CSC 223 Project 1: JUnit and Unit Testing

The objective of this project is to use JUnit framework and write test cases to test a Java program.

# JUnit Basics

JUnit is a simple framework to test Java programs with a set of test cases to see if the programs are implemented correctly under the testing case scenarios. It is a common approach test-driven development process. JUnit is an open source concluded in a jar file. The jar file contains test runner, which is used to run JUnit tests.

You can find some resources about JUnit Testing and all of the data files you need in your Canvas course site.

# JUnit Setup

To run JUnit testing, you need to download jar files that contains all dependent libraries. Please follow the steps to setup JUnit Testing environment:

1. Download two jar files (JUnit and hamcrest-core) and save to local computer, ex. c:\csc202\project01. Save/copy all classes (ex. Triangle.java) and JUnit test (ex. TriangleTest.java) files into this folder. You can write a main method and run it with the class file without test runner.

If you have never used the Windows command line from Command Prompt, check these two tutorials:

* [Windows Command Line Tutorial - 1 - Introduction to the Command Prompt [Length 7:30]](https://youtu.be/MBBWVgE0ewk)
* [Windows Command Line Tutorial - 2 - Listing Files and Directories [Length 3:46]](https://youtu.be/7ABkcHLdG_A)

Practice and get familiar with major commands from the Command Line in Windows (or Terminal in Mac), such as dir (ls in Mac), cd, cd...

1. On PC/Windows, do the following from the command line when you are in the project directory (ex. c:\csc202\project01)
2. To compile Java class: *javac Triangle.java*
3. To compile JUnit test: *javac -cp junit.jar;. TriangleTest.java*
4. To run JUnit test: *java -cp junit.jar;hamcrest-core.jar;. org.junit.runner.JUnitCore TriangleTest*

On Mac, do the following from the Terminal

1. To compile Java class: *javac Triangle.java*
2. To compile JUnit test: *javac -cp junit.jar:. TriangleTest.java*
3. To run JUnit test:

*java -cp junit.jar:hamcrest-core.jar:. org.junit.runner.JUnitCore TriangleTest*

Though you may run a JUnit from some Java IDEs, it is required to run a JUnit from the command line in this project.

# Triangle Class

The following example illustrates how to use testing cases to examine the implementations of Triangle class.

Assume in the Triangle class, the method triangleType takes three string parameters representing three sides of a triangle. The three sides are considered as integers. This method produces different outputs under different conditions.

* If the three sides are the same, it returns “This is an equilateral triangle”.
* If two sides are the same, it returns “This is an isosceles triangle”.
* For other cases with valid input, the method returns the word “This is a scalene triangle”.
* If three sides do not form a valid triangle, such as 10, 10, 4, the method returns “Not a valid triangle”.
* If one or more sides are not a valid integer, the method returns an error message.

For example:

|  |  |  |
| --- | --- | --- |
|  | arguments | triangleType( ) |
|  | 5 5 5 | This is an equilateral triangle. |
|  | 5 5 6 | This is an isosceles triangle. |
|  | 4 5 6 | This is a scalene triangle. |
|  | 8 8 1 | Not a valid triangle. |
|  |  |  |

# Triangle Testing

You are about to test the implementations of triangleType method with JUnit Testing. To test this Triangle class, you write a JUnit test class named TriangleTest.java. This test class contains multiple test cases in the format of methods. The test runner executes TriangleTest to generate a JUnit test report.

In this project, you do not need to write the code for this Triangle class. Instead, you will write JUnit test cases to make sure that the implementation of triangelType method meets the requirements. The main method will not be tested; you may use it any way you want.

Download Triangle.java and the template TriangleTests.txt. Rename the template as FirstnameLastnameTriangleTest.java (ex. JohnSmithTriangleTest.java). The template file contains 20 test cases. The first three test cases are given; you need to complete rest of the test cases. Do not remove any test cases from the template, just change the contents of the methods to test various situations. The goal is to write a comprehensive set of test cases (i.e. reasonable and non-repeated) that covers all the different possible situations. For example, (5 5 5) and (6 6 6) are considered repeated test cases with equilateral type of arguments.

Your test cases will be graded based on the output from JUnit. A Triangle class file will be provided for testing purpose. You cannot modify the Triangle class.

There are two types of test cases:

1. Valid types - You will receive credit for submitting a test set for a correct implementation of the triangleType method from the class provided (i.e. an equilateral, isosceles or scalene triangle). The JUnit test run tells you which of your test numbers is failing.
2. Invalid types/error messages – You will need to design your test set for some bugs that we have intentionally placed into the triangleType method (i.e. error message). This part measures the quality of your test set. Have too few or too similar tests, and you won't be able to find all the bugs.