CS 3305: Data Structures

Assignment 1 – OOP and Recursion

Note: If you re-upload the files, you must re-upload **ALL** files as the system keeps the most recent uploaded submission only. No zip files!

Note 2: Never hard-code test data in the test program, unless explicitly stated in the assignment. Always allow the user to enter the test data using menu option.

The goal of this assignment is to reinforce the concept of Object-Oriented Programming and implement classes in Java.

Every program **must** include author header as shown, **no header, no points!**

```
// Name:             <your name>
// Class:            CS3305/w03
// Term:             Spring 2025
//Instructor:        Prof. Wang
// Assignment:         #1
// IDE:              Intellij
```

Note: Below, *(…) means the method/function takes parameters. () means the method/function takes no parameters.*

Exercise #1 (50 points): Design and implement Java class ***Rectangle***. The class defines the following data fields and member functions:

1. Private `double` data field named `width` to store the rectangle width (default value is 1.00).
2. Private `double` data field named `height` to store the rectangle height (default value is 1.00).
3. Non-argument (default) constructor function that creates a default rectangle (with default values above).
4. Constructor function that creates a rectangle with specified width and height values from the user.
5. Get functions for the data fields `width` and `height` to read their values (i.e., const functions).
6. Function named `getArea()` that returns the area of the rectangle.
7. Function named `getPerimeter()` that returns the perimeter of the rectangle.
8. Function `printRectangle(String objectName)` to printout a meaningful description of a rectangle object. Assuming rectangle ABC is15.00 units wide and 20.00 units high, calling `printRectangle("ABC")` would display the following output:

```
Rectangle ABC is 15.0 units wide and 20.0 units high.
```

Write a test program (with main() function), named ***TestRectangle***, to create 2 rectangle objects named `myRectangle` and `yourRectangle` as follows:

- `myRectangle` is a default object. (i.e., uses default width and height).
- `yourRectangle` has width and height values specified by the user.

Using proper class functions, display the width, height, area, and perimeter for each object as follows (the example shows you the output format. Shown values are just for illustration. See Appendix F, page 785 on how to format program outputs. Make sure your outputs are formatted as shown):

Test data:

```
myRectangle:
------------
Width:     1.00
Height:    1.00
Area:      1.00
Perimeter: 4.00

Rectangle myRectangle is 1.0 unit wide and 1.0 unit high.

yourRectangle:
--------------
Width:     3.00
Height:    4.00
Area:      12.00
Perimeter: 14.00

Rectangle yourRectangle is 3.0 units wide and 4.0 units high.
```

Allow the user to enter the data values for object `yourRectangle`. Test all class functions on at least one of the objects in a logical order and display meaningful information about the object after each function call.

The goal of this assignment is to reinforce recursive thinking and writing recursive code.

Exercise #2 (50 points): Write a Java program (in file named vowels.java) as follows. The main function of the program prompts the user to enter a string. The main function then passes the string as argument to another recursive function named countVowels(…)(part of file vowels.java). The recursive function takes the string argument as a parameter and recursively counts and returns the number of vowels (a, e, i, o, u) in the passed string. Format the output as shown below and use the sample test data below to test your code. Allow the user to enter the input string using proper input prompt.

The class defines the following member functions:

Integrate the following simple menu in the program with these options:
-----MAIN MENU-----
1. Read input string
2. Compute number of vowels
3. Exit program
Enter option number:

Always re-display the menu after an option (other than option 3) is fully exercised with blank lines before and after the menu.

Test data is entered using the menu options. Do Not use output labels as input prompts. Test your code with the following sample inputs and format the output as shown below. Notice how output values are lined-up after the labels.

```
You entered string: This is a test input
Number of vowels:   6

You entered string: Hello world
Number of vowels:   3

You entered string: My list of Objects
Number of vowels:   4
```

Submission:

Before submitting your programs, make sure you review the assignment submission requirements and grading guidelines on the course webpage. The grading guidelines explain some of the common errors found in programming assignments.

1. The assignment is due no later than **11:59 PM** on the due day posted in D2L.
2. Please compile and run your Java files (only the .java files) right before you upload them to the assignment submission folder in D2L.
3. Please submit all individual .java files.