```
//Que1

class BankAccountProtectionTest{
    public static void main(String args[])
    {
        BankAccountProtection bap1 = new BankAccountProtection(3000,2000,100000);
        bap1.deposit(400);
        System.out.println("deposit -> "+bap1.deposit+"widraw ->
"+bap1.widraw+"details -> "+bap1.balance);
        bap1.widraw(200);
        bap1.setDeposit(2000);
        BankAccountProtection bap2 = new BankAccountProtection(3000,2000,100000);

        System.out.println("deposit -> "+bap1.deposit+"widraw ->
"+bap1.widraw+"details -> "+bap1.balance);
        System.out.println("deposit -> "+bap2.deposit+"widraw ->
"+bap2.widraw+"details -> "+bap2.balance);

    }
}


class BankAccountProtection
{
    int deposit, widraw;
    long balance;

    BankAccountProtection(int deposit, int widraw, long balance)
    {
        this.deposit=deposit;
        this.widraw=widraw;
        this.balance=balance;
    }

    public void deposit(double amount)
    {
        if(this.balance<0)
        {
            balance=this.balance+this.deposit;
        }

    }
    public void widraw(double amount)
    {
        if(balance>0 && balance>=balance-widraw)
        {
            this.balance=balance-widraw;
            System.out.println("current balance - "+this.balance);
        }
        else{
            System.out.println("ensuffiecient balc. ");
```

```
        }

    }

    //getter
    public long getbalance()
    {
        return balance;
    }
    public int getWidraw()
    {
        return widraw;
    }
    public int getDeposit()
    {
        return deposit;
    }


    //setter
    public void setbalance(int balance)
    {
        this.balance=balance;
    }
    public void setWidraw(int widraw)
    {
        this.widraw=widraw;
    }
    public void setDeposit(int deposit)
    {
        this.deposit=deposit;
    }

}


================================================================================
===
//Que 2 :-

class Students
{
    int stdId, marks;

    Students(int stdId, int marks)
    {
        this.stdId=stdId;
        this.marks=marks;
    }

    //setter
    public void setMarks(int marks)
    {
        if(marks>0)
```

```
        {
            this.marks=marks;
        }
        if(marks<0)
        {
            System.out.println("fail..");
        }
    }

    //getter
    public int getMarks()
    {
        return marks;
    }
    public static void main(String args[])
    {
        Students std1= new Students(020,85);
        System.out.println("marks = "+std1.getMarks());
        std1.setMarks(99);
        System.out.println("updated marks = "+std1.getMarks());

    }
}


================================================================================
===
//Que 3:
class BookLab
{
    private int copiesAvailable;

    BookLab(int copiesAvailable)
    {
        this.copiesAvailable=copiesAvailable;
    }

    public void setcopiesAva(int copiesAvailable)
    {
        this.copiesAvailable=copiesAvailable;
    }

    public int getcopies()
    {
        return copiesAvailable;
    }
    //methods
    public void addCopy(int n)
    {
        if(n>0)
        {
            this.copiesAvailable=copiesAvailable+n;
        }
        if(n<0)
```

```
        {
            System.out.println("enter a valide num ..");
        }
    }

    public void removeCopy(int n)
    {
        if(copiesAvailable>=n || copiesAvailable>0)
        {
            copiesAvailable=copiesAvailable-n;
        }
        else
        {
            System.out.println("availabele copies = "+this.copiesAvailable);
        }
    }
    public static void main(String args[])
    {
        BookLab bl1= new BookLab(8);
        System.out.println(bl1.getcopies());
        bl1.addCopy(5);
        System.out.println(bl1.getcopies());
        bl1.removeCopy(3);
        System.out.println(bl1.getcopies());


    }
}

================================================================================
===
//Que 4:
public class Employee
{
    public static void main(String args[])
    {
        Employee1 emp1 = new Employee1("ragini1",1000.0);
        System.out.println(emp1.getSalary());
        emp1.setSalary(2000.0);
        System.out.println(emp1.getSalary());

        RegularEmployee empr1 = new RegularEmployee("ragini", 20000.0);
        System.out.println(empr1.getSalary());
        empr1.setSalary(500.0);
        System.out.println(empr1.getSalary());


        ConstractEmployee empc1= new ConstractEmployee("ashish", 300000.0);
        System.out.println(empc1.getSalary());
        empc1.setSalary(1000.0);
        System.out.println(empc1.getSalary());

    }
}
```

```
class Employee1
{
        String empName;
        double basicSalary;
        Employee1(String empName, double basicSalary)
        {
            this.empName=empName;
            this.basicSalary=basicSalary;
        }


    void setSalary(double basicSalary)
    {
        this.basicSalary=basicSalary;
    }

    double getSalary()
    {
        return basicSalary;
    }
}

//RegularEmployee → HRA 10%
class RegularEmployee extends Employee1
{
    RegularEmployee(String RegEmpName, double salary)
    {
        super(RegEmpName, salary);

    }

        public double getSalary()
        {

        if(basicSalary>5000)
            {
                basicSalary = super.basicSalary + (super.basicSalary * 0.10); //
for RegularEmployee
                return basicSalary;
            }
            else
            {
                System.out.println("cannot deduct..");
                return basicSalary;
            }
        }
}

//ContractEmployee → allowance 5%
class ConstractEmployee extends Employee1
{
    ConstractEmployee(String conEmpname, double basicSalary)
    {
```

```
        super(conEmpname, basicSalary);

    }

    public double getSalary()
    {
        if(basicSalary>5000)
            {
                basicSalary = super.basicSalary + (super.basicSalary * 0.05);
                return basicSalary;
            }
            else
            {
                System.out.println("cannot deduct..");
                return basicSalary;
            }

    }


}
```

```
================================================================================
===
//Que 5:
public class TestAnimal
{

    public static void main(String args[]){

            Animal an = new Cat("meow");
            an.makeSound("bark");

            Animal an1 = new Dog("bark");
            an1.makeSound("bark");

    }
}

class Animal
{
    String sound;
    Animal(String sound)
    {
        this.sound=sound;
    }
    public void makeSound(String sound) {
        System.out.println("Animal sound: " + sound);
    }
```

```
}

class Dog extends Animal
{

    Dog(String sound)
    {
        super(sound);

    }

     @Override
    public void makeSound(String sound)
    {
        if(this.sound.equals(sound)){
            System.out.println("Dog -> "+sound);
        }else {
            System.out.println("Dog doesn’t recognize this sound.");
        }

    }

}

class Cat extends Animal
{
    Cat(String sound)
    {
        super(sound);
    }
    @Override
    public void makeSound(String sound)
    {
        if(this.sound.equals(sound)){
            System.out.println("cat -> "+sound);
        }else {
            System.out.println("Cat doesn’t recognize this sound.");
        }

    }
}

================================================================================
===
//Que 6:
class ShapeAreaOverload{

    public void calculateArea(int side)
    {
        int areaofsq = side * side;
        System.out.println("area of squ - "+areaofsq);
    }
    public void calculateArea(int l, int b)
```

```
    {
        int areaofrect=l*b;
        System.out.println("area of rectagle - "+areaofrect);
    }
    public void calculateArea(double r)
    {
        double areaCircle=2.14*r*r;
        System.out.println("area of circle - "+areaCircle);
    }

    public static void main(String args[])
    {
        ShapeAreaOverload sao = new ShapeAreaOverload();
        sao.calculateArea(4,3);
        sao.calculateArea(6);
        sao.calculateArea(4.4);
    }
}

================================================================================
===
//Que 8:
public class TestVehicle
{
    public static void main(String args[])
    {
        Vehicle vh = new Car("Car",120,"farari");
        //System.out.println( vh.getcarModelType()+" speed "+vh.getSpeed()
+"km/h");
        vh.displaySpeed();
        Vehicle vh1 = new Bike("bike",120,"farari");
        //System.out.println( vh1.getBikemodelType()+" speed "+vh1.getSpeed()
+"km/h");
        vh1.displaySpeed();
    }
}


class Vehicle{
    String brand;
    double speed;
    Vehicle(String brand, double speed)
    {
        this.brand=brand;
        this.speed=speed;

    }

    //setter
    public void setBrand(String brand)
    {
        this.brand=brand;
    }
    public void setSpeed(double speed)
```

```
    {
        this.speed=speed;
    }

    //getter
    public String getbrand()
    {
        return brand;
    }
    public double getSpeed(){
        return speed;
    }

    public void displaySpeed(){

    }

}

class Car extends Vehicle
{
    String carModelType;
    Car(String brand, double speed, String carModelType)
    {
        super(brand,speed);
        this.carModelType=carModelType;
    }

    //setter
    public void setcarModelType(String carModelType)
    {
        this.carModelType=carModelType;
    }

    //getter
    public String getcarModelType()
    {
        return carModelType;
    }

    @Override
    public void displaySpeed(){
        System.out.println("Car speed "+ this.speed +"km/h");


    }

}

class Bike extends Vehicle
{
    String bikeModelType;
    Bike(String brand, double speed, String bikeModelType)
    {
```

```
        super(brand, speed);
        this.bikeModelType=bikeModelType;

    }

    public void setBikeModelType(String bikeModelType)
    {
        this.bikeModelType=bikeModelType;
    }

    public String getBikemodelType(){
        return bikeModelType;
    }

    @Override
    public void displaySpeed(){
        System.out.println("Bike speed "+ this.speed +"km/h");


    }

}




================================================================================
===
//Que 8:
import java.util.*;
public class VehicleTest
{
    public static void main(String args[])
    {
        Vehicle vc = new Vehicle("honda",180);
        System.out.println("Car -> brand :- "+vc.getbrand()+", speed -
>"+vc.getSpeed());

        Car cr = new Car("honda",200, "civie");
        System.out.println("Car -> brand :- "+cr.getbrand()+", speed -
>"+cr.getSpeed()+", model type - "+cr.getcarModelType());

        Bike bk = new Bike("yamaha", 300, "R15");
        System.out.println("Car -> brand :- "+bk.getbrand()+", speed -
>"+bk.getSpeed()+", model type - "+bk.getBikemodelType());

    }
}

class Vehicle{
    String brand;
    double speed;
    Vehicle(String brand, double speed)
```

```
    {
        this.brand=brand;
        this.speed=speed;

    }

    //setter
    public void setBrand(String brand)
    {
        this.brand=brand;
    }
    public void setSpeed(double speed)
    {
        this.speed=speed;
    }

    //getter
    public String getbrand()
    {
        return brand;
    }
    public double getSpeed(){
        return speed;
    }

}

class Car extends Vehicle
{
    String carModelType;
    Car(String brand, double speed, String carModelType)
    {
        super(brand,speed);
        this.carModelType=carModelType;
    }

    //setter
    public void setcarModelType(String carModelType)
    {
        this.carModelType=carModelType;
    }

    //getter
    public String getcarModelType()
    {
        return carModelType;
    }
}

class Bike extends Vehicle
{
    String bikeModelType;
    Bike(String brand, double speed, String bikeModelType)
    {
```

```
        super(brand, speed);
        this.bikeModelType=bikeModelType;

    }

    public void setBikeModelType(String bikeModelType)
    {
        this.bikeModelType=bikeModelType;
    }

    public String getBikemodelType(){
        return bikeModelType;
    }
}



================================================================================
===
//Que 7:
public class Employee
{
    public static void main(String args[])
    {
        Employee1 emp1 = new Employee1("ragini1",1000.0);
        System.out.println(emp1.getSalary());
        emp1.setSalary(2000.0);
        System.out.println(emp1.getSalary());

        RegularEmployee empr1 = new RegularEmployee("ragini", 20000.0);
        System.out.println(empr1.getSalary());
        empr1.setSalary(500.0);
        System.out.println(empr1.getSalary());


        ConstractEmployee empc1= new ConstractEmployee("ashish", 300000.0);
        System.out.println(empc1.getSalary());
        empc1.setSalary(1000.0);
        System.out.println(empc1.getSalary());

    }
}

class Employee1
{
        String empName;
        double basicSalary;
        Employee1(String empName, double basicSalary)
        {
            this.empName=empName;
            this.basicSalary=basicSalary;
        }
```

```
    void setSalary(double basicSalary)
    {
        this.basicSalary=basicSalary;
    }

    double getSalary()
    {
        return basicSalary;
    }
}

//RegularEmployee → HRA 10%
class RegularEmployee extends Employee1
{
    RegularEmployee(String RegEmpName, double salary)
    {
        super(RegEmpName, salary);

    }

        public double getSalary()
        {

        if(basicSalary>5000)
            {
                basicSalary = super.basicSalary + (super.basicSalary * 0.10); //
for RegularEmployee
                return basicSalary;
            }
            else
            {
                System.out.println("cannot deduct..");
                return basicSalary;
            }
        }
}

//ContractEmployee → allowance 5%
class ConstractEmployee extends Employee1
{
    ConstractEmployee(String conEmpname, double basicSalary)
    {
        super(conEmpname, basicSalary);

    }

    public double getSalary()
    {
        if(basicSalary>5000)
            {
                basicSalary = super.basicSalary + (super.basicSalary * 0.05);
                return basicSalary;
            }
            else
```

```
            {
                System.out.println("cannot deduct..");
                return basicSalary;
            }

    }


}
```

```
================================================================================
===
//Que 10:
class TestAcademicStaff
{
    public static void main(String args[])
    {
        //Staff sf = new Staff("ragini", 444.4);
        Staff tf = new TeachingStaff("ragini","Maths",444.44);

        tf.stafInfo();

        Staff ntf = new NonTeachingStaff("sindhu","sport",66.66);
        ntf.stafInfo();
    }
}

class Staff
{
    String name;
    double salary;
    Staff(String name, double salary){
        this.name=name;
        this.salary=salary;
    }

    public void stafInfo()
    {
        System.out.println(this.name+" -> "+", salary="+this.salary);
    }
}

class TeachingStaff extends Staff{
    String subject;
    TeachingStaff(String subject,String name, double salary){
        super(name,salary);
    }

    @Override
```

```
    public void stafInfo()
    {
        System.out.println(this.name+" -> "+", salary="+this.salary+", salary -
"+this.salary);
    }
}

class NonTeachingStaff extends Staff{
    String department;
    NonTeachingStaff(String department,String name, double salary)
    {
        super(name,salary);
        this.department=department;
    }
    public void stafInfo()
    {
        System.out.println(this.name+" -> "+", salary="+this.salary+", dipartment
- "+this.department);
    }
}




================================================================================
===
//Que 9:
public class TestAnimal
{

    public static void main(String args[]){

            Animal an = new Cat("meow");
            an.makeSound("bark");

            Animal an1 = new Dog("bark");
            an1.makeSound("bark");

    }
}

class Animal
{
    String sound;
    Animal(String sound)
    {
        this.sound=sound;
    }
    public void makeSound(String sound) {
        System.out.println("Animal sound: " + sound);
    }

}
```

```
class Dog extends Animal
{

    Dog(String sound)
    {
        super(sound);

    }

     @Override
    public void makeSound(String sound)
    {
        if(this.sound.equals(sound)){
            System.out.println("Dog -> "+sound);
        }else {
            System.out.println("Dog doesn’t recognize this sound.");
        }

    }

}

class Cat extends Animal
{
    Cat(String sound)
    {
        super(sound);
    }
    @Override
    public void makeSound(String sound)
    {
        if(this.sound.equals(sound)){
            System.out.println("cat -> "+sound);
        }else {
            System.out.println("Cat doesn’t recognize this sound.");
        }

    }
}


================================================================================
===
//Que 11:

class TestBank
{
    public static void main(String args[])
    {
        Account ac = new Account(505,444.55);
        ac.accountDetails();

        Account ac1= new SavingAccount(404,555.55,5.5);
        ac1.accountDetails();
```

```
        Account ac2= new CurrentAccount(303,665.55,5.5);
        ac2.accountDetails();

    }
}

class Account
{
    int accountNo;
    double balance;
    Account(int accountNo, double balance)
    {
        this.accountNo=accountNo;
        this.balance=balance;
    }

    public void accountDetails()
    {
        System.out.println("Saving → "+accountNo+", Balance="+5000+",
Interest="+5+"%");
    }
}

class SavingAccount extends Account
{
    double intrestRate=5;
    SavingAccount(int accountNo, double balance,double intrestRate)
    {
        super(accountNo,balance);

    }

    @Override
    public void accountDetails()
    {
        System.out.println("Saving → "+accountNo+", Balance="+balance+",
InterestRate="+intrestRate+"%");

    }
}

class CurrentAccount extends Account
{
    double overdraftLimit=2000.0;
    CurrentAccount(int accountNo, double balance, double overdraftLimit){
        super(accountNo,balance);

    }

    @Override
    public void accountDetails()
    {
        System.out.println("Saving → "+accountNo+", Balance="+balance+",
```

```
overdraftLimit="+overdraftLimit+"%");

    }
}


================================================================================
===
//Que 12:
class TestCompany
{
    public static void main(String args[])
    {
        CreditCardPayment ccp1 = new CreditCardPayment(6699,66.66);
        ccp1.pay();
        UPIPayment upt = new UPIPayment("ragini342434",555.55);
        upt.pay();
        CreditCardPayment ccp = new CreditCardPayment(987654399,666.55);
        ccp.pay();
    }
}

abstract class Pay{
    abstract void pay();

}

class CreditCardPayment extends Pay{

    long cardNo;
    double amount;
    CreditCardPayment(long cardNo, double amount){
        this.cardNo=cardNo;
        this.amount=amount;
    }
    public void pay()
    {
        System.out.println("Payment via Credit Card "+cardNo+"→ Rs. "+amount+"
Paid");
    }
}

class UPIPayment extends Pay{
    String UPI_Id;
    double amount;

    UPIPayment(String UPI_Id, double amount)
    {
        this.UPI_Id=UPI_Id;
        this.amount=amount;

    }

    public void pay()
```

```
    {
        System.out.println("Payment via UPI "+UPI_Id+" → Rs. "+amount+"Paid");
    }
}


================================================================================
===
//Que 13:

class TestShape
{
    public static void main(String args[])
    {

        Circle sh1 = new Circle(4);
        sh1.draw();

        Rectangle sh2 = new Rectangle(5,6);
        sh2.draw();
    }
}

abstract class Shape
{
    public abstract void draw();

}

class Circle extends Shape{

    int r;
    Circle(int r)
    {
        this.r=r;
    }

    @Override
    public void draw(){

        System.out.println("Drawing Circle of radius "+r);
    }

    public void setRadius(int r)
    {
        this.r=r;
    }
}

class Rectangle extends Shape{
    int l,b;
    Rectangle(int l, int b)
    {
        this.l=l;
```

```
        this.b=b;
    }

    @Override
    public void draw(){
        System.out.println("Drawing Rectangle of length "+l+" and breadth "+b);

    }

    public void setRadius(int l, int b)
    {
        this.l=l;
        this.b=b;
    }
}

================================================================================
===
//Que 14:
public class TestBonusCal{
    public static void main(String args[])
    {
        Manager mng1 = new Manager("ragini",50000);
        System.out.println("name - "+mng1.getName()+"bonus -
"+mng1.calculateBonus());
        Developer dv1 = new Developer("ashish",40000);
        System.out.println("name - "+dv1.getName()+"bonus -
"+dv1.calculateBonus());

    }

}

abstract class Employee{
    String name;
    double salary;

    abstract double calculateBonus();

    //getter
    public String getName()
    {
        return name;
    }

    public double getSalary()
    {
        return salary;
    }

}

class Manager extends Employee{
```

```
    Manager(String name, double salary)
    {
        super.name=name;
        super.salary=salary;

    }

    @Override
    public double calculateBonus()
    {
        if(super.salary>1000)
            return salary*0.2;
        else
            return salary;
    }



}

class Developer extends Employee{

    Developer(String name, double salary)
    {
        super.name=name;
        super.salary=salary;
    }


    @Override

    public double calculateBonus()
    {
        if(super.salary>1000)
            return salary*0.1;
        else
            return salary;
    }

}


================================================================================
===
//Que 15:
class ShapeAreaOverload{

    public void calculateArea(int side)
    {
        int areaofsq = side * side;
        System.out.println("area of squ - "+areaofsq);
    }
    public void calculateArea(int l, int b)
    {
```

```
        int areaofrect=l*b;
        System.out.println("area of rectagle - "+areaofrect);
    }
    public void calculateArea(double r)
    {
        double areaCircle=2.14*r*r;
        System.out.println("area of circle - "+areaCircle);
    }

    public static void main(String args[])
    {
        ShapeAreaOverload sao = new ShapeAreaOverload();
        sao.calculateArea(4,3);
        sao.calculateArea(6);
        sao.calculateArea(4.4);
    }
}



================================================================================
===
//Que 16:
class TestEmployeeSalary{

    public static void main(String args[])
    {
    Manager mn1 = new Manager("ragini",40000);
    System.out.println(mn1.getName()+" total salary - "+mn1.getSalary());
    Developer dv1 = new Developer("ashish",60000);
    System.out.println(dv1.getName()+" total salary - "+dv1.getSalary());
    }
}


abstract class Employee{
    String name;
    double salary;

    abstract double displaySalary();


    //getter
    public String getName()
    {
        return name;
    }

    public double getSalary()
    {
        return salary;
    }

}
```

```
class Manager extends Employee{

    Manager(String name, double salary)
    {
        super.name=name;
        super.salary=salary;

    }

    @Override
    public double displaySalary()
    {
        if(super.salary>1000)
        {
            salary=salary*0.2;
            return salary;
        }
        else
        {
            return salary;
        }
    }

}

class Developer extends Employee{

    Developer(String name, double salary)
    {
        super.name=name;
        super.salary=salary;
    }


    @Override
    public double displaySalary()
    {
        if(super.salary>1000)
        {
            salary=salary*0.1;
            return salary;
        }
        else
        {
            return salary;
        }
    }

}

================================================================================
===
//Que 17:
```

```
public class TestVehicle
{
    public static void main(String args[])
    {
        Vehicle vh = new Car("Car",120,"farari");
        //System.out.println( vh.getcarModelType()+" speed "+vh.getSpeed()
+"km/h");
        vh.displaySpeed();
        Vehicle vh1 = new Bike("bike",120,"farari");
        //System.out.println( vh1.getBikemodelType()+" speed "+vh1.getSpeed()
+"km/h");
        vh1.displaySpeed();
    }
}


class Vehicle{
    String brand;
    double speed;
    Vehicle(String brand, double speed)
    {
        this.brand=brand;
        this.speed=speed;

    }

    //setter
    public void setBrand(String brand)
    {
        this.brand=brand;
    }
    public void setSpeed(double speed)
    {
        this.speed=speed;
    }

    //getter
    public String getbrand()
    {
        return brand;
    }
    public double getSpeed(){
        return speed;
    }

    public void displaySpeed(){

    }

}

class Car extends Vehicle
{
    String carModelType;
```

```
    Car(String brand, double speed, String carModelType)
    {
        super(brand,speed);
        this.carModelType=carModelType;
    }

    //setter
    public void setcarModelType(String carModelType)
    {
        this.carModelType=carModelType;
    }

    //getter
    public String getcarModelType()
    {
        return carModelType;
    }

    @Override
    public void displaySpeed(){
        System.out.println("Car speed "+ this.speed +"km/h");


    }

}

class Bike extends Vehicle
{
    String bikeModelType;
    Bike(String brand, double speed, String bikeModelType)
    {
        super(brand, speed);
        this.bikeModelType=bikeModelType;

    }

    public void setBikeModelType(String bikeModelType)
    {
        this.bikeModelType=bikeModelType;
    }

    public String getBikemodelType(){
        return bikeModelType;
    }

    @Override
    public void displaySpeed(){
        System.out.println("Bike speed "+ this.speed +"km/h");


    }

}
```

```
================================================================================
===
//Que 19:
public class BankAcTest {
    public static void main(String args[]) {
        SavingAC bnk1 = new SavingAC(101, "ragini", 5000.0, 5.0);
        System.out.println("Interest: " + bnk1.calculateInterest());
        bnk1.displayBalance();

        CurrentAc bnk2 = new CurrentAc(102, "ragini99", 5000.0, 500.0);
        bnk2.checkOverdraft();
        bnk2.displayBalance();
    }
}

class Bank {
    long acountNo;
    String acHolder;
    double balance;

    Bank(long acountNo, String acHolder, double balance) {
        this.acountNo = acountNo;
        this.acHolder = acHolder;
        this.balance = balance;
    }

    public void displayBalance() {
        System.out.println("Bank → accountNumber=" + acountNo +
            ", accountHolder=" + acHolder +
            ", balance=" + balance);
    }
}

class SavingAC extends Bank {
    double interestRate;

    SavingAC(long acountNo, String acHolder, double balance, double interestRate)
{
        super(acountNo, acHolder, balance);
        this.interestRate = interestRate;
    }

    public double calculateInterest() {
        return super.balance * interestRate / 100;
    }

    public void setInterestRate(double interestRate) {
        if (super.balance > 1000) {
            this.interestRate = interestRate;
        }
```

```
    }

    @Override
    public void displayBalance() {
        System.out.println("SavingsAccount → accountNumber=" + super.acountNo +
            ", accountHolder=" + super.acHolder +
            ", balance=" + super.balance +
            ", interestRate=" + interestRate + "%");
    }
}

class CurrentAc extends Bank {
    double overdraftLimit;

    CurrentAc(long acountNo, String acHolder, double balance, double
overdraftLimit) {
        super(acountNo, acHolder, balance);
        this.overdraftLimit = overdraftLimit;
    }

    public void checkOverdraft() {
        overdraftLimit = super.balance * 0.05;
        System.out.println("OverdraftLimit = " + overdraftLimit);
    }

    @Override
    public void displayBalance() {
        System.out.println("CurrentAccount → accountNumber=" + super.acountNo +
            ", accountHolder=" + super.acHolder +
            ", balance=" + super.balance +
            ", overdraftLimit=" + overdraftLimit);
    }
}


================================================================================
===
//Que 19:



================================================================================
===
//Que 20:
public class CollegeStaffTest {
    public static void main(String[] args) {
        Professor prof = new Professor("Dr. Sharma", 80000, "Math", "Algebra");
        Lecturer lec = new Lecturer("Ms. Mehta", 50000, "Physics", "Science");

        prof.displaySalary();
        lec.displaySalary();
    }
}
```

```java
// Base class
class Employee {
    String name;
    double salary;

    Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }

    public void displaySalary() {
        System.out.println(name + " → Salary=" + salary);
    }
}

// Teaching staff (inherits Employee)
class TeachingStaff extends Employee {
    String subject;

    TeachingStaff(String name, double salary, String subject) {
        super(name, salary);
        this.subject = subject;
    }

    @Override
    public void displaySalary() {
        System.out.println(name + " → Subject=" + subject + ", Salary=" + salary);
    }
}

// Professor (inherits TeachingStaff)
class Professor extends TeachingStaff {
    String specialization;

    Professor(String name, double salary, String subject, String specialization) {
        super(name, salary, subject);
        this.specialization = specialization;
    }

    @Override
    public void displaySalary() {
        System.out.println(name + " → Subject=" + subject +
                ", Specialization=" + specialization +
                ", Salary=" + salary);
    }
}

// Lecturer (inherits TeachingStaff)
class Lecturer extends TeachingStaff {
    String department;

    Lecturer(String name, double salary, String subject, String department) {
        super(name, salary, subject);
        this.department = department;
```

```
    }

    @Override
    public void displaySalary() {
        System.out.println(name + " → Subject=" + subject +
                ", Department=" + department +
                ", Salary=" + salary);
    }
}
```

```
================================================================================
===
//Que 21:
public class TestHospital{
    public static void main(String args[]){


    }
}

abstract class Staff
{
    String name;
    int staffId;

    Staff(String name, int staffId)
    {
        this.name=name;
        this.staffId=staffId;
    }
    public void displayDetails(String name, int staffId);
}

class Doctor extends Staff{

    @Override
    public void displayDetails(){

    }
}

class Nurses extends Staff{


    @Override
    public void displayDetails(){

    }
}
```

```
================================================================================
===
//Que 22:
public class TestVehiclLandWater
{
    public static void main(String args[])
    {
        AmphibiousVehicle av1= new AmphibiousVehicle();
        av1.driveOnLand("hydroWater");

        AmphibiousVehicle av2 = new AmphibiousVehicle();
        av2.driveOnWater("sss");
    }

}

interface ILandVehicle{
    public void driveOnLand(String name);
}

interface IWaterVehicle{

    public void driveOnWater(String name);

}

class AmphibiousVehicle implements ILandVehicle,IWaterVehicle{

    String name;

    @Override
    public void driveOnLand(String name){
            this.name=name;
        System.out.println(this.name+" driving on lnad");
    }

    @Override
    public void driveOnWater(String name){
        this.name=name;
        System.out.println(this.name+" driving on water");
    }


}



================================================================================
===
//Que 23:
```

```
public class Que23Test{
    public static void main(String args[]){
        Memeber m1 = new Students("abd",43,'A');
        m1.displayInfo();

        Memeber m2 = new Teachers("bbb",43,"abcd");
        m2.displayInfo();

        Memeber m3 = new Staff("ccc",43,"ddd");
        m3.displayInfo();


    }

}

class Memeber{
    String name;
    int Id;
    Memeber(String name, int Id)
    {
        this.name=name;
        this.Id=Id;
    }
    public void displayInfo(){

    }

}

class Students extends Memeber{
    char grade;
    Students(String name,int Id,char grade){
        super(name,Id);
        this.grade=grade;
    }

    @Override
    public void displayInfo(){
            System.out.println("student id - "+super.Id+", Student name -
"+super.name+", student grade - "+this.grade);
    }

}

class Teachers extends Memeber{
    String subject;
    Teachers(String name,int Id,String subject){
        super(name,Id);
        this.subject=subject;
    }

    @Override
```

```
    public void displayInfo(){
            System.out.println("teacher id - "+super.Id+", teacher name -
"+super.name+", teacher grade - "+this.subject);


    }

}

class Staff extends Memeber{
    String department;
    Staff(String name,int Id,String department){
        super(name,Id);
        this.department=department;
    }
    @Override
    public void displayInfo(){
            System.out.println("staff id - "+super.Id+", staff name -
"+super.name+", staff grade - "+this.department);

    }


}



================================================================================
===
//Que 24:
/*
Scenario: An e-commerce platform supports multiple payment methods like CreditCard
and PayPal. All
payments must implement a pay() method.
Problem Statement:
● Create an interface Payment → method pay(double amount)
● Classes CreditCardPayment and PayPalPayment implement Payment → provide their
own pay()
implementation
● In main(), take payment amount and process payment using both methods
Sample Input:
CreditCardPayment → amount=2500
PayPalPayment → amount=1500
Sample Output:
Processing Credit Card Payment of 2500
Processing PayPal Payment of 1500

*/

public class TestPayment{

    public static void main(String args[]){
            IPayment ip = new PayPalPayment(777.7);
            ip.Pay(555.00);
```

```
    }
}

interface IPayment
{

    public void Pay(double amount);
}

class PayPalPayment implements IPayment
{
    double amount;
    PayPalPayment(double amount)
    {
        this.amount=amount;
    }
    @Override
    public void Pay(double amount){
        if(this.amount>0)
            System.out.println("current credit card payment of - "+amount);
        else
            System.out.println("invalide ammount");
    }
}

class CreditCardPayment implements IPayment{

    double amount;
        CreditCardPayment(double amount){
            this.amount=amount;
        }

    @Override
    public void Pay(double amount){
        if(this.amount>0)
            System.out.println("payment done");
        else
            System.out.println("invalide ammount");

    }
}




================================================================================
===
//Que 25:
public class TestQue25{
    public static void main(String args[]){
        IAudioPlayer ia = new MediaPlayer();
        ia.playAudio("shaper of you ");
        IVideoPlayer iv = new MediaPlayer();
        iv.playVideo("inception");
```

```
    }
}

interface IAudioPlayer{
    public void playAudio(String audio);

}

interface IVideoPlayer{
    public void playVideo(String video);

}

class MediaPlayer implements IAudioPlayer,IVideoPlayer{

    public void playAudio(String audio){
            System.out.println("play audio - >"+audio);
    }

    public void playVideo(String video){
    System.out.println("play video - >"+video);
    }


}




================================================================================
===
//Que 26:
class TestQue26{
    public static void main(String args[]){
        ICallable ic = new SmartPhone();
        ic.makeCall("98987493");

        IMessaging im = new SmartPhone();
        im.sendMessage(48989833,"hello world");

        Iinternet in = new SmartPhone();
        in.browser("www.dreamland.com");
    }
}

interface ICallable {
    public void makeCall(String call);
}

interface IMessaging{
    public void sendMessage(long no, String msg);

}
```

```
interface Iinternet{
    public void browser(String website);
}

class SmartPhone implements ICallable,IMessaging,Iinternet{

        public void makeCall(String call){
            System.out.println("calling - "+call);
        }
        public void sendMessage(long no, String msg){
        System.out.println("sending message - "+no+", message - "+msg);
        }
        public void browser(String website){

            System.out.println("Browser website - "+website);
        }


}

================================================================================
===
//Que 27:
public class TestQue27 {
    public static void main(String args[]) {
        Shape sh1 = new Rectangle(4, 6);
        sh1.calculateArea();

        Shape sh2 = new Circle(5.5);
        sh2.calculateArea();

        Shape sh3 = new Square(3.3);
        sh3.calculateArea();
    }
}

interface Shape {
    public void calculateArea();
}

class Circle implements Shape {
    double r;
    Circle(double r) {
        this.r = r;
    }
    public void calculateArea() {
        double areaofcircle = 3.14 * r * r;
        System.out.println("Circle area = " + areaofcircle);
    }
}

class Rectangle implements Shape {
    int l, b;
    Rectangle(int l, int b) {
```

```
        this.l = l;
        this.b = b;
    }
    public void calculateArea() {
        int areaofRect = l * b;
        System.out.println("Rectangle area = " + areaofRect);
    }
}

class Square implements Shape {
    double s;
    Square(double s) {
        this.s = s;
    }
    public void calculateArea() {
        double areaOfSquare = s * s;
        System.out.println("Square area = " + areaOfSquare);
    }
}



================================================================================
===
//Que 28:
public class TestQue28 {

    // Simple Product class
    static class Product {
        int id;
        String name;
        double price;

        // Constructor
        Product(int id, String name, double price) {
            this.id = id;
            this.name = name;
            this.price = price;
        }
    }

    // Simple Cart class using array
    static class Cart {
        Product[] products;
        int count;  // how many products are added

        Cart(int size) {
            products = new Product[size];
            count = 0;
        }

        // Add product
        void addProduct(Product p) {
            products[count] = p;
```

```
            count++;
        }

        // Apply discount only to Laptop
        void applyDiscount(double percentage) {
            for (int i = 0; i < count; i++) {
                if (products[i].name.equalsIgnoreCase("Laptop")) {
                    double discount = products[i].price * percentage / 100;
                    products[i].price = products[i].price - discount;
                    System.out.println("Applied " + percentage + "% discount on
Laptop");
                }
            }
        }

        // Calculate total price
        double getTotalPrice() {
            double total = 0;
            for (int i = 0; i < count; i++) {
                total += products[i].price;
            }
            return total;
        }
    }

    // Main method
    public static void main(String[] args) {
        // Create cart with 3 slots
        Cart cart = new Cart(3);

        // Create products (no input, just fixed values)
        Product p1 = new Product(1, "Laptop", 50000);
        Product p2 = new Product(2, "Mouse", 500);
        Product p3 = new Product(3, "Keyboard", 1200);

        // Add products to cart
        cart.addProduct(p1);
        cart.addProduct(p2);
        cart.addProduct(p3);

        // Apply discount on Laptop
        cart.applyDiscount(10);

        // Print total price
        System.out.println("Total Cart Price = " + cart.getTotalPrice());
    }
}
```

```
================================================================================
===
```

```java
//Que 29:
public class TestQue29 {

    // Abstract Employee class
    static abstract class Employee {
        String name;

        Employee(String name) {
            this.name = name;
        }

        // Abstract method (must be implemented by subclasses)
        abstract double calculateSalary();
    }

    // Interface for bonus
    interface BonusEligible {
        double calculateBonus();
    }

    // Permanent Employee
    static class PermanentEmployee extends Employee implements BonusEligible {
        double basicSalary;
        double hra;

        PermanentEmployee(String name, double basicSalary, double hra) {
            super(name);
            this.basicSalary = basicSalary;
            this.hra = hra;
        }

        @Override
        double calculateSalary() {
            return basicSalary + hra;
        }

        @Override
        public double calculateBonus() {
            return calculateSalary() * 0.10; // 10% bonus
        }
    }

    // Contract Employee
    static class ContractEmployee extends Employee {
        double hourlyRate;
        int hoursWorked;

        ContractEmployee(String name, double hourlyRate, int hoursWorked) {
            super(name);
            this.hourlyRate = hourlyRate;
            this.hoursWorked = hoursWorked;
        }

        @Override
```

```
        double calculateSalary() {
            return hourlyRate * hoursWorked;
        }
    }

    // Main method
    public static void main(String[] args) {
        // Create employees
        PermanentEmployee emp1 = new PermanentEmployee("Amit", 50000, 5000);
        PermanentEmployee emp2 = new PermanentEmployee("Ravi", 40000, 4000);

        ContractEmployee emp3 = new ContractEmployee("Neha", 300, 100);
        ContractEmployee emp4 = new ContractEmployee("Sonal", 250, 120);

        // Print details directly (no array, no loop)
        System.out.println(emp1.name + " Salary = " + emp1.calculateSalary() + ",
Bonus = " + emp1.calculateBonus());
        System.out.println(emp2.name + " Salary = " + emp2.calculateSalary() + ",
Bonus = " + emp2.calculateBonus());
        System.out.println(emp3.name + " Salary = " + emp3.calculateSalary());
        System.out.println(emp4.name + " Salary = " + emp4.calculateSalary());
    }
}
```

```
================================================================================
===
//Que 30:

import java.util.ArrayList;
import java.util.List;

public class TestQue30 {

    // Book class with encapsulated fields
    static class Book {
        private int bookId;
        private String title;
        private String author;

        public Book(int bookId, String title, String author) {
            this.bookId = bookId;
            this.title = title;
            this.author = author;
        }

        // Getters
        public int getBookId() {
            return bookId;
        }
```

```
        public String getTitle() {
            return title;
        }

        public String getAuthor() {
            return author;
        }
    }

    // Abstract LibraryMember class
    static abstract class LibraryMember {
        private int memberId;
        private String name;
        private static int idCounter = 1;

        protected List<Book> borrowedBooks;

        public LibraryMember(String name) {
            this.name = name;
            this.memberId = idCounter++;
            this.borrowedBooks = new ArrayList<>();
        }

        public String getName() {
            return name;
        }

        public int getMemberId() {
            return memberId;
        }

        // Abstract method to borrow book
        public abstract boolean borrowBook(Book book);

        public int getBorrowedBookCount() {
            return borrowedBooks.size();
        }
    }

    // Interface Notifyable
    interface Notifyable {
        void sendNotification(String message);
    }

    // StudentMember class
    static class StudentMember extends LibraryMember implements Notifyable {
        private static final int MAX_BOOKS = 3;

        public StudentMember(String name) {
            super(name);
        }

        @Override
        public boolean borrowBook(Book book) {
```

```
            if (borrowedBooks.size() < MAX_BOOKS) {
                borrowedBooks.add(book);
                return true;
            } else {
                System.out.println("StudentMember " + getName() + " cannot borrow
more than " + MAX_BOOKS + " books.");
                return false;
            }
        }

        @Override
        public void sendNotification(String message) {
            System.out.println("Notification sent to " + getName() + ": " +
message);
        }
    }

    // FacultyMember class
    static class FacultyMember extends LibraryMember implements Notifyable {
        private static final int MAX_BOOKS = 5;

        public FacultyMember(String name) {
            super(name);
        }

        @Override
        public boolean borrowBook(Book book) {
            if (borrowedBooks.size() < MAX_BOOKS) {
                borrowedBooks.add(book);
                return true;
            } else {
                System.out.println("FacultyMember " + getName() + " cannot borrow
more than " + MAX_BOOKS + " books.");
                return false;
            }
        }

        @Override
        public void sendNotification(String message) {
            System.out.println("Notification sent to " + getName() + ": " +
message);
        }
    }

    // Main method
    public static void main(String[] args) {
        // Create books
        Book book1 = new Book(1, "Java Programming", "Author A");
        Book book2 = new Book(2, "Data Structures", "Author B");
        Book book3 = new Book(3, "Operating Systems", "Author C");
        Book book4 = new Book(4, "Database Systems", "Author D");
        Book book5 = new Book(5, "Computer Networks", "Author E");

        // Create members
```

```
        StudentMember student = new StudentMember("Amit");
        FacultyMember faculty = new FacultyMember("Prof. Singh");

        // Student borrows 2 books
        student.borrowBook(book1);
        student.borrowBook(book2);
        System.out.println("StudentMember " + student.getName() + " borrowed " +
student.getBorrowedBookCount() + " books");

        // Faculty borrows 4 books
        faculty.borrowBook(book1);
        faculty.borrowBook(book2);
        faculty.borrowBook(book3);
        faculty.borrowBook(book4);
        System.out.println("FacultyMember " + faculty.getName() + " borrowed " +
faculty.getBorrowedBookCount() + " books");

        // Send notifications
        student.sendNotification("Return books within 7 days");
        faculty.sendNotification("Return books within 7 days");
    }
}

================================================================================
===
```