Test report

Name: Shuai Peng

How I am starting my unit test:

First, I check the API in dominion, and understand each function what there do. I also check the structure, make sure I know what it is. Next I pick some basic function in dominion.c, and after I understand what the author means, and I write a code to test them. I used printf to make sure each step is what I expect. I test card effect by printHand and printState in each step, and I go to the website of dominion to check the card effect is correct or not.

Pre-condition: All unitTest and cardTest is assume that initialized game function is correct without any bugs.

My first unitTest test about the scoreFor function in dominion.c. I find bugs in scoreFor function, after each game initialized, there should have 3 estates, so this mean that we should at least have 3 score at the beginning of the game, but scoreFor return 1 score at the beginning of the game. The coverage of unitTest1 is “Lines executed:17.69% of 571”, it is not really high coverage, because I just test a simple function in this game.

My Second unitTest test about the numHandcards function in dominion.c. There are no bugs for numHandcards function. The coverage of unitTest2 is “Lines executed:16.29% of 571”. It is small function in dominion.c, so I get small coverage for this test.

My third unitTest test about the fullDeckCount function in dominion.c. There are no bugs for fullDeckCount function. The coverage of unitTest3 is “Lines executed:16.99% of 571”. It is still small part of dominion.c, and I get small coverage for this test.

My fourth unitTest test about the supplyCount function. Also there are no bugs for supplyCount function. The coverage of unitTest4 is “Lines executed:15.94% of 571”. Because I take small function in dominion.c, I get small coverage here.

My first cardTest test about the smithy card effect in dominion.c. No bugs for this card effect. The coverage of cardTest1 is “Lines executed:23.64% of 571”. It’s better than unitTest, because run line more than simple function.

My second cardTest test about the council room card effect in dominion.c. No bugs for this card effect. The coverage of cardTest2 is “Lines executed:24.34% of 571”

My Third cardTest test about the village card effect in dominion.c. Still no bug is found in this card effect. I use this cardTest3 to test my classmate’s dominion.c, and I found bugs in his

dominion.c, I have commute to him. The coverage of cardTest2 is “Lines executed:23.12% of 571”, It’s same as before.

My Fourth cardTest test about the baron card effect in dominion.c. I found bugs in this card effect. After I use baron, I can’t throw it out of my hand, baron defends on my hand, and if I choose to discard estate, I can get +4 coins too. I also can’t get one more estate if I don’t discard estate. The coverage of cardTest4 is “Lines executed:25.57% of 571”.

==========

How I am starting my random test.

First, I create file to test the cardEffect in dominion.c, and then I use the random function which is contain in standard C library. For example srand() and rand() function. Finally, for each random test, I check the number of the deck, the number of the hand, and also the number of buy, to make sure everything is correct. I also print those value on the screen for my view debugging.

For the smithy card: There is no bugs, I check the number of hand, and the number of deck. It’s completely correct. The total coverage is “Lines executed:26.62% of 571”

The function coverage of this test is 100% cover.
Function 'cardSmithy'
Lines executed:100.00% of 5
Branches executed:100.00% of 2
Taken at least once:100.00% of 2
Calls executed:100.00% of 2

For the council room card: There is also no bugs, I check the number of hand, and also check the buy. It’s correct. The total coverage is “Lines executed:24.69% of 571”.

The function coverage of this test is also 100% cover.
Function 'cardCouncil_room'
Lines executed:100.00% of 8
Branches executed:100.00% of 6
Taken at least once:100.00% of 6
Calls executed:100.00% of 2

For adventurer card: I found the number of hand is not correct, there is bug for adventurer card, but the it gets treasure card from the deck, so that’s part of function correct. After I check the value, it’s like that adventurer after used is still in our hands, function didn’t delete this card after used. The total coverage is “Lines executed:26.62% of 571”

The function coverage of this test is also 100% cover.
Function 'cardAdventurer'
Lines executed:100.00% of 15
Branches executed:100.00% of 12
Taken at least once:100.00% of 12
Calls executed:100.00% of 2

The coverage of random test is little higher than unit test, but random test give some surprise value input, so it may find some stranger bugs in our test.

==========

How I am starting my whole game generator test.

First, I make random generator, so I can have different deck card for each game, and also generate random players in this game, but it will no bigger than max player. And I use initializeGame function to set up the base of the game. I limit user buying and play action, each card in supply only can bought and played once. This is make sure my higher coverage for this test. Finally, starting this game until game over.

The coverage of this whole game generator random test is normally bigger than 60% in whole dominion.c

Using diff tools in Unix is hard to local the bugs, because we don't know who is really correct action in this test.

==========

The status and view of the reliability of the Dominion code of my classmate.

I check Chenchon's dominion game. After he fix bug which is found by him, the whole game usually run correct. After I using my whole game test, most of card effect is correct, but there are still little bugs inside of card effect, hope he can fix those bug. The reliability of his dominion code is mostly reliable.

I also check Duanyi's dominion game. I using my own unit test, random test and whole game generate test in his file. It's like that she didn't fix the dominion.c file a lot. I still can find the same bug as she mentioned before. The reliability of her dominion code is not high.