CS362

Assignment-2: **Due date is Monday Feb. 27th (23:59 pm deadline) (week 7)**
Late submission: **The assignment will be accepted up to 2 days late for a 10% penalty for each day.**

The goals of this assignment are: (a) to learn how to use a random test generation tool; or (b) to learn how to use a search-based test generation tool.

1. You are asked to select, download and use any random test case generation (RT) tool **OR** search-based test case generation (SBST) tool usable for Java language.
   a) Use the **SELECTED** tool to test your dominion code.
   b) After the tool generates a test suite(s), you should execute it using a code coverage tool and create a code coverage report (html), as you did in the first assignment.
2. Write a random tester that plays complete games of dominion, with a random number of players (from 2-4) and a random set of kingdom cards with a seed of 10. Check it in as RandomTestDominion.Java. The goal of this exercise is not to write a "good" AI dominion player, but to write a good test generator that plays valid (or mostly valid) games of dominion, but may play in ways no human or sensible robot would ever dream of playing.

   Helpful hint:
   a) to avoid making it hard to collect coverage when a test fails, use your own asserttrue function instead of the Junit assert (which basically crashes the code and fails to collect coverage). Your assert can also print out more information, and maybe have an option that controls whether it exits the program or not. I find it helpful to make all tests print "TEST SUCCESSFULLY COMPLETED" or some other message if and only if the entire test passes, and usually (this isn't always possible for crashing bugs) print "TEST FAILED" for a failure. This makes it easy to process failing and passing tests.
   b) Instead of testing a complete specification (though you can test as much as your wish in the tester), write your tester that it tests and prints key information about the game state. Just document and justify your choices in your report (pdf file)

3. Submitting your solution
   a) **Canvas**: A document "**Assignment-2.pdf**" in Canvas. The document contains:
      i. **Section Tool**:
         1. Comment on how well the tool cover the source code of the dominion, and contrast these results to those in Assignment-1. Include any graph or

data from the code coverage tool.

2. Did the tool miss some bugs which you already knew were actually bugs, how many?
3. What problems did you face while using this tool (If any)?
4. Discuss your overall experience and findings with using this tool, includes the time budget for testing, the length of a test case and any other parameters that can affect your findings/results.
5. Cite sources and list the tools you looked at before selecting this tool.
6. In a few paragraphs, comment on the utility of the selected tool. Why this tool? What worked well? What worked poorly? When would you want to use it (if ever!)? How do you think it should be improved?

ii. **Section Random Tester**: Write up the development of your random tester (RandomTestDominion .Java), including improvements in coverage and efforts to check the correctness of your specification by breaking/understanding the code. Include some pictures/graphs from the html report generated by code coverage tool.

iii. **Section Bugs**: Document the process of identifying and fixing a bug in your own code. Show how you used a debugger to understand the problem and fix the bug.

b) **The class github repository** https://github.com/aburasali/CS362W17Section-001.git. Under projects/your-onid/

i. A README file that explains the directory structure of your assignment documents how to compile the code, run the test suite, generate the html report, and so on., in the repository in a directory: projects/<onid-id>/dominion.

ii. The unit tests that were generated by the selected tool.

iii. An html report generated by code coverage tool indicating the code coverage when executing the generated test suites for each test tool.

c) Submit your complete maven project of assignment-2 to your onid-folder on our class github. The common maven project should contain:

i. src/main/java package

ii. src/test/java package

iii. pom.xml