Lab 8: Classes and Methods
20 pts

Distribute on November 17, 2025
Due November 23, 2025 (Sunday) at 11:59 pm before 12:00 midnight

Learning Outcomes (CLO vs SO Mapping)

- Utilize Java syntax in fundamental programming algorithms (3)vs(1)
- Recognize and apply the various control structures (5)vs(1)
- Design and implement elementary multi-class solutions to programming problems (6)vs(2, 6)
- Recognize and apply the basic debugging strategies in programming (8)vs(2)
- Recognize the need for arrays in the solutions of programming problems and manipulate data in one-dimensional arrays (7)vs(1, 6)

Requirements

This lab focuses on further experience with Java user-defined classes, methods, and testing.

Preliminaries

1. Create a Java project; the name of the project must be **lab08_<your FirstNameLastName>**.
2. For example, my project would be named, lab08_PeterNg

3. Add **two** Java classes in the order named **Applications and Rectangle** in your project; when using the NetBeans editor, add the Applications class first and mark it as your main class

4. Add the following comment block to the beginning of each of your classes:

```
/*
 * <your name>
 * CS 16000-01 02/03, Fall Semester 2025
 * (Note: Write either 02 or 03, depending on which section your section is.)
 * Lab 8
 *
 */
```

5. Make additional documentation about the purpose of the given classes.

6. **Ask the TA to verify your work**

Exercises

1. Implement the **Rectangle** class described in the book and explained in lectures, and add more properties described below. The requirements are as follows.

The class has

a. Two private data fields of type double, named **length** and **width**

b. Each field has an associated accessor method (getter) and a mutator method (setter); the mutator method takes a parameter and sets the field by passing the parameter, but only if the parameter is not negative. For a negative parameter, the field is assigned zero

c. Implement a method named **computeArea( )** (replacing getArea of the book) such that, using the fields, it computes and **returns** the area of the rectangle

d. Implement a method named **computePerimeter( )** such that, using the fields, it computes and **returns** the perimeter of the rectangle

e. Implement a method named **toString( ),** which **returns** a string message. The message is built using the method to reveal the field values, with references such as "The length is: " and "The width is: ". This method does NOT print the message! A sample display when the message is printed may look like this:

```
The length is: 29.66
The width is: 17.03
```

f. Implement a method named **displayRectangle( ).** The method is void.

The method calls the toString( ) method and prints the returned value to the console.

g. Implement a method named **equals( ),** which **returns** a **boolean** value. The method takes a Rectangle type parameter and compares the class fields with the parameter Rectangle. Returns true if the fields are equal and false otherwise. The header shall be

```
public boolean equals(Rectangle other){
```

The method uses a boolean expression built from

```
length == other.length
```

and the like for **width**.
Or, you may apply an if-else logic to determine the return value.

h. The class contains an **initializer** constructor and the **default** constructor:

```
public Rectangle(double len, double w){
      length = len;
      width = w;
}

public Rectangle(){

}
```

i. Add a constructor to the class for initializing a rectangle, which is an instance of the class rectangle.

```
public Rectangle_A(Rectangle_A rectangle){
      this.length = rectangle.length;
      width = rectangle.width;
}
```

2. The **Applications** class contains the main method and no other methods or fields. In the method

a) Declare local variables length and width of type double

b) Using a Scanner object, solicit two numbers from the console and save the values in the variables' length and width

c) Declare and instantiate a Rectangle object named **box**. Use the initializer constructor; the parameters are the length and width

d) Declare and instantiate another Rectangle object named box2. Use the default constructor.

e) Use box2 to call the setLength() and setWidth() methods. Choose numbers for parameters at will.

f) Call the display( ) method for the box and box2. Copy the console output into a comment block placed after the class. Check and comment on whether the field values correspond to the input.

g) Use box to call the equals( ) method, use box2 for the parameter. Print the returned boolean to the console. Comment on the output.

h) Declare and instantiate a third Rectangle object named box3. Use the default constructor.

i) Use the box to call the getter methods and assign the local variables' width and length to the values returned by the getters.

j) Use box3 to call the setters (like above for box2), the setters this time take the local variables for the parameter.

k) Use the box to call the equals method and use box3 for the parameter. Print the returned boolean to the console. Comment on the output in the comment block you already created above.

l) Call the computeArea( ) and computePerimeter( ) methods with respect to all your boxes. Print the return values to the console, each with a short explanation, and copy the output into the comment block.

m) Declare and instantiate a Rectangle object named **box4**. Use the constructor to initialize the rectangle, which is an instance of the class Rectangle. The instance is **box2**. Use box4 to call the setLength() and setWidth() methods to print the length and width of box4. Then, call the display( ) method for box4. Do box2 and box4 have the same length and width?

Testing and Applications

As described for the Applications class above

Submit

- Zip your project folder and submit your assignment on Brightspace at the designated link