

**Lab Report**

Course Code : **CSE-215**

Course Title : **Object Oriented Programming Lab.**

Lab Report Topic : Object Oriented Programming

**Submitted To**

Ms. Nasima Islam Bithi

**Submitted By**

Rakibul Hasan Akash

ID: 221-15-5688

Sec : V

Department of CSE  
Daffodil International University**.**

Department of CSE,

Daffodil International University

**Date of submission : 28-05-2023**

**Problem-1:**

public class Room {  
 public int roomNo;  
 public String roomType;  
 public double roomArea;  
 public boolean acMachine;  
  
 public void SetData(int roomNo, String roomType, double roomArea, boolean acMachine) {  
 this.roomNo = roomNo;  
 this.roomType = roomType;  
 this.roomArea = roomArea;  
 this.acMachine = acMachine;  
 }  
  
 public void DisPlayData() {  
 System.*out*.println("Room No: " + roomNo);  
 System.*out*.println("Room Type: " + roomType);  
 System.*out*.println("Room Area: " + roomArea + " Meter");  
 System.*out*.println("Acmachine is= " + acMachine);  
 }  
  
}

public class Main {  
 public static void main(String[] args) {  
 Room Labroom = new Room();  
 Labroom.SetData(814, "Lab", 1000, false);  
 Labroom.DisPlayData();  
  
  
 }  
}

**Problem-2:**

public class Simpleobject {  
 public Simpleobject() {  
 System.*out*.println("Hello, simple object is created");  
 }  
 public static void main(String[] args) {  
 Simpleobject A1 = new Simpleobject();  
 }  
}

**Problem-3:**

public class This {  
 public void FirstMethod(){  
 System.*out*.println("Hello,I am First Method");  
 }  
 public void SecondMethed(){  
 this.FirstMethod();  
 System.*out*.println("Hello,I am Second Method");  
 }  
 public static void main(String[] args) {  
 This T1 = new This();  
 T1.SecondMethed();  
 }  
}

**Problem-4:**

public class Parents {  
 public String name;  
 public int age;  
  
 public Parents(String name, int age) {  
 this.age = age;  
 this.name = name;  
 }  
 public void Display(){  
 System.*out*.println("Peremts Name= "+name);  
 System.*out*.println("Age= "+age);

}  
}

public class Child extends Parents{  
 public String childname;  
  
 public Child(String name, int age, String childname) {  
 super(name, age);  
 this.childname = childname;  
 }  
 public void Display(){  
 super.Display();  
 System.*out*.println("Child name= "+childname);  
 }  
  
}

public class Main {  
 public static void main(String[] args) {  
 Child Akash=new Child("RaZZAk",21,"akash");  
 Akash.Display();  
 }  
}

**Problem-5:**

public class MethodOverloading {  
 int a, b, c;  
  
 public int add(int a, int b) {  
 return a + b;  
 }  
  
 int add(int a, int b, int c) {  
 return a + b + c;  
 }  
  
 public double add(double a, int b) {  
 return a \* b;  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 MethodOverloading obj=new MethodOverloading();  
 System.*out*.println(obj.add(10, 20));  
 System.*out*.println(obj.add(10, 20, 30));  
 System.*out*.println(obj.add(5.3, 2));  
 }  
}

**Problem-6:**

class A {  
 public void displayMessage() {  
 System.*out*.println("Hello from class A");  
 }  
}

class B extends A {  
 @Override  
 public void displayMessage() {  
 System.*out*.println("Hello from class B");  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 A objA = new A();  
 objA.displayMessage();  
  
 B objB = new B();  
 objB.displayMessage();  
 }  
}

**Problem-7:**

public class Shape {

public void draw() {

System.out.println("Drawing a circle");

}

public void erase() {

System.out.println("Erasing a circle");

}

}

class Circle extends Shape {

@Override

public void draw() {

System.out.println("Drawing a circle");

}

@Override

public void erase() {

System.out.println("Erasing a circle");

}

}

class Triangle extends Shape {

@Override

public void draw() {

System.out.println(" I am,Drawing a triangle");

}

@Override

public void erase() {

System.out.println("I am,Erasing a triangle");

}

}

class Square extends Shape {

@Override

public void draw() {

System.out.println("I am,Drawing a square");

}

@Override

public void erase() {

System.out.println("I am,Erasing a square");

}

}

public class ShapeExample {

public static void main(String[] args) {

Shape circle = new Circle();

circle.draw();

circle.erase();

Shape triangle = new Triangle();

triangle.draw();

triangle.erase();

Shape square = new Square();

square.draw();

square.erase();

}

}

**Problem-8:**

abstract class Shape {  
 public abstract void draw();  
 public abstract void erase();  
}

public class Circle extends Shape {  
 @Override  
 public void draw() {  
 System.*out*.println("Drawing a circle");  
 }  
  
 @Override  
 public void erase() {  
 System.*out*.println("Erasing a circle");  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 Shape s1=new Circle();  
 s1.draw();  
 s1.erase();  
 }  
}

**Problelm-9:**

public interface A {  
 void meth1();  
 void meth2();  
}

public class Myclass implements A{  
 public void meth1() {  
 System.*out*.println("Hi ,How are You");  
 }  
  
 @Override  
 public void meth2() {  
 System.*out*.println("I am fine,and you");  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 Myclass obj = new Myclass();  
 obj.meth1();  
 obj.meth2();  
 }  
}

**Problem-10:**

public interface A {  
 void run();  
}

public interface B {  
 void Stop();  
}

public class C implements A,B {  
 @Override  
 public void run() {  
 System.*out*.println(" Hi, i am inside A interface");  
 }  
 @Override  
 public void Stop() {  
 System.*out*.println("HI,i am inside B interface");  
  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 C obj=new C();  
 obj.run();  
 obj.Stop();  
  
 }  
}

**Problem-11:**

public interface Test {  
 int Square(int num);  
}

public class Arithmetic implements Test{  
 @Override  
 public int Square(int num) {  
 return num\*num;  
 }  
  
}

public class ToTestInt {  
 public static void main(String[] args) {  
 Arithmetic a1=new Arithmetic();  
 System.*out*.println("Square value is ="+a1.Square(5));  
 }  
}

**Problem-13:**

public class Fibo {  
 public static int fibo(int i) {  
 if (i == 0) {  
 return 0;  
 } else if (i == 1) {  
 return 1;  
 }  
 return *fibo*(i - 1) + *fibo*(i - 2);  
 }  
  
 public static void main(String[] args) {  
 for (int i = 0; i <= 5; i++) {  
 System.*out*.print(*fibo*(i) + "\t");  
 }  
 }  
}

**Problem-14:**

public class Point {  
 private int x;  
 private int y;  
 public Point(){  
 System.*out*.println("Hi, iam Default constructor");  
 }  
  
 public Point(int x, int y) {  
 this.x = x;  
 this.y = y;  
 }  
  
 public int getX() {  
 return x;  
 }  
  
 public void setX(int x) {  
 this.x = x;  
 }  
  
 public int getY() {  
 return y;  
 }  
  
 public void setY(int y) {  
 this.y = y;  
 }  
 public void SetXY(int x,int y){  
 this.x = x;  
 this.y = y;  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 Point p1=new Point(8,9);  
 System.*out*.println(p1.getX());  
 System.*out*.println(p1.getY());  
 }  
}

**Problem-15:**

public class Box {  
 private double length, breadth, height;  
 public Box(){  
 }  
 public Box(double length, double breadth, double height) {  
 this.length = length;  
 this.breadth = breadth;  
 this.height = height;  
 }  
 public void Area(){  
 System.*out*.println("Area is = "+breadth\*length);  
 }  
 public void Volume(){  
 System.*out*.println("Area is = "+breadth\*length\*height);  
 }  
}

public class Box3d extends Box {  
 private double length, breadth, height;  
  
 public Box3d(double length, double breadth, double height) {  
  
 this.length = length;  
 this.breadth = breadth;  
 this.height = height;  
 }  
  
 public void Area() {  
 System.*out*.println("Area is = " + breadth \* length);  
 }  
  
 public void Volume() {  
 System.*out*.println("Area is = " + breadth \* length \* height);  
 }  
  
}

public class Main {  
 public static void main(String[] args) {  
 Box b1 =new Box(12,5,2);  
 b1.Area();  
 b1.Volume();  
 Box3d b2=new Box3d(60,10,6);  
 b2.Area();  
 b2.Volume();  
  
 }  
}

**Problem-16:**

public class AddressInfo {

    String street,city,state,postalCode;

    public AddressInfo(String street, String city,String state,String postalCode) {

        this.street=street;

        this.city=city;

        this.state=state;

        this.postalCode=postalCode;

    }

}

public class Employeee {

    String name;

    AddressInfo addressInfo;

    InsuranceInfo insuranceInfo;

    public Employeee(String name, AddressInfo addressInfo, InsuranceInfo insuranceInfo)

    {

        this.name=name;

        this.addressInfo=addressInfo;

        this.insuranceInfo=insuranceInfo;

    }

    public void display()

    {

        System.out.println("Name of Employee : "+this.name);

        System.out.println("Address of Employee: "+

                addressInfo.street+", "+addressInfo.city+", "+

                addressInfo.state+", "+addressInfo.postalCode);

        System.out.println("Insurance information: "+

                insuranceInfo.policyName+", "+insuranceInfo.policyId);

    }

}

public class Main {

    public static void main(String[] args) {

        AddressInfo addressInfo= new AddressInfo("Sector 4", "Uttara", "Dhaka", "1711");

        InsuranceInfo insuranceInfo =new InsuranceInfo("Life Insurance", 123);

        Employeee employeee =new Employeee("Mr. Rahim",addressInfo, insuranceInfo );

        employeee.display();

    }

**Problem-17:**

public class Exception {  
 public static void main(String[] args) {  
 try {  
 int[] array = new int[-5];  
 } catch (NegativeArraySizeException e) {  
  
 System.*out*.println("Neagitve array size not possible");  
 System.*out*.println(e);  
 } finally {  
 System.*out*.println("I am always execute");  
 }  
 }  
}

**Problem-18:**

import java.util.Scanner;  
  
public class Exception {  
 public static void main(String[] args) {  
 try {  
 int a=10;  
 int b=0;  
 int div=a/b;  
 throw new ArithmeticException("Akash");  
  
 }  
 catch (ArithmeticException ee){  
 System.*out*.println(ee);  
 }  
 }  
}

**Problem-19:**

public class Exception {  
 public void validate(int age) {  
 if(age<18) {  
  
 throw new ArithmeticException("Person is not eligible to vote");  
 }  
 else {  
 System.*out*.println("Person is eligible to vote!!");  
 }  
 }  
 }

public class Main {  
 public static void main(String[] args) {  
 Exception e1=new Exception();  
 e1.validate(23);  
 }  
 }

**Problem-20:**

import java.util.ArrayList;  
import java.util.Collection;  
import java.util.Collections;  
  
public class Sort {  
 public static void main(String[] args) {  
 ArrayList<Integer>list=new ArrayList<>();  
 list.add(5);  
 list.add(4);  
 list.add(3);  
 list.add(2);  
 list.add(1);  
 list.add(0);  
 list.add(6);  
 Collections.*sort*(list);  
 System.*out*.println(list);  
 }  
}