CS 362 Software Engineering Fall 2018
Student: Daniel Krueger
Oct 27, 2018
Assignment 3 – Unit Tests / Card Tests Dominion Code

Assignment Checklist

- ❑ code: 4 Unit Tests, 4 Card Tests
- ❑ Makefile
- ❑ gcov – Figure out what is not covered by testing
- ❑ testResults – combined file called 'unittestresults.out'

Written Assignment

- ❑ Submit "Assignment-3.pdf"
  - o 3 sections Bugs, Unit Testing, and Unit Testing Efforts
- ❑ Submit your complete dominion code under /projects/kruegerd/dominion
- ❑ Submit under youronid-assignment-3 before due date
- ❑ Add Comment in Canvas to give URL of your fork
  - o https://github.com/dwkrueger2/CS362-004-F2018.git
  - o code is on branch kruegerd-assignment-3

Unit Testing Efforts

8 Unit Tests were develop to test the functionality of the provided source code dominion.c. Since the documentation is not very specific on the goals of each function tested the rational for expected behavior was derived from dominion game rules and sometimes the program flow its self.

The 4 functions chosen for testing include: shuffle, buyCard, endTurn, and isGameOver.

The tests run are indicated in the tables below. The approach to the shuffle was to test whether the count and card quantities in the possession of the player stayed the same. The Shuffle appeared to pass all tests. The Buy Card Function was tested against not letting buy attempts when there was no supply, no player buy options left, or insufficient coins for the desired card purchase. All tests appeared to pass.

The strategy for the end of the turn was to check the state of the player whose turn ended and see how their game state changed. The rules for the cleanup phase of the game were evaluated. There were numerous little details that did not seem to match with the proper end turn activites that the dominion game indicates in its cleanup phase.

In each of the card effect the effect of the introduced bug was felt. The strategy for the smithy card was to test whether the game state for actions and buys were not affected, while the cards in hand were affected and the playedCardCount and associated variables were effected.

The affect of the Adventurer playing was difficult. The playing of this card could force a shuffle to occur with the discarded deck. So we looked at the potential permutations associated with having a deck that has up to no coins in the first 5 cards. The deck and discard piles were contrived to evaluate the effect of the adventurer card when a shuffle did and did not occur.

The cutpurse strategy was similar to the permutation problem but this time we had to alter the players hands according to the desired states we wanted to check. It was not chosen to look at every permutation for the players cards. But we did look at whether the person playing the card was effected. The total hand counts and hand count values of the non-current players were evaluated.

The strategy for the Gardens effect was to contrive sever hands that should have different scores. These scores were strategically chose to include a range of card possession. The scores of the hand was then compared with the game state before and after. The game state should be unchanged because this card is not a playable card.

Function	Test: Description
Shuffle	T1: Shuffle was run 50 times and evaluated to make sure it did not return the same deck. (used probability < 5% as success.
	T2: No Errors during 50 runs meaning no shuffled deck was the same as the previous deck.
	T3: No change in card set (i.e. same number duchy's, smithy's copper's etc..
	All Tests: **PASS**

Function	Test: Description
BuyCard	T1: Purchasing all smithy cards increases player discard pile by 8
	T2: Purchasing Card Reduces smithy supply level by 8
	T3: Smithy Cards Left: 0 attempt to purchase should fails.
	T4: Number of Buys Left: 0 attempt to purchase should fail.
	T5: Number of coins left: 2 attempt to purchase province should fail.
	T6: Game Phase: Action attempt to purchase province should fail for purchase not in Buy Phase of the game.
	T7: gameStates before and after attempted buys that fail should equal eachother
	All Tests: **PASS**

Function	Test: Description
endTurn	T1: **PASS** Previous Hand Cards (#6) added to discard, previous discard (#0) result discard pile (#6) T2a: **FAIL** Card Content is exactly the same before and after end turn for player 0
	T2b: **FAIL** Total Cards before end turn (# 17) equal count at end of turn (#16) for player: 0
	T3: **PASS** Top Discard pile card was in hand at end of turn
	T4a: **PASS** Next Player Correct after endTurn. Current Player: 0 Next Player: 1
	T4b: **FAIL** Coins are set to 0: state.coins = 4
	T4c: **PASS** Number of Actions is 1: state.numActions = 1
	T4d: **PASS** Number of Buys is 1: state.numBuys = 1
	T5: **FAIL** Player has 5 cards in hand after endTurn
	T6: **PASS** gameState.phase set to Action(0): state.phase = 0
	Unit Tests 3 - all tests: **FAIL**

Function	Test: Description
isGameOver	T1: **PASS** Returns False (0) after initial setup
	T2: **PASS** Returns True when Province Supply = 0
	T3: **PASS** Returns True when Estate,Duchy, & Smithy Supply = 0
	T4: **PASS** Returns True when Estate,Duchy, & Curse Supply = 0
	T5: **PASS** Returns True when smithy, adventurer, cutpurse Supply = 0
	T6: **PASS** Returns True when smithy, adventurer, cutpurse, gardens, village Supply = 0

CS 362 Software Engineering Fall 2018
Student: Daniel Krueger
Oct 27, 2018
Assignment 3 – Unit Tests / Card Tests Dominion Code

Function	Test: Description
cardEffectSmity	T1: **FAIL** Returns true when handCount increases by 3
	T2: **FAIL** Returns true when deckCount decreases by 3
	T3: **PASS** Returns true when numActions remains the same
	T4: **PASS** Returns true when coins remains the same
	T5: **PASS** Returns true when numBuys remains the same
	T6: **PASS** Returns true when playedCardCount increases by 1
	T7: **PASS** Returns true when Total Cards owned by player remains the same.
	Card Test 1- Smithy - all tests: **FAIL**

Function	Test: Description
cardEffectAdventurer	T1: **PASS** Card count in Players possession remains the same
	T2: **PASS** Number of Coins in Hand increases
	T3: **PASS** Value coins in Hand also increase
	T4: **FAIL** Card Count Remains the Same
	T5: **PASS** Card Count in Hand = +2
	* 32 Permutations run on each of the above tests. It should be noted that T4 is a false negative and T5 was a false positive.

Function	Test: Description
cardEffectCutPurse	T1: **PASS** Player 1's handCount reduces by 1
	T2: **FAIL** Player 1's discardCount reduces by 1
	T3: **FAIL** Player 1's total card count remains the same.
	T4: **PASS** Player 2's handCount reduces by 1
	T5: **FAIL** Player 2's discardCount reduces by 1
	T6: **FAIL** Player 2's total card count remains the same.
	T7: **PASS** Player 3's (the attacker) handCount reduces by 1
	T8: **PASS** Played Card Count increases by 1
	T9: **PASS** Player 3 total card count remains the same.
	T10: **PASS** Player 4's handCount remains the same
	T11: **PASS** Player 4's discardCount remains the same
	T12: **PASS** Player 4's total card count remains the same.
	Card Test 3 - CutPurse - all tests: **FAIL**

Function	Test: Description
cardEffectGardens	T1: **FAIL** Card unplayable - Gamestate should be unchanged.
	T2: **FAIL** Score for player 1 is 1 and should be: 0
	T3: **FAIL** Score for player 2 is 0 and should be: 10
	T4: **FAIL** Score for player 3 is 0 and should be: 12
	T5: **FAIL** Score for player 4 is 22 and should be: 18
	Card Test 4- Gardens - all tests: **FAIL**

BUGS

1. Shuffle - No bugs detected
2. BuyCard – No Bugs detected

3. endTurn – The number of cards the player has changes –
4. endTurn – The end state of coins should be 0 not 4
5. endTurn – the ending player does not retake up cards. – The affects players when attack cards are played
6. isGameOver = No bugs detected
7. Smithy = hand count does not increase by 3 and deck does not decrease by 3- this detected the inserted bug.
8. Adventurer – The card count in the had remained the same but this is a false positive because the player state actually changed to match the pre play state.
9. Adventurer – The hand count did not increase by 2 – This is a false negative because of the current player state changing for the game. The playing player's hand actually does increase by 2 but since the next player is being triggered, that player's hand count is less than the starting players.
10. cutpurse – Caught the error that the owned card count was changing, and the discard count was not increasing by 1. The bug is caught, and it will just take a little digging to see that this card is trashed instead of discarded.
11. GardenEffect = This really should do nothing and giving the trivial nature of this I went ahead and calculated the score variant for the cards the patient has and it was clear that the scores were not calculated correctly.

Current Player does not draw hand at the end of the turn. There are a number of attack cards in dominion that affect or are affected by the players having populated hands before their turn starts. Rules of dominion clearly state

> "Then, the player draws a new hand of 5 cards from his Deck. If there are not enough cards in his Deck, he draws as many as he can, shuffles his Discard pile to form a new face-down Deck, and then draws the rest of his new hand. Once the player has drawn a new hand of 5 cards, the next player starts his turn."

GCOV – Discussion

40.93% of the 557 lines of code were executed. The functions tested were specifically examined to see if there was a segment of the analyzed functions that were not executed during the testing.

shuffle – every line had at lease 2 executions. All of buyCard appears to have coverage. EndTurn appears to have had coverage and isGameOver all appeared to have coverage. The effect functions appeared to have coverage of every statement as well. I don't feel like we can have a fulminate discussion on the point of coverage with this project. The project appears to have all areas covered that were intended for us to test.

Table 1: Intentional Bug Insertions

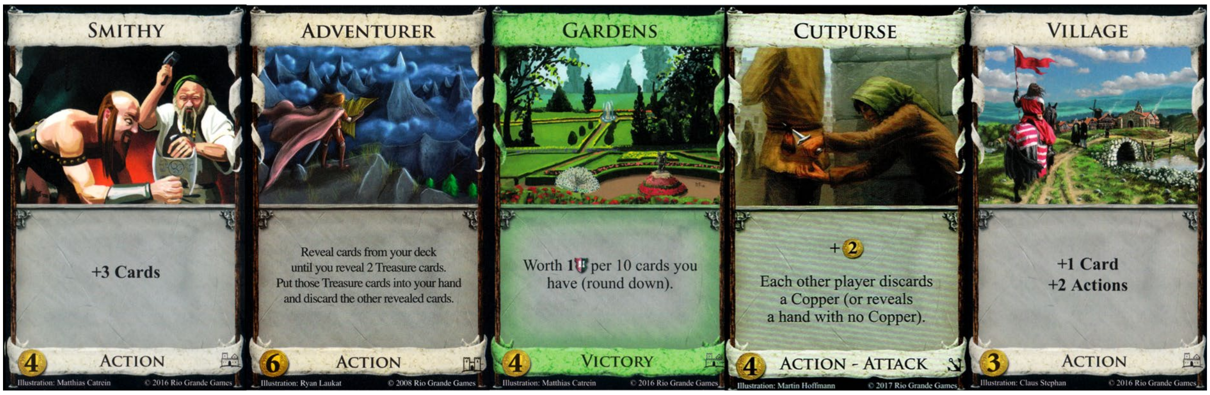
Function	Bug

CS 362 Software Engineering Fall 2018
Student: Daniel Krueger
Oct 27, 2018
Assignment 3 – Unit Tests / Card Tests Dominion Code

dominion.c/cardEffectAdventurer	Because code was copied and pasted, the player was incremented a second time. The following code is duplicated.
	
	int nextPlayer = currentPlayer + 1;
	if (nextPlayer > (state->numPlayers - 1)) {
	nextPlayer = 0;
	}

dominion.c/ cardEffectAdventurer	while (drawntreasure <= 2) { // BUG should cause an extra treasure card-- orignial --> while (drawntreasure < 2)
dominion.c/cardEffectCutPurse	Card is trashed instead of discarded. discardCard(j, i, state, 1); // BUG last integer – 0 indicates discard, 1 will cause it to trash
dominion.c/cardEffectGardens	Added drawCard instead of no effect
dominion.c/cardEffectVillage	Added an extra Draw Card

Figure 1: Cards Selected


SMITHY
+3 Cards
4
ACTION
ADVENTURER
Reveal cards from your deck until you reveal 2 Treasure cards. Put those Treasure cards into your hand and discard the other revealed cards.
6
ACTION
GARDENS
Worth 1 per 10 cards you have (round down).
4
VICTORY
CUTPURSE
+2
Each other player discards a Copper (or reveals a hand with no Copper).
4
ACTION - ATTACK
VILLAGE
+1 Card
+2 Actions
3
ACTION