CS362, Software Engineering II

Assignment 3

Assignment 3 - Learn how to create unit tests

1. **Github**
   a. Merge your changes from Assignment 2 into Master**.**
   b. Create a new Assignment-3 branch from Master.
2. **Unit Tests**
   a. Write five (5) Unit Tests, **one** for each of the five functions that you created in Assignment 2. The five functions are the refactored code for 1) baron, 2) minion, 3) ambassador, 4) tribute, and 5) mine. Implement each Unit Test in a separate file and check the five source files in as **unittest1**.**c**, **unittest2**.**c**, **unittest3**.**c**, **unittest4.c**, and **unittest5**.**c**.
      Each Unit Test should initialize variables, call the refactored function, and assert the results like the Sample for testing updateCoins provided in the Week 3 Modules. (**25 points**)
   b. Write a Unit Test for each of the following functions in Dominion.c: 1) initializeGame, 2) shuffle, 3) endTurn, 4) getWinners, 5) drawCard. Implement each Unit Test in a separate file and check the five source files in as **cardtest1**.**c, cardtest2**.**c**, **cardtest3**.**c**, **cardtest4**.**c**, and **cardtest5**.**c**.
      Each Unit Test should initialize variables, call the function, and assert the results like the Sample for testing updateCoins provided in the Week 3 Modules. (**25 points**)
3. **Code Coverage and MakeFile**
   a. Add a rule in Makefile named "**unittestresults"** that will generate and execute all the Unit Tests and append complete test results, including % coverage using the gcov branch option, into a file called **unittestresults.out**. The rule should be named **unittestresults** and should depend on all your Unit Test source files as well as the dominion code. The .out file will contain the output of your running tests along with coverage information. (**10 points**)

4. **Documentation**
    a. Describe each Unit Tests **in detail** in a section called “**Unit Testing**“. Document your strategy for testing the function thoroughly and your expected results. Provide code snippets that show your test code and assertions. (**10 points**)
    b. Describe **in detail** the bugs you found in your Unit Testing in a section named “**Bugs**“**.** Provide a description of the effect of the bug on processing and the root cause of the bug. Provide code snippets of the code where the bug was found. (**10 points**)
    c. Describe each Unit Test coverage (statement, branch, boundary, etc.), and describe their implications for the tests in a section called “**Code Coverage**“. Discuss what parts of your code are not covered and how you would improve your unit tests to increase coverage. (**20 points**).
5. **Notes**: To avoid making it hard to collect coverage when a test fails, use your own asserttrue function instead of the standard C assert (which basically terminates the process and fails to collect coverage).

Deliverables:

1. Assignment-3 branch with your Makefile and Unit Tests pushed to your github repository. This branch must be created before the due date to receive credit.
2. Assignment-3.pdf document uploaded to Canvas.
   Note: When you submit your PDF to Canvas, add a Comment in the Comments Box and provide the URL for the Assignment-3 branch in your GitHub repository. (-10 points for missing it).