Individual Project:
Replace Utility -- Deliverable 2

Goals

- Develop a Java application for replacing strings within a file.
- Get experience with an agile, test-driven process.

Details

For this project, you must develop, using the Java language, a simple **command-line** utility called *replace*, which is the same utility for which you developed test frames in the category-partition assignment. A concise specification for the `replace` utility was provided with that assignment and is repeated here for your convenience:

Concise Specification of the `replace` Utility

- NAME:
  `replace` - looks for all occurrences of string *from* and replaces it with string *to*. It is possible to specify one or more files on which to perform the replace operation(s) in a single replace command.
- SYNOPSIS
  `replace OPT <from> <to> -- <filename> [<filename>]*`

  where `OPT` can be zero or more of
  - `-b`
  - `-f`
  - `-l`
  - `-i`
- COMMAND-LINE ARGUMENTS AND OPTIONS

  `from`: string to be replaced with string `to`.

  `to`: string that will replace string `from`.

  `filename`: the file(s) on which the replace operation has to be performed.

  `-b`: if specified, the `replace` utility creates a backup copy of each file on which a replace operation is performed before modifying it.

  `-f`: if specified, the `replace` utility only replaces the first occurrence of string

`from` in each file.

`-l`: if specified, the `replace` utility only replaces the last occurrence of string `from` in each file.

`-i`: if specified, the `replace` utility performs the search for string `from` in a case insensitive way.

- EXAMPLES OF USAGE

  Example 1:
  ```
  replace -i Howdy Hello -- file1.txt file2.txt file3.txt
  ```
  would replace all occurrences of "Howdy" with "Hello" in the files specified. Because the "`-i`" option is specified, occurrences of "howdy", "HOwdy", "HOWDY", and so on would also be replaced.

  Example 2:
  ```
  replace -b -f Bill William -- file1.txt file2.txt
  ```
  would replace the first occurrence of "Bill" with "William" in the two files specified. It would also create a backup copy of the two files before performing the replace.

  Example 3:
  ```
  replace -f -l abc ABC -- file1.txt
  ```
  would replace both the first and the last occurrences of "abc" with "ABC" in the file specified.

You must develop the `replace` utility using a test-driven development approach such as the one we saw in lesson. Specifically, you will perform three activities within this project:

- For Deliverable 1, you will generate 40 JUnit test cases for `replace`. At least 10 of the test cases that you generate should be instantiations of the test frames that you created for the category-partition assignment. (Each test frame is a test spec that can be instantiated as a concrete test case.) To generate the remaining 30 tests, you can (1) leverage additional test frames, (2) generate the test cases from scratch, or (3) do a combination of these two approaches (i.e., implement some of the test frames and create some new test cases).
- For Deliverable 2, you will develop the actual `replace` utility and make sure that all of the test cases that you developed for Deliverable 1, plus some test cases added by us, pass.
- The details of Deliverable 3 will be revealed in due time.

Deliverables:

~~DELIVERABLE 1~~

- ~~**provided:**~~
  - ~~Skeleton of main class~~
  - ~~Initial set of JUnit test cases~~
- ~~**expected:**~~
  - ~~40 or more **additional** JUnit test cases~~

DELIVERABLE 2 (this deliverable)

- **provided:**
  - Additional set of JUnit test cases (to be run in addition to yours)
- **expected:**
  - Implementation of replace that makes **all** test cases pass

DELIVERABLE 3

- **provided:** TBD
- **expected:** TBD

Deliverable 2: Instructions

1. Download archive individualProject-d2.zip
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:
   - `<root>/IndividualProject/.../replace/MainTestAddOn.java`
     This is a test class with an additional set of test cases for the `replace` utility. Imagine that these are test cases developed by coworkers who were also working on the project and developed tests for the `replace` utility based on (1) their understanding of the requirements and (2) some of the discussion about the semantics of `replace` that took place on Piazza during the week. As it was the case for the test cases we provided for Deliverable 1, all these tests must also pass, **unmodified**, on your implementation of `replace`.
     - Please note that these tests, as stated above, make some assumptions on the way `replace` works. Please ask privately on Piazza if you have doubts about specific tests. We will make your question public if it is OK to do so.
     - In most cases, we expect that these assumptions will not violate the assumptions you made when developing your test cases for Deliverable 1, but they might in some cases.
     - If they do, please adapt your test cases (and possibly your code) accordingly, which may also give you an opportunity to get some experience with refactoring.
   - `<root>/IndividualProject/.../replace/MainTestSuite.java`
     This is a simple test suite that invokes all the test cases in test classes `MainTest`, `MyMainTest`, and `MainTestAddOn`. You can use this class to run all the test cases for `replace` at once and check that they all pass. This is how we will run all the test cases on your submission.
3. Implement the `replace` utility by completing the `Main` class, whose skeleton we provided last week, and adding any other class you may need. The only requirement for your code is that it passes all the test cases you created and we provided and that it can be executed as follows:
   `java -cp <classpath> edu.qc.seclass.replace.Main <arguments>`
4. Commit and push your code. You should commit, at a minimum, the content of directories `IndividualProject/replace/test` and `IndividualProject/replace/src`. As for previous assignments and projects, committing the whole IntelliJ IDEA or Eclipse project, in case you used one of those IDEs, is fine.
5. Paste the commit ID for your submission on Blackboard, as usual.