**LAB # 8**

**Question:**

Implement below hierarchal diagram in java using interfaces.

Interface Y

(Abstract mull)

Interface X Extends Y

(Abstract add and sub)

Class A Implements X

(Implements add, sub and mull)

**Source Code:**

package javaapplication14;

interface y{

void mull(int a,int b);

}

interface x extends y{

void add(int a,int b);

void sub(int a,int b);

}

class A implements x{

@Override

public void add(int a, int b) {

int c=a+b;

System.out.println("This is addition method "+c);

}

@Override

public void sub(int a, int b) {

int c=a-b;

System.out.println("This is subtraction method "+c);

}

@Override

public void mull(int a, int b) {

int c=a\*b;

System.out.println("This is multiplication method "+c);

}

}

public class JavaApplication14 {

public static void main(String[] args) {

A ob=new A();

x ob1=new A();

ob.add(2, 5);

ob.sub(10, 8);

ob.mull(25, 5);

System.out.println("\nAdd, Subtract & Multiplication of A on Interface reference\n");

ob1.add(3, 4);

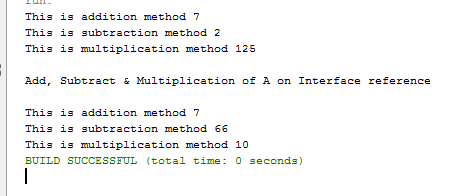
ob1.sub(88, 22);

ob1.mull(2, 5);

}

}

**Output:**



Create an Interface Shape with three child classes Circle, Square and Rectangle. Create a method Drvaw in each class. Create an Object of each class that can call the method Draw and it will call automatically the Draw method of each class. The text of Draw method e.g. Circle is "This is a Circle".

**Source code:**

package lab11\_2;

interface Shape{

void draw();

}

class circle implements Shape{

@Override

public void draw() {

System.out.println("This is circle");

}

}

class square implements Shape{

@Override

public void draw() {

System.out.println("This is square");

}

}

class rectangle implements Shape{

@Override

public void draw() {

System.out.println("This is rectangle");

}

}

public class Lab11\_2 {

public static void main(String[] args) {

circle cir=new circle();

square sqr=new square();

rectangle rec=new rectangle();

cir.draw();

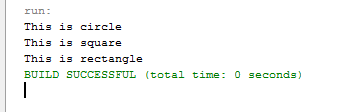
sqr.draw();

rec.draw();

}

}

**Output:**



Define an abstract base class Shape that includes protected data members for the (x, y) position of a shape, a public method to move a shape, and a public abstract method show() to output a shape. Derive subclasses for lines, circles, and rectangles. Also, define the class PolyLine as its base class. You can represent a line as two points, a circle as a center and a radius, and a rectangle as two points on diagonally opposite corners. Implement the toString() method for each class. Test the classes by selecting ten random objects of the derived classes, and then invoking the show() method for each. Use the toString() methods in the derived classes.

**Source Code:**

package lab11\_3;

import java.util.Scanner;

abstract class Shape{

protected double x1,y1;

Scanner scan=new Scanner(System.in);

abstract void move(double x1,double y1);

abstract void show();

}

class circle extends Shape{

double x2,y2;

@Override

void move(double x2, double y2) {

System.out.println("Enter Remaining quadrants of circle : ");

x1=scan.nextDouble();

y1=scan.nextDouble();

}

@Override

void show() {

System.out.println("Quadrants of circle are : ");

x2=1+Math.random()\*4;

y2=1+Math.random()\*4;

System.out.println("x1 , y1 = "+x1 + " , "+y1);

System.out.println("x2 , y2 = "+Math.ceil(x2) + " , "+Math.ceil(y2));

}

}

class lines extends Shape{

double x2,y2;

@Override

void move(double x2, double y2) {

System.out.println("Enter Remaining quadrants of lines : ");

x1=scan.nextDouble();

y1=scan.nextDouble();

}

@Override

void show() {

System.out.println("Quadrants of lines are : ");

x2=1+Math.random()\*4;

y2=1+Math.random()\*4;

System.out.println("x1 , y1 = "+x1 + " , "+y1);

System.out.println("x2 , y2 = "+Math.ceil(x2) + " , "+Math.ceil(y2));

}

}

class rectangle extends Shape{

double x2,y2;

@Override

void move(double x2, double y2) {

System.out.println("Enter Remaining quadrants of rectangle : ");

x1=scan.nextDouble();

y1=scan.nextDouble();

}

@Override

void show() {

System.out.println("Quadrants of rectangle are : ");

x2=1+Math.random()\*4;

y2=1+Math.random()\*4;

System.out.println("x1 , y1 = "+x1 + " , "+y1);

System.out.println("x2 , y2 = "+Math.ceil(x2) + " , "+Math.ceil(y2));

}

}

public class Lab11\_3 {

public static void main(String[] args) {

circle c=new circle();

lines l=new lines();

rectangle r =new rectangle();

c.move(3,4);

c.show();

System.out.println("");

l.move(0,1);

l.show();

System.out.println("");

r.move(0,1);

r.show();

}

}

**Output:**

