Name: Spencer Hubbard
Date: 6/9/15
CS 362 Project Report

This is a test report documenting my experience testing the dominion code in the projects repository with the URL `https://github.com/cs362sp15/projects`.

I wrote a test suite consisting of unit tests for four functions and four kingdom cards documented in the the public header file `dominion.h`. The functions that I wrote unit tests for are `initializeGame`, `numHandCards`, `handCard`, and `fullDeckCount`. The kingdom cards that I wrote unit tests for are the smithy, mine, village, and adventurer cards. The unit test for `initializeGame` checks that it returns a negative number—which indicates failure according to the documentation given in `dominion.h`—when called with a number less than two for the number of players. The unit test for `numHandCards` checks that it returns zero for the number of cards in an empty hand. The unit test for `handCard` checks that it returns a negative number when called with a hand index that is greater than the size of the hand. The unit test for `fullDeckCount` checks that it returns the correct number of cards when one of the hand, deck, or discard are empty. The unit tests for the smithy and village cards check for the correct card effect. The unit test for the mine card checks that the player is not allowed to gain a treasure card costing greater than three more when playing the mine card. The unit test for the adventurer card checks for the correct card effect when there are no treasure cards in the deck and two copper cards in the discard.

I ran my test suite on two mutants of the dominion code:
`projects/trunk/hubbarsp/dominion/dominion.c`
`projects/trunk/Cronisej/dominion/dominion.c`
The unit tests for the `handCard` function and the mine card both failed in the first implementation. This means that the first implementation of dominion is not reliable. The bug identified by the unit test for the mine card was corrected in the first implementation and is documented in the text file `debugging.txt`. The unit tests for the `handCard` function and the smithy, mine, and village cards failed in the second implementation. This means that the second implementation of dominion is also not reliable. The bugs in the second implementation discovered by the unit tests for the kingdom cards are documented in the following bug reports:
`projects/trunk/Cronisej/bugreports/hubbarsp1.txt`
`projects/trunk/Cronisej/bugreports/hubbarsp2.txt`
`projects/trunk/Cronisej/bugreports/hubbarsp3.txt`

I wrote random testers for the mine and adventurer cards. The random tester for the mine card picks two random treasure cards and attempts to play the mine card by trashing one of the treasure cards and gaining the other. Note that if the player should not be allowed to play the mine card with these two treasure cards, then the game state should not change. The random tester for the adventure card inserts two copper cards into a stack of kingdom cards, splits the resulting stack of cards between the deck and discard, and attempts to play the adventurer card. Note that the split may occur so that one of the deck or discard contains all the cards in the stack and the other contains none.

I wrote a random tester that plays a complete game of dominion. My implementation of this random tester is actually quite elegant. At the beginning of the game and after every action or buy, a copy of the game state is enqueued into a game state queue. When the game is finished, each game state in the queue is dequeued and recorded in a text file. This means that the entire sequence of changes to the game state during the course of the game is recorded in this text file.

I ran my random tester on two mutants of the dominion code:
`projects/trunk/hubbarsp/dominion/dominion.c`
`projects/trunk/alzamilb/dominion/dominion.c`
The first difference in the output text file between these two mutants occurs on line 229. This difference occurs when a player attempts to play the mine card by trashing a silver card and gaining a gold card. In both implementations, the value returned by `playCard` is `-1`. This indicates failure and should not change game state according to the documentation. However, the player should be allowed to play the

mine card in this situation. Moreover, the game state does changes in both mutants, albeit in different ways. In one mutant, the player's coin count increases by one, the played card count increases by one, the hand count decreases by one, and the mine and silver cards are removed from the player's hand while a gold card is added to the player's hand. In the other mutant, the player's coin count increases by three, the played card count increases by one, the hand count remains the same, and the mine card is replaced by a gold card in the player's hand.

I also implemented tarantula for `projects/trunk/hubbarsp/dominion/dominion.c`. The development of tarantula and it's use in debugging is documented in the text file `tarantula.txt`.