Assessment: CST8116 Lab Exercise 03(22W)

Student Name: Yongli Dai

Lab Professor Name: *Stanley Pieda*

Lab Section Number: *311*

Due Date: February 20

# Part One Understand the Problem

I should supply a program which can calculate the hypotenuse, perimeter and area of a triangle given the length of the adjacent and opposite sides. I will create two classes to solve the problem. And the UML Class Diagrams have already been supplied. One class is RightAngleTriangle with constructors, accessors, mutators and worker methods to calculate hypotenuse, perimeter and area. The other one is the main class named Exercise03.

Specifically speaking, in the class RightAngleTriangle, there should be instance variables like adjacent and opposite as well as get-set methods and worker methods. And then the class with method main should accept input for adjacent and opposite. There will be three worker methods to calculate hypotenuse, perimeter, area and report result respectively.

For example, if the adjacent side length of the triangle is a and the length of opposite side is b. The method calculateHypotenuse() will do calculations to get the length of the hypotenuse h, namely, h= Math.sqrt(a\*a+b\*b).

For method calculatePerimeter(), it will do calculation like perimeter = adjacent+opposite+h. Method calculateArea() would show that area = a\*b/2.

# Part Two Pseudocodes

// The following two methods are part of class RightAngleTriangle. To review the class-level variables, please refer to the UML Class Diagram.

// calculate the length of hypotenuse

calculateHypotenuse()

declaration

num hypotenuse //number containing the length of hypotenuse

hypotenuse = Math.sqrt(Math.pow(adjacent,2) +Math.pow(opposite,2))// calculate the length of hypotenuse

return hypotenuse

// calculate the length of perimeter

calculatePerimeter()

declaration

num perimeter //number containing the length of perimeter

perimeter = adjacent+opposite+calculateHypotenuse()// calculate the length of perimeter

return perimeter

// calculate the area of the triangle

calculateArea()

declaration

num area //number containing the area of the triangle

double area = adjacent\*opposite/2 //calculate the area of the triangle

return area

// create report for the result

createReport()

declaration

String adjacentStr //String containing the formatted content of the adjacent with 4 decimal places

String oppositeStr //String containing the formatted content of the opposite with 4 decimal places

String hypotenuseStr //String containing the formatted content of hypotenuse with 4 decimal places

String perimeterStr //String containing the formatted content of perimeter with 4 decimal places

String areaStr //String containing the formatted content of the area with 4 decimal places

String res // String containing the formatted report we need

res = "adjacent "+adjacentStr+", "+"opposite "+ oppositeStr+", "+"hypotenuse "+hypotenuseStr

+"\n"+"Hypotenuse: "+hypotenuseStr+"\n"+"Perimeter: "+perimeterStr+

"\n"+"Area: "+areaStr+"\n"+"Program by Yongli Dai" // formatted report of the result

return res

// pseudocode for method main of class Exercise03

start

declarations

num adjacent // number of adjacent side length

num opposite // number of opposite side length

RightAngleTriangle triangle // Reference to object of class RightAngleTriangle

output “Right angle triangle program\nEnter adjacent length: ” //prompt for input of adjacent

Input adjacent and assign it to the variable adjacent

output “Enter opposite length: ” // prompt for input of opposite

Input opposite and assign it to the variable opposite

// test set methods by setting values first

triangle.setAdjacent(adjacent)

triangle.setOpposite(opposite)

// test get methods by getting values back

adjacent = triangle.getAdjacent()

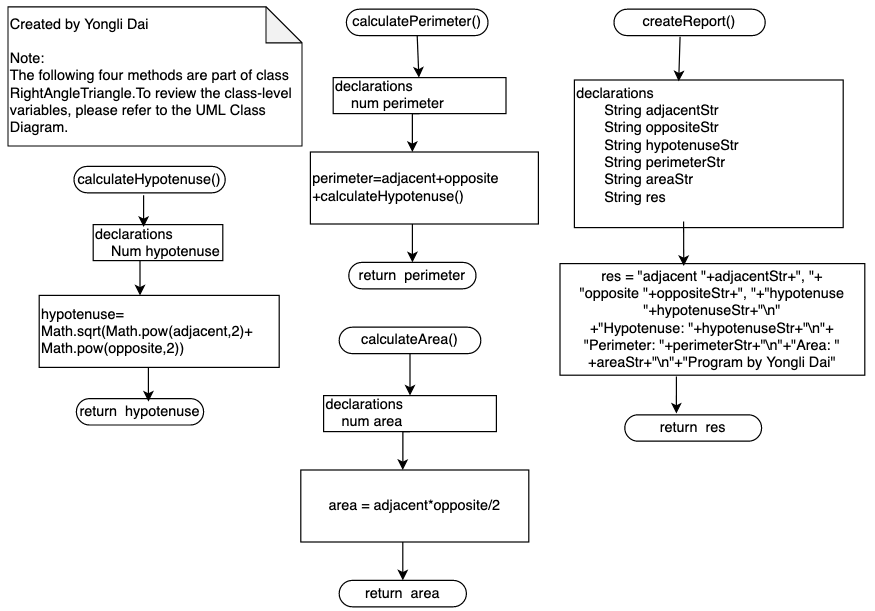
opposite = triangle.getOpposite()

// test the worker methods

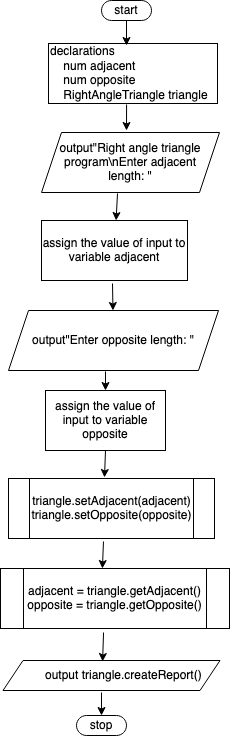
output triangle.createReport() // output the calculation results in customized format with 4 decimal places

stop

# Part Three Flowcharts



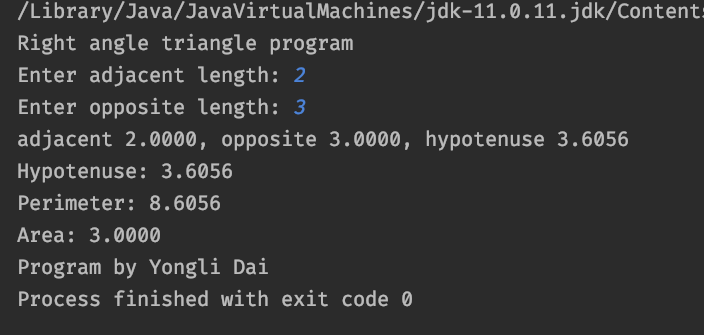
Flowchart for the main method:



# Part Four Test Plan Algorithm

|  |  |  |  |
| --- | --- | --- | --- |
| input | Expected output | Actual output | Description |
| 2  3 | Right angle triangle program  Enter adjacent length: 2  Enter opposite length: 3  adjacent 2.0000, opposite 3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: 8.6056  Area: 3.0000  Program by Yongli Dai | Right angle triangle program  Enter adjacent length: 2  Enter opposite length: 3  adjacent 2.0000, opposite 3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: 8.6056  Area: 3.0000  Program by Yongli Dai | After hand tracing of the method calls, it is found that actual output matches the expected output. |
| -2  -3 | Right angle triangle program  Enter adjacent length: -2  Enter opposite length: -3  adjacent -2.0000, opposite -3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: -1.3944  Area: 3.0000  Program by Yongli Dai | Right angle triangle program  Enter adjacent length: -2  Enter opposite length: -3  adjacent -2.0000, opposite -3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: -1.3944  Area: 3.0000  Program by Yongli Dai | Since the limit for input is just any number of double type, then passive numbers with double type would also be regarded as valid input.  The program still needs to be updated. |
| “hello” | Exception in thread "main" java.util.InputMismatchException | Exception in thread "main" java.util.InputMismatchException | The input can not be text. |

# Part Five Screen Shot of Program Execution



# Part Six Test Plan Program

|  |  |  |  |
| --- | --- | --- | --- |
| input | Expected output | Actual output | Description |
| 3  4 | Right angle triangle program  Enter adjacent length: 3  Enter opposite length: 4  adjacent 3.0000, opposite 4.0000, hypotenuse 5.0000  Hypotenuse: 5.0000  Perimeter: 12.0000  Area: 6.0000  Program by Yongli Dai | Right angle triangle program  Enter adjacent length: 3  Enter opposite length: 4  adjacent 3.0000, opposite 4.0000, hypotenuse 5.0000  Hypotenuse: 5.0000  Perimeter: 12.0000  Area: 6.0000  Program by Yongli Dai | After hand tracing of the method calls, it is found that actual output matches the expected output. |
| hi | Right angle triangle program  Enter adjacent length: hi  Exception in thread "main" java.util.InputMismatchException | Right angle triangle program  Enter adjacent length: hi  Exception in thread "main" java.util.InputMismatchException  at java.base/java.util.Scanner.throwFor(Scanner.java:939)  at java.base/java.util.Scanner.next(Scanner.java:1594)  at java.base/java.util.Scanner.nextDouble(Scanner.java:2564)  at Exercise03.main(Exercise03.java:36) | After expectation of the error, the actual output matches expected one, and supplies more information about the compilation process. |
| -3.5  -4 | Right angle triangle program  Enter adjacent length: -3.5  Enter opposite length: -4  adjacent -3.5000, opposite -4.0000, hypotenuse 5.3151  Hypotenuse: 5.3151  Perimeter: -2.1849  Area: 7.0000  Program by Yongli Dai | Right angle triangle program  Enter adjacent length: -3.5  Enter opposite length: -4  adjacent -3.5000, opposite -4.0000, hypotenuse 5.3151  Hypotenuse: 5.3151  Perimeter: -2.1849  Area: 7.0000  Program by Yongli Dai | Since the limit for input is just any number of double type, then passive numbers with double type would also be regarded as valid input.  The program still needs to be updated. |