Stephen Townsend
CS 362-400
11/5/17

Assignment 4

Random Testing

For my random testers I wanted to keep it similar to mu unit tests. I feel that my unit tests had some good results and I wanted to see what would happen when I took out the static aspect. As a result, I first took my unit test and implemented a random generator in it. Allowed the generator to give me a random number of players and a random seed. When I ran the test for 10 passes, I did not see any change from what I produced in the Unit test result.

I then decided to up the runs and allowed the program to make 50 passes. This is where I noticed the changes from the unit tests. I ran gcov on my random tester and found I had similar coverage. I then made a few modifications to how I would call some of the functions. Making some minor modifications I was able to get a higher coverage.

Code Coverage

I was able to cover 92.59% of my code for radomtestcard 1 and 2. After reviewing the gcov file, it looks like the if else statements are the portions that do not get covered completely. The reason being is if the code does not fail the test then it will only touch the if or the else, but not both. I will discuss this a little more later in the next section, but I was unable to get the adventurer to not seg fault. As a result, I was still unable to get any coverage here. There is something wrong in my source files that is causing this. The only solution is to get a new copy and attempt to use the original code and then look to refactor as we did in the first assignment.

I was not able to get 100% coverage, but the closest I could get was 96.30%. I was able to increase card1 random tester by 3.7%. I have attempted to modify some functions and some of my tests to utilize more of the code. I feel that the test could have been 100% if I didn't use if else statements. I feel that it is what is keeping the coverage down; similar to the Unit tests. I tried to do more random test to get a higher coverage. I went from 50 to 1000 and ended up at 10,000. However, from about 5000+ my coverage would not go past 96.30%. However, having so many passes feels a little pointless because having to look back at all the information would not be probable.

Unit vs Random

Unittest1:	Lines Executed: 86.96% of 23	Randomtestcard1	Lines Executed: 96.30% of 27
Unittest2:	Lines Executed: 76.72% of 116	Randomtestcard2	Lines Executed: 92.59% of 27
Unittest3:	Lines Executed: 87.88% of 33	Randomtestadventurer	Seg Fault
Unittest2:	Lines Executed: 83.33% of 12		

After looking at my result. I was able to get better coverage with my random testing. I feel that random testing did better in finding problems. I noticed that when I ran a set of 50 runs with random numbers for both # of players and the seed I was getting more failed messages. This shows me that running a wide range of random numbers can be very useful in seeing where there may be issues that a single Unit test run may not pick up. The reason being, unit tests deal with static runs and only run once. While the limit on using random is unparalleled.

However, as in the Unit tests, there seems to be an issue with my adventurer that I cannot find. I attempted to re download the working Domain folder to fix my adventurer. However, I am still having issues with it seg faulting. I attempted to restore it back to how it originally was in the card effect function, but that did not help. I do not know where this issue is yet. Until I can fix this problem, unit tests and random tests are not viable with a section of code that only runs one line then anything after it faults.