Lab Practice Assignment 2

Save each test script and screenshots of test result as per question number. Make a zip folder of all the files. Upload the zip file to Moodle.

Problem 1. (20%) Circle Radius

Read the following problem then look at its implementation in 'Circle-Radius.java'.

Write a Java program calculates the circumference of a circle (c)based on the radius (r) input by the user. The return value must be rounded up to the nearest integer. Program should return -1 if the input value is invalid.
Input:
r: 0<r<10^9
Output:
c

Write at least one test case to check if 'Circle-Radius.java' passes the following conditions:

a. Does the program return the correct result when the input is c = 4

b Does the program return the correct result when the input is c <=0

c. Does the program return the correct result when the input is c =10^9

d. Does the program return the correct result when the input is c = = ‘f’

Problem 2. (20%) Roman to Integer

Read the following problem then look at its implementation in 'RomantoInteger.java'.

Given a string in roman no format (s) your task is to convert it to an integer . Various symbols and their values are given below. Return -1 if the input string is invalid.

SYMBOL	***VALUE***
I	*1*
V	*5*
X	*10*
L	*50*
C	*100*
D	*500*
L	*1000*

Input:
S: String.lengh <10^9
Output:
I: Integer number

Here are some rules that a Roman number must follow:

a. It can’t have more than three of the same Roman symbols together

b. If a symbol comes after a symbol that is larger or equal in value, then it must be added to the symbols before it.

c. Smaller symbols placed before a larger numeral indicates subtraction of that smaller numeral from the larger one.

d. A number can’t be created by subtracting more than one value from a Roman numeral. (e.g., IV for 4, IX for 9). You cannot, for example, write something like IIV for 3.

Write at least one test case to check if the implementation produces the correct result.

Problem 3. (20%) Triangle

Consider “Triangle.java” file. There is at least one error in the implementation. Design testcases to expose that error.

Problem 4. (20%) Fibonacci

Look at the Fibonacci class. This class is an attempt at implementing the recursive method fib, which should generate the nth Fibonacci number. Create at least one test cases for this class and run those test cases using Junit.

Problem 5. (20%) Quadrilateral

The Quadrilateral class denotes a polygon with four sides. It has two methods, isRectangle() and isSquare(). Furthermore, it also uses the classes Point, Line and Vector2D. To find if the polygon is a rectangle, vectors and dot products are used to determine if every corner forms a right angle. To find if the polygon is a square, isRectangle() is used and check if the lengths of all sides are equal.

Create test cases for **each** class and execute using Junit.

~~The end~~