Minh Nguyen

Assignment 3 Testing and Reflection

CS 162

Testing

To test that my program works, I've set up all the possible fights in 10 different functions. When the program is executed, there is a menu to choose which of the following fights you'd like to see. To make sure that my game works as intended, I have a print statement for every roll and every action of the battle in progress. I print out the rolls of the dies, then print out the damage that occurs after subtracting from the defense. This means that the number printed out is the damage before applying armor.

Then I also print out the damage after armor, and then print out the remaining strength point of each monster during each turn. As I run through the program, I can see each numbers and was able to fix a few bugs when the numbers did not look right. Such as a portion where a defense roll was higher than an attack roll that lead to the defenders strength point increasing because of the math error. So I had to implement an if statement that the attack roll had to be higher than the defense roll to apply any other math.

Reflection

I began my implementation following the design plan as far as creating the classes go. Then I realized that I didn't need the set functions as I thought I did, and directly put the random number generators in the get functions and then returned the dice rolls that way. It was a little more efficient because I could just call the random number with one call per turn and saved me some space in coding.

The next thing that was different was I didn't have the need to set a player value to = 1 or 2 as I thought I did in my original plans. I realized I could just run an entire sequence, one attack, one defense, two attack, two defense, in one while loop. What I had in mind when I thought of the original idea was that 2 users would be playing against each other, so I was thinking of giving the user some control in when they were rolling and seeing their rolls being played out. But that was not needed, so I scrapped that idea.

I also had to create new variables in each fighting functions because I had to figure out a way to store the classes' strength values as it was changing. In my design document , I stated that I wasn't sure how I would save the strength value to go to the next round, so I found a way around this by every attack and defense value to a variable in each function, and then just manipulating that variable was much easier. Then, each monsters strength can be strength1 for one monster, and strength2 for the next. And it would simply be subtraction of total damage inflicted from the previous strength value. This worked out well and I was able to save the strength value for every turn of the loop.

The rest of the design went as planned. I created a several ways to exit the game, such as if statements when a strength value becomes less than or equal to 0. Then leaving the while loop, I also have 2 more if statements with the condition which strength value is 0 to declare the winner.