1. Define a abstract class figure. Define the area and volume method in the child classes. Use dynamic method dispatch.

Implenent the following design with suitable example classes.

**Program**

import java.io.\*;

interface Area{

public void area();

}

interface Volume{

public void volume();

}

abstract class Display implements Area ,Volume{

public int a,vol,l;

public void display(){

System.out.println("The area is "+a+" the volume "+vol);

}

}

public class Devrun extends Display implements Volume,Area{

public void area(){

a= l\*l;

}

public void volume(){

vol=l\*l\*l;

}

public static void main (String args[]){

Devrun ob = new Devrun();

ob.l=3;

ob.area();

ob.volume();

ob.display();

Display obj = new Devrun(); //Dynamic Method Despatch

obj.a=4;

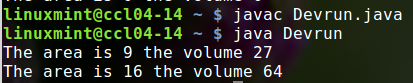
obj.area();

obj.display();

}

}

Output:



Extra Program---

Bank Problem

Program:

interface Bank{

float repo=5.5f;

float rateOfInterest();

float minbalance();

}

class SBI implements Bank{

public float rateOfInterest(){return repo+1.5f;}

public float minbalance(){return 3000;}

}

class PNB implements Bank{

public float rateOfInterest(){return repo+0.65f;}

public float minbalance(){return 500;}

}

class Devrun2{

public static void main(String[] args){

Bank b=new SBI();

System.out.println("ROI: "+b.rateOfInterest());

System.out.println("MinBalance: "+b.minbalance());

b =new PNB();

System.out.println("ROI: "+b.rateOfInterest());

System.out.println("MinBalance: "+b.minbalance());

}

}

Output:

