Kevin Busch
CSCI-1300 Spring Semester
Tony Wong / Thanika Reddy - Section 300 / Recitation 301
Project 3 Reflection Paper

Reflection Paper

The only preparation I did mainly consisted of just brainstorming ideas about what to actually do for the project and some ideas for classes. Almost a code skeleton for the code skeleton, if you will. Then from there, before I started any actual development of the project, I commented some entire class ideas, and then after I made some of the main parts of each of my classes, I went into the main game function, and walked through every step that the game would go through logically, commenting how it would work and how it would need to be developed. This was extremely helpful, as it helped me organize my workflow and my ideas much more than what I was used to. With a full game plan of what the program should look like, it really helped to keep me on track and not get confused within the scope of the program, and it let me prioritize what needed to be done first.

During the project, I wish I had planned for the game to be more simple to start with, and then added more to the game after I had it working at a very basic level. I spent a lot of time during development removing aspects of the game that I thought I would have time to implement, only to decide later that I either couldn't implement it in time, or that it would be to difficult to make it work, at least while just trying to get simple parts of the game working. Originally I had planned for the game to have 4 races and 4 classes, allowing for much more possible variance in gameplay. Along with the 2 other races and classes that I couldn't include, I wanted for there to be another stat for each hero, and I wanted to have fights use a more complicated system to determine how much damage a hero could do. Some other things I wanted to do, but had to go back on after realizing I didn't have time included: random events based on turn count (which could be avoided if certain stats were high enough), items such as rings or armor, and possibly a map system of some sort.

After eventually deciding to just simplify many parts of my game to make the task more feasible, the actual development of my game went really smoothly. I think the main reason for this was because I changed my development strategy for this project compared to what I usually would do for my other projects. Instead of writing a large portion of code, and then trying to compile it all later, then have to bug fix a ton of errors, this time I would test what I wrote after every little section, to make sure it worked. This strategy allowed me to catch way more errors, much faster. By developing like this, I was much more productive, because the bugs that I had to fix were generally a much easier problem to solve. Also, because I knew all the other code worked, I don't have to make a ton of guesses about where the error is or what is actually wrong with the code. The last little thing that really helped me was making stat dump functions for almost every object, so that I could easily see what was happening to each object within my code. This made tracking down and pinpointing where errors were happening much easier.