## Bugs:

Unittest1.c : No bugs were found, test was completed successfully.

Unittest2.c : No bugs were found, test was completed successfully.

Unittest3.c : No bugs were found, test was completed successfully.

Unittest4.c : No bugs were found, test was completed successfully.

Cardtest1.c: this was testing the adventurer card and there was no error found unfortunately but I know there was an error in this card since I was the one who introduced the bug during assignment 2. I tried to make sure that the unit test was able to catch this bug but there was not enough time to make the unit test robust enough to go through all of the bugs.

Cardtest2.c: This was testing the Smithy card and there was error when I tested the function. The player is supposed to draw 3 cards. So my unit test for this card was able to catch this error as there was one less card change due to my bug that was introduced during assignment 2.

Cardtest3.c: This was testing the Minion card and I know there was an error in this card which was caught during the unit test. I changed the value of a if statement from 4 to 2 which means the redraw happens when it is not supposed to happen. Due to the bug the redraw happens when it is not supposed to happen.

Cardtest4.c: This was testing the steward function where there is a bug. I changed the == sign to a = sign which means instead of comparing it is now assigning a value. This bug was not caught by the tester because the tester only tests when a certain function runs correctly or not. In this case the if statement is supposed to execute only when the value of choice1 is 1 but in this case because of the bug the value of choice1 is always 1 so the if statement executes all the time. The tester is not able to test that because it checks when the if statement runs, does it run correctly.

## Unit Testing:

Unittest1.c : Code coverage: 5%

Unittest2.c : Code coverage: 6%

Unittest3.c : Code coverage: 25%

Unittest4.c : Code coverage: 25%

Cardtest1.c: Code coverage: 20%

Cardtest2.c: Code coverage: 30%

Cardtest3.c: Code coverage: 30%

Cardtest4.c: Code coverage: 25%

Values may not be exact as they tend to change everytime I run the code.

**Unit Testing Efforts:**

The unit tests were easy to design initially but making them go through more coverage were hard.