tier system introduced another layer of complexity, requiring careful consideration of resource management and upgrade progression. We designed a hierarchical upgrade tree, with each tier offering increasingly powerful abilities and enhancements. Implementing this system necessitated robust data structures and algorithms to track player progress, manage upgrade availability, and ensure a smooth and intuitive user experience.

In summary, strategic spacing in "Ultimate Tower Defence" is not just a gameplay feature but a technical achievement. It underscores the intricacies of game development, requiring the implementation of advanced algorithms and systems to create a balanced and engaging experience for players. Through meticulous planning and technical innovation, we were able to realize the vision of a dynamic and challenging tower defense game that continues to captivate players worldwide.

This is the current state of the game after the 5th week



### TESTING:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No. | Description | Expected Outcome | Justification | Actual Outcome | Actions required |
| 1 | Verify the background successfully loads | There should be a background for the game | Improves the quality of the game. | The background successfully loaded | N/A |
| 2 | Wave of entities load successfully | Their should be multiple entities spawning at gaps | Need waves to be spread out for no collision | Wave is correctly spread out | N/A |
| 3 | Tower is shooting arrows | Tower is animation is shooting arrows at the entities | Tower detects enemies | Tower animation correctly loops | N/A |

# Evaluation

## Post development testing

### MAIN MENU:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test | Expected Outcome | Justification | Conclusion |
| 1 | To verify image is loaded onto the main menu screen | Image should be seen on the main menu | Introduces players to the game, and gives instructions to said players | image is successfully loaded onto the main menu screen |
| 2 | To verify that a background image is loaded when the game is launched | There should be a textured background to the main menu screen | Improves upon the refinement and quality of my game. | A background was successfully loaded on launch |
| 3 | Pressing “Play” will start the main game part of my project | The menu should close and the game should start | The player has to have a way to access the gameplay part of a game | The game runs as expected when “Play” is pressed |
| 4 | Pressing “Quit” will close the program | The program should be closed when Quitis pressed | Allows the player to exit the game with ease | The game successfully closed upon entering “Quit” |

### GAMEPLAY:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test | Expected Outcome | Justification | Conclusion |
| 1 | Clicking platform will build a tower | The platform and hitbox for the platform will build tower | Allows the player to move the ship around the screen when used in tandem with the thrusters | Tower was build after clicking on platform which are spread throughout the map |
| 2 | The entities follow their given path and are spread out correctly | Entities followed their path correctly | Used for movement of entities | Entities successfully traversed throughout the map |
| 3 | The entities should lose health when hit by arrow | Health went down when hit by arrows | Without the entities losing health the user will automatically lose | The entities died ones health reached 0 |
| 9 | A background should be loaded during the main game loop | There should be a space background during gameplay | Improves upon the polish of my game, as it is nicer to view then a screen filled with a set colour | The background was successfully loaded when |
| 10 | The score should be added to each second survived | The score should be reset dying | Allows players to compete for a high score | The score successfully increments to each second survived |

## Usability testing

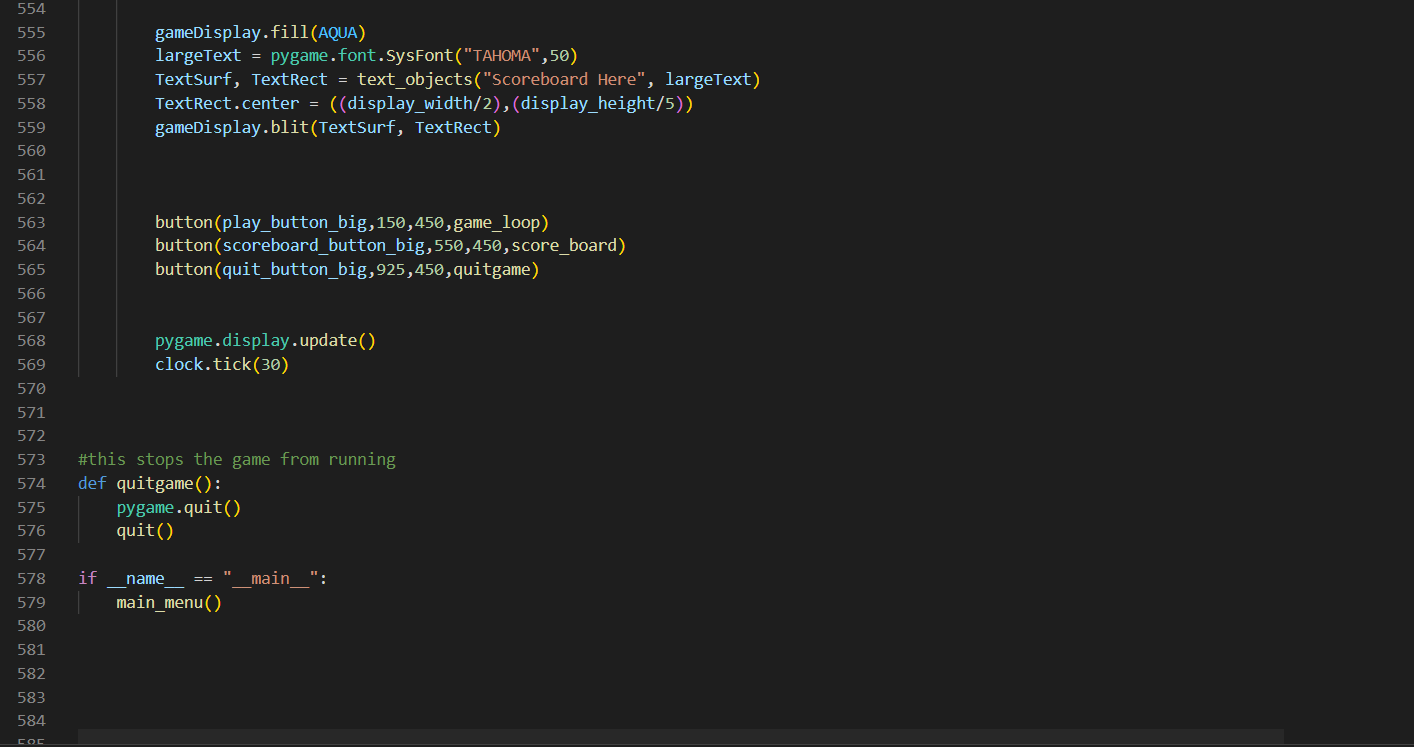
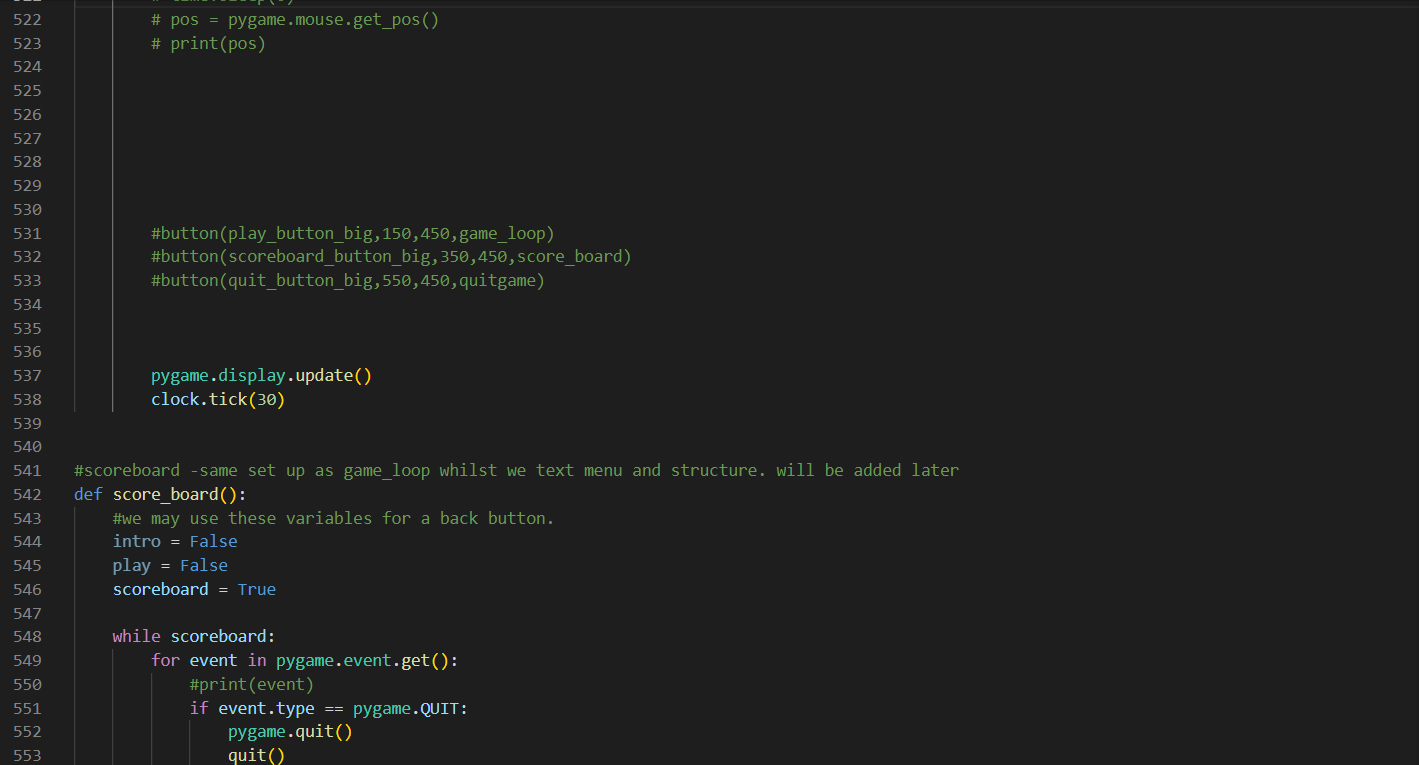
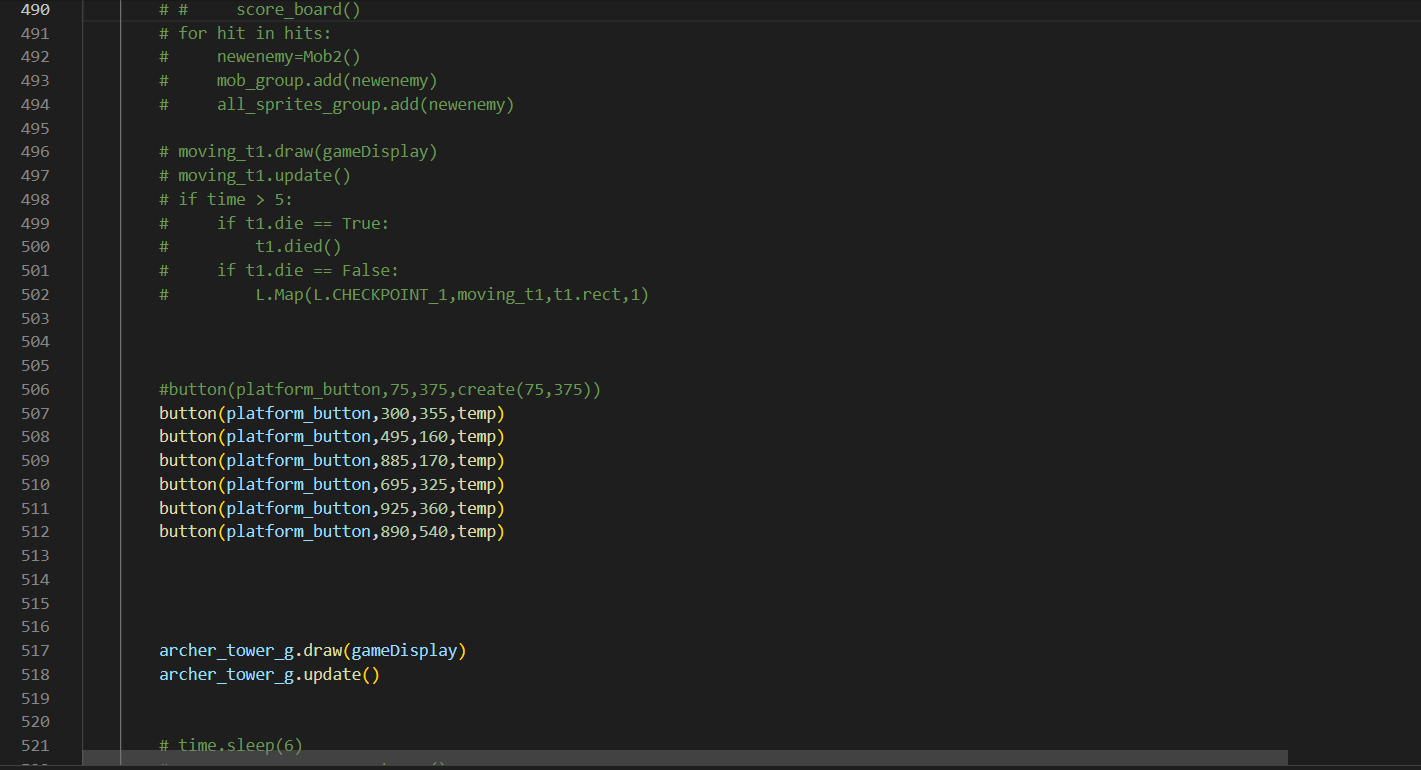
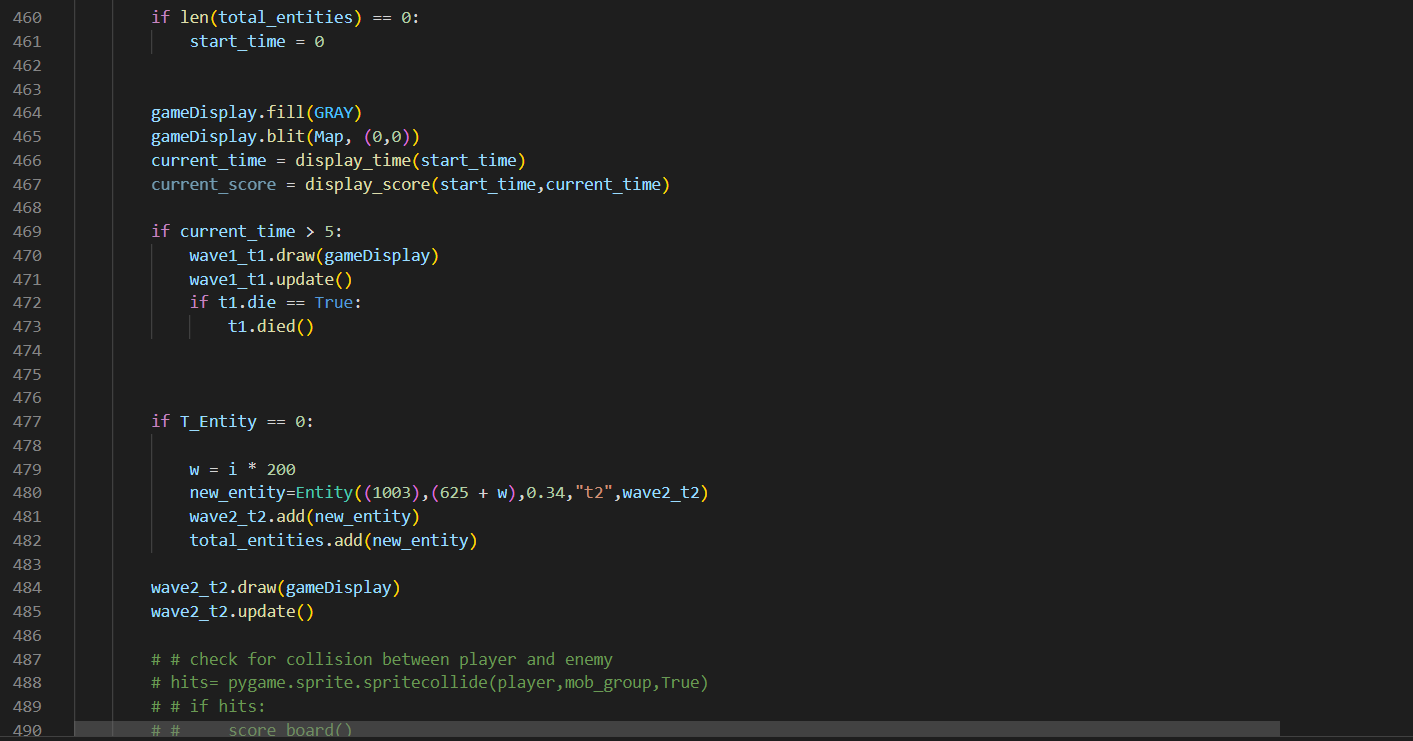
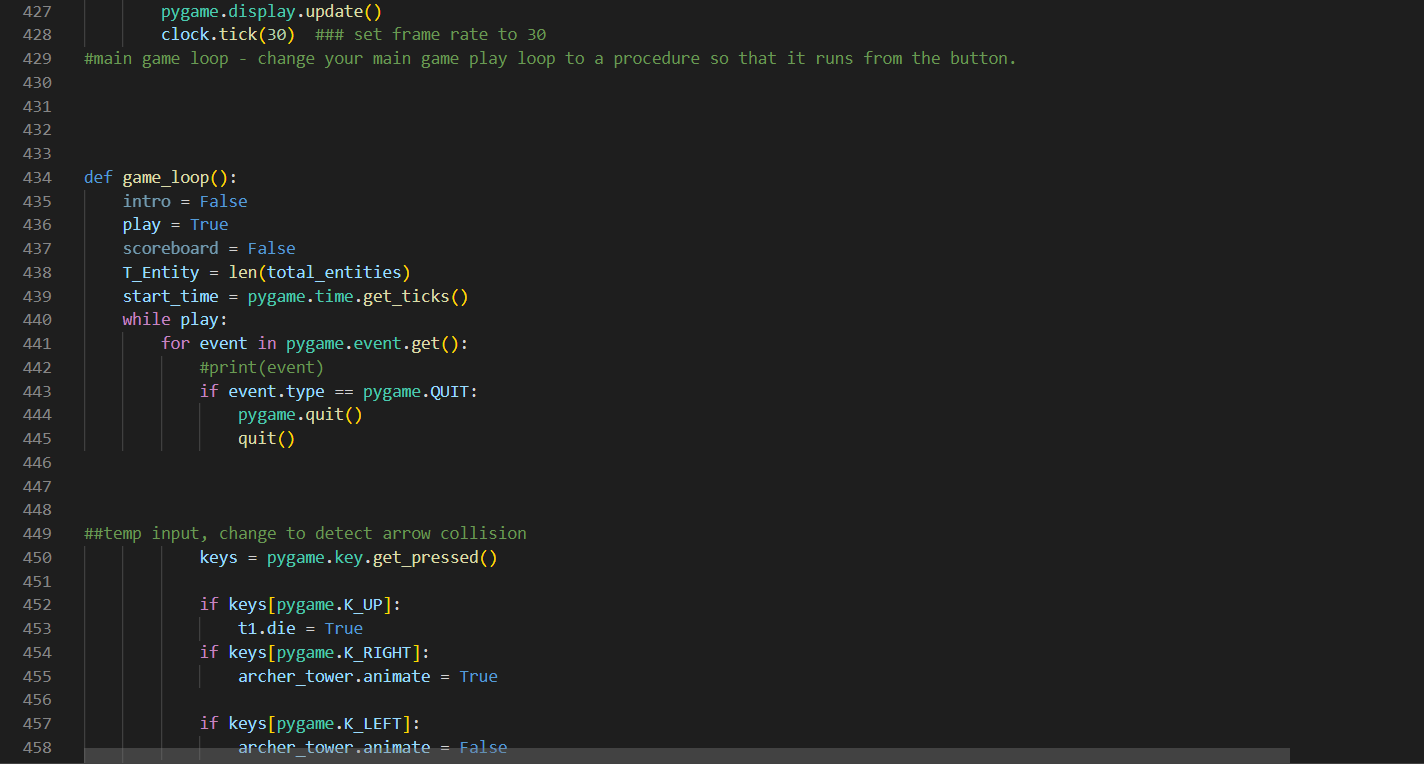
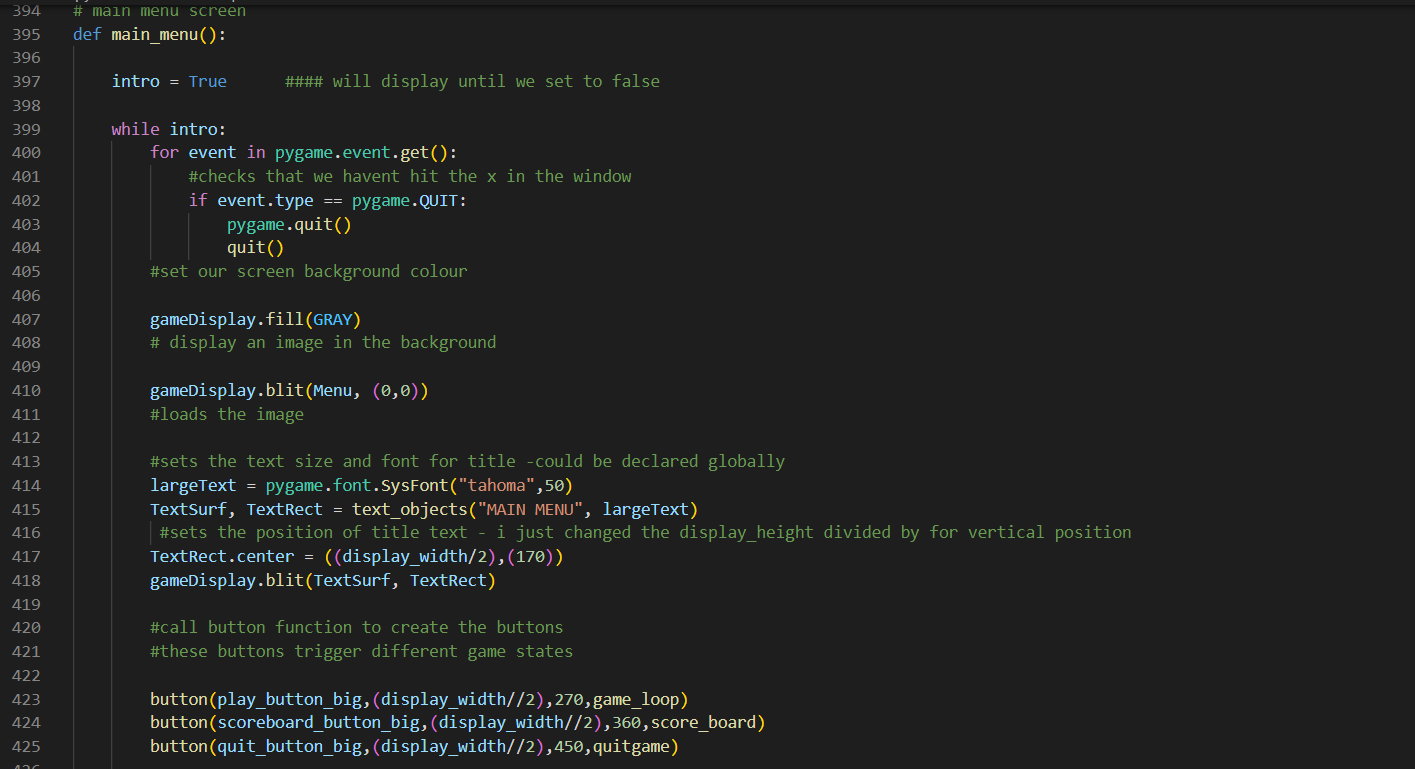
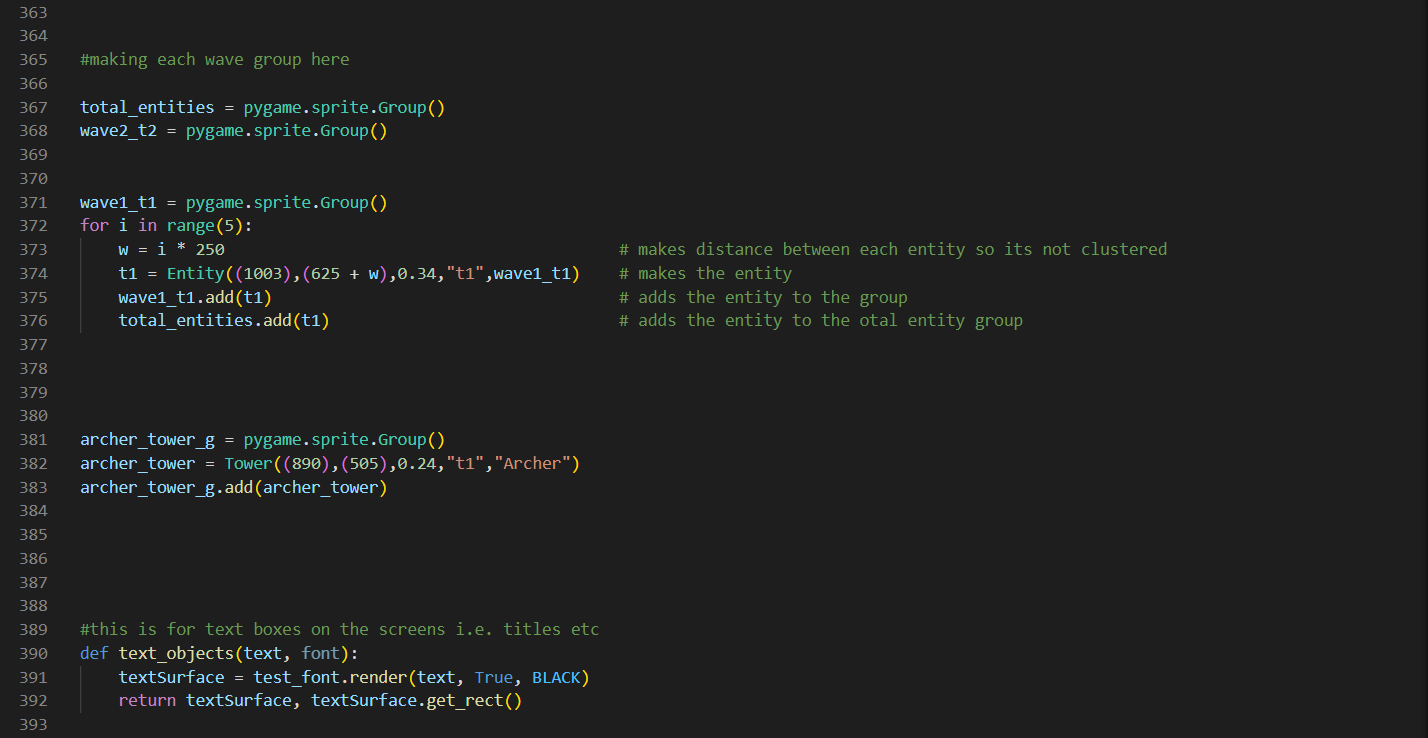
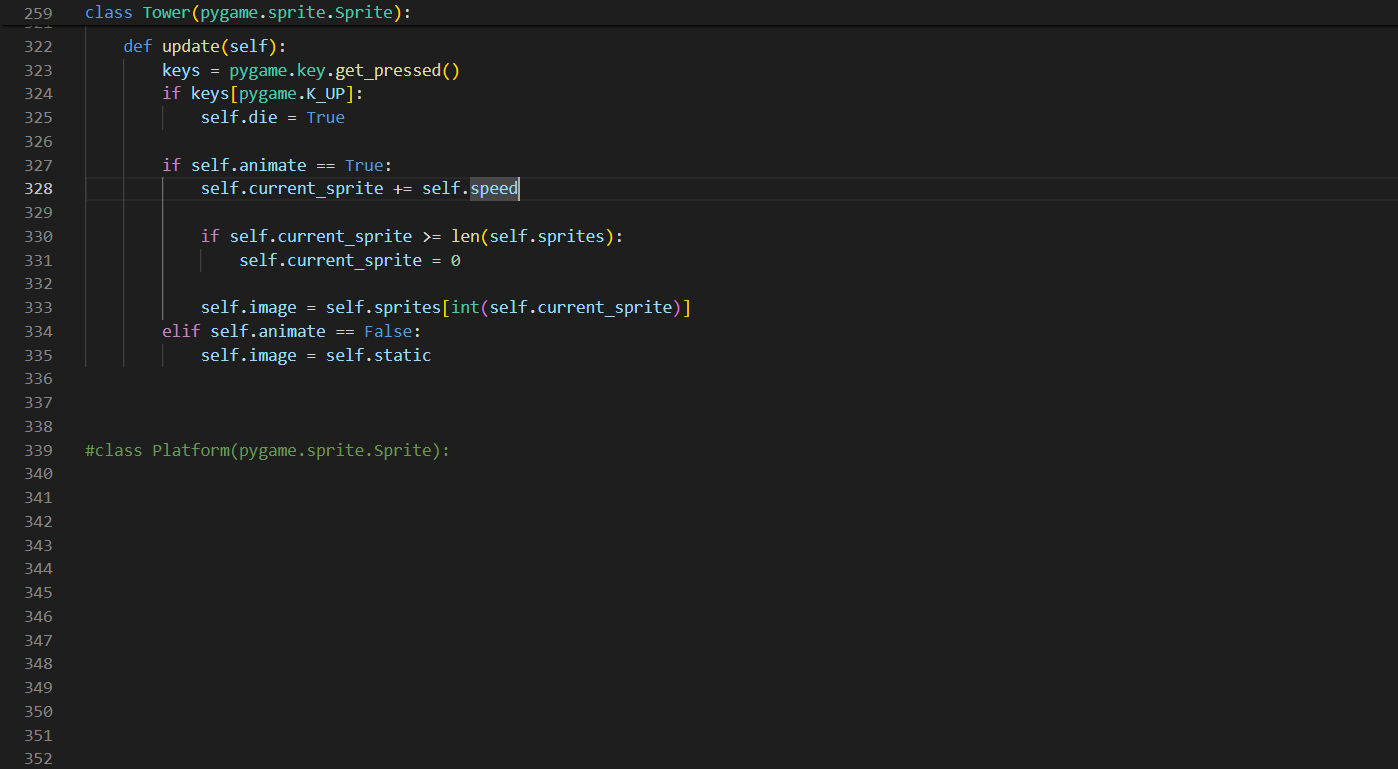
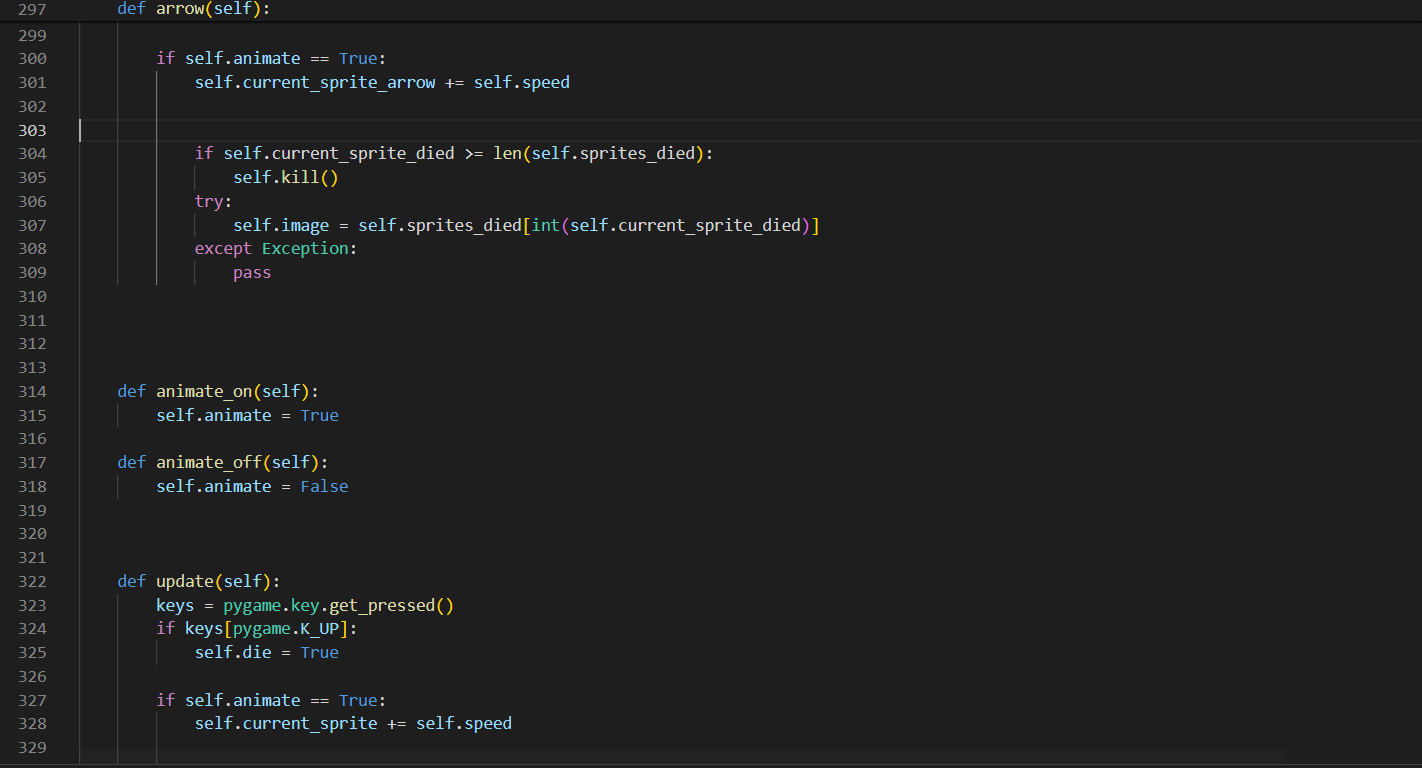
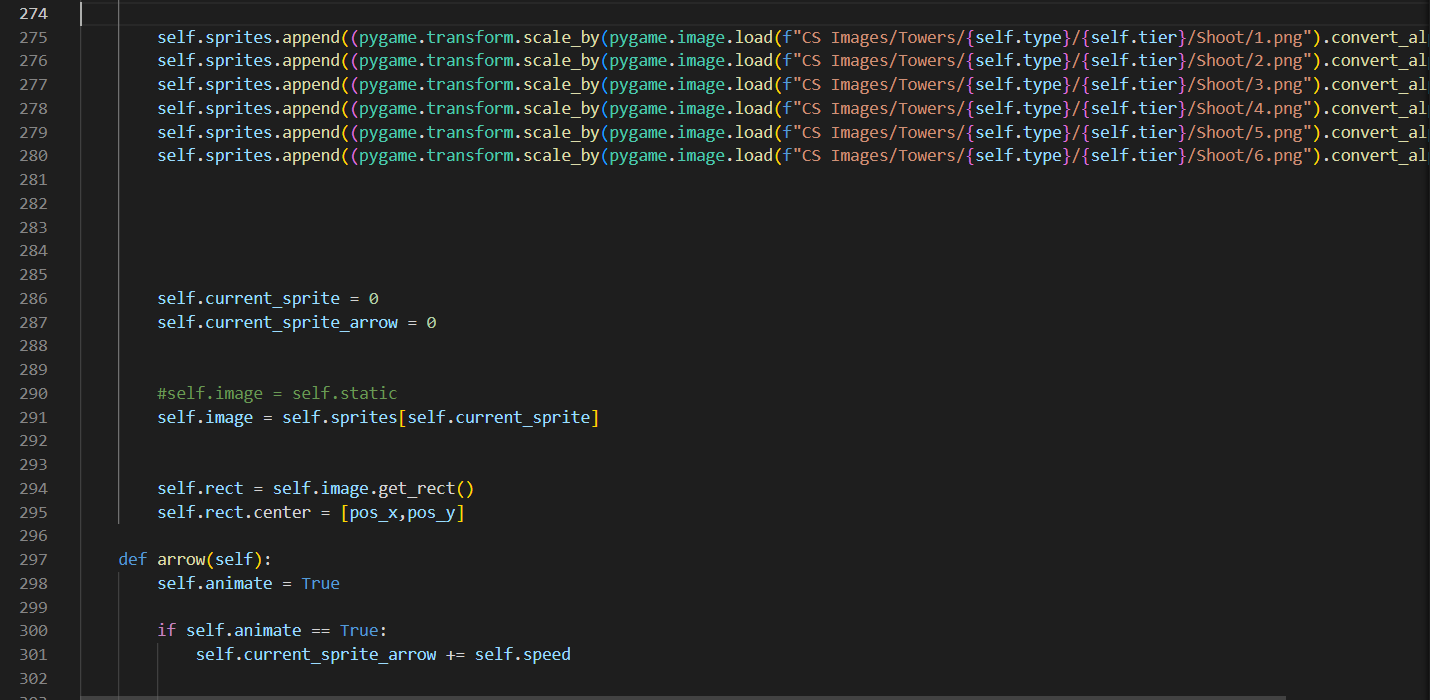
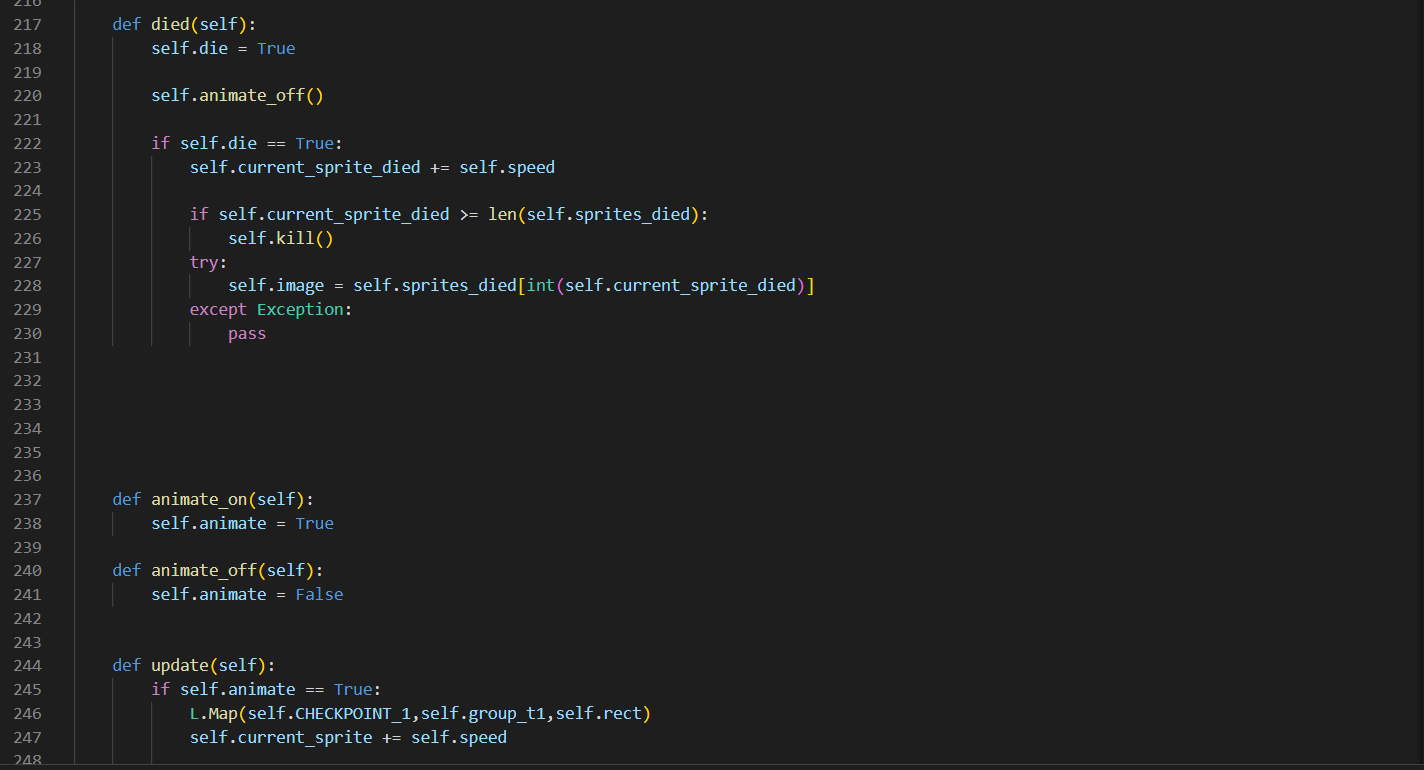
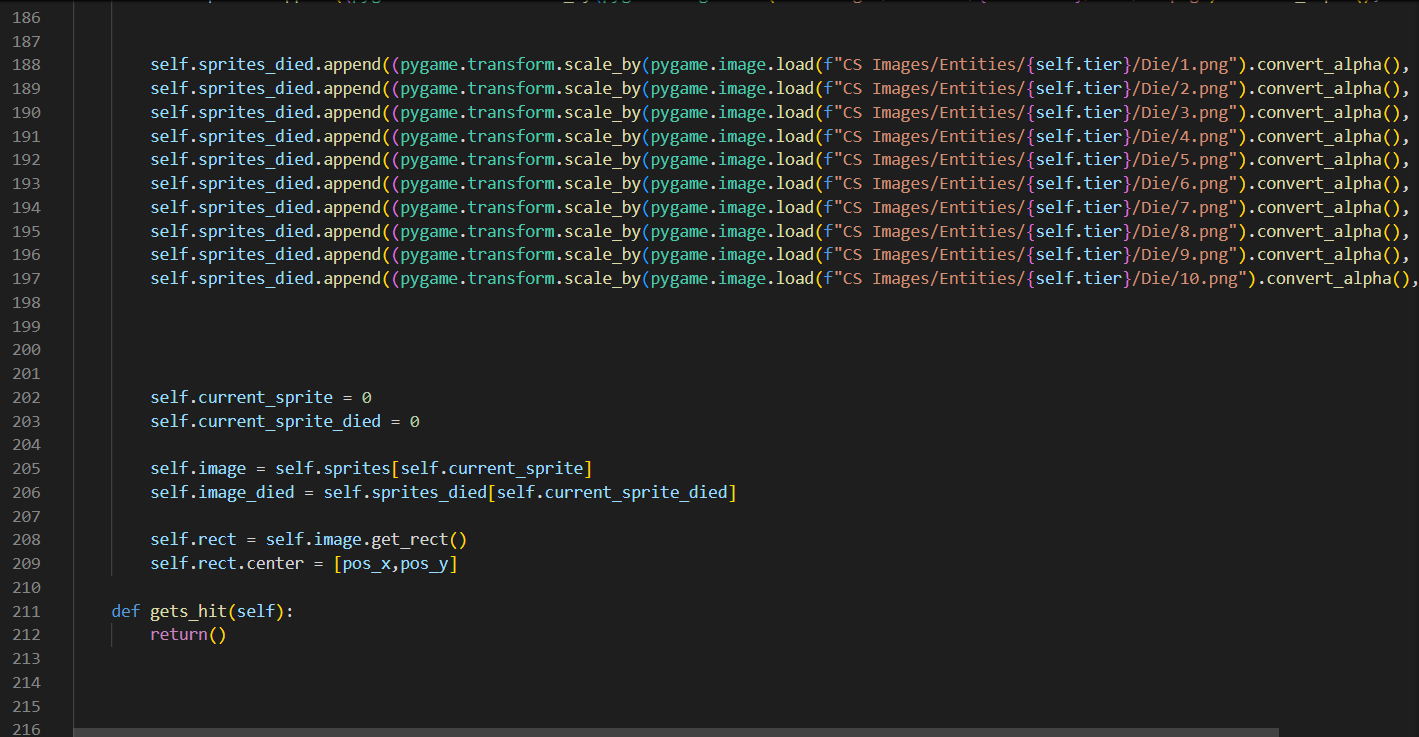
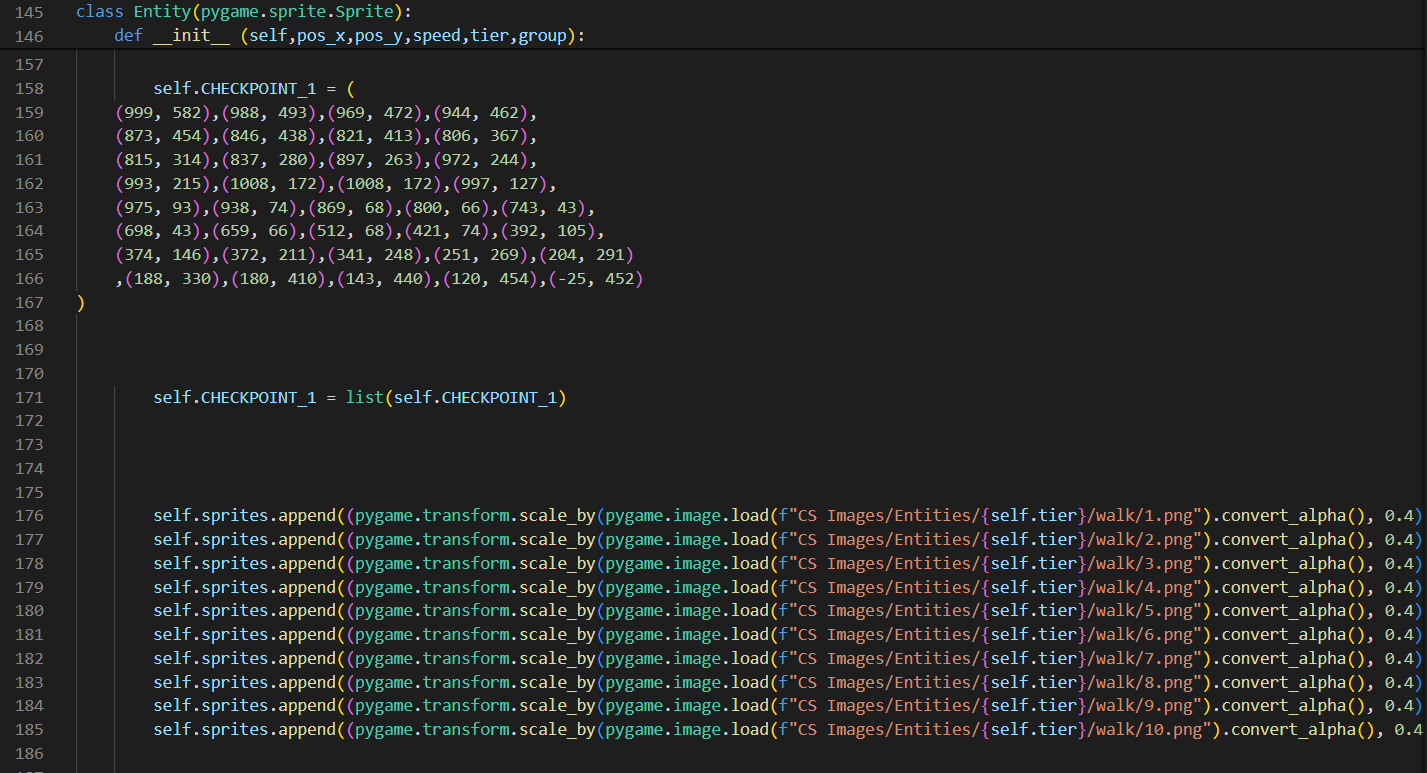
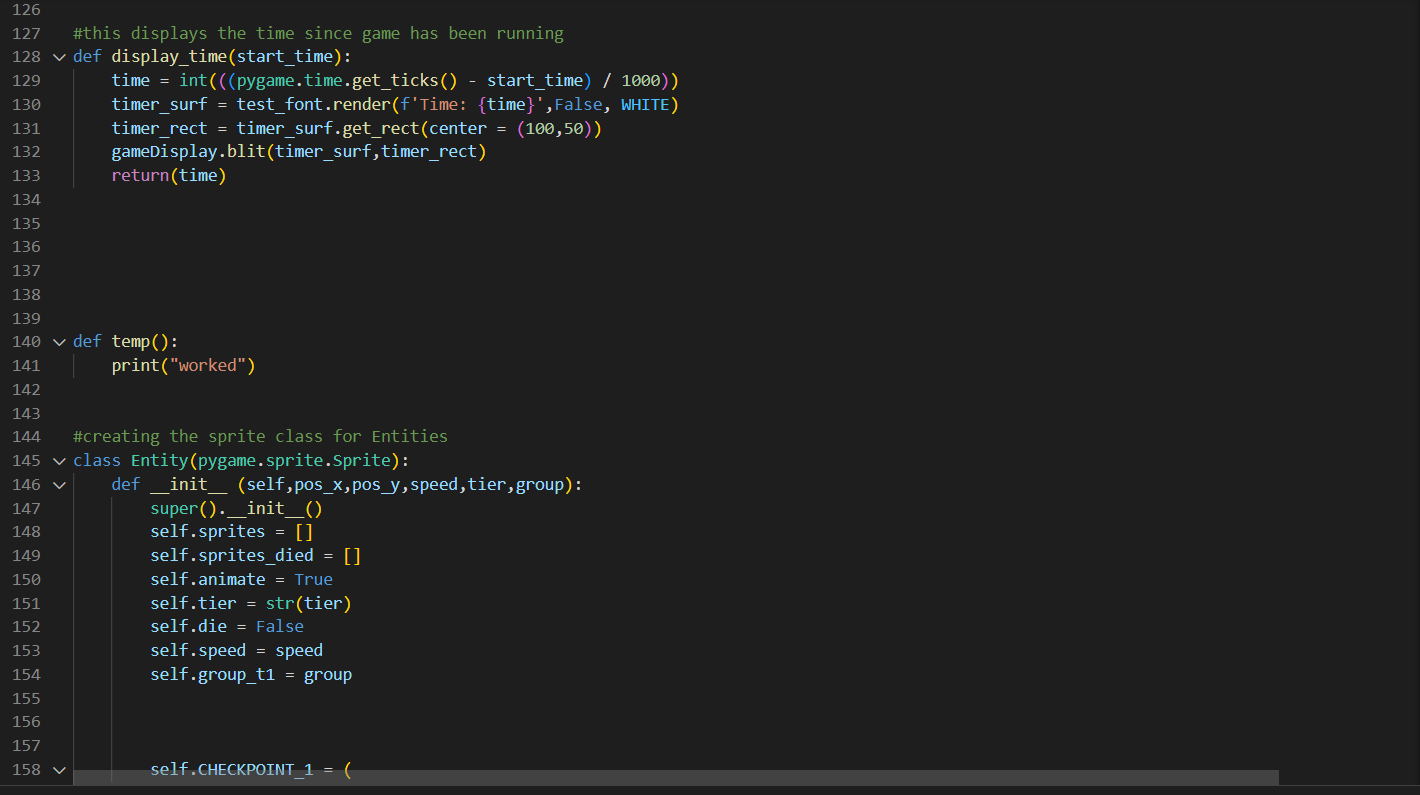
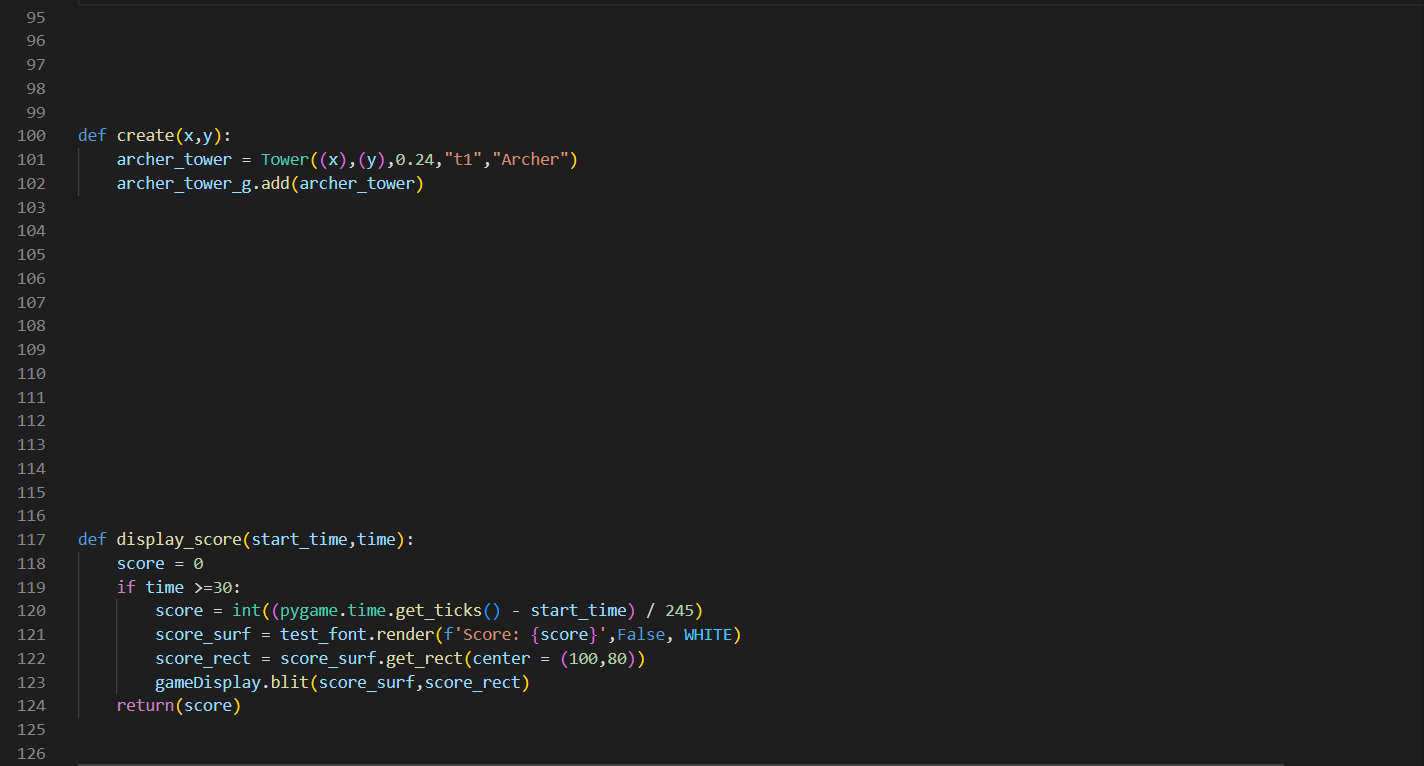
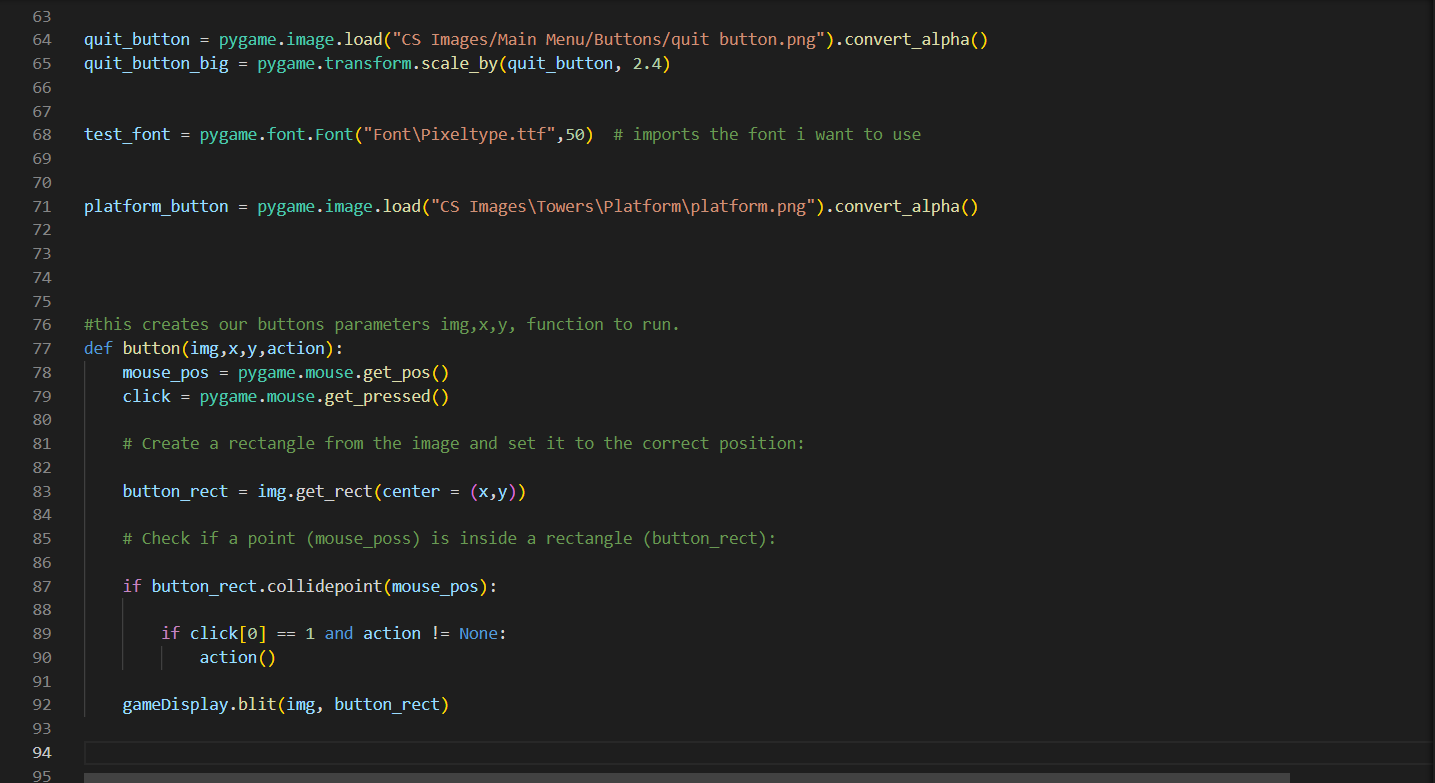
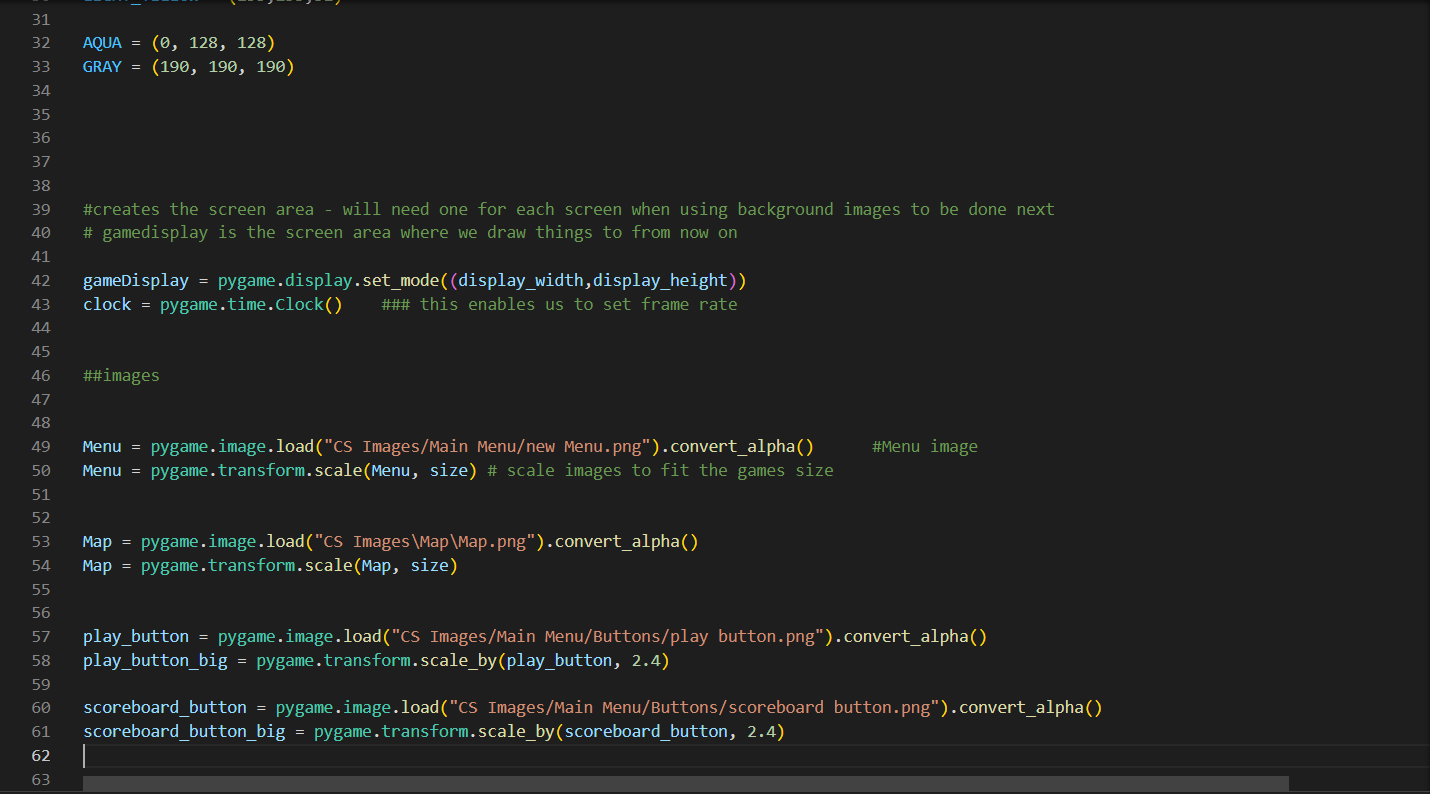
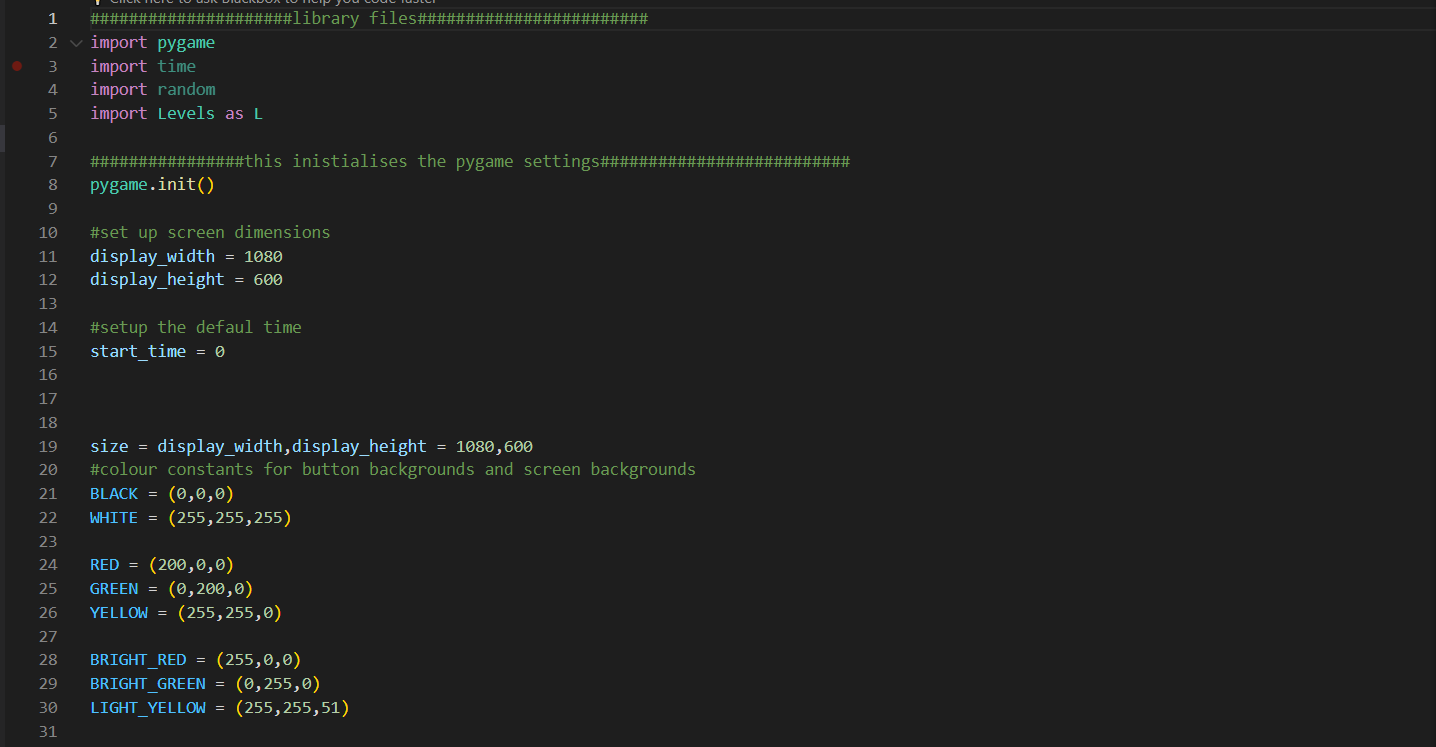
## Success criteria

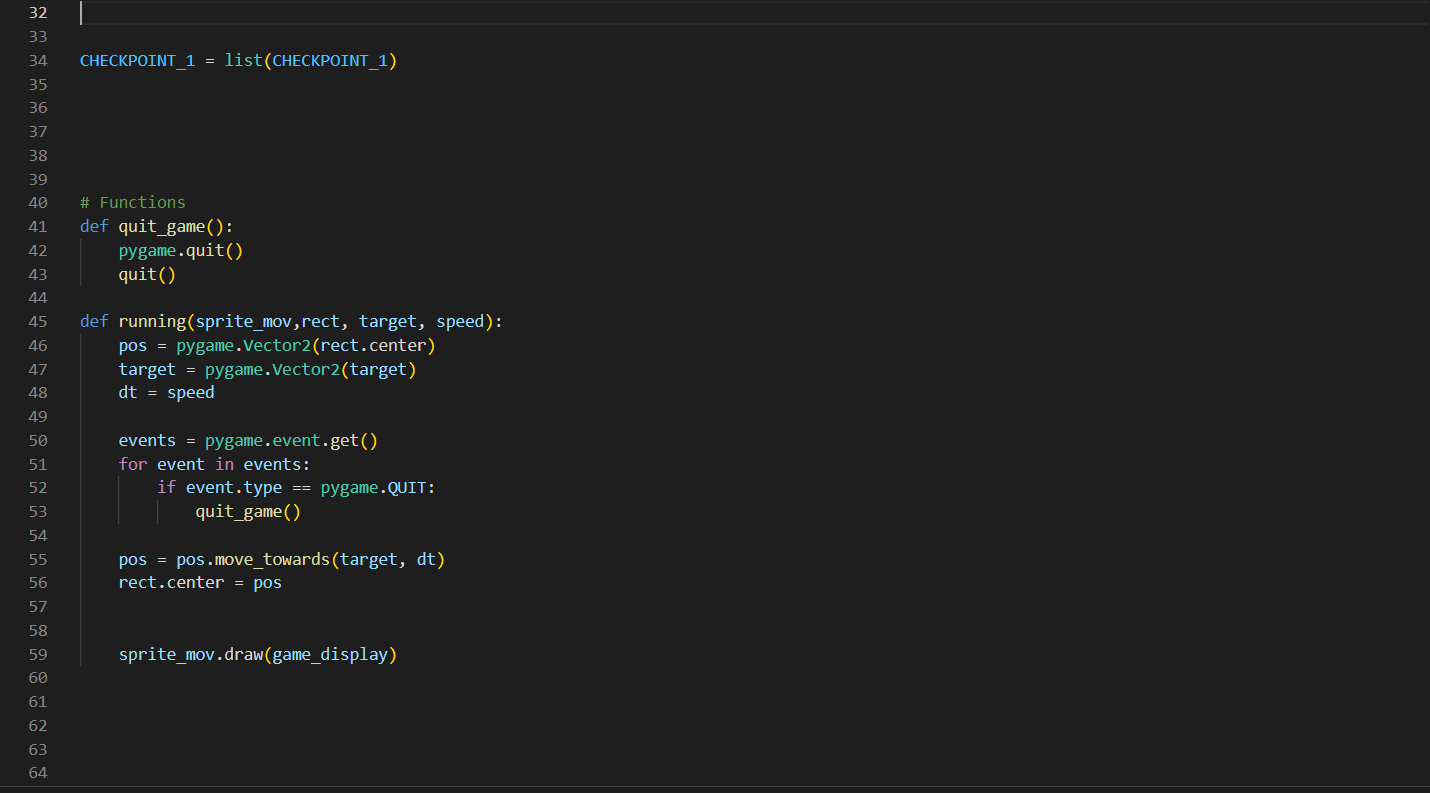
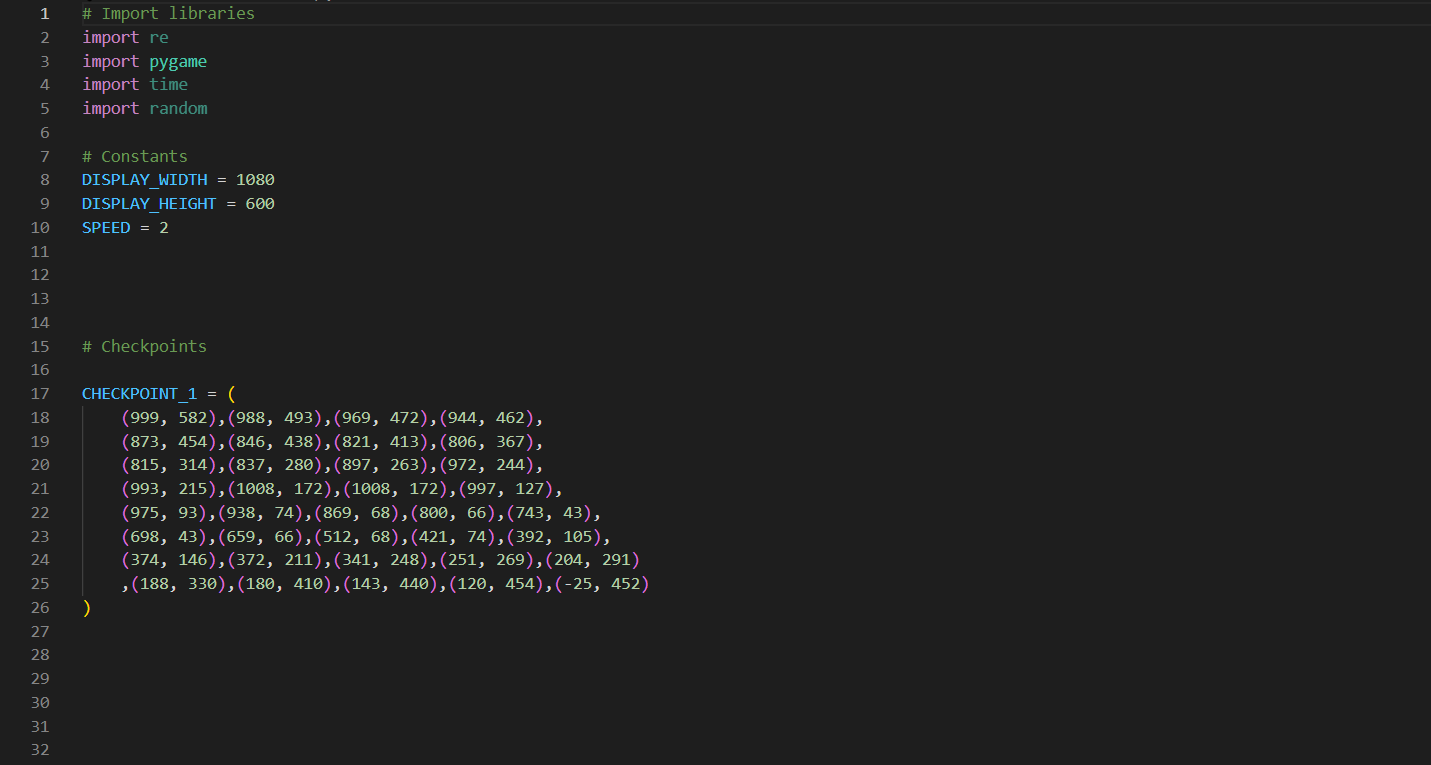
## Limitations/ future development

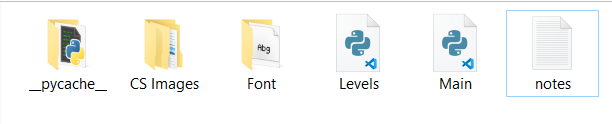
## maintenance

# Project Appendixes

Main.py :

Levels.py:

Folder:



Stakeholder Questions:

### INTERVIEW

To get a better idea of how others view my project, and both the positive and negative aspects of it, I asked my stakeholders some questions pertaining to the game and its mechanics:

|  |  |  |  |
| --- | --- | --- | --- |
| Questions | Joe | Stephen | Conclusion |
| How will the towers be upgraded? | You should click on them to upgrade them | Make a shop button which lets u choose upgrade | Ones clicking on tower it will upgrade the tower. |
| Should the Entities be animated? | Animations aren't a necessary feature, so you could include them when you're game has finished. | You should add them for a better player experience. | Animations may be added to refine the game, but first I should focus on the key aspects of the game itself first. |
| Will time be recorded? | Allowing user track time will be cool to be display and track | Tracking time seems pointless | Time will be tracked and possibly displayed on scoreboard |