Individual Project: Replace Utility -- Deliverable 3

For the last deliverable of this project, your goals are to

- evaluate your current set of test cases on an alternative implementation of `replace` and
- possibly extend the test set in a black-box fashion to improve the coverage they achieve.

Details

In this deliverable, you will be provided with a precompiled, alternative version of the `replace` utility that was developed by one of your colleagues in parallel to yours. The goal of this task is to use this version of `replace` to evaluate the tests that you developed for the previous deliverables. To complete this task you must (1) run the tests you defined (class `MyMainTest`) against the provided version of `replace` and measure the coverage they achieve and (2) extend your set of tests to try to achieve 100% line (i.e., statement) coverage. In the process, you will also keep track of how many bugs your tests can find in this version of `replace` (there are several, none of which are revealed by the test cases **we** provided). You must find at least two bugs to get full points, and finding more than three bugs will qualify you for up to 10 extra points for this deliverable.

Detailed instructions

Setup

1. Download archive individualProject-d3.tar.gz
2. Unpack the archive in the root directory of the **personal GitHub repo that we assigned to you**. After unpacking, you should see the following files:

   Under `<root>/IndividualProject/replace/D3`:

   - `compileAndRunTests.bat` and `compileAndRunTests.sh`
     These scripts for Windows and Unix/Mac, respectively, will compile and run against the provided version of `replace` the set of tests in `.../D3/testsrc/edu/.../MyMainTest.java` and save the corresponding coverage information.
   - `generateReport.bat` and `generateReport.sh`
     These scripts for Windows and Unix/Mac, respectively, will take the coverage information generated using the previous scripts and generate a coverage report in file `report.txt`.

- `lib/*`
  Various libraries used to run tests, compute coverage, generate reports, and so on. You can safely ignore these files.
- `template.xml`
  This file contains metadata used when computing coverage. You can safely ignore this file.
- `.../D3/testsrc/edu/.../MyMainTest.java`
  This is a placeholder that you will have to replace with your own version of this test class.
- `testclasses/*`
  This is the directory where the compiled version of your tests (i.e., class `MyMainTest`) will be saved. You can safely ignore this directory.

Testing

3. Before beginning this task you should make sure that the provided files work as expected, by doing the following:
    - Open a command shell
    - Go to directory `.../D3`
    - Run `./compileAndRunTests.sh`
      (if on Windows, run the corresponding bat file)
      You should see the following output (time may vary):

      ```
      JUnit version 4.12
      .......
      Time: 0.122

      OK (7 tests)
      ```

    - Run `./generateReport.sh`
      (if on Windows, run the corresponding bat file)
      This command should produce no output.
    - Check the content of file `report.txt`, which should be as follows:

      ```
      Coverage Report:
      ALL: classes: 100% (2/2); branches:  49% (36/73); lines:  70%
      (71/102);

      PKG+: edu.gatech.seclass.replace classes: 100% (2/2); branches:
      49% (36/73); lines:  70% (71/102);

      CLS+: Main  branches:  74% (17/23); lines:  81% (38/47);


      CLS+: Replace  branches:  38% (19/50); lines:  60% (33/55);
      ```

- If some of these steps do not work as expected, please post a public question on Piazza, as other student may have solved similar issues and may be able to help.

4. Copy your latest version of `MyMainTest.java` to `.../D3/testsrc/edu/.../replace/MyMainTest.java`, thus replacing the placeholder file currently there.
5. Run your test suite by executing `compileAndRunTests` (`.sh` or `.bat`, depending on your platform). If one of your test cases fails, it could be for one of several reasons:
   - Your test case hits a corner case that is not defined in the requirements and for which your version of `replace` makes different assumptions than the version we provided. You should (1) modify the test case (**in this copy of `MyMainTest.java` only**) so that it passes and (2) add the following comment right before the `@test` annotation for that test: “// Type a”.
   - Your test behaves incorrectly, which means that there is a bug in your version of `replace` that was not caught by our test suite (class `MainTest`). You should (1) modify the test case (**in this copy of `MyMainTest.java` only**) so that it passes and (2) add the following comment right before the `@test` annotation for that test: “// Type b”. **Note**: you will not be penalized for this, so there is no need to fix the bug; we are actually interested in seeing what kind of bugs our test suite missed in your code.
   - Your test triggered a bug in our version of `replace` and caused a failure: good job! In this case, you should leave the test case as is but add the following comment right before the `@test` annotation for that test: “// Type c: <short explanation of the failure and what you think it’s the corresponding bug>”. Do not worry too much about the explanation and just make your best guess (e.g., “// Type c: replace fails when passed more than 3 options, probably due to the storing of the options in a fixed-size array), as you will not be penalized for getting the explanation wrong. **Note**: the fixed-size array example is not a hint...
6. Generate the coverage report for your test suite by running `generateReport` (`.sh` or `.bat`, depending on your platform).
7. Save this report as `report-initial.txt` under directory `.../D3`.

8. If your **line coverage** of both class `Main` and class `Replace` is already 100%, and you found at least two bugs in the version of `replace` we provided, you are done.
9. Otherwise, you should add test cases to your test suite (`.../D3/testsrc/edu/.../replace/MyMainTest.java`) to try to (1) get to 100% **line coverage** for both class `Main` and class `Replace` and (2) make our version of `replace` fail at least twice. If one of the new tests you add does cause a failure in our version of `replace`, you should add the following comment right before the `@test` annotation for that test: "// Type d: <short explanation of the failure and what you think it's the corresponding bug>".

   Note: To extend your test suite, you can either use test cases from the test suite we provided (`MainTest`) or add new tests. The advantage of adding new tests is that our tests will not reveal any bug in our code.
10. This task will be completed when either your tests will achieve 100% line coverage and cause at least two failures (both goals are possible) or you are stuck and cannot achieve one or both of these goals. (In this latter case, you will still get partial credit depending on how much you accomplished.)
11. Generate the coverage report for the final version of your test suite and save it as `report-final.txt` under directory `.../D3`.
12. Commit and push the following files:
    - `.../D3/testsrc/edu/gatech/seclass/replace/MyMainTest.java`
    - `.../D3/report-initial.txt`
    - `.../D3/report-final.txt` (unless you didn't have to modify your test suite, so you only have the initial report)
    - There is no need to push any other file, as none of the other existing files under `D3` is supposed to be modified.
13. Paste this last commit ID on T-Square, as usual.