Mark Moore

CS362-HW3

**Bugs**

The first bug I found was in the scoreFor(). The third of four loop should you use “state→deck[player]” for the limit of the loop instead of “state->discardCount[player]”. This error will have the Loop cycle only the amount of discard cards that player has at that time. So if one player has few of their cards in their discount pile at the end of a game, they are more likely to lose the game.

**Unit Testing**

All of my Unit Testing fell in a range of 15-20%.

**Unit Testing Efforts**

I spent the better part of 2 days just reading and understanding the dominion.c code. This code is very difficult to understand without commentary! I started to think that the major point of this assignment is to convince us students to verbosely comment on any and all code we write in the future. I put some new commentary into my branch of the code.

The first function I tested was the initializeGame() function. It took a while to figure out how the code worked together but, finally, I was able to organize a test sufficiently rigorous to display the efficacy of the initializeGame() function.

My test checks all the cards in the hand and deck of the player to sum the total amounts and arrive at the final number.

UnitTest2.c proves the scoreFor() function. My test incrementally adds cards which are worth a specific amount and records that amount with each addition.

UnitTest3.c tests the shuffle() function. I create an array of 30 elements containing various cards. Then, I copy that array into the array for Player 1. I then shuffle() that array and compare it with its previous version.

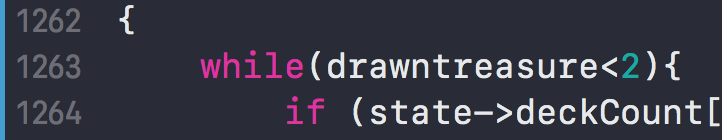
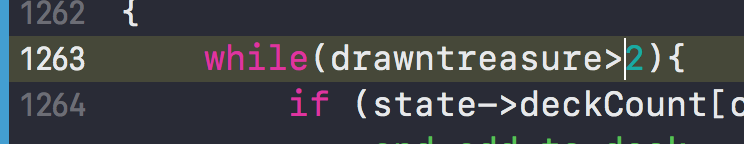
I repeat this process three times, showing that Random() function sufficiently shuffles the deck as to make it completely unpredictable.

Last, in UnitTest4.c, I prove the numHandCards() function. I manually change the size of the deck recorded for P1 and P2, then check the number though the numHandCards() function to see if it gives the correct number of cards for that player.

**Bugs**

Adventurer –

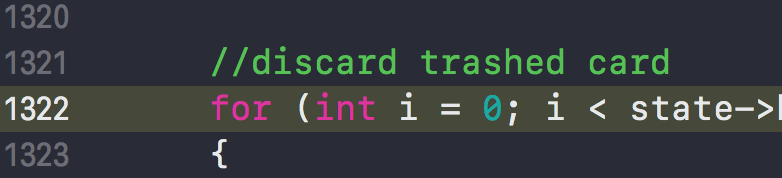
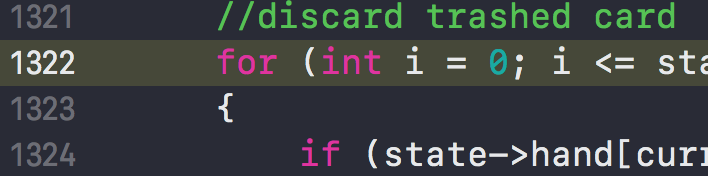
before after

For Adventurer, I simply changed the if statement checking if the drawntreasure is < to > 2. this won’t create a problem in every hand but over the course of a game it should create problems.

Mine –

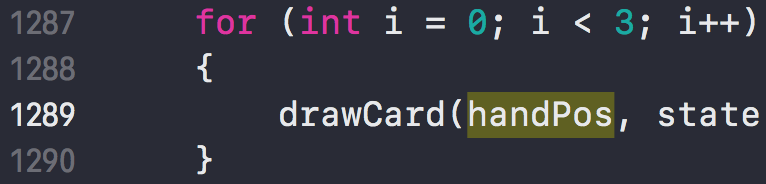
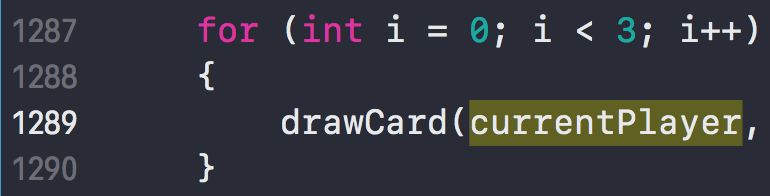
before after

For Mine, I changed the for-loop to discard cards to cycle one time more than it should by changing the “<” to a “≤” in the conditional part of the loop declaration.

Smithy –

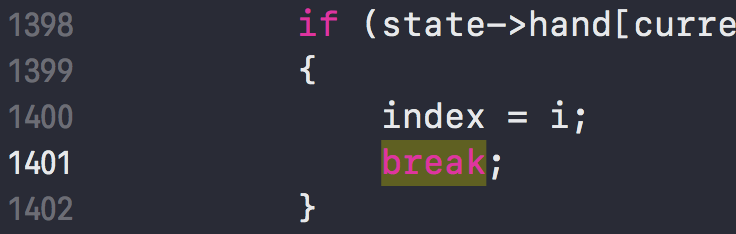
before after



For Smithy, I changed the integer sent to the drawcard function from “currentPlayer” to “handPos”. I’m not sure how this will affect the game as I don’t know enough about the long-term effects yet but I know current player only has 0 or 1 so a higher number will either crash the program or cause an error if proper checks are in place.

Treasure Map –

before after

For Treasure map, I commented out the “break;” in the if-statement nested in the for-loop so that index will keep rising in value as long as the condition is met.