CS362 Software Engineering II Section 400 Fall 2019
Final Project Part B

Table of Contents

Team Information:

Member Name	Email	ONID
Alexander Drath	<dratha@oregonstate.edu>	dratha
Eric Morgan	<morganer@oregonstate.edu>	morganer
Serena Tay	<tays@oregonstate.edu>	tays

Github (5 points)

1. Decide which student repository to use as a team. You will create two branches in this repository as directed below.

Repository URL:

https://github.com/AneresArsenal/CS362-F2019-FinalProject

Studentonid-finalproject-bugs branch URL:

https://github.com/AneresArsenal/CS362-F2019-FinalProject/tree/tays-finalproject-bugs

Files can be founds in this folder:

https://github.com/AneresArsenal/CS362-F2019-FinalProject/tree/tays-finalproject-bugs/projects/FinalProject-Bugs/dominion

Studentonid-finalproject-bugfree branch URL:

https://github.com/AneresArsenal/CS362-F2019-FinalProject/tree/tays-finalproject-bugfree

Files can be founds in this folder:

https://github.com/AneresArsenal/CS362-F2019-FinalProject/tree/tays-finalproject-bugfree/projects/FinalProject-BugFree

CS362 Software Engineering II Section 400 Fall 2019
Final Project Part B

Unit Tests (45 points)

Use your Test Plans from the Final Project Part A to create Unit Tests for each bug.
1. Create a studentonid-finalproject-bugs branch from the Master branch. Check out the studentonid-finalproject-bugs branch and create a folder under the projects folder called FinalProject-Bugs. Copy the dominion folder from the root folder into the FinalProject-Bugs folder. This dominion folder will have the original dominion source code but will not have any refactored code.

2. Create Unit Tests for each Bug Report using the Test Plan developed in Part A. Create the Unit Tests in the projects\FinalProject-Bugs\dominion folder.

3. Create a makefile and add a rule named “**unittestresults”** that will generate all the Unit Test source files, the dominion code, execute all the Unit Tests, and append complete test results including code coverage into a file called **unittestresults.out**. Use the gcov command options to produce coverage on **branches** and **functions** (-b -f). The .out file will contain the output of your tests along with coverage information.

4. Run the makefile to execute the Unit Tests and report the results. You should expect your asserts to fail.

5. In a document called ‘Final Project PartB’, document each Unit Test in a section called ‘Unit Tests’.

6. Push the studentonid-finalproject-bugs branch to github.

Dominion Bug 1 Unit Test

================================
Description
===========
“The last discardCard function call within the mine case statement of the cardEffect function is not actually trashing the card it is just putting it into the currentPlayers discard pile. The issue can be seen on line 836. The card requirements that the chosen card is trashed. The correction to the bug is to change the trashFlag value within the discardCard function call from 0 to 1.” **bug description by T Grasse extracted from Piazza post “Find and Fix Dominion Bugs”**

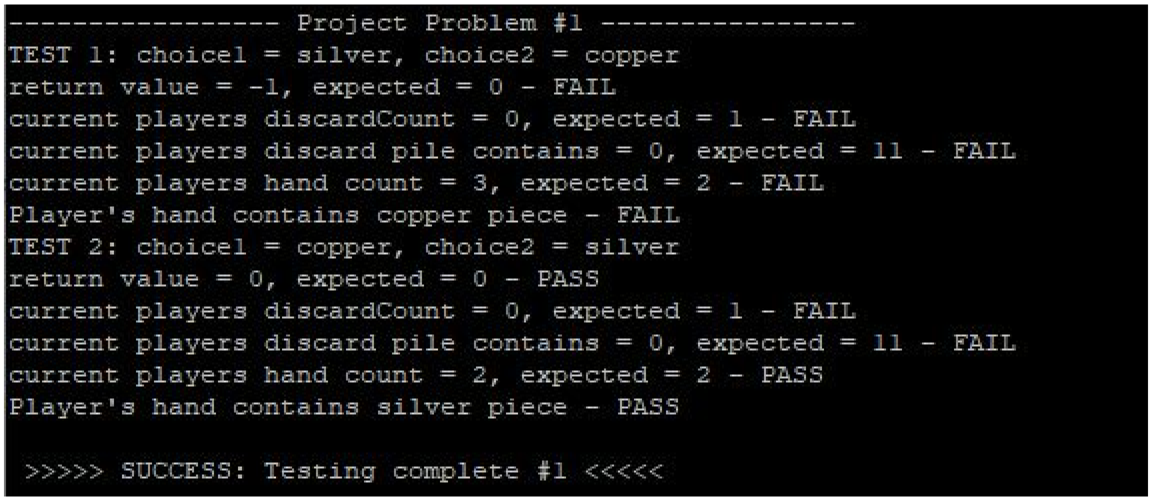
The tribute card does not trash the card when the discardCard function is called either but that is not a bug and therefore does not need to be addressed.

What is another bug that is tied to both of these issues is the fact that the discardCard function doesn’t even have a trash pile nor does it update the discard pile for the player with the discarded card as well. The discardCard function will need to be altered to actually function properly before any sort of testing can be performed on the mine card function. One shouldn't really be doing testing on the cards at all if basic necessary functions such as discardCard function isn't even written correctly.

Results

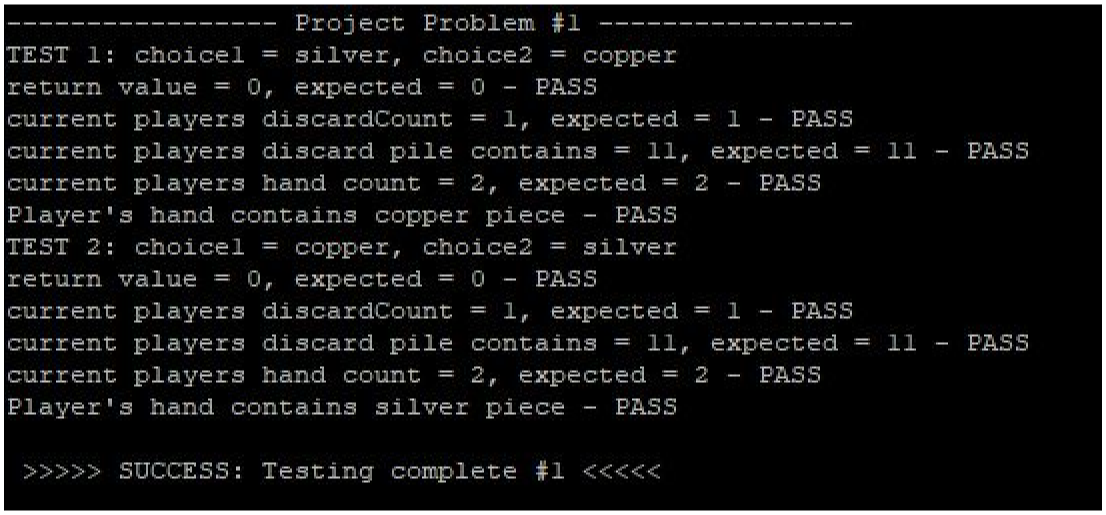
===========

Bugs:



```
----------------- Project Problem #1 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = -1, expected = 0 - FAIL
current players discardCount = 0, expected = 1 - FAIL
current players discard pile contains = 0, expected = 11 - FAIL
current players hand count = 3, expected = 2 - FAIL
Player's hand contains copper piece - FAIL
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 0, expected = 1 - FAIL
current players discard pile contains = 0, expected = 11 - FAIL
current players hand count = 2, expected = 2 - PASS
Player's hand contains silver piece - PASS

 >>>>> SUCCESS: Testing complete #1 <<<<<
```

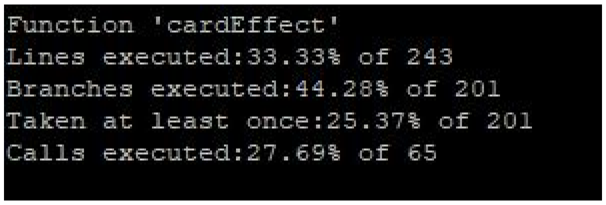
BugFree:



```
----------------- Project Problem #1 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains copper piece - PASS
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains silver piece - PASS

 >>>>> SUCCESS: Testing complete #1 <<<<<
```

Coverage

===========

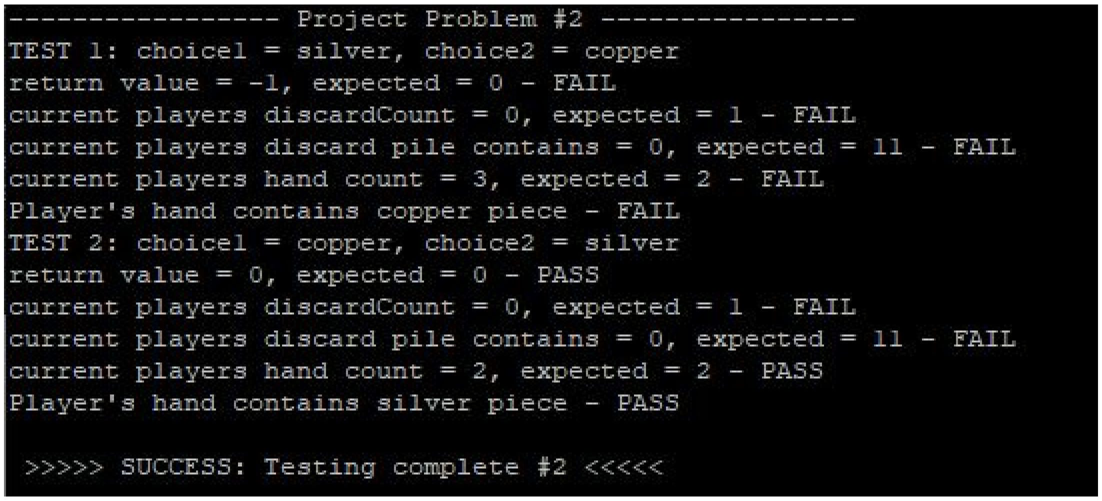


```
Function 'cardEffect'
Lines executed:33.33% of 243
Branches executed:44.28% of 201
Taken at least once:25.37% of 201
Calls executed:27.69% of 65
```

Dominion Bug 2 Unit Test

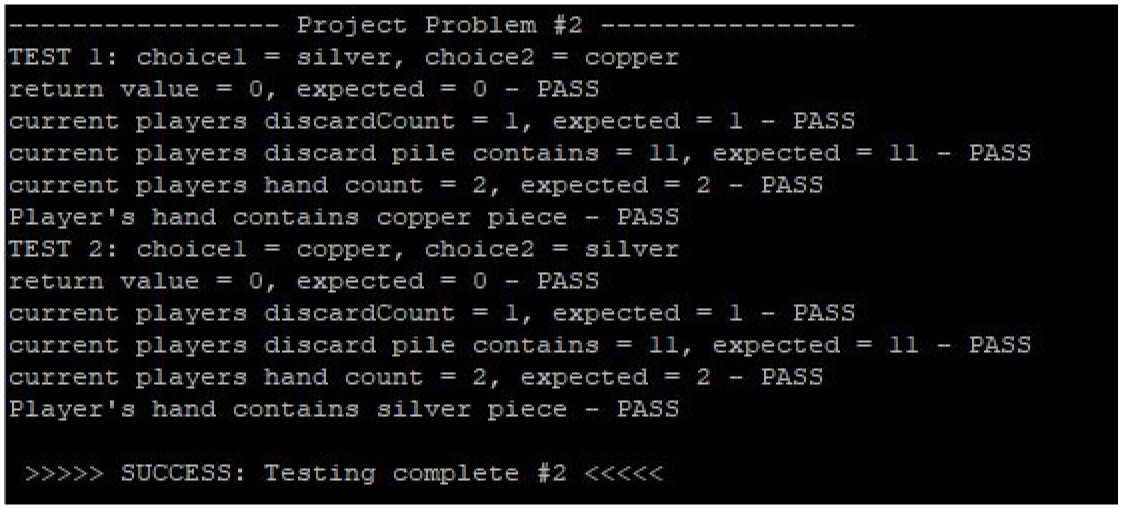
================================
Description
===========
“For the Mine switch statement in the cardEffect function (around line 821), it seems that it checks the cost of the card they trash against the cost of the card they want to buy incorrectly. It looks like if the cost of the treasure that they choose to trash plus 3, is greater than the cost of the card they want to buy, then it will return -1.” **bug description by Robert Saraceno extracted from Piazza post “Find and Fix Dominion Bugs”**
The statement is saying that if the choice1+2 is greater than choice2 then return -1 but this means that choosing an item that costs less than choice1 will fail when it shouldn’t. If you reverse the statements the proper action will occur

Results
===========
Bugs:



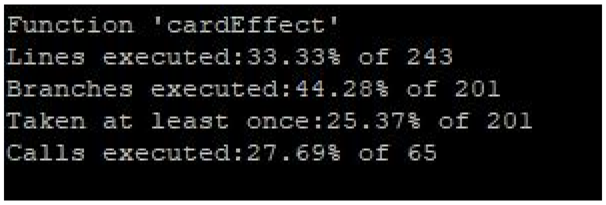
```
----------------- Project Problem #2 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = -1, expected = 0 - FAIL
current players discardCount = 0, expected = 1 - FAIL
current players discard pile contains = 0, expected = 11 - FAIL
current players hand count = 3, expected = 2 - FAIL
Player's hand contains copper piece - FAIL
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 0, expected = 1 - FAIL
current players discard pile contains = 0, expected = 11 - FAIL
current players hand count = 2, expected = 2 - PASS
Player's hand contains silver piece - PASS

 >>>>> SUCCESS: Testing complete #2 <<<<<
```

BugFree:



```
----------------- Project Problem #2 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains copper piece - PASS
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains silver piece - PASS

 >>>>> SUCCESS: Testing complete #2 <<<<<
```

Coverage
===========

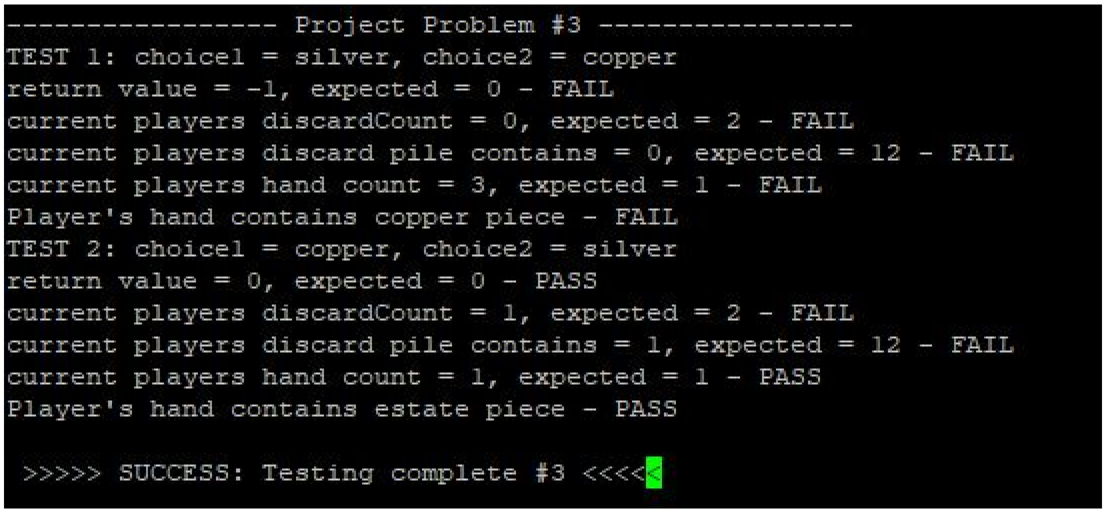


```
Function 'cardEffect'
Lines executed:33.33% of 243
Branches executed:44.28% of 201
Taken at least once:25.37% of 201
Calls executed:27.69% of 65
```

Dominion Bug 3 Unit Test

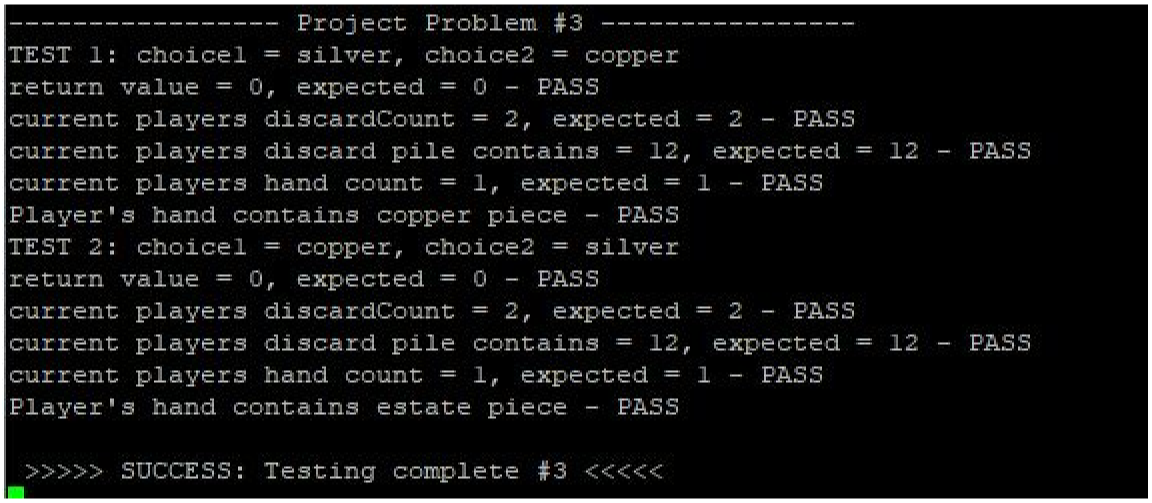
================================
Description
===========
"In the remodel case within the cardEffect function, the if statement that compares the two choice statements needs to be switched (around Line #846)." **bug description by Alexander Goodman extracted from Piazza post "Find and Fix Dominion Bugs"**
The statement is saying that if the choice1+2 is greater than choice2 then return -1 but this means that choosing an item that costs less than choice1 will fail when it shouldn't. If you reverse the statements the proper action will occur.

Results
===========
Bugs:



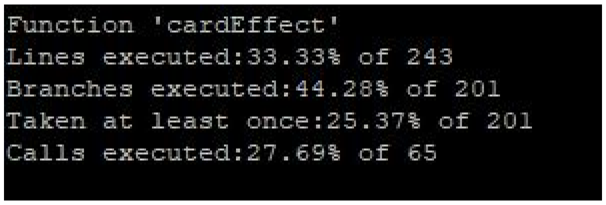
```
----------------- Project Problem #3 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = -1, expected = 0 - FAIL
current players discardCount = 0, expected = 2 - FAIL
current players discard pile contains = 0, expected = 12 - FAIL
current players hand count = 3, expected = 1 - FAIL
Player's hand contains copper piece - FAIL
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 2 - FAIL
current players discard pile contains = 1, expected = 12 - FAIL
current players hand count = 1, expected = 1 - PASS
Player's hand contains estate piece - PASS

 >>>>> SUCCESS: Testing complete #3 <<<<<
```

BugFree:



```
----------------- Project Problem #3 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = 0, expected = 0 - PASS
current players discardCount = 2, expected = 2 - PASS
current players discard pile contains = 12, expected = 12 - PASS
current players hand count = 1, expected = 1 - PASS
Player's hand contains copper piece - PASS
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 2, expected = 2 - PASS
current players discard pile contains = 12, expected = 12 - PASS
current players hand count = 1, expected = 1 - PASS
Player's hand contains estate piece - PASS

 >>>>> SUCCESS: Testing complete #3 <<<<<
```

Coverage
===========



```
Function 'cardEffect'
Lines executed:33.33% of 243
Branches executed:44.28% of 201
Taken at least once:25.37% of 201
Calls executed:27.69% of 65
```

Dominion Bug 4 Unit Test

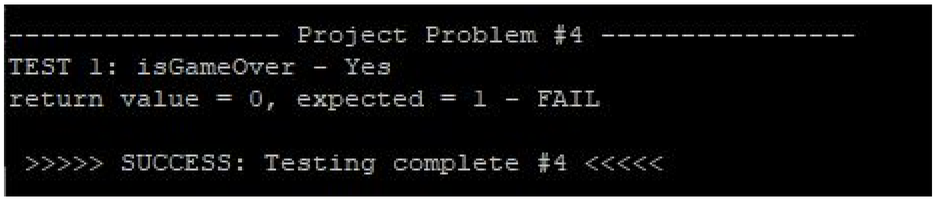
============================

Description

===========

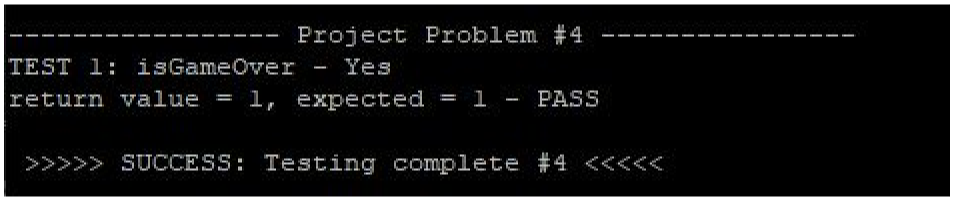
“There is a bug in the isGameOver function. While checking if there are 3 cards with a card count of 0, it only loops through 25 cards. The problem is that the supply count array has 27 items. treasure map is equal to 26 in the cards enumeration so the loop never properly loops through all the supply cards, it only loops through 25 of the cards (0-24) so two cards are never checked, it should loop through 27 cards (0-26).” **bug description by George Lenz extracted from Piazza post “Find and Fix Dominion Bugs”**

Results
===========
Bugs:



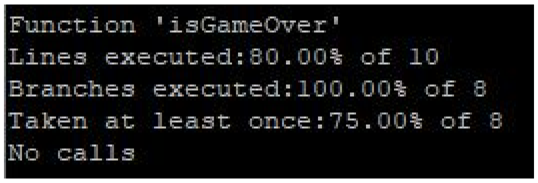
```
----------------- Project Problem #4 ----------------
TEST 1: isGameOver - Yes
return value = 0, expected = 1 - FAIL

 >>>>> SUCCESS: Testing complete #4 <<<<<
```

BugFree:



```
----------------- Project Problem #4 ----------------
TEST 1: isGameOver - Yes
return value = 1, expected = 1 - PASS

 >>>>> SUCCESS: Testing complete #4 <<<<<
```

Coverage
===========



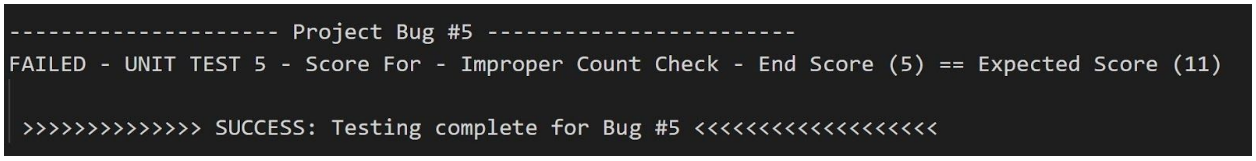
```
Function 'isGameOver'
Lines executed:80.00% of 10
Branches executed:100.00% of 8
Taken at least once:75.00% of 8
No calls
```

Dominion Bug 5 Unit Test

==============================
Description
===========
In the scoreFor function, the discardCount is being used for the deck count. This causes the score to be incorrectly counted. A unit test should be setup that places a known number of victory cards in both the discard pile, deck, and hand and checks to ensure the proper count is returned by the function.

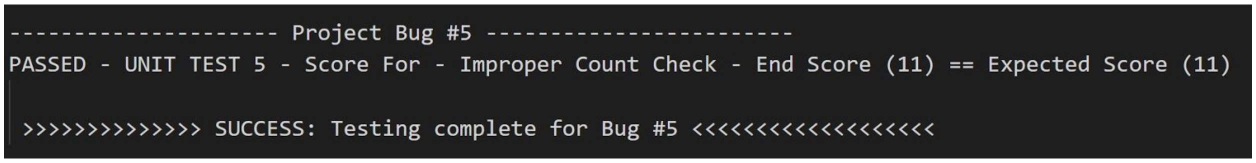
This bug is encountered when the number of cards in the deck exceeds the number number of cards in the discard pile, and when victory cards are in a position greater than discard count. The test, we will place estate cards, which are worth a single point, in the player’s hand, discard pile, and deck. 1 estate card will be placed in the player’s hand, 2 cards will be placed in the player's discard pile, and 8 cards will be placed in the deck. This should result in a final score of 8. As the discard pile will only contain two cards, the current scoreFor functionality will not count the full range of cards in the deck.

Results
===========
Bugs:



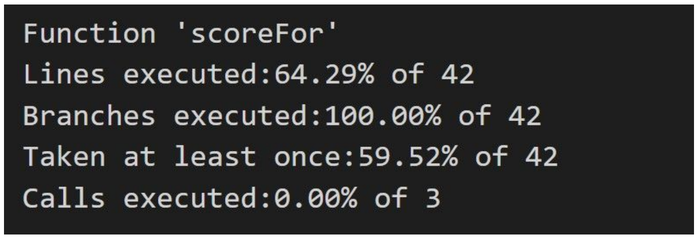
```
--------------------- Project Bug #5 ------------------------
FAILED - UNIT TEST 5 - Score For - Improper Count Check - End Score (5) == Expected Score (11)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #5 <<<<<<<<<<<<<<<<<<<
```

Bug Free:



```
--------------------- Project Bug #5 ------------------------
PASSED - UNIT TEST 5 - Score For - Improper Count Check - End Score (11) == Expected Score (11)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #5 <<<<<<<<<<<<<<<<<<<
```

Coverage
===========



```
Function 'scoreFor'
Lines executed:64.29% of 42
Branches executed:100.00% of 42
Taken at least once:59.52% of 42
Calls executed:0.00% of 3
```

Dominion Bug 6 Unit Test

================================
Description
===========
The Feast card allows a player to trash the card to gain a card worth up to 5. On line 762 Instead of updating the coins tally, it should enter the while loop and allow players to gain a card up to 5 coins without taking into account how much coins the player has. This effect updates the user's coin state, and does not restore it when complete. As such, the user could lose currently earned coins. Additionally, the card is not trashed when it is played.

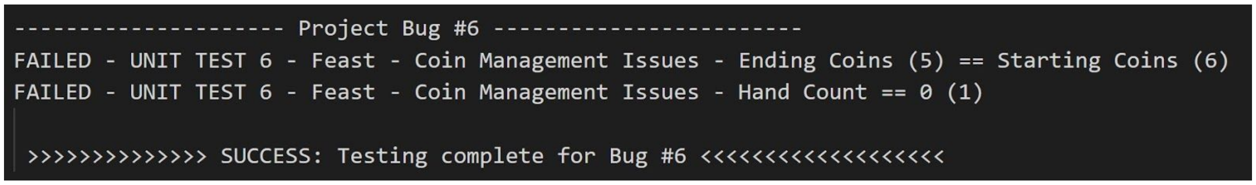
Players can earn additional coins when they play actions. These coins are bonus coins above those earned by holding treasure in their hand. To implement this card's effect the code changes the player's coins, and allows them to select a card. Unfortunately, the current implementation erases any bonus the player may have earned.

To trigger this bug and to ensure we have caught it, we will setup a game state that contains a known number of coins. We will then play the feast card and examine the number of coins after the card is played. The number of coins in the player's hand should remain unchanged after playing this card.

Results

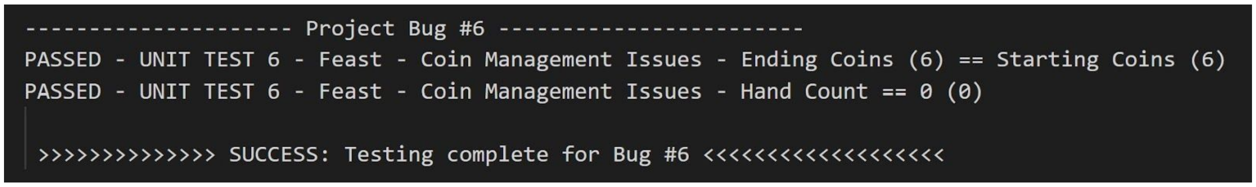
===========

Bugs:



```
--------------------- Project Bug #6 ------------------------
FAILED - UNIT TEST 6 - Feast - Coin Management Issues - Ending Coins (5) == Starting Coins (6)
FAILED - UNIT TEST 6 - Feast - Coin Management Issues - Hand Count == 0 (1)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #6 <<<<<<<<<<<<<<<<<<<
```

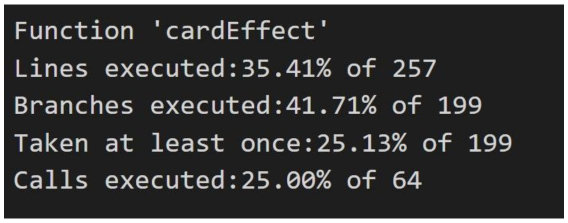
Bug Free:



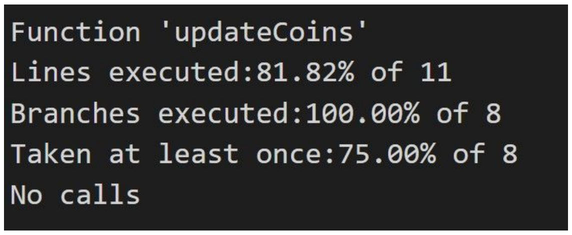
```
 --------------------- Project Bug #6 ------------------------
 PASSED - UNIT TEST 6 - Feast - Coin Management Issues - Ending Coins (6) == Starting Coins (6)
 PASSED - UNIT TEST 6 - Feast - Coin Management Issues - Hand Count == 0 (0)

  >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #6 <<<<<<<<<<<<<<<<<<<
```

Coverage

===========



```
Function 'cardEffect'
Lines executed:35.41% of 257
Branches executed:41.71% of 199
Taken at least once:25.13% of 199
Calls executed:25.00% of 64
```



```
Function 'updateCoins'
Lines executed:81.82% of 11
Branches executed:100.00% of 8
Taken at least once:75.00% of 8
No calls
```

Dominion Bug 7 Unit Test

==============================

Description

===========

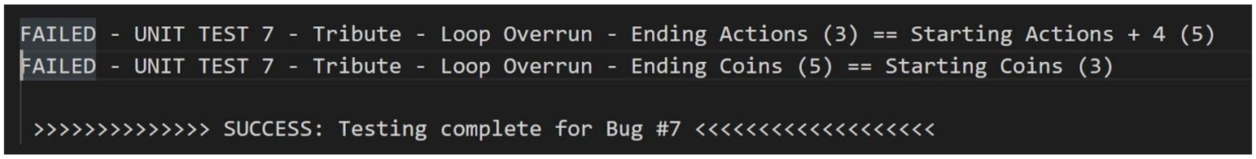
Tribute card bug around line 1075. The for loop overruns the array. The tributeRevealedCards array has two elements and the loop is written to iterate 3 times. This currently awards the player three bonuses.

This bug is currently triggered anytime the card effect is run. To test for specific bonuses we will add two different action cards to the top of the next player's deck. This should grant the player a +4 action bonus only.

Results

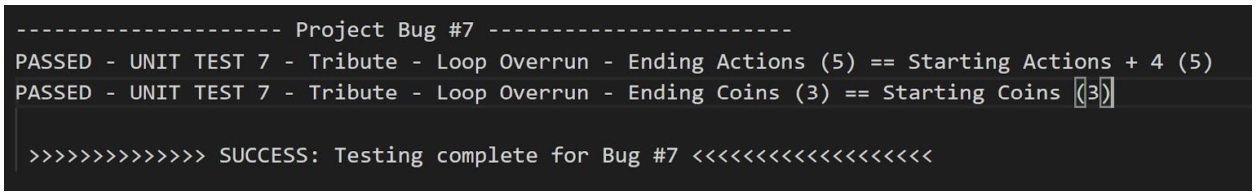
===========

Bugs:



```
FAILED - UNIT TEST 7 - Tribute - Loop Overrun - Ending Actions (3) == Starting Actions + 4 (5)
FAILED - UNIT TEST 7 - Tribute - Loop Overrun - Ending Coins (5) == Starting Coins (3)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #7 <<<<<<<<<<<<<<<<<<<
```

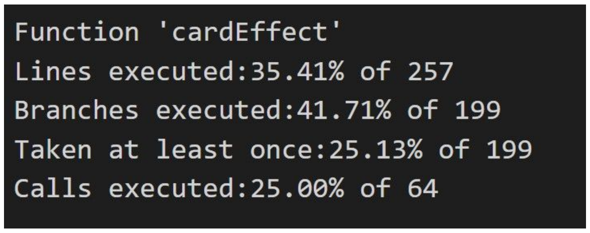
Bug Free:



```
--------------------- Project Bug #7 ------------------------
PASSED - UNIT TEST 7 - Tribute - Loop Overrun - Ending Actions (5) == Starting Actions + 4 (5)
PASSED - UNIT TEST 7 - Tribute - Loop Overrun - Ending Coins (3) == Starting Coins (3)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #7 <<<<<<<<<<<<<<<<<<<
```

Coverage

===========



```
Function 'cardEffect'
Lines executed:35.41% of 257
Branches executed:41.71% of 199
Taken at least once:25.13% of 199
Calls executed:25.00% of 64
```

Dominion Bug 8 Unit Test

================================

Description

===========

The number of bonus coins from actions does not appear to be recorded correctly. Originally this bug was reported as being isolated to card effect, though this bug exists in any place within the code where a bonus is being granted as part of a card effect. The following effects are affected:

1. Baron
2. Cutpurse
3. Embargo
4. Minion
5. Salvager
6. Steward

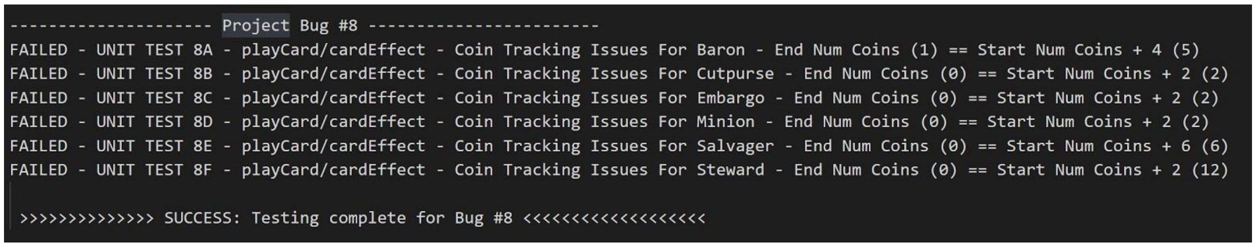
The current workflow invokes a card through the playCard function. This function then invokes the selected card through the cardEffect function passing in a coin_bonus argument that tracks coin bonuses added from played cards. The above detailed cards to not utilize the bonus parameter when added coins, but instead mutate the coin state directly. When code execution returns to the playCard function, we immediately run the updateCoins function. This has the effect of removing any coin bonuses that have been added to the coin state. In addition, as players can earn bonus across multiple actions, this flow removes all bonuses from the player's coin state.

Six separate unit tests will be setup for each of the cards above. Each test will initialize a game state and add the applicable card as the first in the player's hand, and will then call the playCard function to play the card with the relevant choices made to trigger the card's coin bonus. We will then check that the applicable coin bonus has been added.

Results

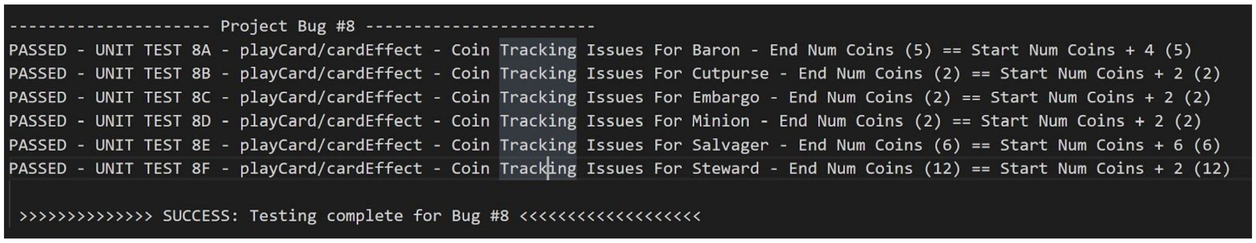
===========

Bugs:



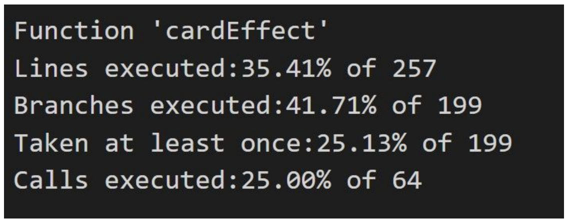
```
--------------------- Project Bug #8 -----------------------
FAILED - UNIT TEST 8A - playCard/cardEffect - Coin Tracking Issues For Baron - End Num Coins (1) == Start Num Coins + 4 (5)
FAILED - UNIT TEST 8B - playCard/cardEffect - Coin Tracking Issues For Cutpurse - End Num Coins (0) == Start Num Coins + 2 (2)
FAILED - UNIT TEST 8C - playCard/cardEffect - Coin Tracking Issues For Embargo - End Num Coins (0) == Start Num Coins + 2 (2)
FAILED - UNIT TEST 8D - playCard/cardEffect - Coin Tracking Issues For Minion - End Num Coins (0) == Start Num Coins + 2 (2)
FAILED - UNIT TEST 8E - playCard/cardEffect - Coin Tracking Issues For Salvager - End Num Coins (0) == Start Num Coins + 6 (6)
FAILED - UNIT TEST 8F - playCard/cardEffect - Coin Tracking Issues For Steward - End Num Coins (0) == Start Num Coins + 2 (12)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #8 <<<<<<<<<<<<<<<<<<<
```

Bug Free:

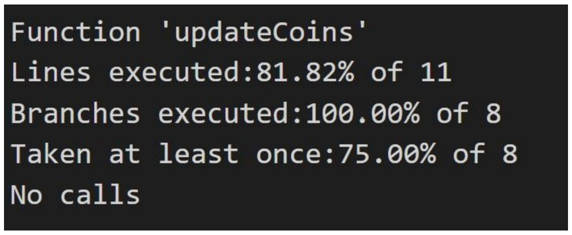


```
--------------------- Project Bug #8 -----------------------
PASSED - UNIT TEST 8A - playCard/cardEffect - Coin Tracking Issues For Baron - End Num Coins (5) == Start Num Coins + 4 (5)
PASSED - UNIT TEST 8B - playCard/cardEffect - Coin Tracking Issues For Cutpurse - End Num Coins (2) == Start Num Coins + 2 (2)
PASSED - UNIT TEST 8C - playCard/cardEffect - Coin Tracking Issues For Embargo - End Num Coins (2) == Start Num Coins + 2 (2)
PASSED - UNIT TEST 8D - playCard/cardEffect - Coin Tracking Issues For Minion - End Num Coins (2) == Start Num Coins + 2 (2)
PASSED - UNIT TEST 8E - playCard/cardEffect - Coin Tracking Issues For Salvager - End Num Coins (6) == Start Num Coins + 6 (6)
PASSED - UNIT TEST 8F - playCard/cardEffect - Coin Tracking Issues For Steward - End Num Coins (12) == Start Num Coins + 2 (12)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #8 <<<<<<<<<<<<<<<<<<<
```

Coverage
===========



```
Function 'cardEffect'
Lines executed:35.41% of 257
Branches executed:41.71% of 199
Taken at least once:25.13% of 199
Calls executed:25.00% of 64
```



```
Function 'updateCoins'
Lines executed:81.82% of 11
Branches executed:100.00% of 8
Taken at least once:75.00% of 8
No calls
```

Dominion Bug 9 Unit Test

=============================

Description
===========

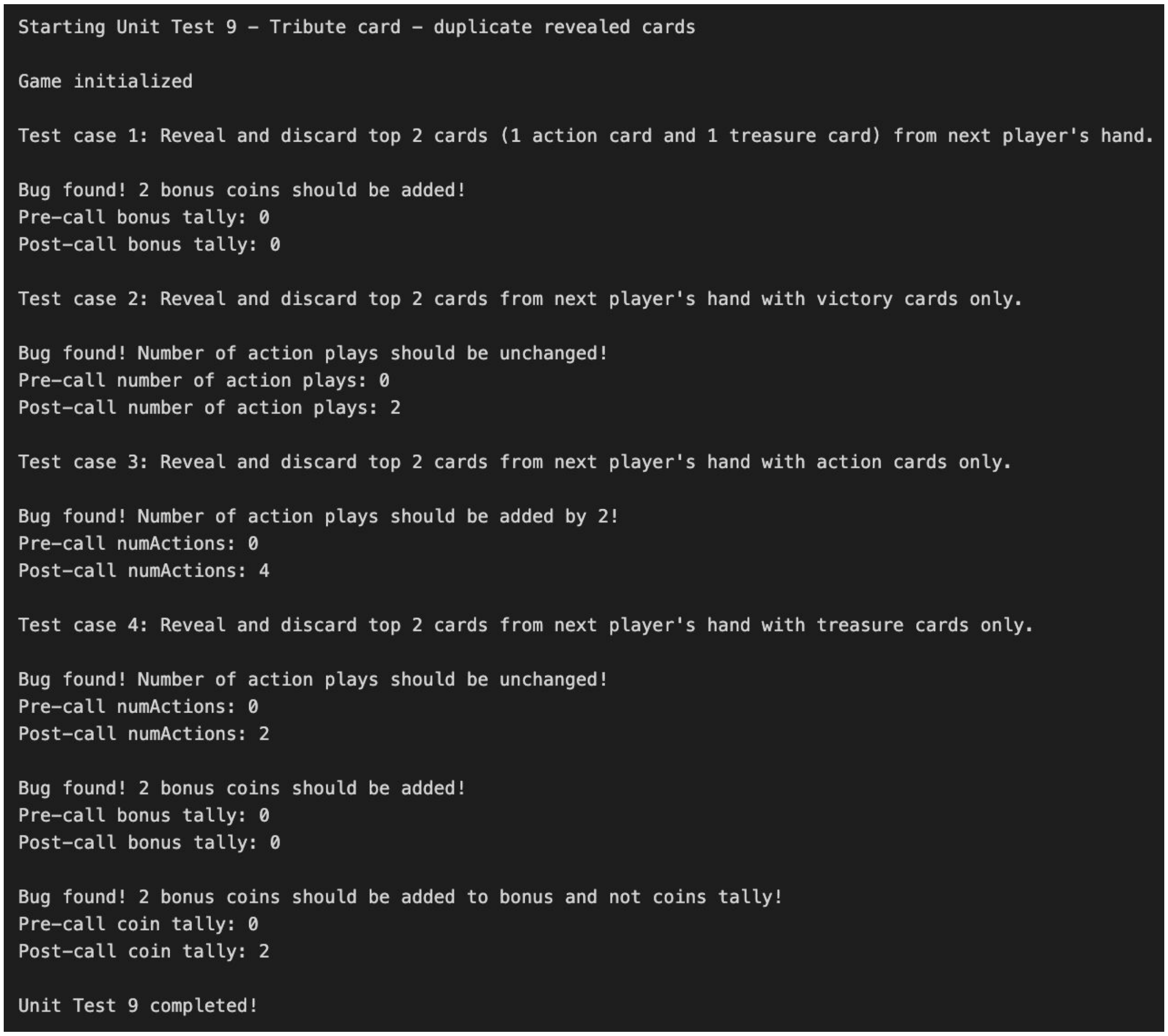
“Possible bug with the way duplicate "revealed cards" are handled for the Tribute card. If tributeRevealedCards[0] is the same as tributeRevealedCards[1], the code sets tributeRevealedCards[1] to -1. The loop that follows expects 2 entries in the array, which is fine - there are still two entries: index 0 contains the card and index 1 contains -1.

What isn't fine is there is no condition to catch the -1. On the first trip through the loop it'll (hopefully correctly) identify the card in index 0 as a Treasure, Victory, or Action card. But on the next iteration, it'll identify that -1 as an Action card, since that falls under the "else" condition.”
bug description by Mandi Grant extracted from Piazza post “Find and Fix Dominion Bugs”

Results
===========

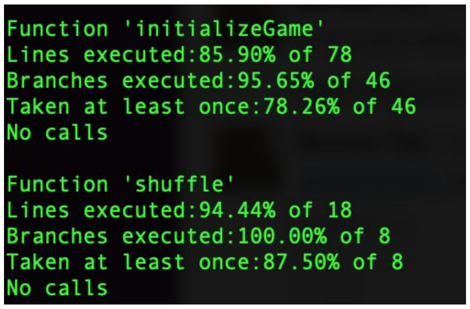
As expected, the bug would only be triggered when the top 2 cards in the deck are duplicates. One of the duplicate cards will be marked “-1” which will trigger the else branch in the test case, resulting in additional numActions provided to player (which is incorrect).

CS362 Software Engineering II Section 400 Fall 2019
Final Project Part B

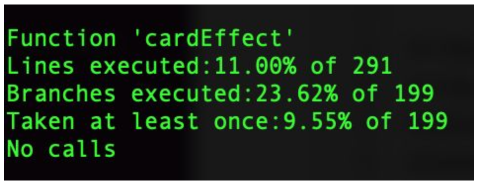


```
Starting Unit Test 9 - Tribute card - duplicate revealed cards

Game initialized

Test case 1: Reveal and discard top 2 cards (1 action card and 1 treasure card) from next player's hand.

Bug found! 2 bonus coins should be added!
Pre-call bonus tally: 0
Post-call bonus tally: 0

Test case 2: Reveal and discard top 2 cards from next player's hand with victory cards only.

Bug found! Number of action plays should be unchanged!
Pre-call number of action plays: 0
Post-call number of action plays: 2

Test case 3: Reveal and discard top 2 cards from next player's hand with action cards only.

Bug found! Number of action plays should be added by 2!
Pre-call numActions: 0
Post-call numActions: 4

Test case 4: Reveal and discard top 2 cards from next player's hand with treasure cards only.

Bug found! Number of action plays should be unchanged!
Pre-call numActions: 0
Post-call numActions: 2

Bug found! 2 bonus coins should be added!
Pre-call bonus tally: 0
Post-call bonus tally: 0

Bug found! 2 bonus coins should be added to bonus and not coins tally!
Pre-call coin tally: 0
Post-call coin tally: 2

Unit Test 9 completed!
```

Coverage
===========



```
Function 'initializeGame'
Lines executed:85.90% of 78
Branches executed:95.65% of 46
Taken at least once:78.26% of 46
No calls

Function 'shuffle'
Lines executed:94.44% of 18
Branches executed:100.00% of 8
Taken at least once:87.50% of 8
No calls
```



```
Function 'cardEffect'
Lines executed:11.00% of 291
Branches executed:23.62% of 199
Taken at least once:9.55% of 199
No calls
```

Dominion Bug 10 Unit Test

==============================

Description

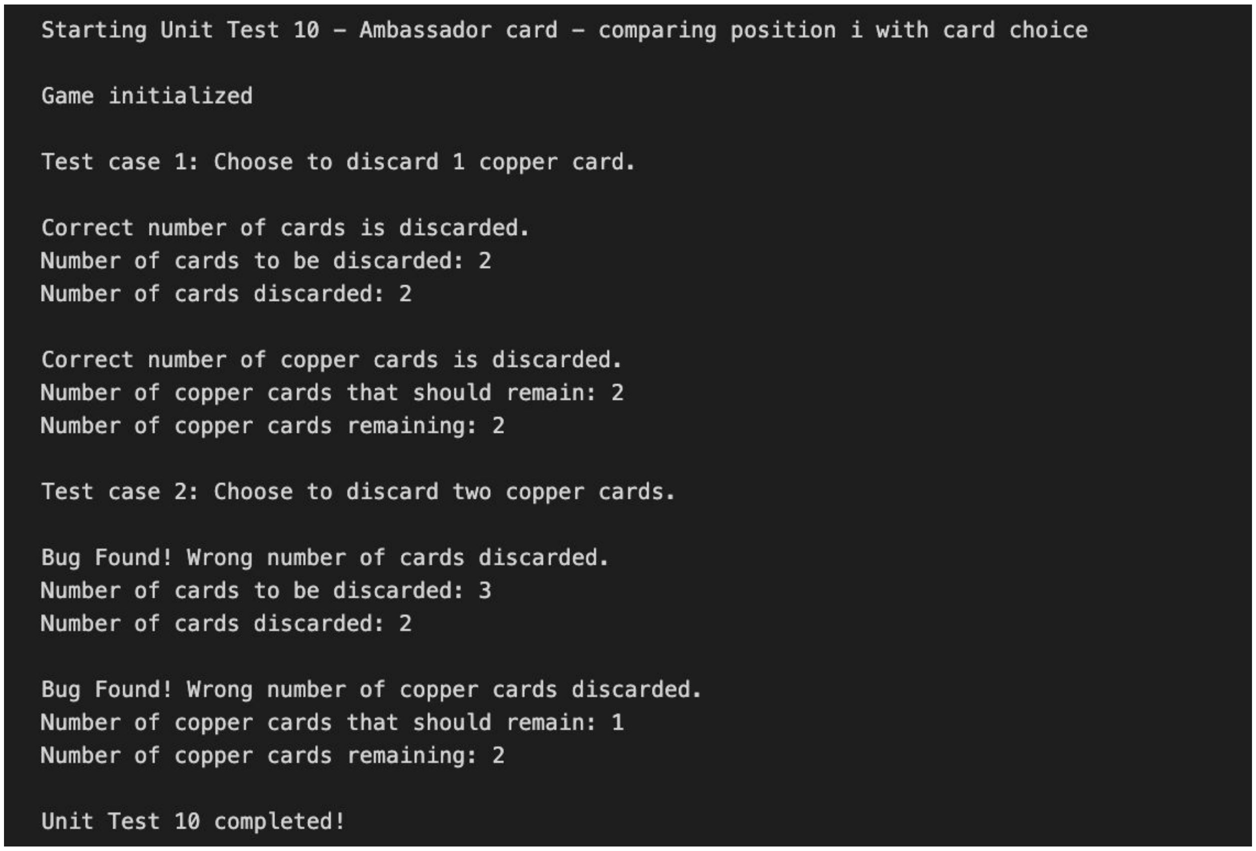
===========

In the dominion.c cardEffect function, ambassador case, line 1100 to 1106, the program goes through the currentPlayer's hand, and tries to find copies of choice1 card. In this loop, if the i-th card is not "ambassador", the same kind of card as the choice1-th card and "i != choice1", count up "j". But this line compares the position "i" with the card in choice1-th position. [This is incorrect] because we want to compare the card, not the position. **bug description by Akifumi Komori extracted from Piazza post "Find and Fix Dominion Bugs"**

Results

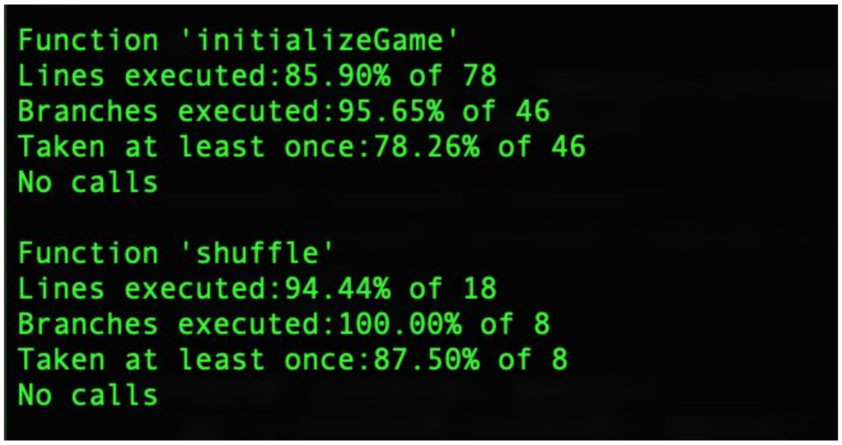
===========

As expected the assert functions shows that if the choice2 is to discard one card, given the iteration will pick up one match, the bug will not be detected nor triggered. On the contrary, if the current player chooses to discard 2 cards, only 1 card will be detected at position i where i equals the enum of the card type. This would result in j being 1 and causes one card to be discarded instead of two.

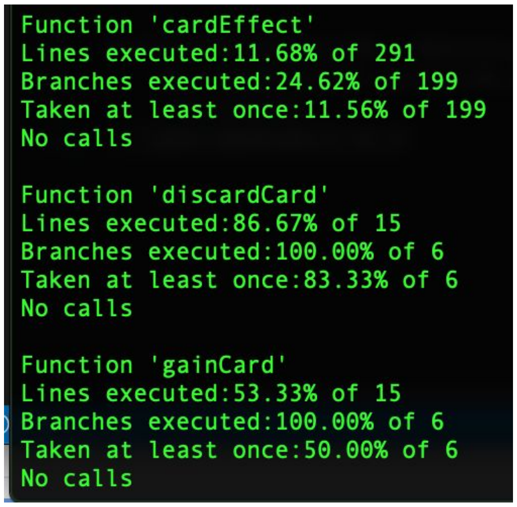


```
Starting Unit Test 10 - Ambassador card - comparing position i with card choice

Game initialized

Test case 1: Choose to discard 1 copper card.

Correct number of cards is discarded.
Number of cards to be discarded: 2
Number of cards discarded: 2

Correct number of copper cards is discarded.
Number of copper cards that should remain: 2
Number of copper cards remaining: 2

Test case 2: Choose to discard two copper cards.

Bug Found! Wrong number of cards discarded.
Number of cards to be discarded: 3
Number of cards discarded: 2

Bug Found! Wrong number of copper cards discarded.
Number of copper cards that should remain: 1
Number of copper cards remaining: 2

Unit Test 10 completed!
```

Coverage
===========



```
Function 'initializeGame'
Lines executed:85.90% of 78
Branches executed:95.65% of 46
Taken at least once:78.26% of 46
No calls

Function 'shuffle'
Lines executed:94.44% of 18
Branches executed:100.00% of 8
Taken at least once:87.50% of 8
No calls
```



```
Function 'cardEffect'
Lines executed:11.68% of 291
Branches executed:24.62% of 199
Taken at least once:11.56% of 199
No calls

Function 'discardCard'
Lines executed:86.67% of 15
Branches executed:100.00% of 6
Taken at least once:83.33% of 6
No calls

Function 'gainCard'
Lines executed:53.33% of 15
Branches executed:100.00% of 6
Taken at least once:50.00% of 6
No calls
```

Dominion Bug 11 Unit Test

==============================

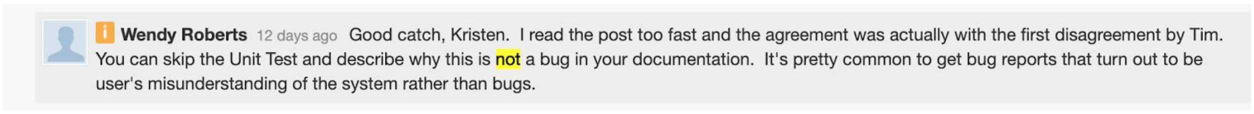
Description

===========

In the case statement for minion starting around line 955, as the rule says, it will increase action by 1, and then player can make choice either to gain 2 money or draw cards. The code around line 960 calls the function discardCard() before the player makes the choice. And the second issue is the 'if, else if' for choice 1 and choice 2 statement is not right, and should be 'if, else'.

As discussed on Piazza, the noted bugs are not valid bugs as the two statements achieve the results the code intends to produce mainly for the following reasons:

- The discardCard call is needed to remove the played card which is a valid move
- Changing the “if else if” to “if, else” does not change the logic flow unless there is intention to create a catch all else block for errors detected (i.e to ensure user chooses choice1 or choice2 as mentioned by Tim Palecek on Piazza)



Wendy Roberts 12 days ago Good catch, Kristen. I read the post too fast and the agreement was actually with the first disagreement by Tim. You can skip the Unit Test and describe why this is not a bug in your documentation. It's pretty common to get bug reports that turn out to be user's misunderstanding of the system rather than bugs.

Results

===========

NA - not a valid bug

Coverage

===========

NA - not a valid bug

CS362 Software Engineering II Section 400 Fall 2019
Final Project Part B

Bug Fixes (45 points)

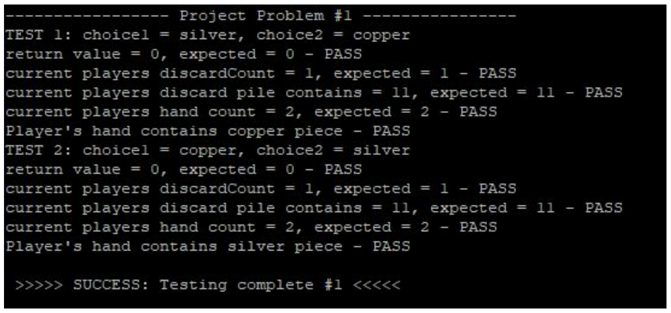
Fix the bugs and run the Unit Tests to verify the bugs are fixed.
1. Create a studentonid-finalproject-bugfree branch from the studentonid-finalproject-bugs branch. Check out the studentonid-finalproject-bugfree branch and create a folder under Projects called FinalProject-BugFree. You should now have two folders under projects called FinalProject-Bugs and FinalProject-BugFree. Copy the dominion Folder from the FinalProject-Bugs folder into the FinalProject-BugFree folder. This dominion folder will have the Unit Tests you created.

2. Fix the bugs identified in the Bug Reports. The fixes will be made in the projects\FinalProject-BugFree\dominion folder.
3. Run the makefile in the projects\FinalProject-BugFree\dominion to execute the Unit Tests and report the results. You should expect your asserts to pass.

4. In the document called ‘Final Project PartB’, document each Bug Fix in a section called ‘Bug Fixes’.

5. Push the studentonid-finalproject-bugfree branch to github.

6. Play the game several times and verify that the game now works according to the game documentation for the bugs identified in the Bug Reports. Report the results in the ‘Bug Fixes’ section in the format below.

	Verified	Notes
Bug 1		Fixes made: Fixed the discardCard() function:

```
---------------- Project Problem #1 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains copper piece - PASS
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains silver piece - PASS

 >>>>> SUCCESS: Testing complete #1 <<<<<
```

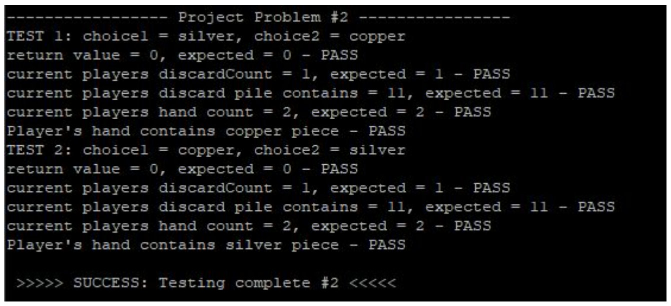
```
int discardCard(int handPos, int currentPlayer,
struct gameState *state, int trashFlag)
{
    //if trash flag is set, add to trash pile
    if (trashFlag != 0)
    {
        //add card to trash pile
        state->trash[state->trashCount] =
state->hand[currentPlayer][handPos];
        state->trashCount++;
    }

    //if trash flag is not set, add to player's
discard pile
    else
    {

state->discard[currentPlayer][state->discardCoun
t[currentPlayer]] =
state->hand[currentPlayer][handPos];
        state->discardCount[currentPlayer]++;
    }

    //remove played card from player's hand
    //set played card to -1
    state->hand[currentPlayer][handPos] = -1;

    //if discarded card is not the last card in
the player's hand or the only card in the
player's hand -> move cards up to fill gap
    if ((handPos !=
(state->handCount[currentPlayer] - 1)) &&
(state->handCount[currentPlayer] != 1))
    {
        //need to maintain hand order -> set
hand[p] = hand[p+1]
        for (int p = handPos; p <
state->handCount[currentPlayer]; p++)
        {
            state->hand[currentPlayer][p] =
state->hand[currentPlayer][p + 1];
        }

state->hand[currentPlayer][state->handCount[curr
entPlayer]] = -1;
    }

    //reduce number of cards in hand
    state->handCount[currentPlayer]--;

    return 0;
}
```

After fixing the discardCard() function I was able to change the mine card function by changing the trashFlag in the discardCard() function from 0 to 1:

```
//discard trashed card
        for (i = 0; i <
state->handCount[currentPlayer]; i++)
        {
            if (state->hand[currentPlayer][i] ==
j)
            {
                discardCard(i, currentPlayer,
state, 1);
                break;
            }
        }
```

CS362 Software Engineering II Section 400 Fall 2019
Final Project Part B

Bug 2

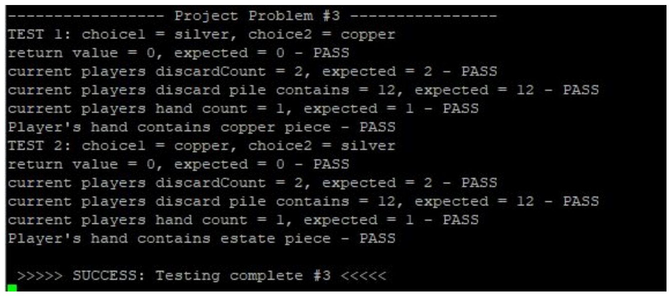


```
----------------- Project Problem #2 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains copper piece - PASS
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 1, expected = 1 - PASS
current players discard pile contains = 11, expected = 11 - PASS
current players hand count = 2, expected = 2 - PASS
Player's hand contains silver piece - PASS

 >>>>> SUCCESS: Testing complete #2 <<<<<
```

Fixed By:
Changing the > to < in this portion of the code of the mine card:

```
if
((getCost(state->hand[currentPlayer][choice1]) +
3) < getCost(choice2))
        {
            return -1;
        }
```

Bug 3

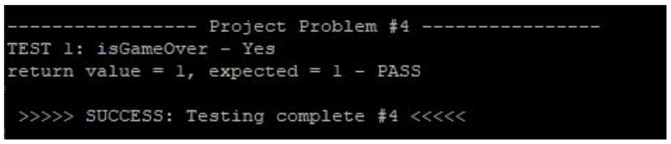


```
----------------- Project Problem #3 ----------------
TEST 1: choice1 = silver, choice2 = copper
return value = 0, expected = 0 - PASS
current players discardCount = 2, expected = 2 - PASS
current players discard pile contains = 12, expected = 12 - PASS
current players hand count = 1, expected = 1 - PASS
Player's hand contains copper piece - PASS
TEST 2: choice1 = copper, choice2 = silver
return value = 0, expected = 0 - PASS
current players discardCount = 2, expected = 2 - PASS
current players discard pile contains = 12, expected = 12 - PASS
current players hand count = 1, expected = 1 - PASS
Player's hand contains estate piece - PASS

 >>>>> SUCCESS: Testing complete #3 <<<<<
```

Fixed By:
Changing the > to < in this portion of the code of the remodel card:

```
if
((getCost(state->hand[currentPlayer][choice1]) +
2) < getCost(choice2))
        {
            return -1;
        }
```

Bug 4

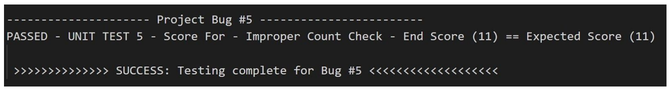


```
----------------- Project Problem #4 ----------------
TEST 1: isGameOver - Yes
return value = 1, expected = 1 - PASS

 >>>>> SUCCESS: Testing complete #4 <<<<<
```

Fixed Bug By:
Changing the i < 25 to i <= treasure_map. This makes sure that it cycles through all of the supply piles:

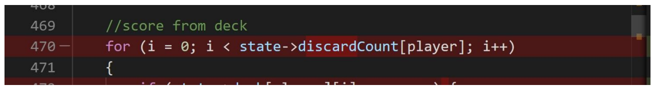
```
//if three supply pile are at 0, the game ends
    j = 0;
    for (i = 0; i <= treasure_map; i++)
    {
        if (state->supplyCount[i] == 0)
        {
            j++;
        }
    }
```

Bug 5



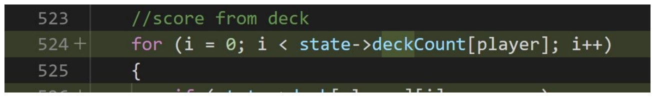
```
-------------------- Project Bug #5 ----------------------
PASSED - UNIT TEST 5 - Score For - Improper Count Check - End Score (11) == Expected Score (11)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #5 <<<<<<<<<<<<<<<<<<
```

Fixed bug by updating the discardCount to deckCount:
Before:



```
    //score from deck
-   for (i = 0; i < state->discardCount[player]; i++)
    {
```

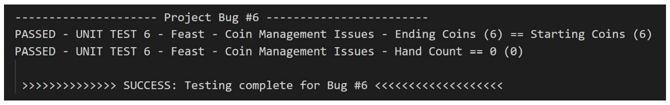
After:



```
    //score from deck
+   for (i = 0; i < state->deckCount[player]; i++)
    {
```

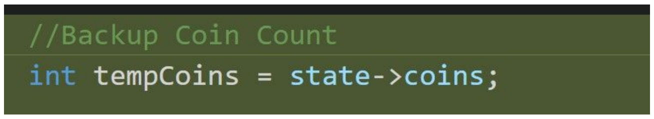
CS362 Software Engineering II Section 400 Fall 2019
Final Project Part B

Bug 6

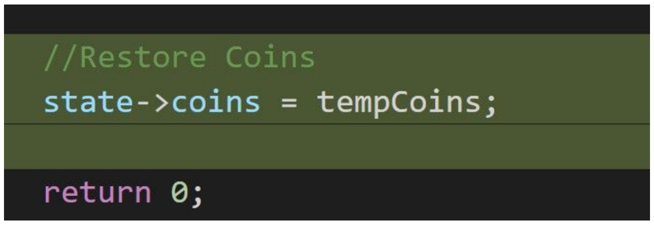


```
--------------------- Project Bug #6 ------------------------
PASSED - UNIT TEST 6 - Feast - Coin Management Issues - Ending Coins (6) == Starting Coins (6)
PASSED - UNIT TEST 6 - Feast - Coin Management Issues - Hand Count == 0 (0)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #6 <<<<<<<<<<<<<<<<<<<
```

This bug was fixed by adding a new variable called tempCoins to the feast card effect logic. This tracks the number of coins the player currently has immediately after the code in which the player's hand is backed up. After the card effect logic has allowed a player to gain the card, the player's coins are reset to the number of coins they had prior to running the effect. The code then returns 0.

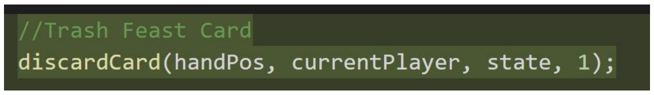


```
//Backup Coin Count
int tempCoins = state->coins;
```



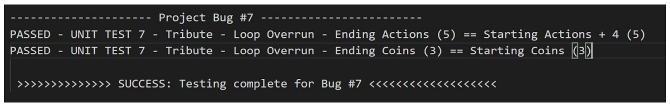
```
 //Restore Coins
 state->coins = tempCoins;

 return 0;
```

In addition to the above update, the card effect also did not trash the feast card. As such, I added logic to properly trash the feast card.



```
//Trash Feast Card
discardCard(handPos, currentPlayer, state, 1);
```

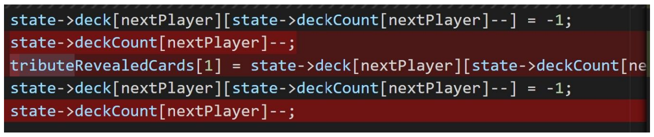
Bug 7



```
--------------------- Project Bug #7 ------------------------
PASSED - UNIT TEST 7 - Tribute - Loop Overrun - Ending Actions (5) == Starting Actions + 4 (5)
PASSED - UNIT TEST 7 - Tribute - Loop Overrun - Ending Coins (3) == Starting Coins (3)

 >>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #7 <<<<<<<<<<<<<<<<<<<
```

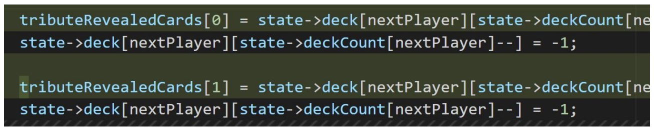
The first issue with this card related to how it navigated through the next player's deck. As shown in the before picture below, the code decrements the state->deckCount twice when it reveals a card. I removed these duplicative decrements to deckCount.

Before:



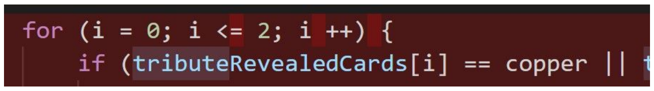
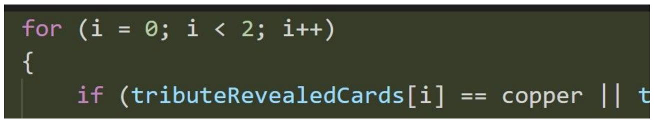
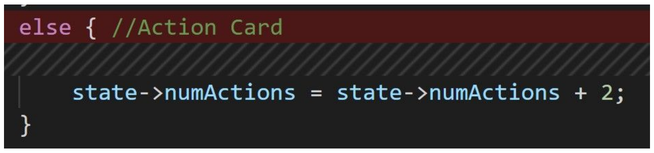
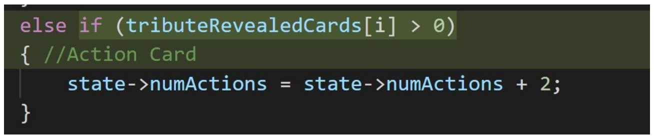
```
state->deck[nextPlayer][state->deckCount[nextPlayer]--] = -1;
state->deckCount[nextPlayer]--;
tributeRevealedCards[1] = state->deck[nextPlayer][state->deckCount[ne
state->deck[nextPlayer][state->deckCount[nextPlayer]--] = -1;
state->deckCount[nextPlayer]--;
```

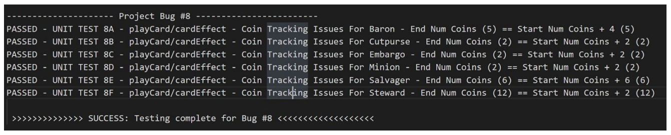
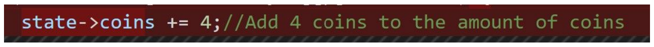
After:

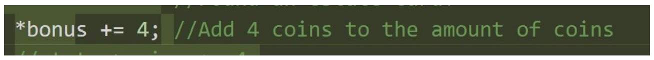


```
 tributeRevealedCards[0] = state->deck[nextPlayer][state->deckCount[ne
 state->deck[nextPlayer][state->deckCount[nextPlayer]--] = -1;

 tributeRevealedCards[1] = state->deck[nextPlayer][state->deckCount[ne
 state->deck[nextPlayer][state->deckCount[nextPlayer]--] = -1;
```

The next issue with this card was that it did not properly iterate through the revealed cards array. This array has a size of 2. The code was

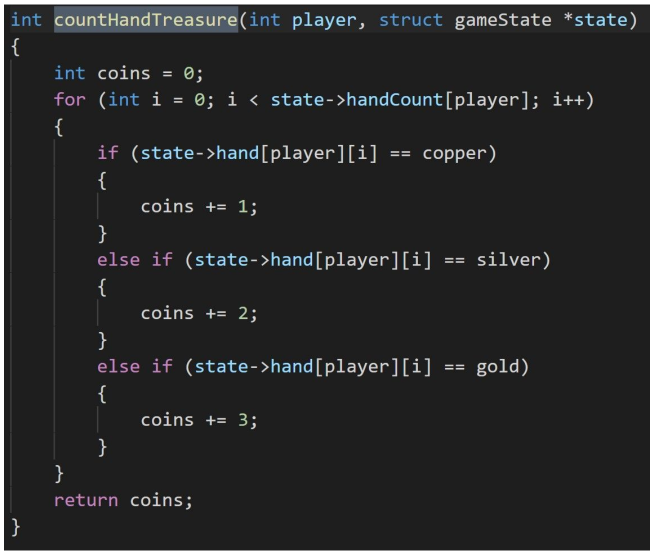
		previously iterating when size index was <= 2 as shown below in the before section. I updated the code to iterate when the index is < 2.
		Before:
		
		`for (i = 0; i <= 2; i ++) {`
		`    if (tributeRevealedCards[i] == copper || t`
		After:
		
		`for (i = 0; i < 2; i++)`
		`{`
		`    if (tributeRevealedCards[i] == copper || t`
		
		Finally, the code was granting an action bonus whenever the current int in the revealed cards list was not indicative of a Treasure or Victory card. This caused the code to grant a bonus when the revealed card was -1 (in cases like a duplicative card, or not additional cards in the deck). I updated the code to ensure the revealed card was > 0.
		
		Before:
		
		`else { //Action Card`
		`    state->numActions = state->numActions + 2;`
		`}`
		
		After:
		
		`else if (tributeRevealedCards[i] > 0)`
		`{ //Action Card`
		`    state->numActions = state->numActions + 2;`
		`}`

Bug 8	
	`--------------------- Project Bug #8 ------------------------`
	`PASSED - UNIT TEST 8A - playCard/cardEffect - Coin Tracking Issues For Baron - End Num Coins (5) == Start Num Coins + 4 (5)`
	`PASSED - UNIT TEST 8B - playCard/cardEffect - Coin Tracking Issues For Cutpurse - End Num Coins (2) == Start Num Coins + 2 (2)`
	`PASSED - UNIT TEST 8C - playCard/cardEffect - Coin Tracking Issues For Embargo - End Num Coins (2) == Start Num Coins + 2 (2)`
	`PASSED - UNIT TEST 8D - playCard/cardEffect - Coin Tracking Issues For Minion - End Num Coins (2) == Start Num Coins + 2 (2)`
	`PASSED - UNIT TEST 8E - playCard/cardEffect - Coin Tracking Issues For Salvager - End Num Coins (6) == Start Num Coins + 6 (6)`
	`PASSED - UNIT TEST 8F - playCard/cardEffect - Coin Tracking Issues For Steward - End Num Coins (12) == Start Num Coins + 2 (12)`
	
	`>>>>>>>>>>>>>> SUCCESS: Testing complete for Bug #8 <<<<<<<<<<<<<<<<<<<`
		First, all of the documented cards were not properly utilizing the bonus argument when applying a card effect bonus. Instead, all card effects mutated the coin state directly. The first fix was to address this issue. The following before/after example below shows how the change was made to the card effect for all cards:
		
		Before:
		
		`state->coins += 4;//Add 4 coins to the amount of coins`
		
		After:



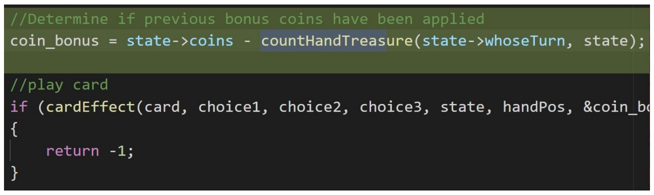
```
*bonus += 4; //Add 4 coins to the amount of coins
```

Next, the game currently runs updateCoins to update the coin state based on the treasure cards in a player’s hand. I refactored the code that counts the treasure cards in a player’s hand into a new function called countHandTreasure



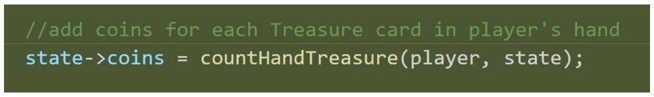
```
int countHandTreasure(int player, struct gameState *state)
{
    int coins = 0;
    for (int i = 0; i < state->handCount[player]; i++)
    {
        if (state->hand[player][i] == copper)
        {
            coins += 1;
        }
        else if (state->hand[player][i] == silver)
        {
            coins += 2;
        }
        else if (state->hand[player][i] == gold)
        {
            coins += 3;
        }
    }
    return coins;
}
```

As players could perform multiple actions in a turn, we need a way to track previous bonuses granted by actions. As such, I further edited the playCard function to determine the current coin_bonus by determining the delta between the current coin state and the current coins granted by treasures in the player’s hand.



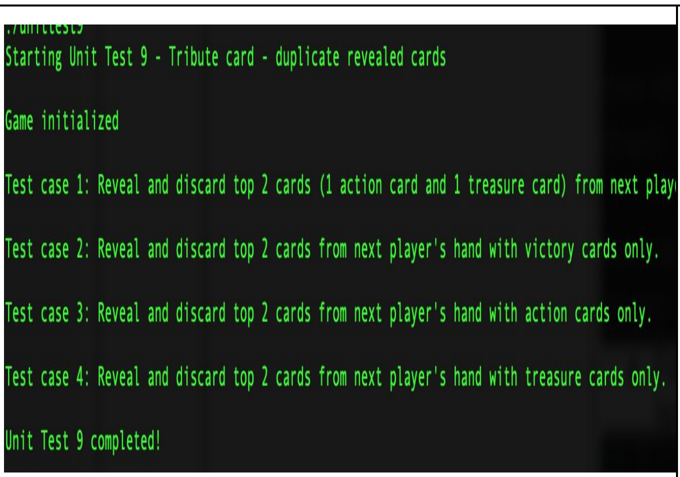
```
//Determine if previous bonus coins have been applied
coin_bonus = state->coins - countHandTreasure(state->whoseTurn, state);

//play card
if (cardEffect(card, choice1, choice2, choice3, state, handPos, &coin_bo
{
    return -1;
}
```

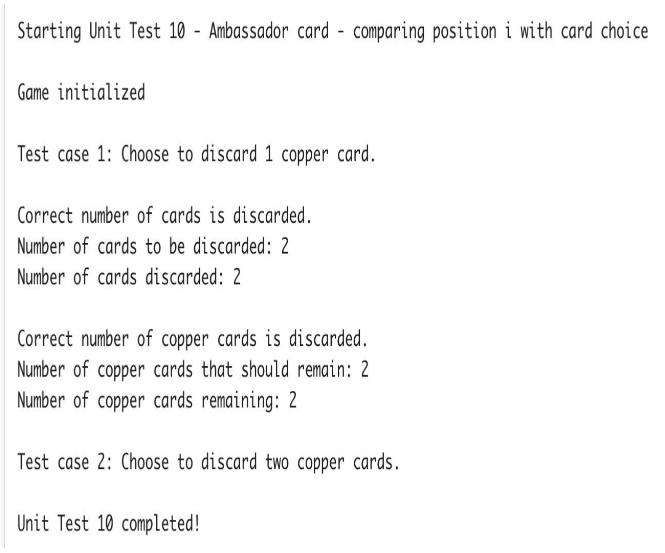
Finally, to ease code maintainability, I updated the updateCoins function to also make use of the countHandTreasure function when updating a player’s coin count.



```
  //add coins for each Treasure card in player's hand
  state->coins = countHandTreasure(player, state);
```

CS362 Software Engineering II Section 400 Fall 2019
Final Project Part B

Bug 9	
	`Starting Unit Test 9 - Tribute card - duplicate revealed cards`
	
	`Game initialized`
	
	`Test case 1: Reveal and discard top 2 cards (1 action card and 1 treasure card) from next play`
	
	`Test case 2: Reveal and discard top 2 cards from next player's hand with victory cards only.`
	
	`Test case 3: Reveal and discard top 2 cards from next player's hand with action cards only.`
	
	`Test case 4: Reveal and discard top 2 cards from next player's hand with treasure cards only.`
	
	`Unit Test 9 completed!`
		Fixes made:
		● How the revealed cards are handled and stored in the array is modified as the initial handling skipped one iteration
		` tributeRevealedCards[0] =`
		`state->deck[nextPlayer][state->deckCount[nextPlayer] - 1];`
		
		`state->deck[nextPlayer][state->deckCount[nextPlayer]--] = -1;`
		
		` tributeRevealedCards[1] =`
		`state->deck[nextPlayer][state->deckCount[nextPlayer] - 1];`
		
		`state->deck[nextPlayer][state->deckCount[nextPlayer]--] = -1;`
		● For loop fixed as initially loop through 3 iterations when there should only be two
		`for (i = 0; i < 2; i++)`
		
		All assertions passed.

Bug 10	
	`Starting Unit Test 10 - Ambassador card - comparing position i with card choice`
	
	`Game initialized`
	
	`Test case 1: Choose to discard 1 copper card.`
	
	`Correct number of cards is discarded.`
	`Number of cards to be discarded: 2`
	`Number of cards discarded: 2`
	
	`Correct number of copper cards is discarded.`
	`Number of copper cards that should remain: 2`
	`Number of copper cards remaining: 2`
	
	`Test case 2: Choose to discard two copper cards.`
	
	`Unit Test 10 completed!`
		Fixes made:
		● Instead of using i equals the card choice made to discard, compare card at the ith position in the hand.
		`if (i != handPos && state->hand[currentPlayer][i] == state->hand[currentPlayer][choice1] && i != choice1)`
		● Discard loop not functioning properly, redesigned discard loop by setting a variable to equal the card type for choice1 (as the card might get deleted once the first is detected)
		`int cardChoice1 =`
		`state->hand[currentPlayer][choice1];`
		
		`//trash copies of cards returned to supply`
		` for (j = 0; j < choice2; j++)`

		` {`
		` for (i = 0; i <`
		`state->handCount[currentPlayer]; i++)`
		`   {`
		`     if (state->hand[currentPlayer][i] ==`
		`cardChoice1)`
		`    {discardCard(i, currentPlayer, state,`
		`1);`
		`     break;}`
		`    }`
		`}`
		
		All assertions passed.

Bug 11	NA - not a valid bug	NA - not a valid bug

Team Work: (10 points)

Document the contribution of each of the team member in a section called ‘Team Work’.

	Unit Test	**Bug Fixes**
Bug 1	Alex created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Alex performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 2	Alex created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Alex performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 3	Alex created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Alex performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 4	Alex created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Alex performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 5	Eric created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Eric performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 6	Eric created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Eric performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 7	Eric created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Eric performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 8	Eric created and performed all the unit testing for this bug. All assertions passed once the bug was fixed.	Eric performed all the testing and fixes. All assertions passed once the bug was fixed.
Bug 9	Serena created and performed all the unit test for this bug. All assertions passed once the bug is fixed.	This bug was fixed by Eric as it relates to Bug 8 so no further fixes was required by Serena.
Bug 10	Serena created and performed all the testing and fixes. All assertions passed once the bug is fixed.	Serena performed all the testing and fixes. All assertions passed once the bug is fixed.

Bug 11	As mentioned in Part A, Serena investigated and confirmed this is not a bug	As mentioned in Part A, Serena investigated and confirmed this is not a bug