The 5 Legendary Sannin

Professor Bills

INST 326

13 November, 2021

Class/Function Descriptions

Lauren Liberati:

The problem I worked on solving was how to determine the location of the ship in our game (lines 1-64). I created a class called Battleship which uses random number generation (the randint function) to determine if a ship is horizontal or vertical. Based on its orientation, it determines the next point which must be adjacent to the first point because the ship is two units long. It then appends these two coordinates to the self.location attribute which is a list. The final output returns both positions held in that list (ex. 4A 4B).

The file with my solution is “check_in.py” and there are no additional files required to run it.

As a note, our final game might be called Airship as we are considering modifying certain parts of the game, but for this check in I used traditional battleship to model my solution

Arfa Sheikh:

The problem I worked on is the ability to shoot the bullets and the position of the bullets. Instead of uploading single individual bullets or an image file, I had the Bullets class inherit a module of Pygame called “Sprite” which allows it to group related objects in the game and allows us to do actions on all the group elements at once. For example, firing all bullets instead of one by one. The rect() class is also part of pygrame. I used this to solve the issue of the position of the bullets

because the position of the bullets depends on the ship/plane's positions. I tried to match both of them to make it look like the belly is being fired from the plane/ship.

Resha Ahmed:

I worked on trying to manage the position of the bullets and then drawing the bullets to the screen (lines 100-112). In the Bullets class, I used the "Update" method to get the bullets to move up the screen. Once the bullets are fired, the y value is decreased. In order to fix the position of the y value, I needed to subtract the amount that is stored in settings.bullet_speed from self.y, then, update the position using self.rect.y = self.y. This would help fix the positioning of the bullets. Lastly, I used the "draw_bullets" function to draw the rectangle.

Mohammud Mossa:

The problem that I worked on was how to upload the plane image to the game. I created a class that focused on uploading a plane image to the program. The class image includes a function that initializes the image and sets it to the position given to us from the Plane class. We upload the image onto the screen rect given to us from the AirBattle class.

Sajjad Razvi:

I worked on the playing board function. This was needed in order to run our Airship game. I used one function called "playing_board" that would create an image(2d board) in the terminal that was set to 5x5(the size of our board). Alongside the columns and rows I numbered them 1-5 and letters A-E.