

2019/2020

Lab 1: Object and Class

FAKULTI TEKNOLOGI KEJURUTERAAN KELAUTAN DAN INFORMATIK

**DATA STRUCTURE & ALGORITHM**



**VERSION 1**

**Name:** Click or tap here to enter text.  
**Matric Number:** Click or tap here to enter text.   
**Lab:** Choose an item.   
**Date:** Click or tap to enter a date.

Table of Contents

[INSTRUCTIONS 1](#_Toc16069964)

[TASK 1: BASIC KNOWLEDGE ON OBJECT AND CLASS 2](#_Toc16069965)

[TASK 2: POST-LABORATORY PROBLEMS](#_Toc16069966) 10

# INSTRUCTIONS

Manual makmal ini adalah untuk kegunaan pelajar-pelajar Fakulti Teknologi Kejuruteraan Kelautan dan Informatik, Universiti Malaysia Terengganu (UMT) sahaja. Tidak dibenarkan mencetak dan mengedar manual ini tanpa kebenaran rasmi daripada penulis.

Sila ikuti langkah demi langkah sebagaimana yang dinyatakan di dalam manual.

This laboratory manual is for use by the students of the Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu (UMT) only. It is not permissible to print and distribute this manual without the official authorisation of the author.

Please follow step by step as described in the manual.

# TASK 1: BASIC KNOWLEDGE ON OBJECT AND CLASS

## ObjectiveS

At the end of this lab, the students are able to

1. Write a class with instance variable fields of a user
2. Write constructors with and without parameters.
3. Access class members from member methods of the class.

## Estimated Time

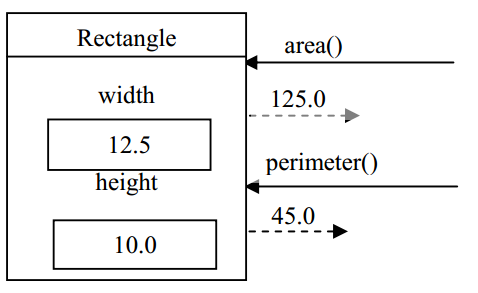
[120 Minutes]

### steps:

In this lab, the Rectangle class will be defined.

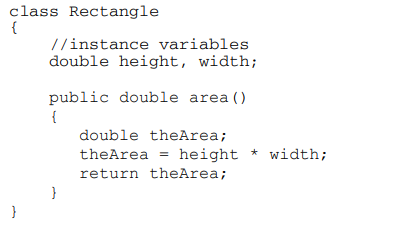
A rectangle has a height and width, and given this data, the area and perimeter of a rectangle can be determined. Therefore, the class Rectangle should define data members: height and width. Since each Rectangle object stores its own values for these variables, height and width are instance variables. The methods are the method's that determine the area and perimeter of a Rectangle object and are, therefore, instance methods. The Rectangle class contains only instance variables and instance methods.

The following diagram is a diagram of a Rectangle object showing the instance variables width and height and the methods area and perimeter. The state of the object are the values of its data members. Since the state of this object has been set, also illustrated are the values returned when the area and perimeter methods are invoked on the object.



**1.1 Testing the utility class**

1. Run the following code and place as Rectangle.java



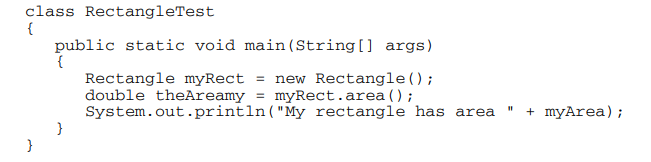
1. Record the error message.

|  |
| --- |
|  |

1. Why does the error happen?

|  |
| --- |
| Because there is no main method in this class to run the coding. |

1. Enter and save this in a file called RectangleTest.java.



1. Compile the program RectangleTest.java. Execute the program and record the results.

|  |
| --- |
|  |

1. Currently, you can access the values of width and height directly by joining the variable to the name of the object using the dot operator. Add the following statements to the end of the main method.



Predict the output of the new program.

|  |
| --- |
| A runtime error message will pop up. Because the height and width values is neither initiated nor assigned a value for it. |

1. Compile and execute the program. Record the results. Was your Step 5 prediction correct? If not, correct your answers.

|  |
| --- |
| My prediction was wrong. |

1. Modify the program by adding these statements at an appropriate place in the main method so that the area of myRect is no longer 0.



Compile and execute the program. Record the results.

|  |
| --- |
|  |

**1.2 Access Modifier: private**

1. Being able to directly access the instance variables of an object (Rectangle) from an outside class (RectangleTest) is considered to be an inappropriate practice in object-oriented languages. To prevent this, the instance variables of a class should be modified by the access modifier private. Modify the Rectangle class by inserting the private modifier in the data member declaration statement:

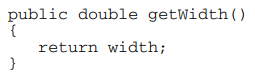
private double width, height;

Compile the modified program. Record the compiler error messages.

|  |
| --- |
|  |

**1.3 Accessor Methods**

1. In the Rectangle class, insert the code for the method getWidth which has no parameters and returns a double



Compile the code. Then, insert a similar method to give the user, or client, access to the height of a Rectangle.

1. Modify the client class, RectangleTest, to correctly access the width and height of the Rectangle object.

This is an example of code to access the width of the Rectangle object.



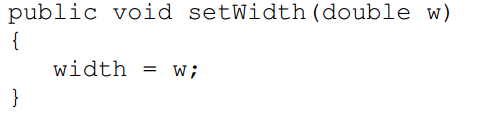
Insert the similar code to access the height of the Rectangle object.

Compile RectangleTest.java and execute the program. Record the results.

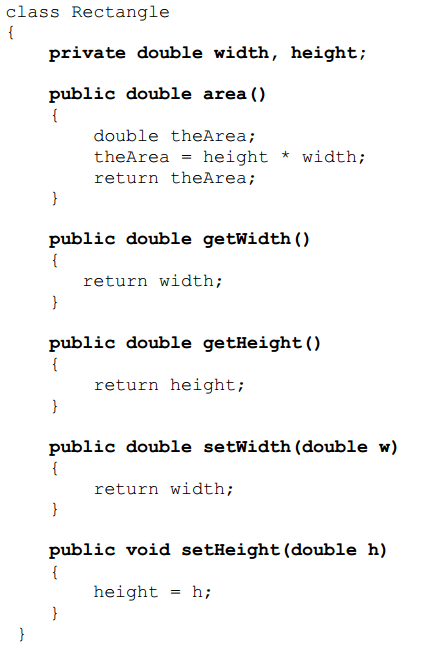
|  |
| --- |
|  |

**1.4 Mutator Methods**

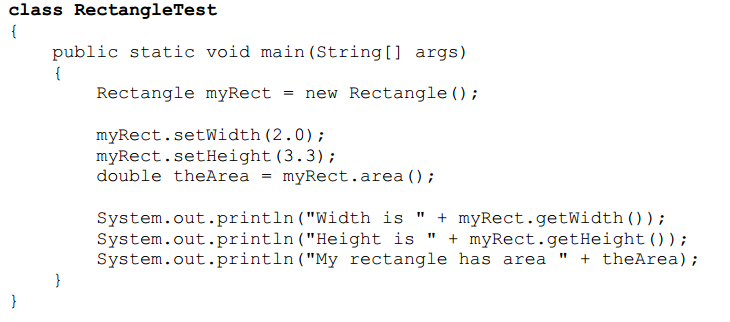
1. The completed method should be added to the Rectangle class:



Add the methods setWidth and setHeight to the Rectangle class. Make changes to RectangleTest to correctly use these methods. Your Rectangle class should now be:



And, your RectangleTest class should now be:



Compile RectangleTest. Execute the program and record the results.

|  |
| --- |
|  |

Modify the Rectangle class by adding the constructor, placing it after the declaration of the instance variables, and before the definitions of the existing methods. This location is not mandatory, but it makes the code more readable.

Also, do the following to modify the class RectangleTest class

* 1. Comment out the two statements in main that invoke the set methods.
  2. Change the statement that creates the Rectangle object from



to

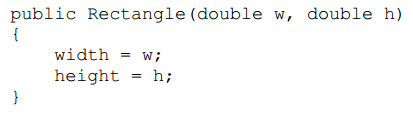


Compile and execute the modified program. Record the results

|  |
| --- |
|  |

* 1. **Writing a Constructor**

1. A constructor must have the same name as the class name. Therefore, a constructor used to construct a Rectangle object, must be named Rectangle. We say that a constructor is a special type of method because it does not have a return type and because it can only be used in conjunction with the new operator. A constructor that initializes the height and width of a Rectangle object would take the form



Add the constructor to the Rectangle.java.

Compile and execute the modified program. Record the results

|  |
| --- |
|  |

* 1. **Completing the Rectangle class**

Now, complete the Rectangle class by

* Adding a method to find the perimeter of a Rectangle object with the header that calculates and returns the perimeter of the rectangle.



Compile and execute the modified program. Record the results

|  |
| --- |
|  |

# TASK 2: POST-LABORATORY PROBLEM

## objective

To test the understanding of basic concepts and terminologies in object and class.

## task description

This task is basically to test the understanding of basic concepts and terminologies in object and class.

## estimated time

[60 Minutes]

Design a class named Kipas to represent a fan. The class contains:

1. Three constants named PERLAHAN, SEDERHANA, and LAJU with the values 1, 2, and 3 to denote the fan speed.
2. A private int data field named speed that specifies the speed of the fan (the default is PERLAHAN).
3. A private boolean data field named on that specifies whether the fan is on (the default is false).
4. A private double data field named radius that specifies the radius of the fan (the default is 5).
5. A string data field named colour that specifies the colour of the fan (the default is biru).
6. The accessor and mutator methods for all four data fields.
7. A no-arg constructor that creates a default fan.
8. A method named toString() that returns a string description for the fan. If the fan is on, the method returns the fan speed, colour, and radius in one combined string. If the fan is not on, the method returns the fan color and radius along with the string “fan is off” in one combined string.

Write a test program that creates two Fan objects. Assign maximum speed, radius 10, color merah, and turn it on to the first object. Assign medium speed, radius 5, color biru, and turn it off to the second object. Display the objects by invoking their toString method.

Both Fan.java and TestFan.java should be submitted via **epembelajaran.**