CMPT 276 Phase 4 Report

2.1 The Game

Include a brief overall description of your game in your report. Discuss how much you have been faithful to your original plan and design, and how the final product varies from that plan. Justify what has changed, and what are the most important lessons you have learned.

Game Concept:

The original game concept consists of a kid on Halloween being trapped in a Haunted House. The game follows the kid looking for a way out of the Haunted House, while collecting candy and avoiding Ghosts and Zombies within the house. The kid also needs to avoid bad candy such as black licorice.

Upon collecting the different types of objects, you receive or lose points based on the rewards collected. For example, if you collect a normal candy (regular reward) you get +50 to your score, whereas if you collect a Candy Basket (bonus reward) you get +150 to your score. Furthermore, collecting the black licorice (trap) or bumping into a Zombie or Ghost (enemies) will reduce your score by 50, 100, and 100 respectively. Also if your score reaches below 0, you die and the game ends. You clear the game by collecting all the regular and bonus rewards in the Haunted House.

We actually stayed quite faithful to the original game concept. We implemented the same theme, enemies, environment, traps, and rewards as stated in our phase 1 report. However, there are few minor changes such as both enemies being able to end the game if the player's score reaches below 0, instead of just the Zombie being able to do this. This change would give the game more difficulty. We also altered the design of the main character in order to differentiate him from the enemies. Another change is that we opted for a massive single floor level instead of multiple levels accessed by staircases. This change allowed for less cramped and more fluid movement, as well as made the game easier to play through. The last notable change was the game's clear logic. Initially, we wanted the player to collect all the candy, which would open a door he could exit through. However, we found that the door functionality created alot of branching cases which decreased our overall test coverage. For example, all cases where the player would try to open the door with 0, 1, 2, 3, ….4 or more candies. Another case would be how it considers combinations of regular candy and black licorice. As a result, to make testing more simple without changing core game aspects we decided to remove the door and just have the game clear when all candies were collected.

Game Implementation:

Our initial design changed several times throughout the game’s implementation. Only when we started programming the initial classes in our UML diagram, we found that we were missing classes we didn’t know we needed or we had extra redundant classes. Therefore, throughout the project, some classes were removed completely, some were combined into others, and some new ones were introduced.

For example, we combined the wall and room classes into a single class called Tile, as well as added new classes to draw/update the frame and handle the GUI. We decided to combine the wall and room classes since the logic for rooms could have been implemented using a sequence of wall tiles, making the room class redundant. Furthermore, the wall class was changed to Tile in order to make it more general. With this change, a tile could be a wall or a floor tile. Another change is the implementation of the TileController, which replaced the Haunted House class in our old UML diagram. TileController essentially handled the tiles of our map, and since it held the fundamentals of map creation, the HauntedHouse class became redundant.

Next, we added a bunch of classes that dealt with functionality we didn't consider during phase 1 such as the DetectCollision class and ObjectSettings class. These classes were used to determine the collision of the player with enemies, rewards, walls, traps along with the generation of these enemies, rewards and traps in the actual map.

Lessons Learned:

Throughout the development of our game, we have learned many lessons through the multitude of issues we've faced. Firstly, a big takeaway would be the importance of planning and thoughtful design. In this project, we had to tweak our UML diagram numerous times and changed our code accordingly. All the time spent making these changes could have been saved if we spent more time thinking about the design in the initial stages of the project. We also learned about the importance of planning, and how it helps you meet deadlines while organizing the massive amount of tasks associated with big projects.

Another important lesson was communication. You need to make sure you update your team continuously on things that have been completed and issues you are facing. This helps the entire team work together more efficiently by reducing repetition of work. Furthermore, your team is there to help you. If you have issues, getting other people's opinions and thoughts may help you solve the issue quicker. In order to do this, we had weekly group calls on discord, where we would update each other on outgoing and completed tasks.

Time management is also an important lesson. Although we delegated tasks to each person and made a plan, we seemed to always be right on the deadline with our submissions. This is definitely due to every student's busy schedules with work, other classes and personal reasons. That said, we've learned that it's always best to start as soon as possible.

We also learned a lot of technical information. For a few students in the group, this was their first experience with Java in general. Furthermore, we got to learn about and implement threading, JFrame, JPanel and general Object Oriented Programming practices.

2.2 Tutorial

2.2.1 In the Report Provide a tutorial/demo of main features of your game and highlight scenarios that you deem more important. Use screenshots along with text for explaining the features and teach users how to play.

Main Menu

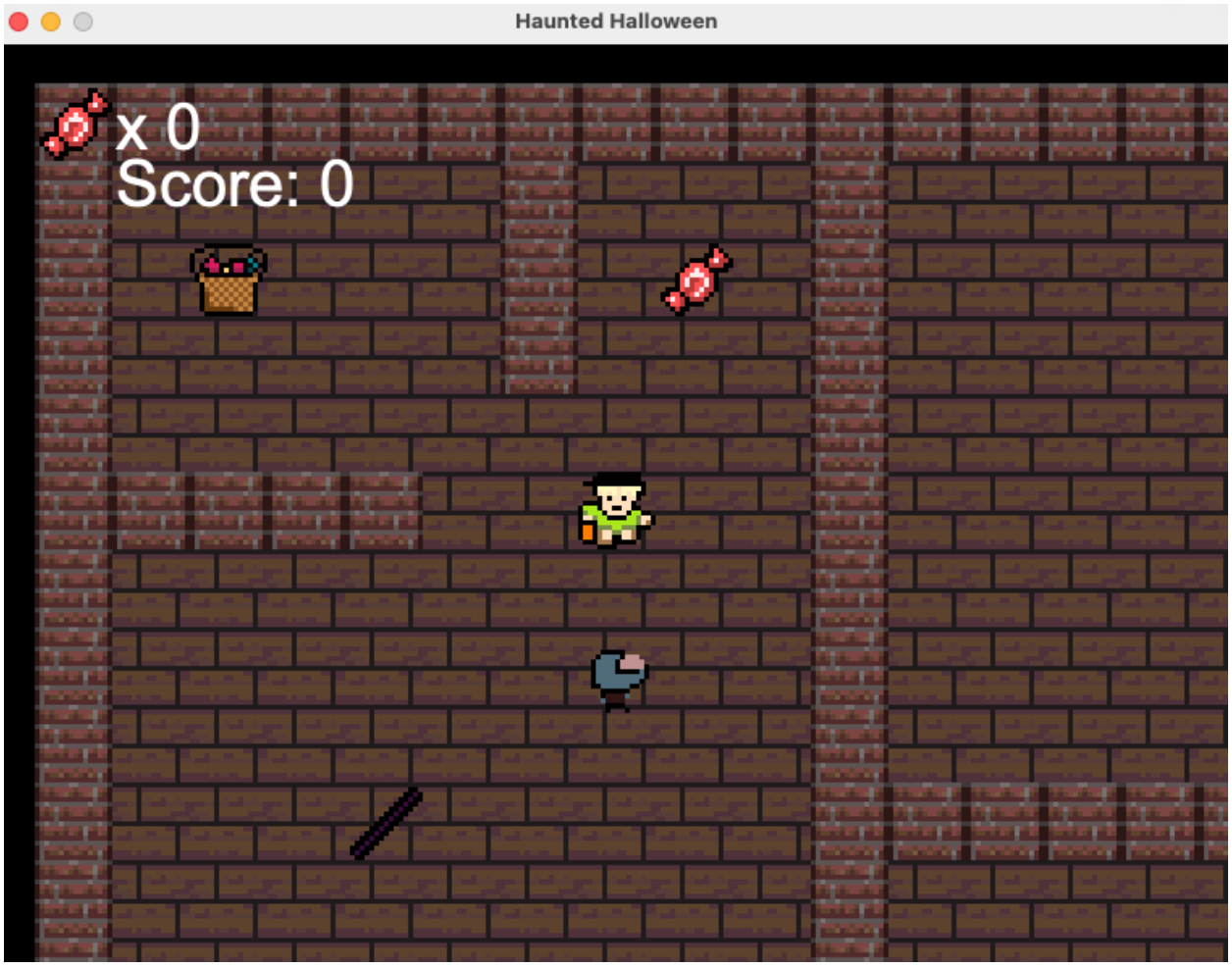
1. NEW GAME : click to start the game.
2. QUIT: Click to exit out of the game.

Use the “w” and “s” keys to scroll down the menu. Press “enter” to select an option.

The image shown below is the Main Menu screen

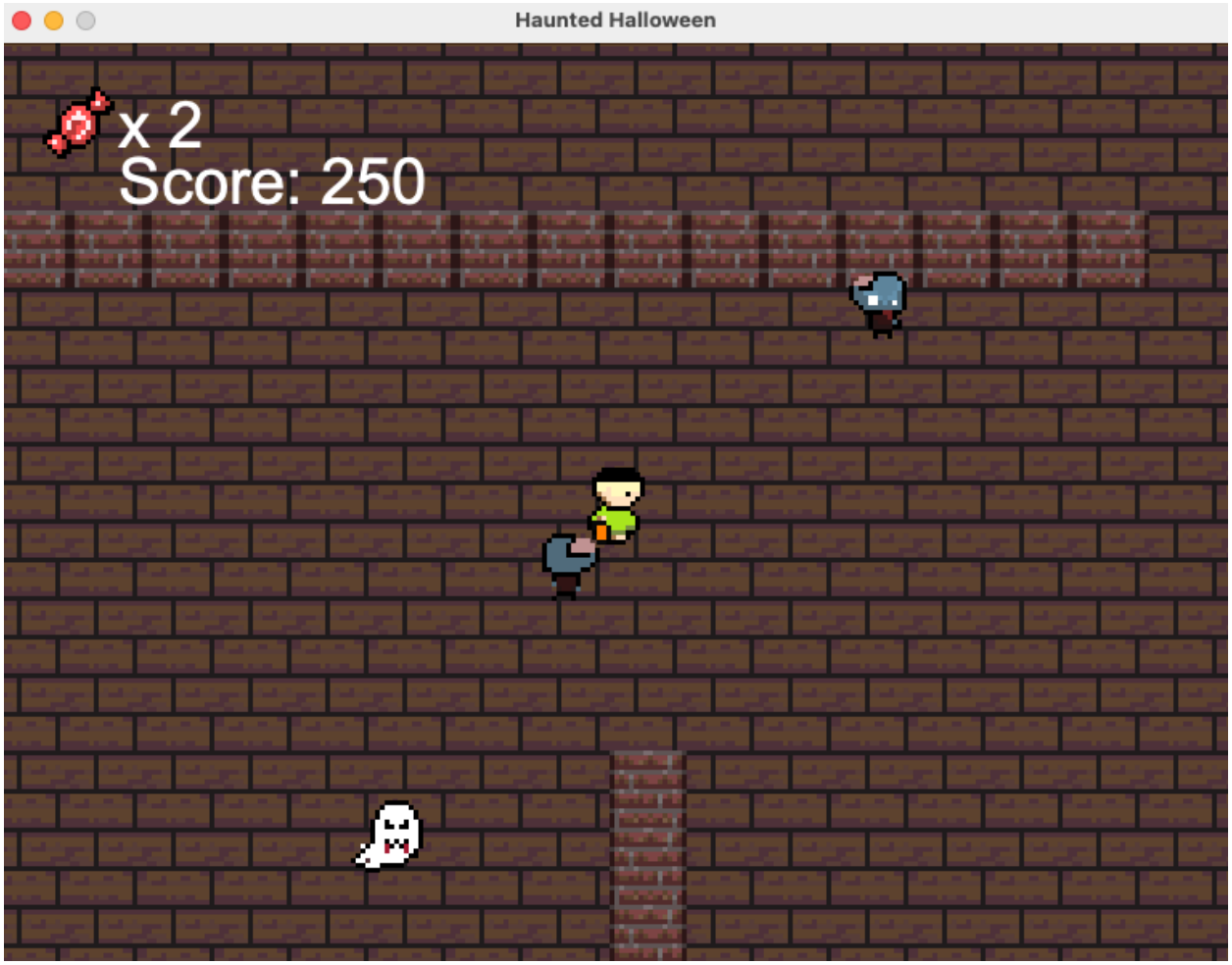
Game Map:

Scattered across the maps are walls, enemies, and rewards. The player has to traverse the map while avoiding enemies and collecting the rewards.



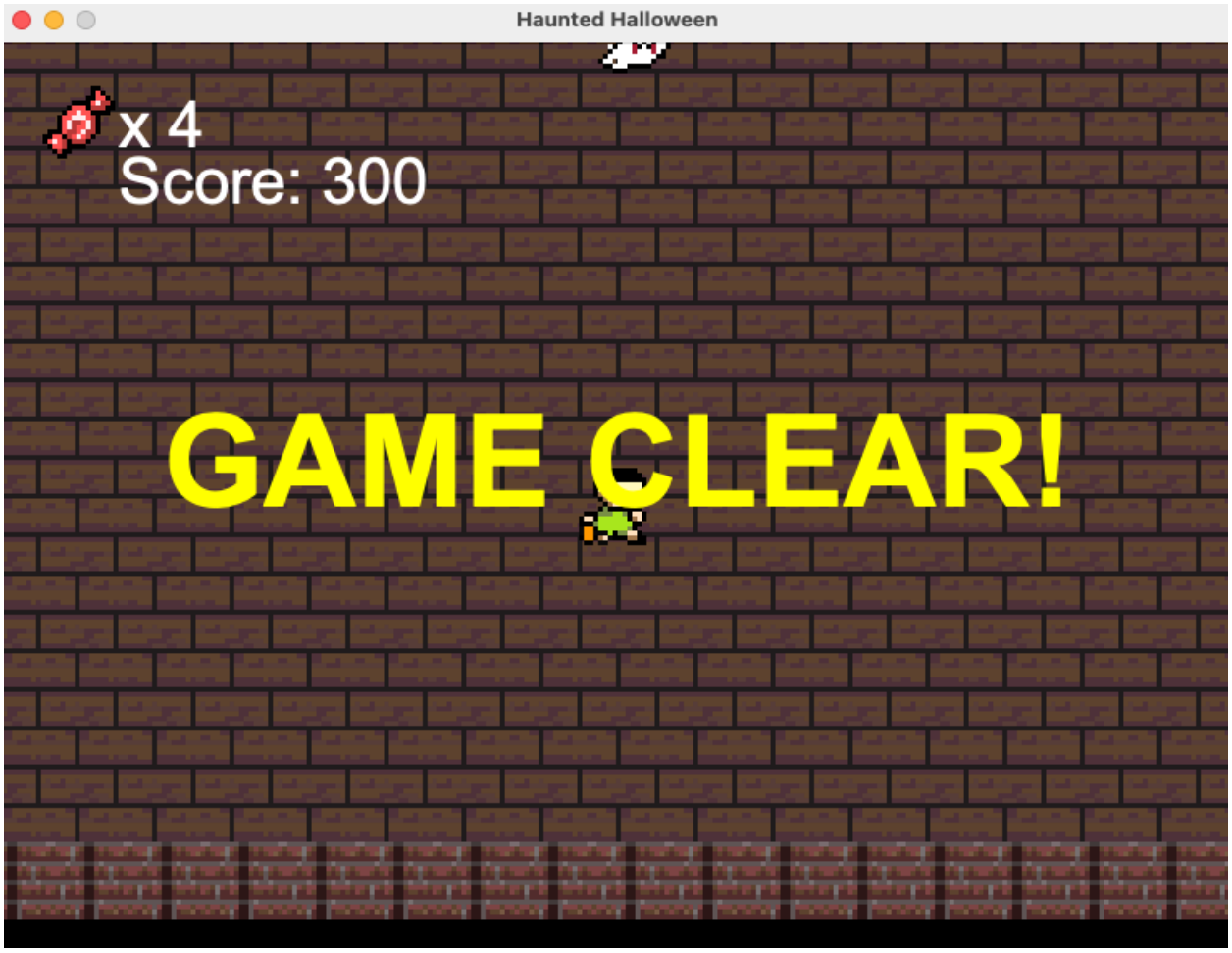
Game Score and Game Enemies

There are two types of moving enemies that track the player. The ghost can move through walls and will reduce the player's score by 100. The zombie can only track when it sees or has line of sight of the player. The ghost will also reduce the player's score by 100.



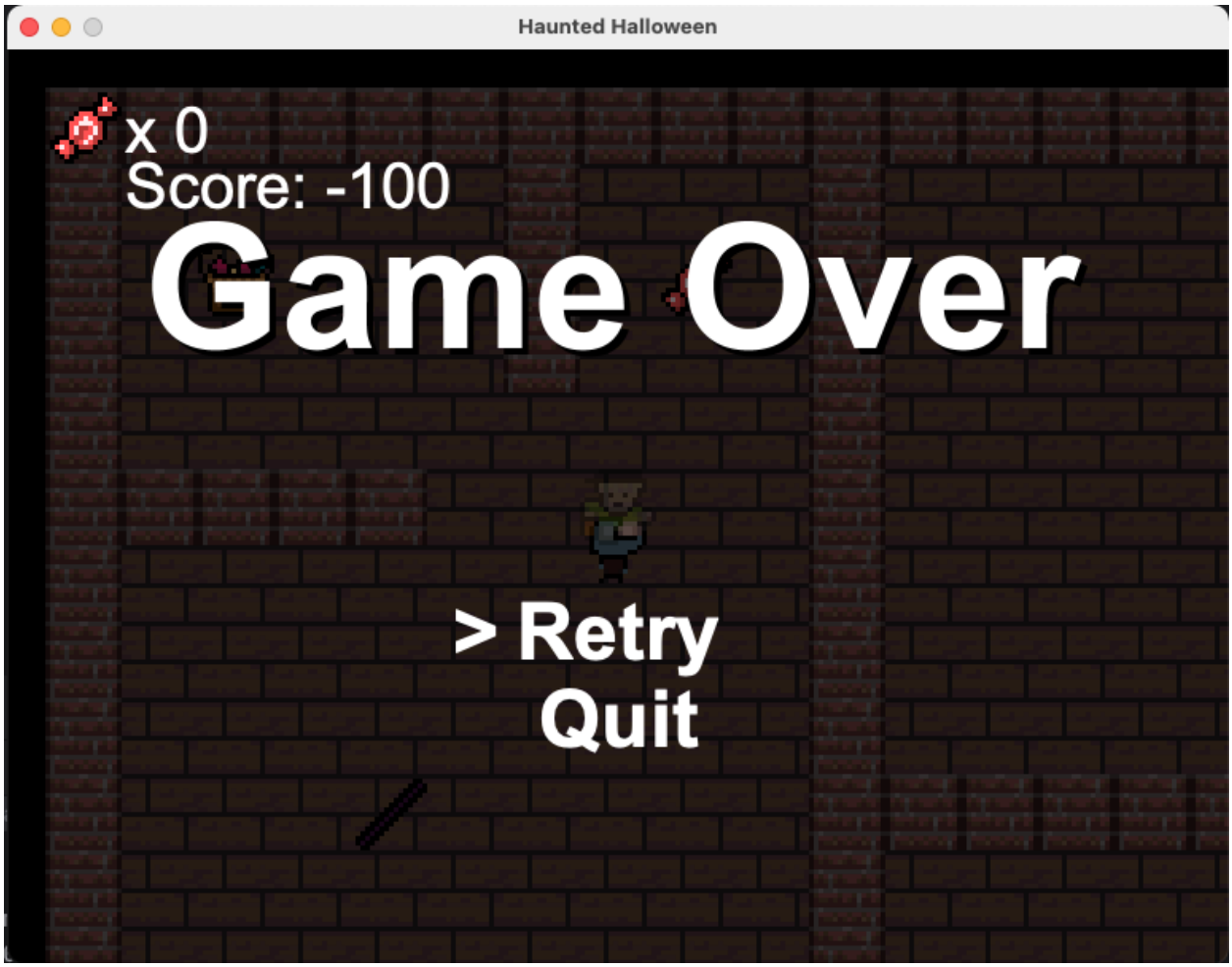
Game Clear Screen and Game Rewards

If your main character collects a normal candy, you will earn 50 points and if the character collects a candy basket then you receive 150 points. If all candies in the map are collected the game is cleared and you win.



Game Losing screen and Game Traps

If the character collects a black Licorice it will reduce the points by 50. If a zombie catches the character you will lose 100 points. Similarly the ghost can reduce the points by 100. If the character's score drops below 0, the game ends. After the game ends, the player can quit the game or retry again using the "w," "s," and "enter" keys to traverse and select an option.



3 The Artifacts

The artifacts and jar file are located in the phase2 directory: ~/CMPT276F23_group3/phase2