Assignment 3
EECS 362 FALL SEMESTER 2018

Oregon State University

Vinny Harris-Riviello

October 27, 2018

OVERVIEW & PURPOSE

Plan

Part 1:

Test 4 functions that are no card implementations or card effects

1. unittest1.c for compare()
2. unittest2.c for getCost()
3. unittest3.c for updateCoins()
4. unittest4.c for gainCard()

Part 2:

Tests 4 Dominion cards implemented in dominion.c

1. cardtest1.c for adventurer effect
2. cardtest2.c for smithy effecr
3. cardtest3.c for
4. cardtest4.c for

.

BUGS

I encountered multiple bugs, some were harder to debug, because the were more in the correctness of the code. With feast for example I got into an infinite loop when I was testing because I wanted to test all the branches, but testing 2 of its branches they have not way out of the loop which made it really hard to figure out a way to test.

I encountered other simple bugs that the unit test made it easy to find that something was wrong with the code. I had a couple seg fault that took awhile to find again, even though I had introduce them, but I resisted looking at my notes because I wanted to do it through the tests,

I found that unit test is very useful, I wish we had started from scratch more like TDD, I feel there is much to learn.

I got another bug because the code is not discarding the player card and the unit test showed that the player does not end with the correct hand it should.

I had introduced a division by zero that got picked right away.

With the help of the makefile build found the segmentation fault I introduced on my bugs last assignment.

UNIT TESTING

My finding can be found in the section of unitestresults.out

1. Unitest1 90
2. Unitest2 1
3. Unitest 3 92
4. Unitest 4 100
5. Adventurer 88
6. Smithy 90
7. Council room 86
8. Feast 90

UNIT TESTING EFFORTS

This activity was very time consuming, I wish there was more instruction in actual practical stuff. I had originally made all my test with assert then I modified to create my own custom version.You can see it in the directory unitTest under testUtility and it made things simple. I pass in a parammeter list that includes 2 values to compare, the signature, something simple to say about the function being tested and a message.

It sill show Success or fail according and it will put the message tha makes it easier to see what the test is about.

This was a lot of work specially the makefile, I had to almost go from super small to continuosly grow it, because when I tried to create it at once it did not run.