SE 333 Software Testing

Assignment 2: Maven and Unit test basics

Due date: April 13, 2020, 11:59pm

Overview
This assignment has 3 goals:

1. **Introduce Maven**
   By completing this assignment correctly you will demonstrate that you have Maven and your IDE setup correctly and that you know the basics of using Maven as needed by this class.

2. **Introduce unit testing with Junit**
   This is a very basic first step in learning test automation with Junit. We will dig deeper into Junit in future assignments.
3. **Make a proposal for your testing project**
   If you already have source code, please make it available to me in some way. You could send it to me of give me access to the repository it's stored in, etc.

   I also need a plain English description of your project. If you are starting from scratch, the description might be all you have at this point, which is fine but then I need a fairly detailed description so I can assess whether the project is suitable.

   Project requirements:
   a. It is written in Java
   b. It is composed of several (minimum 4?) classes that depend on each other in some way.
   c. You are not breaking any laws by giving me access to this software

   We can discuss you project ideas in any form you want (Slack, a voice connection, etc.) but your final proposal needs to be uploaded to the D2L submission folder I will create for these proposals.

Instructions
1. Download and import SE333-week2-triangle.zip. See the maven video and/or the PowerPoint presentation for details on how to do that.

2. Implement at least 1 test representing each of the following results:
   a. A test that uses assertEquals(), that passes
   b. A test that uses assertEquals(), that should pass but doesn't
   c. A test that uses assertThrows() that passes
   d. A test that uses assertThrows() , that should pass but doesn't

3. Tips on tests
    a. There is no need to test the main() function for this particular assignment. It is there to demonstrate the Triangle API
    b. The API is a bit different from the web site. In this case, you first create an instance of Triangle(args) and then call classify() on it. Args is a String array containing side values. The constructor throws an exception if it cannot make a valid triangle…but does it do this correctly?

4. Produce a zip file for submission using Maven. It is critically important that you create the zip file by using maven. Resist the urge to hand craft a zip file if Maven isn't working for you. The goal isn't to create a zip file. The goals is to
    a. Verify maven is configured correctly
    b. Verify you know the basics of how to use it.

    If accomplishing these 2 things takes a little time and effort on your part and on my part, that is ok.

    Keep in mind that normally Maven's lifecycle management stops you from packaging when all your tests aren't passing. In this assignment, failing tests are expected so you need use one of the following techniques to override the lifecycle convention:

    c. Add "-Dmaven.test.failure.ignore=true" to the maven command line
    d. Enable the "skip tests" mode in IDEA

5. Upload your zip file the submission folder.
    If you have made the correct updates to pom.xml and executed the Maven command correctly, there will be a zip file in your project's target folder named:

    SE333-<your-user-id>-triangle.zip

    Upload this file to the submission folder. If the zip file in the submission folder is named SE333-REPLACE-ME-triangle.zip, you have missed a step and need to review the import steps for project. Producing a zip file correctly, including having the correct name is part of the assignment and will be reflected in your grade.