Assessment: Lab exercise 3

Student Name: Yanzhang Wu

Lab Professor Name: *Fedor Ilitchev*

Lab Section Number: *CST8116\_323*

Due Date: June 24th 2022

# Understand the problem

This program will print several lines:

1. length of adjacent, length of opposite, length of hypotenuse
2. length of hypotenuse
3. length of perimeter
4. area of the triangle
5. name

A picture containing text

Description automatically generated

Figure 1. Output of the console

Shape

Description automatically generated

Figure 2. Right angled triangle

For a right-angled triangle with the length of adjacent side **,** the length of opposite side , and the length of hypotenuse (show as figure 2), we have

Therefore,

The perimeter of the triangle is the sum of three side, which means

The area of the triangle could be calculated from

For example, a right-angled triangle has adjacent in nondimensional length 3 and opposite in nondimensional length 4. The hypotenuse could be calculated as

The perimeter  is

And the area is

To solve this problem, I will create two classes. The first class is class RightAngleTriangle(). It will hold adjacent and opposite as properties. There will be get-set methods for adjacent and opposite. calculateHypotenuse() will be used to calculate hypotenuse. calculatePerimeter() will be used to calculate perimeter. calculateArea() will be used to calculate area. createReport() will be used to concatenate and format strings. The second class is class Exercise03 in which main() locates.

# Develop and Describe an Algorithm

## UML Class Diagrams

Text

Description automatically generated with medium confidence

Figure 3. UML digram for RightAngleTriangle

Table

Description automatically generated with medium confidence

Figure 4. UML diagram for Exercise03

## Pseudocode for calculateHypotenuse()

calculateHypotenuse()

/\* hypotenuse is the squareRoot (power(adjacent, 2)+ power(opposite, 2)) \*/

hypotenuse = Math.sqrt((Math.pow(adjacent, 2) + Math.pow(opposite, 2)))

return hypotenuse

calculatePerimeter()

/\* perimeter is the sum of three sides \*/

perimeter = adjacent + opposite + calculateHypotenuse()

return perimeter

calculateArea()

area = 0.5 \* adjacent \* opposite

return area

String createReport()

//use String.format to reserve 4 decimals

String report = String.format("adjacent %.4f, ", adjacent) + String.format("opposite %.4f, ", opposite)

+ String.format("hypotenuse %.4f", calculateHypotenuse())

return report

## Flowchart

Chart, diagram

Description automatically generated

Figure 5. Flowchart for different methods

# Algorithm Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| input | Expected output | Actual output | Description |
| 2  3 | adjacent 2.0000, opposite 3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: 8.6056  Area: 3.0000  Program by Yanzhang Wu | adjacent 2.0000, opposite 3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: 8.6056  Area: 3.0000  Program by Yanzhang Wu | A hand trace of the logic produced the correct results. |

# Source code

Text

Description automatically generated

Figure 6. Source code - class Exercise03 - part 1

Graphical user interface, text, application

Description automatically generated

Figure 7. Source code - class Exercise03 - part 2

Text

Description automatically generated

Figure 8. Source code - class RightAngleTriangle - part 1

Graphical user interface, text, application, email

Description automatically generated

Figure 9. Source code - class RightAngleTriangle - part 2

# Console output

Graphical user interface, text, application

Description automatically generated

Figure 10. Running program

# Test Plan for the Program

|  |  |  |  |
| --- | --- | --- | --- |
| input | Expected output | Actual output | Description |
| 2  3 | adjacent 2.0000, opposite 3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: 8.6056  Area: 3.0000  Program by Yanzhang Wu | adjacent 2.0000, opposite 3.0000, hypotenuse 3.6056  Hypotenuse: 3.6056  Perimeter: 8.6056  Area: 3.0000  Program by Yanzhang Wu | Java  program  output  matches the  expected  output |
| -1  0  asd  (input for adjacent) | Invalid input.  Please enter adjacent: | Invalid input.  Please enter adjacent: | Java  program  output  matches the  expected  output. Java program will ask user to enter adjacent until the input is valid |
| -5  0  qwe  (input for opposite) | Invalid input.  Please enter adjacent: | Invalid input.  Please enter adjacent: | Java  program  output  matches the  expected  output. Java program will ask user to enter opposite until the input is valid |

Text

Description automatically generated

Figure 11. Enter "-1", "0", and "asd" for adjacent

Text

Description automatically generated

Figure 12. Enter "-5", "0", and "qwe" for adjacent