Bug Report:

1. Bug in Smithy Card:

I ran my randomtestcard1.c file which runs the game and test the function of **smithy** card. Here are the results for the last few tests:

```
Random_test number 179
Number of hand before: 224
Number of hand after: 227
Test failed
Number of hand (expected): 226
Number of hand: 227

Random_test number 180
Number of hand before: 1
Number of hand after: 4
Test failed
Number of hand (expected): 3
Number of hand: 4

Random_test number 181
Number of hand before: 355
Number of hand after: 358
Test failed
Number of hand (expected): 357
Number of hand: 358

File 'dominion.c'
Lines executed:23.23% of 564
Creating 'dominion.c.gcov'
File 'randomtestcard1.c'
Lines executed:98.21% of 56
Branches executed:100.00% of 18
Taken at least once:94.44% of 18
Calls executed:95.00% of 20
Creating 'randomtestcard1.c.gcov'
```

As we can see from all the tests, the number of hand is always 1 greater than the expected number of hands, as a result the test always failed. This indicates that there is a bug in the part of the code which is responsible for drawing a new card adding it to the hand. After running the test I compared the code of my partner with the original dominion code and I found out that my predictions were correct. There was a bug in the code which makes the Smithy card function add 4 cards instead of 3. On line 733 of the dominion code the *for* loop goes from 0 to 4 instead of going from 0 to 3. This shows that my test code worked and I was successfully able to find the bug in my teammate’s code. The code can be fixed by changing the code on line 733 from “for (i = 0; i < 4; i++)” to “for (i = 0; i < 3; i++)”

2. Bug in Village Code

I ran my randomtestcard1.c file which runs the game and test the function of **village** card. Here are the results for the last few tests:

```
Test Number: 513
Number of hand before: 135
Number of discard card before: 104
Number of card before: 264
Number of coin before: 5
Number of action before: 1
Number of hand after: 136
| Discard Count: 104
Number of discard card after: 264
Number of coin after: 5
Number of action after: 4
Test failed

Test Number: 514
Number of hand before: 153
Number of discard card before: 208
Number of card before: 207
Number of coin before: 10
Number of action before: 1
Number of hand after: 154
| Discard Count: 208
Number of discard card after: 207
Number of coin after: 10
Number of action after: 4
Test failed

Test Number: 515
Number of hand before: 159
Number of discard card before: 72
Number of card before: 437
Number of coin before: 10
Number of action before: 1
Number of hand after: 160
| Discard Count: 72
Number of discard card after: 437
Number of coin after: 10
Number of action after: 4
Test failed
```

```
File 'dominion.c'
Lines executed:28.11% of 555
Creating 'dominion.c.gcov'

File 'randomtestcard2.c'
Lines executed:97.62% of 42
Branches executed:100.00% of 12
Taken at least once:58.33% of 12
Calls executed:95.83% of 24
Creating 'randomtestcard2.c.gcov'
```

As we can see from the test results the number of hand increases by 1 and the number of coins increases by 3 after the card village is called through the *cardeffect* function. The feature of the village card indicates that the number of coins should be increased by 2 rather than 3. This indicates that there is a bug in the village function which increases the number of action by 3 rather than 2. After looking at the code my conclusion was right, in line 823 it says *state->numActions += 3;* whereas it should be *state->numActions += 2;*. This is how I found the second bug in my teammate's code. The code can be fixed by changing line 823 from *state->numActions += 3 to state->numActions += 2;*.

Test Report:

I ran all my unittests, cardtests, and randomtests on my teammate's code. The process was pretty easy since the unittests were performed on functions that were already provided to us so there was nothing to change. All the functions calls and the parameters were the same as before, so the unittests ran on the first try.

The results for the unittests are:

```
Result for running tests:
unittest1.c:
Performing test on gainCard() function
File 'dominion.c'
Lines executed: 10.21% of 564
Removing 'dominion.c.gcov'

unittest2.c:
Performing test on getCost() function
Test on getCost() function passed
File 'dominion.c'
Lines executed: 5.64% of 564
Removing 'dominion.c.gcov'

unittest3.c:
Performing test on buyCard() function
Test on buyCard() function passed!
```

```
File 'dominion.c'
Lines executed: 20.21% of 564
Creating 'dominion.c.gcov'

unittest4.c:
Performing test on isGameOver() function
Test on isGameOver passed
File 'dominion.c'
Lines executed:16.31% of 564
Creating 'dominion.c.gcov'
```

Running the cardtest files:

The cardtest files did not run as easily as the unittests. At first I was getting errors because the function names and the parameters did not match with my team member's function names and parameters. After fixing the function calls to match the function names of my teammate's I had to look at the parameters to make sure I am passing the right number of arguments. After taking care of this the cardtest files also ran without any problem.

Here are the results for the cardtest files:

```
cardtest1.c:
For this test to pass, expect 2 treasures in hand after.
Cards before: 7, -1, -1, -1, -1,
Cards after: 4, 6, -1, -1, -1,
TEST SUCCESSFULLY COMPLETED.
File 'dominion.c'
Lines executed:20.21% of 564
Creating 'dominion.c.gcov'

cardtest2.c:
For this test to pass, expect a 13 in the hand before,
and 3 random cards in the hand after.
Before: 13, -1, -1, -1, -1,
After: 14, 20, 4, -1, -1,
TEST SUCCESSFULLY COMPLETED.
File 'dominion.c'
Lines executed:20.21% of 564
Creating 'dominion.c.gcov'

cardtest3.c:
Before: 17, -1, -1, -1, -1,
Cards after: 17, -1, -1, -1, -1,
File 'dominion.c'
Lines executed:21.63% of 564
Creating 'dominion.c.gcov'

cardtest4.c:
For the test to pass, there should be one more card in deck and 2 more actions should be available.
```

```
Cards before: 14, -1, -1, -1, -1,
Actions before: 1
Cards after: 14, -1, -1, -1, -1,
Actions after: 4
TEST FAILED.
File 'dominion.c'
Lines executed:22.34% of 564
Creating 'dominion.c.gcov'
```

Running the random test files:

The random test files ran without any issue as well since they were designed to run by calling the cardeffect function rather than individual card functions. My teammate followed the exact procedure for assignment 2 so my random tests ran without any issue.

Here are the results for my randomtests:

Randomtestcard1.c

```
File 'dominion.c'
Lines executed:23.23% of 564
Creating 'dominion.c.gcov'
File 'randomtestcard1.c'
Lines executed:98.21% of 56
Branches executed:100.00% of 18
Taken at least once:94.44% of 18
Calls executed:95.00% of 20
Creating 'randomtestcard1.c.gcov'
```

Randomtestcard2.c

```
File 'randomtestcard2.c'
Lines executed:97.62% of 42
Branches executed:100.00% of 12
Taken at least once:58.33% of 12
Calls executed:95.83% of 24
Creating 'randomtestcard2.c.gcov'
```

Randomtestadventurer.c

```
File 'randomtestadventurer.c'
Lines executed:100.00% of 65
Branches executed:100.00% of 48
Taken at least once:93.75% of 48
Calls executed:100.00% of 22
Creating 'randomtestadventurer.c.gcov'
```

I was able to run all the unittests and the randomtests in my team member’s code. I feel like my teammate’s dominion code is reliable, he did an excellent job in the previous assignments so my tests ran after making simple adjustments.

Debugging:

I tried using GDB for fixing the smithy code which was being tested by both randomtestcard1.c and cardtest2.c. GDB helped me to find the bug in the code that was modified by my teammate. I used the following steps to detect the error using GDB:

1. Compile the program using make cardtest2.
2. Running the program in GDB mode by using the command “gdb cardtest2”
3. Add breakpoints in different parts of the code using ‘break’ command.
4. Run the code by using the command ‘run’
5. Breakpoint 1 which was located at smithyEffect was reached.
6.

```
printf("After: ");
for (i=0; i<5; i++) {
        printf("%i, ", G.hand[1][i]);
}
printf("\n");
```

7. After that I typed ‘continue’ and pressed enter.
8. Then breakpoint 2 was reached.
9. ‘continue’ again.
10. gdb exited normally.

Using these steps I could check if the number of cards that were added to the hand before and after the discardCard() function was executed. The card smithy is supposed to add 3 cards to the hand. Step 6 printed out the initial number of cards in the hand and after step 9 it should print the final number of cards in the hand. For step 6 I got the handcount to be 10 and for step 9 the handcount was 14. This indicated that the number of cards in the player’s hand was increased by 4 rather than 3. After analyzing that part of the code and tracing it back to the dominion.c file I was able to find the error. *for (i = 0; i < 4; i++)* which was drawing 4 cards. Changing the code to *for (i = 0; i < 3; i++)* I ran the debugger again and this time I got the correct set of results that I was expecting.

This is the result of running gdb after fixing the code:

```
flip3 ~/dominion_assignment5 252% gdb cardtest2
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-100.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later
<http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show
copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
```

```
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from
/nfs/stak/users/chowdhua/dominion_assignment5/cardtest2...done.
(gdb) run
Starting program:
/nfs/stak/users/chowdhua/dominion_assignment5/cardtest2
For this test to pass, expect a 13 in the hand before,
and 3 random cards in the hand after.
Before: 13, -1, -1, -1, -1,
After: 16, 15, 4, -1, -1,
TEST SUCCESSFULLY COMPLETED.
[Inferior 1 (process 4213) exited normally]
Missing separate debuginfos, use: debuginfo-install glibc-2.17-
196.el7_4.2.x86_64
(gdb)
(gdb) run
Starting program:
/nfs/stak/users/chowdhua/dominion_assignment5/cardtest2
For this test to pass, expect a 13 in the hand before,
and 3 random cards in the hand after.
Before: 13, -1, -1, -1, -1,
After: 16, 20, 5, -1, -1,
TEST SUCCESSFULLY COMPLETED.
[Inferior 1 (process 4282) exited normally]
```

We can see from the test the number of cards before and after the card is drawn and both of the times there are 3 random cards being added to the hand which tells us the program is running correctly.