CS362 W17

Assignment-1: Due date is Feb. 5th (23:59 pm deadline) (4th week)

The goals of this assignment are: (1) to help you learn or review the Java programming language; (b) use junit to automate the unit tests; and (c) to learn how to use a code coverage tool to assess the code coverage of the unit tests.

Rules

Make sure you read the CS362 and the university dishonesty policies (here is the link) before doing this assignment or any of the course's assignments.

This is NOT part of the team project. Do it on your own!

Your Tasks

1. Download the base C dominion code from Canvas.
2. You should have access to the class github repository.
3. Implement dominion in Java, with an API that matches that of the C program (except where the types in the languages differ, obviously); the implementation of game state does not need to be similar: **initializeGame** should no longer take as input a game state, but return a fresh game used in other calls. You do not have to implement all of the kingdom cards in the C implementation: instead, implement the first 10 in alphabetical order, plus three cards chosen from the remainder.
4. **Ensure that there are at least five bugs (known to you) in this implementation, but not bugs that stop testing (e.g., not such that all runs crash, etc.)**
5. Implement a set of unit tests.
    a. For the unit tests, you can pick the size but I suggest trying to produce **enough unit tests** that it is plausible some developer would view this as an "ok" suite for making sure dominion isn't broken. Calling each top level API at least once sounds like a good minimum!. Execute your tests cases. Use the junit tool to automate the execution of your test suite. You can download junit and learn more about how to install and use the tool by accessing the www.junit.org website.
    b. In order to assess the adequacy of the unit tests you must determine what percentage of the source code statements where executed during the testing process. It is possible that the unit tests provided do not execute every statement of code in the Dominion code. You are asked to submit the html output of any code coverage tool (e.g., http://codecover.org/,

http://www.eclemma.org/jacoco/, or http://cobertura.github.io/cobertura/) along with your testing document. **If your unit tests statement/branch coverage is low you might want to improve it by adding more test cases.**

6. **Submitting your solution**
    a. **Canvas**:
        i. A document "Assignment-1.pdf" in Canvas. The document contains (a) the unit tests; (b) the actual outcomes of the executed tests; (c) explain your choice of the input; (d) describe any bugs you find; and (e) commentary on how well the unit tests cover the source code of the Dominion code and include any graph or data generated by the code coverage tool indicating the code coverage when executing the tests; (f) write pseudo-code for an automated testing system that somehow _generates_ tests without you picking each value/operation.

    b. **The class github repository**
    https://github.com/aburasali/CS362W17Section-001.git
        i. Check your own copy of the Java dominion code, including everything required to compile the code, in the repository in a directory: projects/<onid-id>/dominion.
        ii. a README file with directions describing exactly how to compile the program under test and run the unit tests (with and without code coverage tool).
        iii. An html report generated by the code coverage tool indicating the code coverage when executing the entire test suite.

Helpful hint:

- Make a Maven development project for this assignment. The Apache Maven tool is available for download from Apache Maven Project
- Given some lead time, I may be able to help some with Maven problems that you encounter.
- Please try to avoid adding the target folder to the class github repository.