David Taylor  
CS 362: Software Engineering II  
*Test Report 2*

***Approach:*** *As my plan from the previous test report, I wanted to do more random testing. In this test block I did just that. I implemented a random test function, which tests the code on the 3 different unit tests I did before. I kept my unit tests when testing code just to make sure that the results were the same, as well as to extend my test suite. So first I made a routine that tests randomly to try and purchase a random card, out of all of them that are available. This will run 5000 times on the game to check to make sure that the player can buy all cards available, with 1 buy and 10 coins, enough to buy 1 card. This won’t actually buy the card, but merely test that its purchasable and that the user can in fact purchase it. It cannot confirm a correct purchase, that is what test case 2 and 3 do. The second random test is buying in particular the Adventurer card, though any could have been used, with which a player has a random number of buys and also only has 10 coins. This routine also runs 5000 times. Finally, the last routine tests to see, if you have 1 buy and a random amount of coins that you are able to only buy the cards that you have with the random amount of coins you have available. At the end of all these tests, the programme will tell you how many of the 5000 tests for each subroutine have failed. I also extended my previous plan of doing coverage testing, this time more precise. I ran gcov to see how much of the code I was covering in my tests. As I do my final report, I will try to get as high as possible. Also I did some random testing on buying a specific card, in this case the adventurer card, so I will look specifically in my next report how successful people were with implementing the effects of that card.*

*This below is an example of a random test case:  
printf("Purchase random card\n");  
 for (i = 0; i < 5000; i++) {  
 g.numBuys = 1;  
 g.coins = 10;  
 ret = buyCard(k[rand()%11], &g);  
 if (ret == -1) {  
 Failed++;  
 }  
 Ran++;  
 }  
 printf("%i ran, %i failed\n", Ran, Failed);*

***Test:*** *I ran this on my own code, and noticed that there were still some errors in my code that my original tests weren’t catching.  
 My own code: First off I noticed that my random tests showed me a loophole in my test, mainly that it was still running when I ran out of cards to buy. Once all the cards were bought, it was failing. Not a big deal, I just made a test case that covered this. Once this was implemented, it worked great. But I didn’t want to perfectly fix my dominion.c code, so someone else could catch my mistakes. My code fails on the 1st test case 0%, second test case 12% and the third test 8%. There are very minor bugs which I hopefully fix eventually.*

*Lesliew: So when I tested his code, I noticed a few minor things. First off, when I ran my first instance of tests, the random test on buying every card succeeded, so there is no problem buying a card. The area where the problem arose was in the second and third test cases.In the second test case, 1722 tests of the 5000 failed, so not terrible, but not very good either. Lesliew dominion code failed 34% of the time. The third test case was slightly worse, at 40%. When I took a closer look at the code, I noticed that the player was only buying specific cards, in this case, cards with value, copper, silver or gold. So there was no way to determine whether or not the user was buying the correct card. The reason my first test case succeeded was because it was testing to see if the cards were actually possible to buy, and that they could in fact be bought, not that they were bought. My second 2 test cases actually try buying cards.*

*Ellingsn: Since I noticed that the code is in fact the original code, it has not been changed what so ever, I made these observations. The original code will work in some ways. Using the first set of random tests I noticed that the code will be able to buy every card in the deck. It passed all the random tests of the first test case to test buying any random card. However on the other 2 test cases, it really failed. Test case 2 failed 1717/5000 times, which is a fail rate of 34%. It really doesn’t work. Then the 3rd random test case also failed miserable, with 41% fail rate. So the original code is really broken. I understand why it was offered as one of the code segments that needed to be edited, it is broken.*

*Omaara: When running this person’s code, it was more successful than the other tests. The first test case, of trying to buy every card succeeded with no hiccups. It was 100% successful, meaning that their code will buy every card when run. However test case 2 and 3 were a different story. So test case 2, using a random amount of coins to buy a card, failed 32% of the time. This means that they don’t cover certain cases of buying a card. Looking closer at the code, I notice it was just an extension of the original code. I noticed that it doesn’t look at the number of buys a player has, but instead just the number of coins and the card supply. Finally, the 3rd test case also failed, but not as much, only19% of the time. I believe it failed for the same reason as the second test case. To improve this code, I would recommend changing the code so that it checks to make sure that there is the right number coins, necessary number of buys and that there is the right supply count. This should fix all the problems occurring in the code.*

*Shearini: This code, just like the others, works correctly with the first random test case. It passes the 1st random test case of trying to purchase any random card. However the other 2 test cases failed. For the second random test case it failed 32% of the time. There is probably something wrong with buying a card with various number of available buys. The final test case, checking to see if you can buy a card with random number of coins also failed. It failed 19% of the time. This means that it will allow you to buy a card when you have the number of coins, sometimes or it could let you buy a card even when you don’t have the right number of coins. Looking closer at the code, I made some of the following observations. The code never takes into account the number of buys the player has, nor the number of supply that are left in the pile for the specific card. This is the root problem of the code. It is failing because it doesn’t check to see if your buy count is legitimate, you could have 0 for your num buys. Nor is it checking to see if there is any of that card left. Checking these conditions should solve the problems with the random tests.*

*Nicolgl: when testing this person’s code, I was undable to compile their code. It was givning me errors that I didn’t want to try and solve. I attempted a couple quick fixes, but it seemed more involved and therefore this person can change their code to compile themselves.*

*Tangke: this person’s code also didn’t compile. I noticed that it was certain errors with their makefile accepting C99 standards. Maybe their makefilie is formatted differently, but again, I didn’t want to try and change it. Also they had redefinitions of certain constants in their code which was preventing compilation. It shouldn’t take much to fix. The main problems are on line 661, 682 and 684 of their code, dominion.c, inside their implementation of Adventurer. Once this is fixed, I should be able to test their code for buycard.*

*Mcconnjo: This person’s code actually compiled and gave convincing results. The first random test case for buying any random card, always succeeded. This is good, you can buy any card. However the other 2 test cases failed badly. The second random test case, to see if they can buy the card with a random number of buys fails 32% of the time. They are probably not looking to see if the number of buys if enough to buy the card. Even more surprising is that the 3rd test case, testing with random number of coins failed 40% of the time. This is not very good at all. When I took a closer look at their code, I noticed a lot of minor things. It is very similar to the original code, except for a few things. It makes sure to decrement the coin count, and the buy count, but doesn’t actually check to make sure the player has enough buys or cards prior to buying. This should be something that is fixed in order to get the code working fully. Just add a couple if conditions looking at the numbuys and the numcoins and probably the supply count, and you should have working code.*

*Nguyenta: Since I noticed that the code is in fact the original code, it has not been changed what so ever, I made these observations. The original code will work in some ways. Using the first set of random tests I noticed that the code will be able to buy every card in the deck. It passed all the random tests of the first test case to test buying any random card. However on the other 2 test cases, it really failed. Test case 2 failed 1717/5000 times, which is a fail rate of 34%. It really doesn’t work. Then the 3rd random test case also failed miserable, with 41% fail rate. So the original code is really broken. I understand why it was offered as one of the code segments that needed to be edited, it is broken.*

***Results:*** *This is where I am going to list the information I got from gcov on how much of the code I covered. In the future, I will want to have test cases that test more lines of code. Originally, I was getting only about 18-20% with just with individual unit test cases, but this really helped with the random testing, and increased the percentage. I am hoping by the end I can get it to about 70-80%*

|  |  |
| --- | --- |
| User tested on: | Gcov coverage test %: |
| Taylodav (my code) | 24.13% |
| Lesliew | 28.03% |
| Ellignsn | 21.73% |
| Shearini | 21.30% |
| Omaara | 22.47% |
| Nicolgl | 0% - didn’t compile |
| Tangke | 0% - didn’t compile |
| Mcconnjo | 21.68% |
| nguyenta | 21.73% |

***Plan:*** *In the future test cases, I would like to use some of the delta debugging tools as I really like using python. Also I have noticed we are going to be using some other various testing techniques. This would be really cool to use, expand my knowledge and experience with various test tool chains. Finally, and I mentioned this earlier, I would like to improve my coverage testing quite a lot. I would hope to expand to drawcard, and test the adventurer implementation a bit more. I barely touched it, and it would interesting to test as it has such a variety of effects on the player.*