Project 3 Report
CSCI 1300
Jack Savio

I began development on my project with the initial intention of preparing the project specifically based upon a role-playing game with objectives solely based on finding and defeating bosses in order to win the game. To make this game, I had to plan out HOW I would be able to implement such ideas. The basis of the entire project was essentially using classes to implement the functions and main stream of code to create such goals. Thus, I had to carefully pick my classes that sufficiently support these goals. I had to think about what my classes would be to make the game. I decided initially to work on four classes: an enemy class, an NPC class, a hero class, and an area class. After some quick development on the project, I realized that it was more important to focus on the implementation of the area class, as the main function of the game was to travel through separate areas and all other aspects of the game are supplementary.

After deciding on the class structure behind the code, implementation utilizing the code skeleton was crucial towards success. Defining setters and getters for private variables was an important part of the steps to complete the code, but I ran into a hiccup quite quickly in that the hero and enemy class are dependent on each other in that the combat system is reliant on both classes at the same time. To fix this problem, my classes are independent but linked together through their combat stats alone. For example, hinging the classes on each other could lead to bugs in the mainstream of code. The code skeleton as a whole was crucial in order to completely implement the code of the project. Using a base of four classes led to a significant improvement to the project, as I was able to use these classes to make the interactions between them after the fact.

In reflection, my ability to create the project was seriously hindered by the fact that I was unable to fully bug test my code due to poor planning. I initially planned to be done by fall break, but my project was not finished until far after that, leading to less time to bug test the code. I ran into some issues immediately after testing, and gave a massive amount of time constraint to the project that would not have been there in the first place. Fortunately, I was able to figure out that the stats were not converting properly as the setHealth() command on the hero class was improperly coded. Altering the code led to a solution of creating a common variable that resets based on when a battle ends, and effectively improved the project conclusion. Similarly, I decided to improve the combat system relatively last minute, adding a speed category that decides a form of dodge chance if the player has a massive lead in speed. While improving the combat was necessary in order to improve the aspects of the game, this was a mistake, as it massively increased the time to complete the project as it led to difficulties in making the game.

In the context of CSCI 1300's project guidelines, I had to significantly alter my project in order to fit the requirements for the code. For example, the random selection requirement proved difficult to implement in a way I thought it would be good for the project as a whole. Thus, after some brainstorming, the speed command (as mentioned above) was decided on. Essentially, it gives a random boost to the combat system that does not damage the integrity of the system, but

it adds a chance to outspeed and defeat opponents. This utilizes all categories of code, including NPCS being able to sell items that will increase stats, and both heroes and enemies having these boosts. The idea that I could do this project without clearly deciding on where each part of the project requirements had to be implemented led to a slight hiccup in completion of the project to fit both CSCI 1300 guidelines while also being satisfied with the end result.