Design Reflection

Program name : Project3_Design_Reflection_Schaefer_Kristin.pdf
Author : Kristin Schaefer
Date : 05-12-2019

Design Changes and Problems Encountered

For Project 3, I found the hardest part of the assignment was not to understand the concept of polymorphism, but to find the right sources and examples of how to implement polymorphism in source files, header files and main files. The sources I found most often had examples that were too general and compacted. The abstract class was relatively simple to create, however, it wasn't clear to me when to use "virtual" and "override", and I ended up testing and seeing what worked and what didn't. Ultimately, what I found to work the best was to create the Barbarian class as a derived class of the abstract class Character. Then I made the Vampire, Harry Potter, Blue Men and Medusa classes derived classes of Barbarian.

I had imagined that I would be able to make a comprehensive rollDice() function that all classes could use for both their attack and defense function, but I ran into issues with the program crashing. I had the same error in previous projects with trying to use rand() to generate random values. I learned in the past that it helped to declare and initiate all variables before using them with the rand() function. I tried this approach when I had problems with the rollDice() function, however, the program was still crashing. Due to time constraints I had to scrap the idea of using the rollDice function for all classes and I instead had several of the character classes override the rollDice function and set their own number of dice and sides values. Thankfully, this project consists of two parts so I will try to get the function working for the next half of the assignment.

Compared to previous projects, I did not struggle with dynamic memory allocation and memory leaks in this project, or at least that I am aware of. For the first round of testing I ran into problems with a segmentation fault at the end of the program. I quickly discovered though that I was calling the delete ptr several times, thus trying to delete what had already been deleted! I reduced the delete to one call and the problem was fixed. Fortunately, I think I am improving a bit in terms of understanding dynamic memory allocation!