INSTITUTE OF INFORMATION TECHNOLOGY

JAHANGIRNAGAR UNIVERSITY

Number of Assignment : 01

Submission Date : 02/07/2021

Course Tittle : Object Oriented Programming

Course Code : ICT - 2103

Submitted To

Dr. Jesmin Akhter

Professor

IIT – JU

Submitted By

Md. Shakil Hossain

Class Roll – 2023

Exam Roll – 192340

IIT – JU

Md. Shakil Hossain
Class Roll :- 2023
Exam Roll :- 19 23 40

Course code :- ICT - 1203
Course name :- Object Oriented Programming

Object Oriented Programming is a computer Programming model that organizes software design around data or objects. rather than function and logic.

There are 6 pillars of OOP are

1. Class
2. Object and methods
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation.

Class Roll:- 2023
Exam Roll:- 19 23 40

1. Class :

A Class is a template for manufacturing objects. It can define types of operations or methods that can be performed on a Car Object. We declare a class by specifying the class keyword followed by a non-reserved identifier that names it. A class's body is Populated with fields methods and constructors. An application of O.O.P is implemented by one or more classes. A class defines a new type of data. It's user define type that describes what a certain type of Object will look like.

Roll - 192340

Example:

```
class Box {
        double width;
        double height;
        double depth;
      }
// This class declares an object of type Box.

class BoxDemo {
    Public static void main (String args[]) {
        Box mybox = new box();
        double vol;
        // assign value of mybox instance variable
        mybox.width = 10;
        mybox.height = 15;
        mybox.depth = 5;
        // Compute value of box
        vol = mybox.width * mybox.height * mybox.depth;
        System.out.println("Volume is "+vol);
    }
}
```

2. Object and Methods

A method in Object Oriented Programming is a Procedure associated with a message and an object. It is the equivalent of a function in OOP. The methods are the actions that Perform Operations on a variable. A method accepts Parameters as arguments manipulateds these and then Produces an output when the method is called on an object. Methods are also classified according to their Purpose in the class design.

Object is an instance of a class. An object has three characteristics state, behavior and identity. Object in OOP is an abstract data type created by a developer. It can include multiple Properties and methods and may even contain other objects. Objects Provide the data within an Object is Protected from being modified or destroyed by other

function or methods unles explicitly allowed

Example:

Creating multiple objects by one type Only in which Volume method doesn't return a value.

```
class Box {
        double width;
        double height;
        double depth;
        // display volume of a box.

  Void Volume(){
        System.out.println("Volume is");
        System.out.println(width* height* depth)
      }
}
```

Roll- 1923 40

```
Class BoxDemo {
  Public static void main (String args[]) {
    Box mybox1 = new Box();
    Box mybox2 = new Box();

    mybox1.width = 10;
    mybox1.height = 20;
    mybox1.depth = 15;

    mybox2.width = 3;
    mybox2.height = 6;
    mybox2.depth = 9;

   mybox1.Volume();
   mybox2.Volume();

  }
}
```

3. Inheritance: Inheritance is a mechanism in which one class acquires the property of another class. For example a child inherits the traits of his/her parents. With inheritance we can reuse the fields and methods of the existing class. Hence inheritance facilatates reuseability and is an important concept of OOPs. There are various types of inheritance single inheritance, Multiple inheritance, Multilevel inheritance, Hierarchical inheritance, Hybrid inheritance.

Example:

```
class Doctor {
    Void Doctor_Details () {
        System.Out.println (" Surgeon Detail...");
    }
}
```

```
class Surgeon extends Doctor {
    Void Surgeon_Details () {
        System.out.println ("Surgeon Detail...");
    }
}
Public class Hospital {
    Public static Void main (String args[]) {
        Surgeon s = new Surgeon();
        s. Doctor_Details ();
        s. Surgeon_Details ();
    }
}
```

4. Polymorphism: Polymorphism is the ability forms of an object to take on many forms. The most common use of Polymorphism in OOP occurs when a Parent class reference is used to refer to a child class object. In Java all Java objects are polymorphic since any object will Pass the IS-A test for their own type and for the class object.

Example:

```
Public class Salary extends Employee {
  Private double salary;
  Public Salary (String name, String address, int
                 number, double salary){

  SetSalary (salary);
  Super (name, address, number);
  }
```

```
Public Void mailcheck () {
    System.out.Println ("within mailcheck of salary
                                        class");
    System.out.Println ("Mailing check to "+ getname()
                        +with salary" + salary);
  }
Public double getsalary () {
      return salary;
}
Public Void setsalary (double newsalary) {
        it (new salary >= 0.0) {
            salary = new salary;
      }
}
Public double computePay () {
    System.out.Println ("Computing salary Pay
                        for " + getName ());
    return salary /52.
  }
}
```

5. Abstraction: Abstraction is the concept of Object Oriented Programming that "shows" only essential attributes and "hides" unnecessary information. The main purpose of abstraction is hiding the unnecessary details from users. Abstraction is selecting data from a larger Pool to show only relevant details of the Object to the user. It helps in reducing Programming complexity and efforts. It is one of the most important concepts of OOP. The main benefit of using an Abstraction in Programming is that it allows us to group several related class as siblings.

Example:

```
abstract class Bike {
    abstract void run();
}
```

```
class Honday extends Bike {
    Void run()
    {
      System.out.Println("running safely");
    }
Public Static Void main (String args[]) {
        Bike obj = new Honda4();
        Obj.run();
    }
}
```

6. Encapsulation:

The Process of binding data and Corresponding methods togethem into a single unit is called encapsulation in Java. Encapsulation is a mechanism of Packaging the data and Code acting on the methods togetion as a Single accessed by outside of the Package The whole idea behind encapsulation is to hide the implementation details brom the users. Advantages of encapsulation are data hiding increased flenibility, maintainability reusability, testing of code less error Prone, Provides more contans security etc.

Example:

```
Public class Student {

    Private String name;
    Private int roll;
    double number;
```

```
Public Void setName (String n)
        {
         name = n;
        }
Public Void setRoll (int r)
        {
         roll = r;
        }
Public String getName ()
        {
         return name;
        }
Public int getRoll ()
        {
         return roll;
        }
}
```

THE END