Lab 02
Defining classes and objects in JAVA

Objective:

Objective of this lab is to understand the importance of classes and construction of objects using constructors

Activity Outcomes:

The student will be able to declare a classes and objects.
The student will be able to declare member functions and member variables of
a class. The student will be able to declare overloaded constructors.

Instructor Note:

As pre-lab activity, read Chapter 9 from the text book “Introduction to Java Programming”, Y. Daniel Liang, Pearson, 2019.

1) Useful Concepts

- **Data Abstraction**
  Abstraction is the process of recognizing and focusing on important characteristics of a situation or object and leaving/filtering out the un-wanted characteristics of that situation or object. For example a person will be viewed differently by a doctor and an employer.
  A doctor sees the person as patient. Thus he is interested in name, height, weight, age, blood group, previous or existing diseases etc of a person.
  An employer sees a person as an employee. Therefore, employer is interested in name, age, health, degree of study, work experience etc of a person.
- **Class and Object:**
  The fundamental idea behind object-oriented languages is to combine into a single unit both data and the functions that operate on that data. Such a unit is called an object. A class serves as a plan, or blueprint. It specifies what data and what functions will be included in objects of that class. An object is often called an ―instance‖ of a class.
- **Instance Variables and Methods:**
  Instance variables represent the characteristics of the object and methods represent the behavior of the object. For example length & width are the instance variables of class Rectangle and Calculatearea() is a method.
  Instance variables and methods belong to some class, and are defined inside the class to which they belong.
  Syntax:

```
public class Class_Name
{
Instance_Variable_Declaration_1
Instance_Variable_Declaration_2
. . .
Instance_Variable_Declaration_Last

Method_Definition_1 Method_Definition_2
. . .
Method_Definition_Last
}
```

- **Constructors:**

  It is a special function that is automatically executed when an object of that class is created. It has no return type and has the same name as that of the class. It is normally defined in classes to

initialize data members. A constructor with no parameters is called a no-argument constructor. A constructor may contain arguments which can be used for initiation of data members.

Syntax:

```
class_name( )
{
      Public class_name()
      {
      //body
      }

      Public class_name(type var1, type var2)
      {
      //body
      }
}
```

If your class definition contains absolutely no constructor definitions, then Java will automatically create a no-argument constructor. If your class definition contains one or more constructor definitions, then Java does not automatically generate any constructor; in this case, what you define is what you get. Most of the classes you define should include a definition of a no-argument constructor.

2) Solved Lab Activities (Allocated Time 1 Hr.)

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
Activity 1	*10 mins*	*Low*	*CLO-4*
Activity 2	*20 mins*	*Low*	*CLO-4*
Activity 3	*20 mins*	*Low*	*CLO-4*

Activity 1:

The following example shows the declaration of class Rectangle. It has two data members that represent the length and width of rectangle. The method calculateArea will return the area of rectangle. The runner class will create an object of Rectangle class and area function will be called.

Solution:

```
class Rectangle {

    public int length, width;

    public int Calculatearea() {
        return (length * width);
    }
}

public class runner {

    public static void main(String args[]) {
        Rectangle rect = new Rectangle();
        rect.length = 10;
        rect.width = 5;
        System.out.println(rect.Calculatearea());
    }
}
```

Activity 2:

The following example demonstrates the use of constructors

Solution:

```
class Rectangle {

    public int length, width;

    public Rectangle() {
        length = 5;
        width = 2;
    }

    public Rectangle(int l, int w) {
```

```
        length = l;
        width = w;
    }

    public int Calculatearea() {
        return (length * width);
    }
}
public class runner {

        public static void main(String args[]) {
        Rectangle rect = new Rectangle();
        System.out.println(rect.Calculatearea());
        Rectangle rect1 = new Rectangle(10, 20);
        System.out.println(rect1.Calculatearea());
    }
}
```

Activity 3:

The following example shows the declaration of class Point. It has two data members that represent the x and y coordinate. Create two constructors and a function to move the point. The runner class will create an object of Point class and move function will be called.

Solution:

```
class Point {

    private int x;
    private int y;

    public Point() {
        x = 1;
```

```
        y = 2;
    }

    public Point(int a, int b) {
        x = a;
        y = b;
    }

    public void setX(int a) {
        x = a;
    }

    public void setY(int b) {
        y = b;
    }

    public void display() {
        System.out.println("x coordinate = " + x + " y coordinate =
" + y);
    }

    public void movePoint(int a, int b) {
        x = x + a;
        y = y + b;
        System.out.println("x coordinate after moving = " + x + " y
coordinate after moving = " + y);
    }

}


public class runner {

       public static void main(String args[]) {

        Point p1 = new Point();
        p1.movePoint(2, 3);
```

```
        p1.display();

        Point p2 = new Point();
        p2.movePoint(2, 3);
        p2.display();

    }

}
```

3) Graded Lab Tasks (Allocated Time 2 Hr.)

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Create a class circle class with radius as data member. Create two constructors (no argument, and two arguments) and a method to calculate Circumference.

Lab Task 2

Create a class Account class with balance as data member. Create two constructors (no argument, and two arguments) and methods to withdraw and deposit balance.

Lab Task 3

Create a class –Distance\ with two constructors (no argument, and two argument), two data members (feet and inches). Also create display function which displays all data members.

Lab Task 4

Write a class Marks with three data members to store three marks. Create two constructors and a method to calculate and return the sum.

Lab Task 5

Write a class Time with three data members to store hr, min and seconds. Create two constructors and apply checks to set valid time. Also create display function which displays all data members.