Game Report - Bailey Yi

My game is designed after a typical maze game, in which a player has to navigate through a floor while avoiding monsters in order to find the exit. The game most closely aligns with the field of education: I wanted to design an entertaining way for incoming freshman students, specifically at NYU, to familiarize themselves with the counseling and educational resources NYU offers. While NYU already has information about these resources listed on their website, I thought that a game would be an even more effective way of grabbing students' attention over possible challenges they might encounter during their time here at NYU. With that in mind, the default four monsters in the game are based on the issues that I found both my friends and I struggling with the most during our first year of college: academic stress, homesickness, mental health issues, and time management. The monsters are shown below, in their respective order:

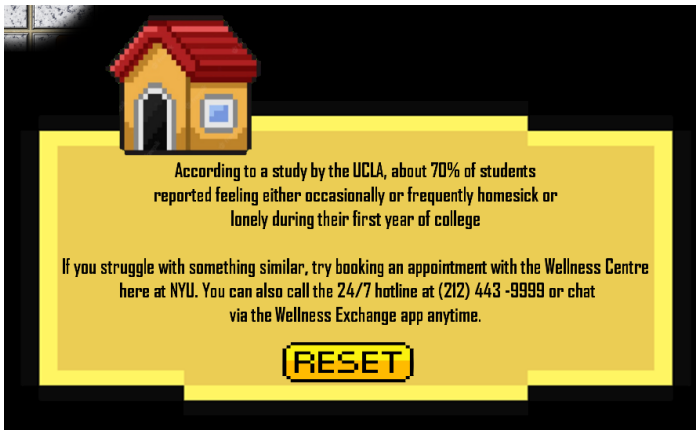
Each "monster" is also an extended class of the parent "Enemy" class, which means that more custom monsters can be added to cover the various struggles students experience in their transition to college. As the game was specifically designed in mind for incoming NYU students to play, the game is set in a library scenario in reference to NYU's Bobst Library. I also decided to replace the blocks/obstacles in the game with images of tables with NYU logos on them as a fun decoration.

A few design choices I would like to note: the game has a "flashlight effect", whereby the player can only see what's a few steps ahead of them. The motivation behind this design choice was to conceal the monsters on the floor, thereby adding an element of unpredictability to the game:however, it was also meant to mimic how often I felt during my freshman year that new problems were "popping up out of nowhere." To achieve this effect, I first used photoshop to draw a black rectangle with a transparent circle in the middle. I then loaded the image into processing and displayed it based on the player's coordinates. To make the rest of the screen black, I drew four other rectangles in processing whose width and height would continuously update relative to the position of the central black rectangle containing the transparent circle. The resulting effect is as shown:

Rather than specifically design a maze, I decided to generate block obstacles and randomly change their location every time the player died. This was to add some variability to the game, such that if the player restarted, they would find themselves faced with a "new" floor to navigate.

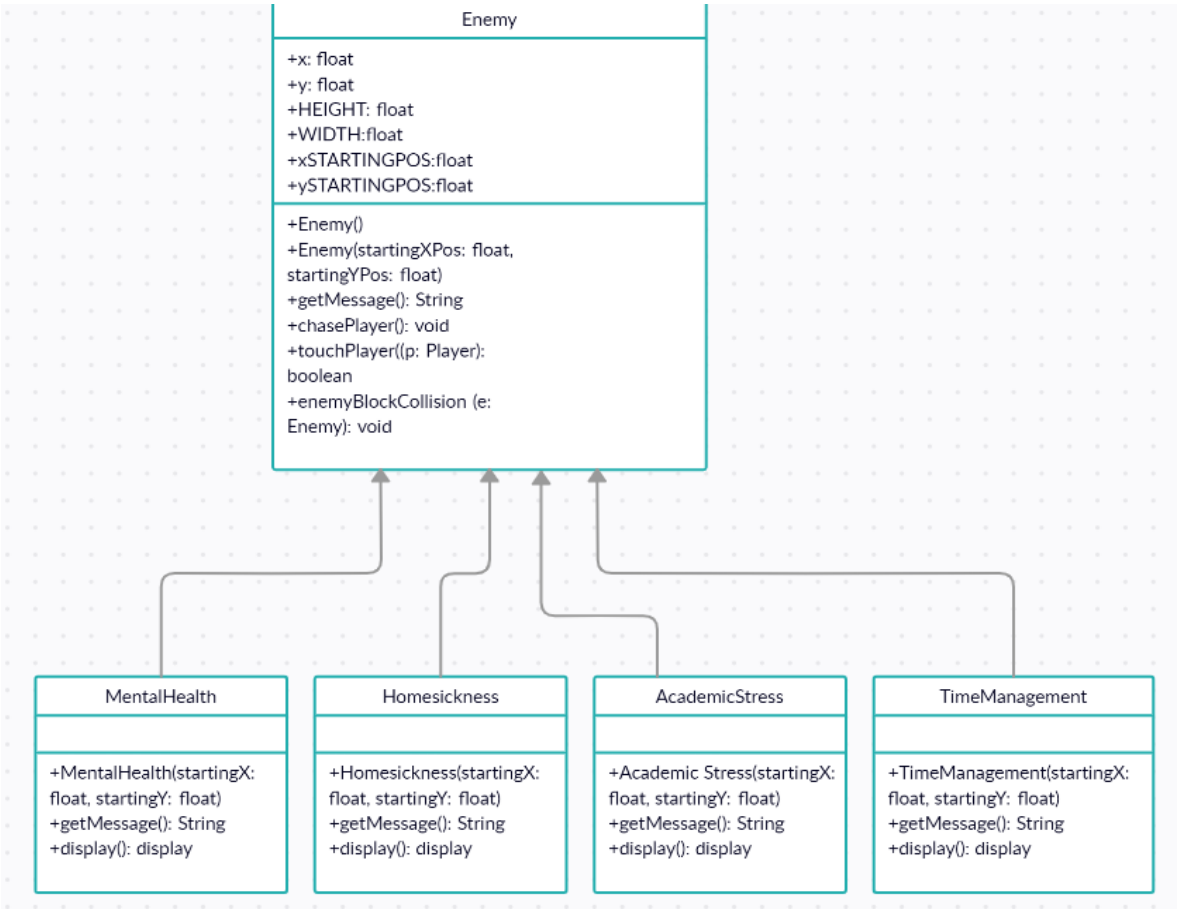
Gameplay:

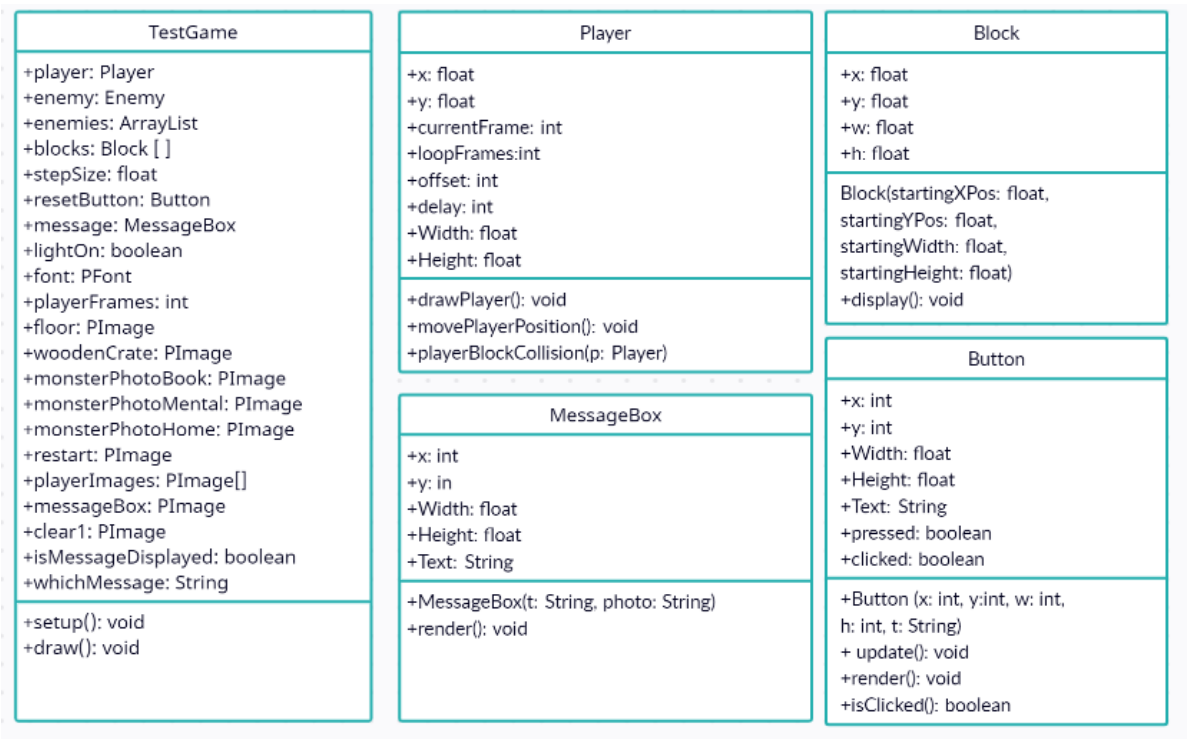
The gameplay itself is relatively simple: the player has to navigate through a floor using their arrow keys and find the exit. If the player enters into a certain radius of the monsters, they begin chasing after the player. In college, it is easy to feel as if your deadlines and responsibilities are constantly "chasing after you," which was the inspiration behind this design voice. If the player is caught, a death screen pops up. Which death screen pops up depends on which monster catches the player. For instance, the following death screen pops up if the player is caught by the "homesickness" monster.



Should the player choose to restart, they are then teleported back to the initial starting position, and the positions of the blocks and monsters are randomized. Given the time constraint for making this game, each monster has only one specific death screen message. In the future, however, I would like to add more and randomize the process, such that a new message is displayed almost every time the player dies. (UML diagrams shown below)

UML Diagrams:




TestGame
+player: Player
+enemy: Enemy
+enemies: ArrayList
+blocks: Block [ ]
+stepSize: float
+resetButton: Button
+message: MessageBox
+lightOn: boolean
+font: PFont
+playerFrames: int
+floor: PImage
+woodenCrate: PImage
+monsterPhotoBook: PImage
+monsterPhotoMental: PImage
+monsterPhotoHome: PImage
+restart: PImage
+playerImages: PImage[]
+messageBox: PImage
+clear1: PImage
+isMessageDisplayed: boolean
+whichMessage: String
+setup(): void
+draw(): void
Player
+x: float
+y: float
+currentFrame: int
+loopFrames:int
+offset: int
+delay: int
+Width: float
+Height: float
+drawPlayer(): void
+movePlayerPosition(): void
+playerBlockCollision(p: Player)
MessageBox
+x: int
+y: in
+Width: float
+Height: float
+Text: String
+MessageBox(t: String, photo: String)
+render(): void
Block
+x: float
+y: float
+w: float
+h: float
Block(startingXPos: float,
startingYPos: float,
startingWidth: float,
startingHeight: float)
+display(): void
Button
+x: int
+y: int
+Width: float
+Height: float
+Text: String
+pressed: boolean
+clicked: boolean
+Button (x: int, y:int, w: int,
h: int, t: String)
+ update(): void
+render(): void
+isClicked(): boolean