Phase 2 Report

CMPT 276 Group 15

1.Describe your overall approach to implement the game.

To implement the game, we first did plenty of research online, such as watching the game video tutorials. This helped us a lot because it points a direction for us on how to start a game. For example, we learned useful java classes like opening a game window, reading character images, displaying images on screen, and so on. After implementing the basic part of the game (E.g., game map, keyboard handler, player movements), we add in features that satisfy our game design requirement.

2.State and justify the adjustments and modifications to the initial design of the project.

The biggest adjustment we made is adding a ControlHandler class that we didn't think of in our initial UML design. We also add new methods that check the collision between objects. For the use cases, we add something new which is once the player presses P key, the game can pause, and it resumes when the player presses P again. We also implemented the title screen differently compared to our original UI mockups that we proposed in Phase 1, as we felt that it would be less intuitive for the title screen UI to be the way it was in our mockups. Old UML diagram and an updated UML diagram is attached at the end of the report.

3.Explain the management process of this phase and the division of roles and responsibilities.

We collaborated on the implementation concurrently via Git and we were able to communicate while we were working simultaneously by using Discord to notify each other when we would be making changes and pushing to the remote repository in order to minimize the amount of potential merge conflicts that would arise from working on the implementation at the same time.

Member	Tasks/progress
Clinton	1. Help window[Done] 2. Title Screen[Done] 3. Help Window and Title Screen Graphics [Done] 4. Restart method for GameOver UI[Done]
Linda	1. Medkit (bonus) and carrot (rewards) pixel arts.[Done] 2. Medkits & carrot placement mechanism.[Done] 3. Medkit random appear[Done] 4. Medkit pickup[Done] 5. Main character collison control [Done] 6. Time control[Done]

Jackson	1. Enemy implementation [Done] 2. Main menu implementation[Done] 3. UI implement [Done] 4. Pick up rewards and disappear from the screen method[Done] 5. Game Over Screen[Done] 6. Control from keyboard for game state[Done]
Joanna	1. Main game thread [DONE] 2. Game resolution and tile size management [DONE] 3. Pixel art for the main character and basic map tile package [DONE] 4. Main character control and animation [DONE] 5. Wolf moving[Done]

4.List external libraries you used, for instance for the GUI and briefly justify the reason(s) for choosing the libraries.

We didn't use any external libraries.

5.Describe the measures you took to enhance the quality of your code.

-Instead of passing new objects as arguments into methods in a loop, we initialize the object outside the loop, and pass that object. This way, we save more memory and make code more efficient.

-We take the advantages of using functions, so as to avoid repetition of code and make the program more readable.

-We tried to modularize aspects of the code via packages to help with functionality and to reduce repetition.

-We added a decent amount of comments in order to help both ourselves work through the code better and for a user to understand what function's purpose is.

-The use of inheritance reduces the amount of duplicated code.

6.Discuss the biggest challenges you faced during this phase.

-Using git is challenging, sometimes there are many merge conflicts that are hard to solve.

-The method calls in the class file are frequent and sometimes a bit confusing

-Implementing the enemy pathing was the most complex as it required the use of a modified Djikstra's algorithm (ie. A* algorithm) in order to determine the path to pursue the player. We have not fully implemented this feature yet, so in phase 3 the pathfinding will be

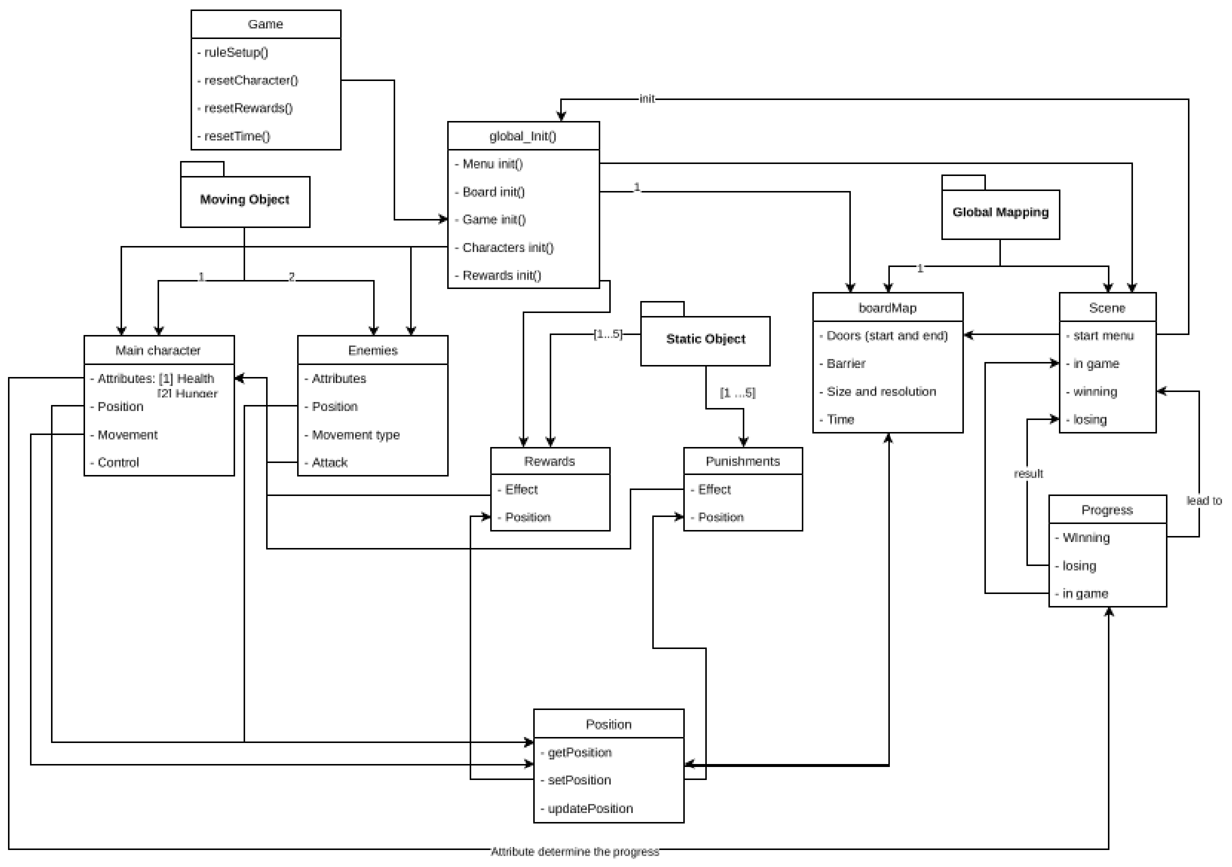

Figure 1. Old UML diagram

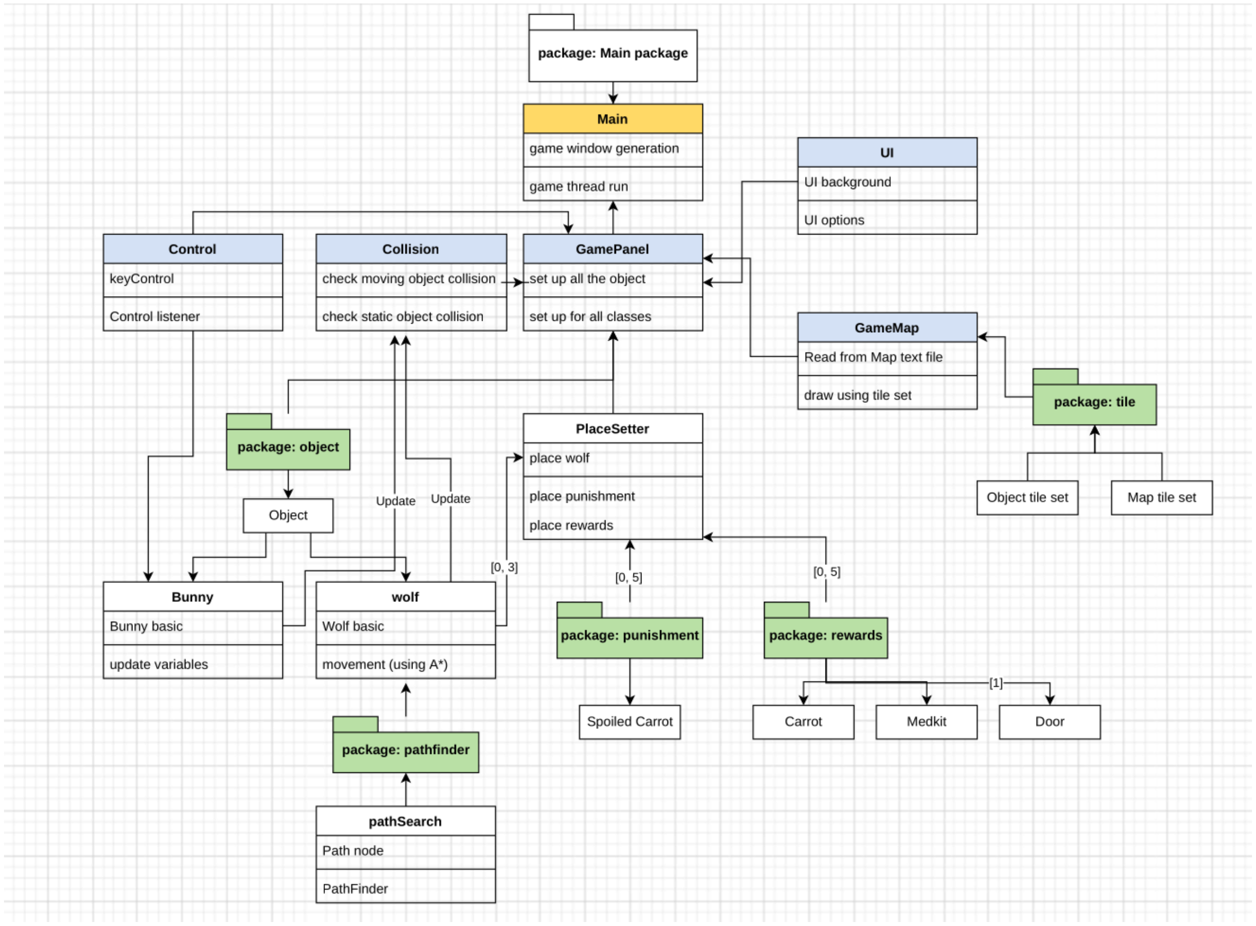

Figure 2. New UML diagram