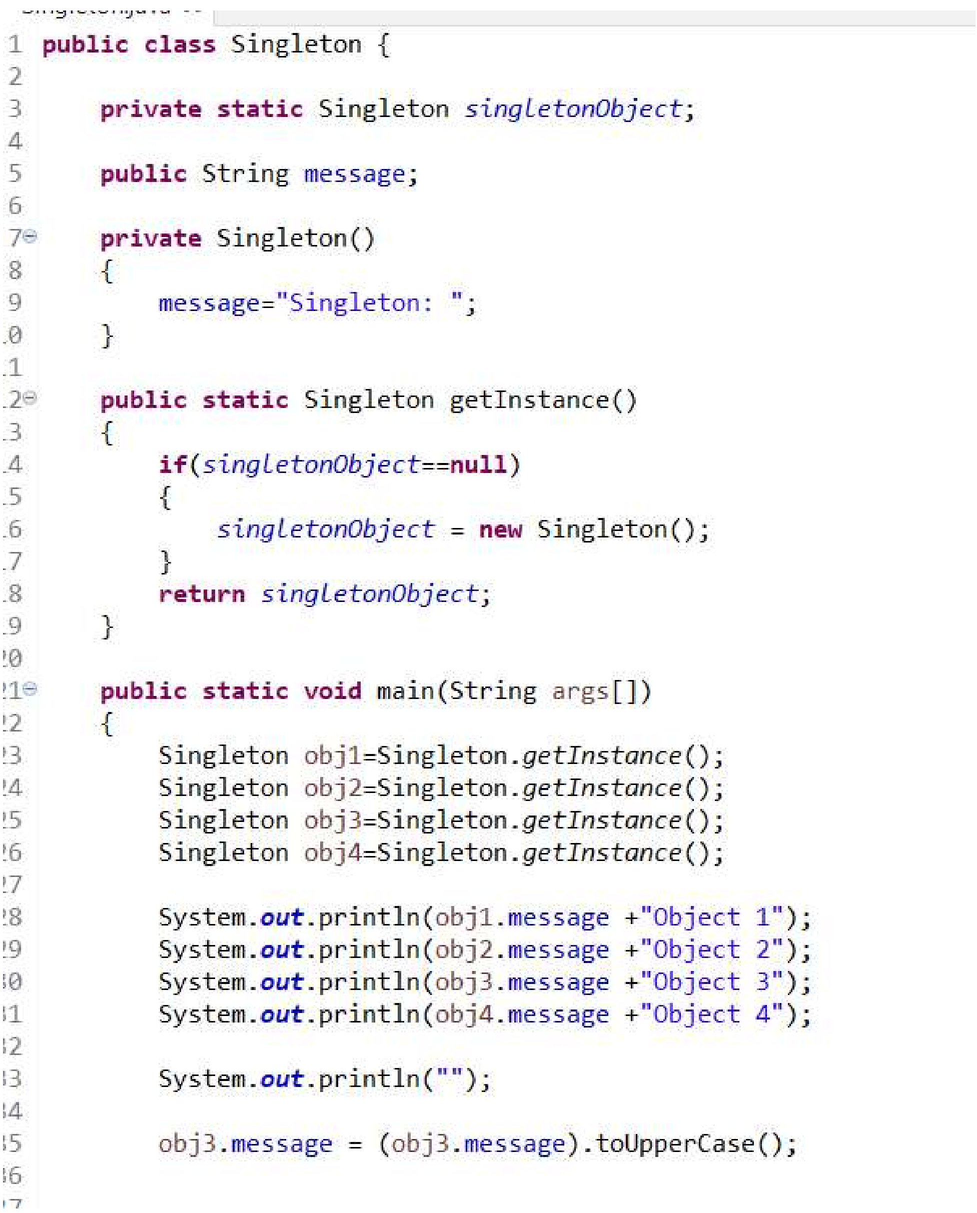
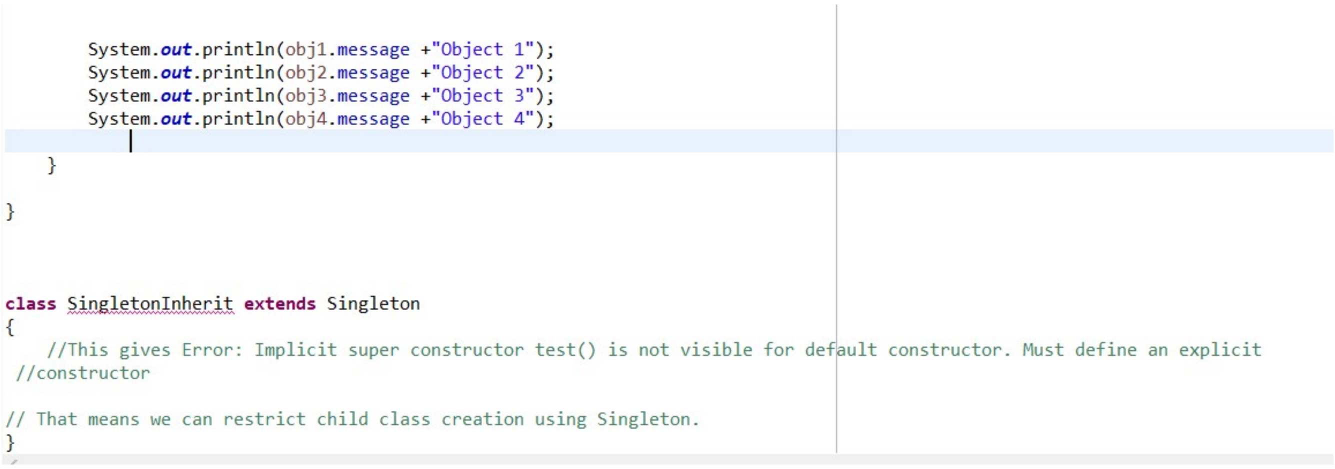
OOPS ASSIGNMENT

1)Singleton Class:

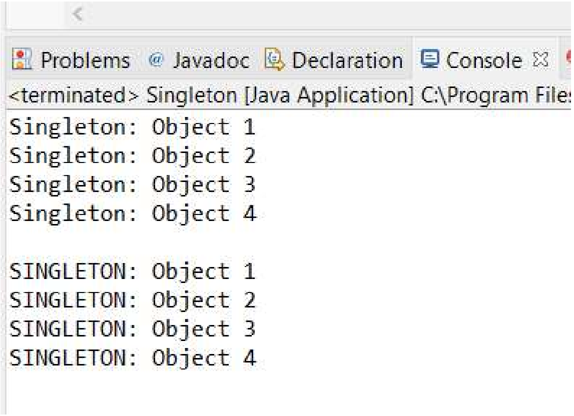


```
public class Singleton {

    private static Singleton singletonObject;

    public String message;

    private Singleton()
    {
        message="Singleton: ";
    }

    public static Singleton getInstance()
    {
        if(singletonObject==null)
        {
            singletonObject = new Singleton();
        }
        return singletonObject;
    }

    public static void main(String args[])
    {
        Singleton obj1=Singleton.getInstance();
        Singleton obj2=Singleton.getInstance();
        Singleton obj3=Singleton.getInstance();
        Singleton obj4=Singleton.getInstance();

        System.out.println(obj1.message +"Object 1");
        System.out.println(obj2.message +"Object 2");
        System.out.println(obj3.message +"Object 3");
        System.out.println(obj4.message +"Object 4");

        System.out.println("");

        obj3.message = (obj3.message).toUpperCase();

```



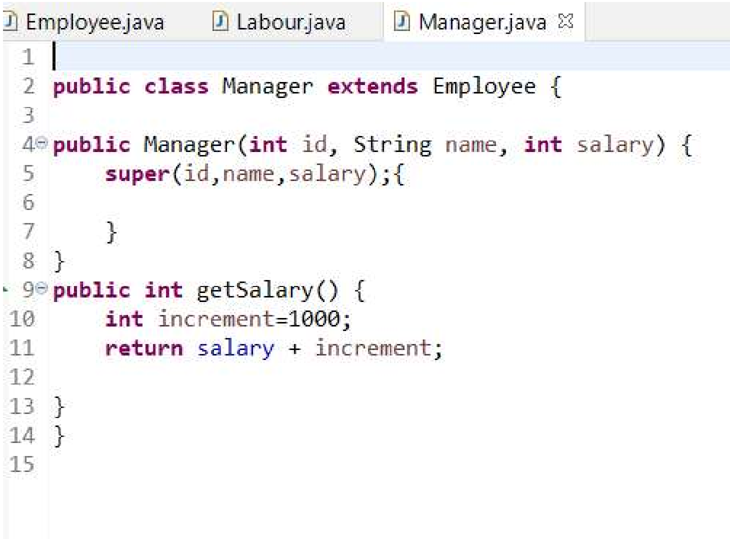
```
        System.out.println(obj1.message +"Object 1");
        System.out.println(obj2.message +"Object 2");
        System.out.println(obj3.message +"Object 3");
        System.out.println(obj4.message +"Object 4");

    }

}


class SingletonInherit extends Singleton
{
    //This gives Error: Implicit super constructor test() is not visible for default constructor. Must define an explicit
 //constructor

// That means we can restrict child class creation using Singleton.
}
```

Output:

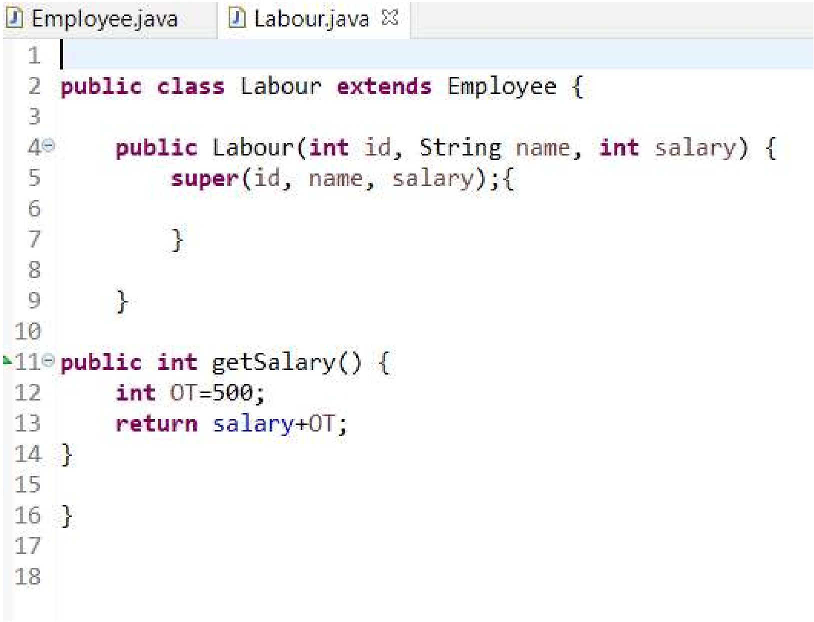


```
<terminated> Singleton [Java Application] C:\Program File:
Singleton: Object 1
Singleton: Object 2
Singleton: Object 3
Singleton: Object 4

SINGLETON: Object 1
SINGLETON: Object 2
SINGLETON: Object 3
SINGLETON: Object 4
```

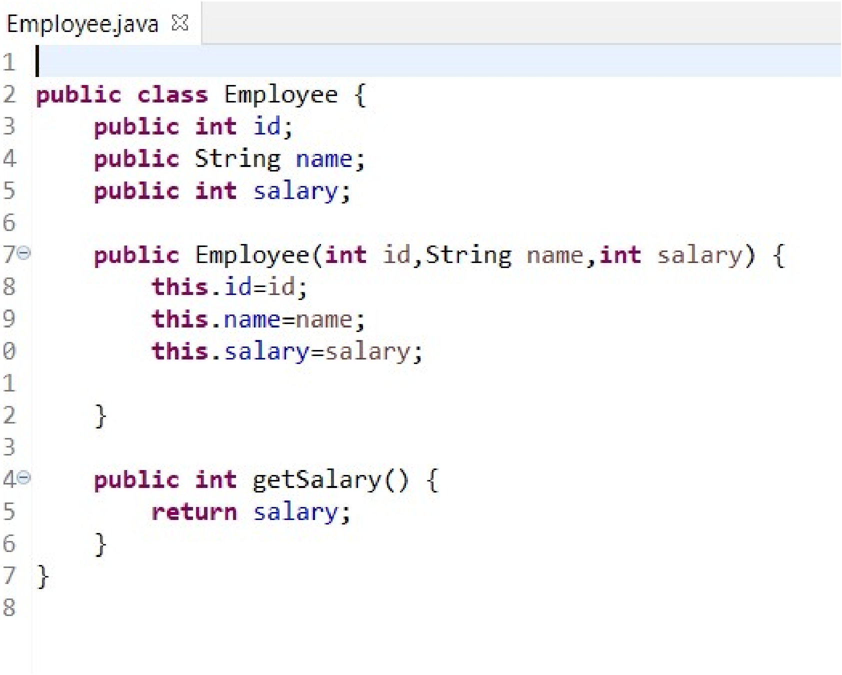
2)Hierarchy of Organization:



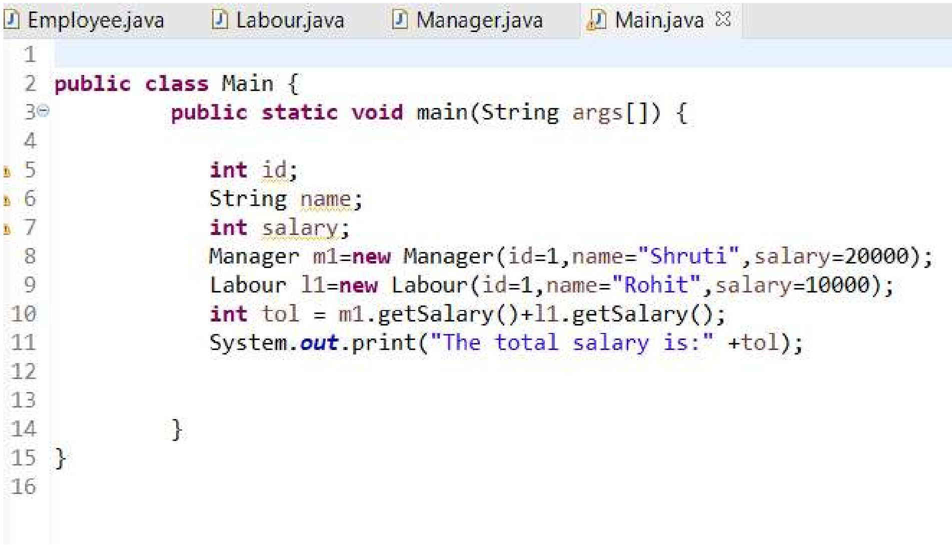
```java

public class Manager extends Employee {

public Manager(int id, String name, int salary) {
    super(id,name,salary);{

    }
}
public int getSalary() {
    int increment=1000;
    return salary + increment;

}
}

```



```java

public class Labour extends Employee {

    public Labour(int id, String name, int salary) {
        super(id, name, salary);{

        }

    }

public int getSalary() {
    int OT=500;
    return salary+OT;
}

}


```

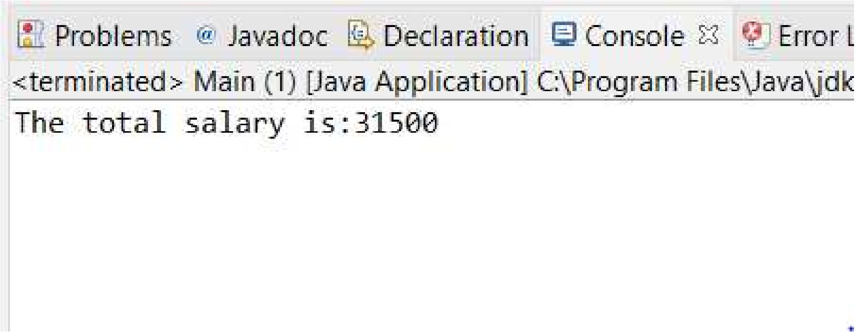


```java

public class Employee {
    public int id;
    public String name;
    public int salary;

    public Employee(int id,String name,int salary) {
        this.id=id;
        this.name=name;
        this.salary=salary;

    }

    public int getSalary() {
        return salary;
    }
}

```



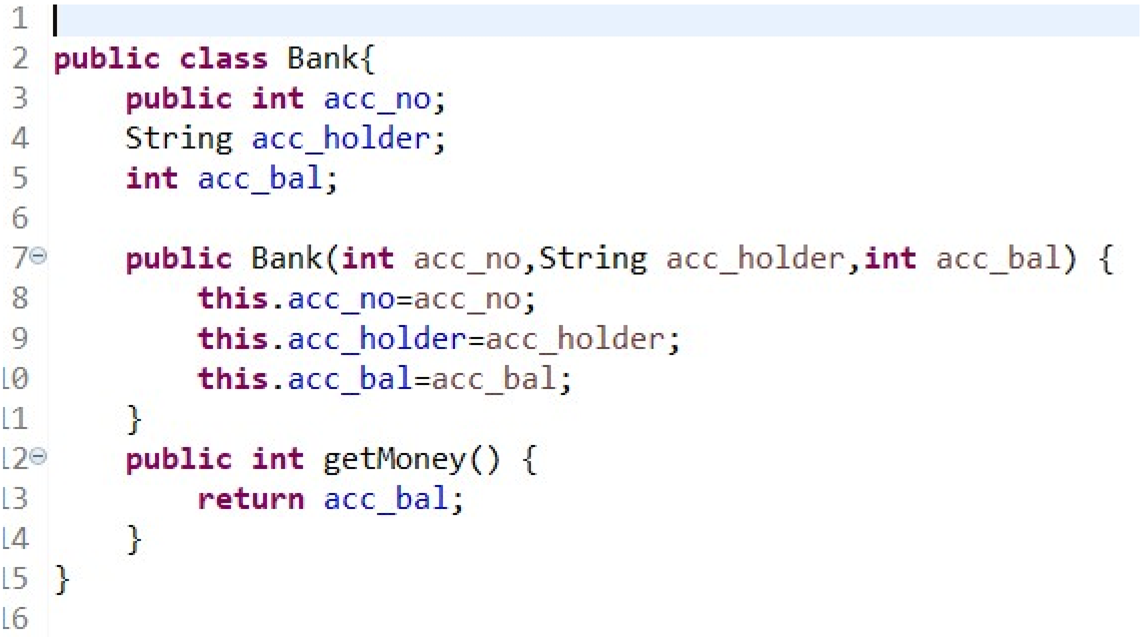
```java

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

            int id;
            String name;
            int salary;
            Manager m1=new Manager(id=1,name="Shruti",salary=20000);
            Labour l1=new Labour(id=1,name="Rohit",salary=10000);
            int tol = m1.getSalary()+l1.getSalary();
            System.out.print("The total salary is:" +tol);


        }
}

```

Output:

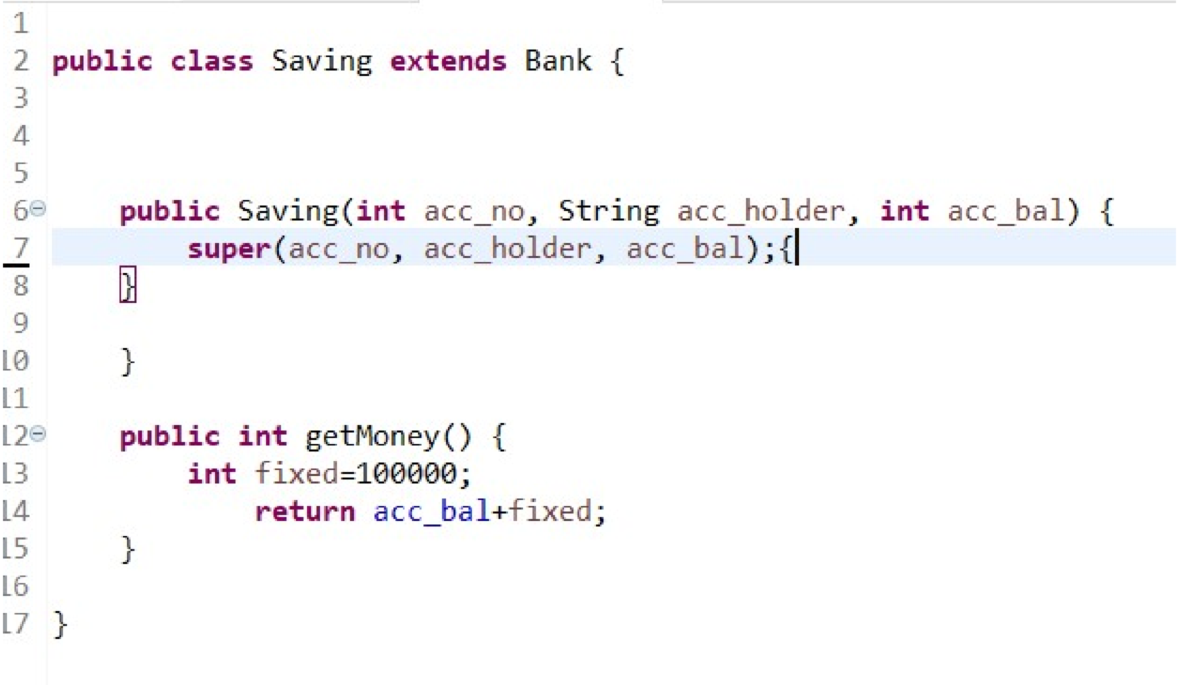


```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk
The total salary is:31500
```

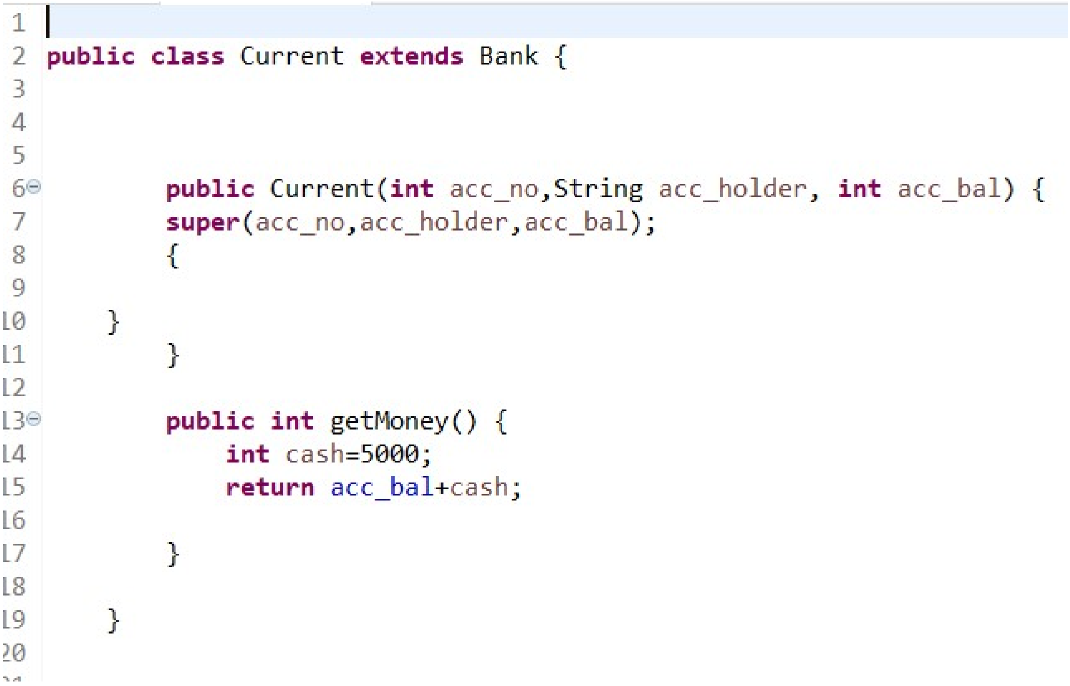
3)Bank:



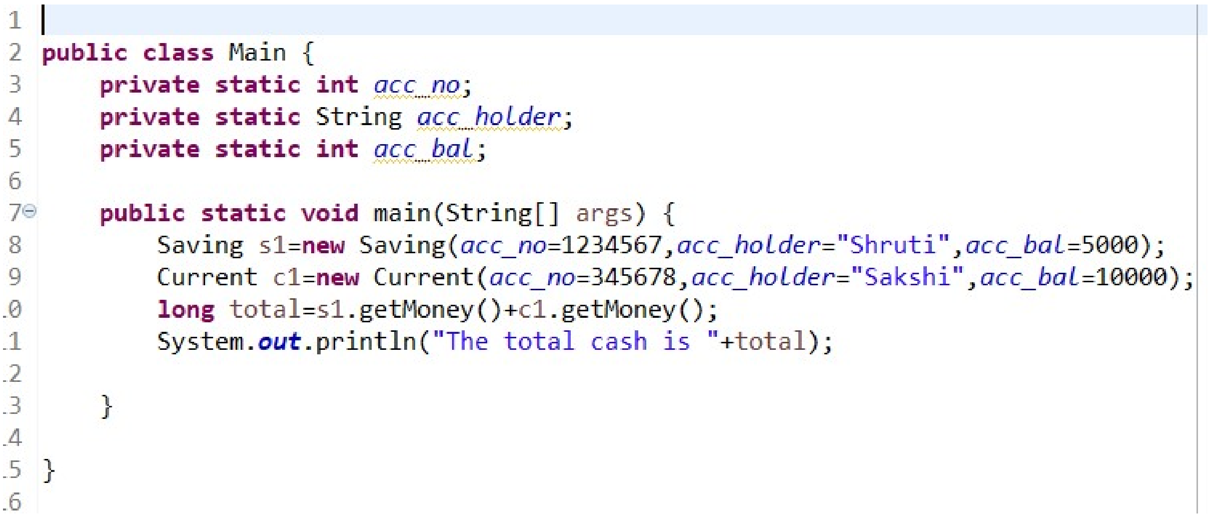
```java

public class Bank{
    public int acc_no;
    String acc_holder;
    int acc_bal;

    public Bank(int acc_no,String acc_holder,int acc_bal) {
        this.acc_no=acc_no;
        this.acc_holder=acc_holder;
        this.acc_bal=acc_bal;
    }
    public int getMoney() {
        return acc_bal;
    }
}

```



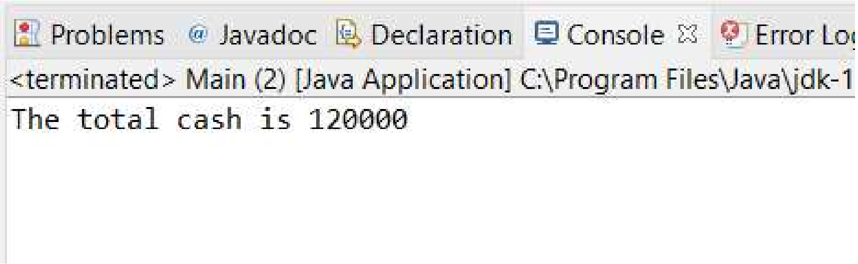
```java

public class Saving extends Bank {



    public Saving(int acc_no, String acc_holder, int acc_bal) {
        super(acc_no, acc_holder, acc_bal);{
    }

    }

    public int getMoney() {
        int fixed=100000;
            return acc_bal+fixed;
    }

}
```



```java

public class Current extends Bank {



        public Current(int acc_no,String acc_holder, int acc_bal) {
        super(acc_no,acc_holder,acc_bal);
        {

    }
        }

        public int getMoney() {
            int cash=5000;
            return acc_bal+cash;

        }

    }

```

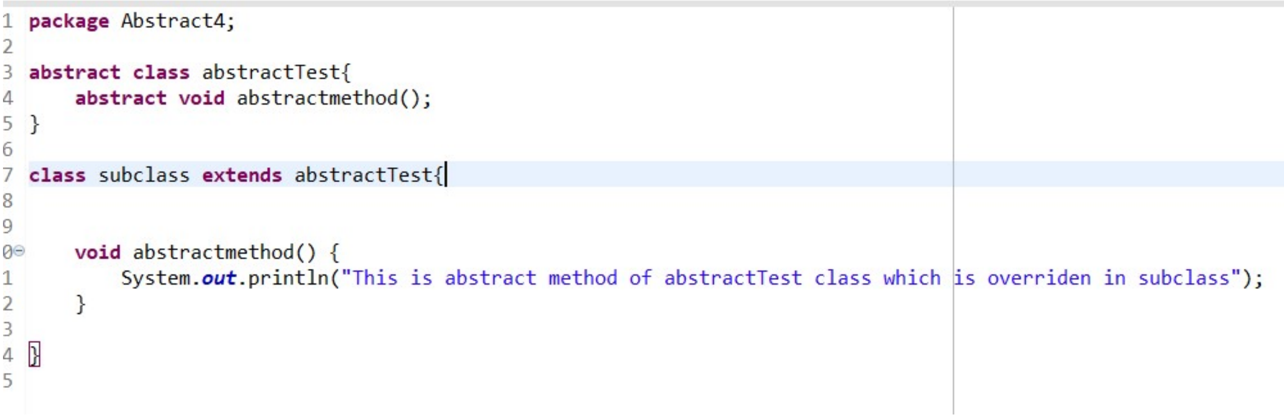


```java

public class Main {
    private static int acc_no;
    private static String acc_holder;
    private static int acc_bal;

    public static void main(String[] args) {
        Saving s1=new Saving(acc_no=1234567,acc_holder="Shruti",acc_bal=5000);
        Current c1=new Current(acc_no=345678,acc_holder="Sakshi",acc_bal=10000);
        long total=s1.getMoney()+c1.getMoney();
        System.out.println("The total cash is "+total);

    }

}

```

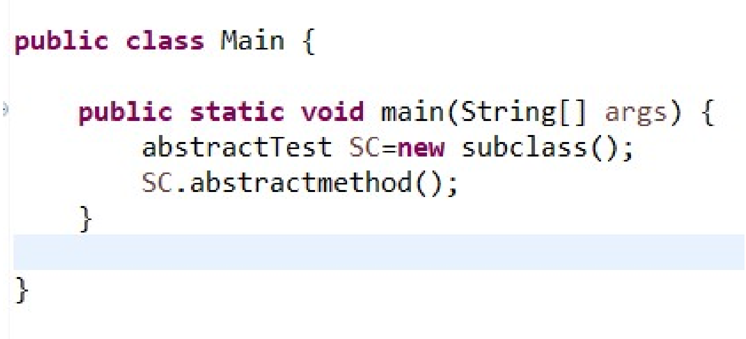


```
<terminated> Main (2) [Java Application] C:\Program Files\Java\jdk-1
The total cash is 120000
```

4)Abstract:

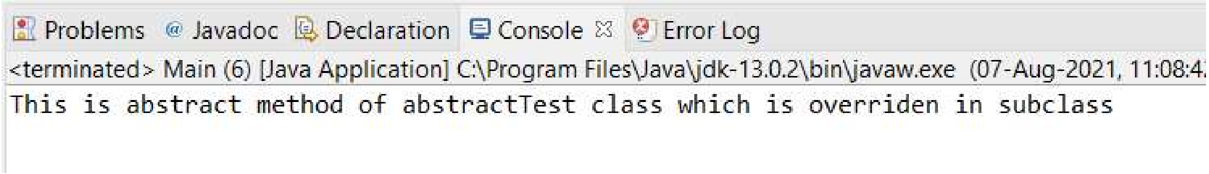


```java
package Abstract4;

abstract class abstractTest{
    abstract void abstractmethod();
}

class subclass extends abstractTest{


    void abstractmethod() {
        System.out.println("This is abstract method of abstractTest class which is overriden in subclass");
    }

}

```



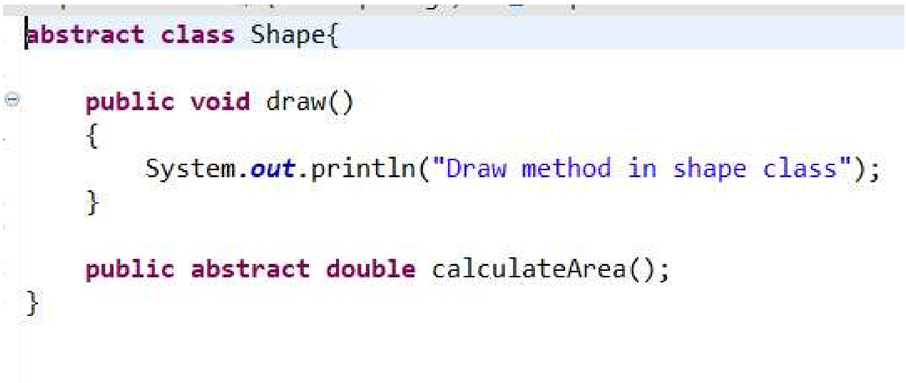
```java
public class Main {

    public static void main(String[] args) {
        abstractTest SC=new subclass();
        SC.abstractmethod();
    }

}
```

Output:

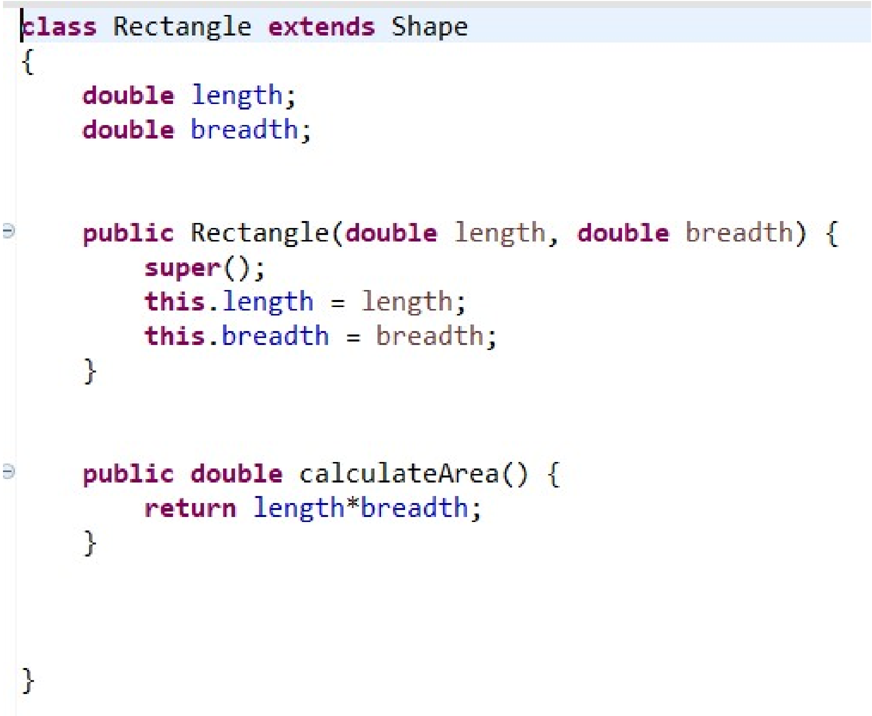


```
<terminated> Main (6) [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe  (07-Aug-2021, 11:08:4
This is abstract method of abstractTest class which is overriden in subclass
```

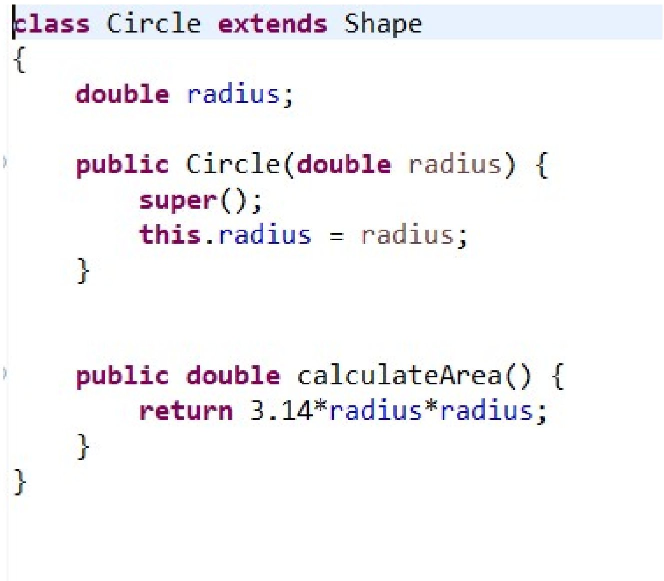
5)Shapes:



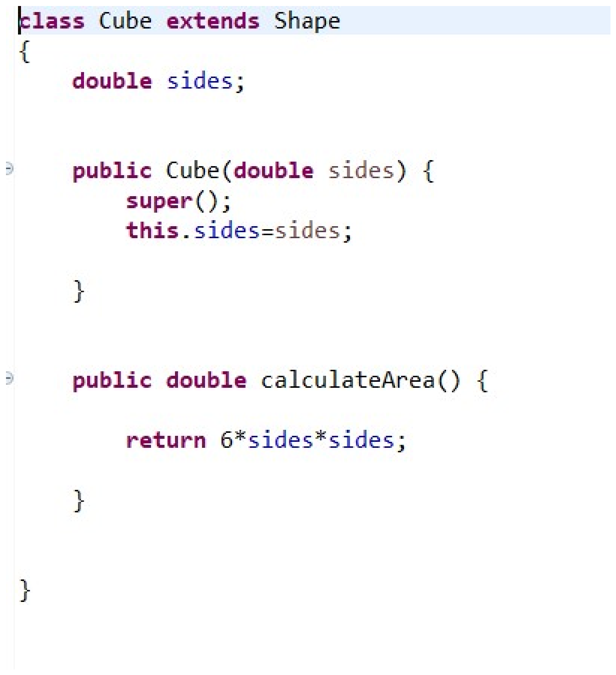
```java
abstract class Shape{

    public void draw()
    {
        System.out.println("Draw method in shape class");
    }

    public abstract double calculateArea();
}
```



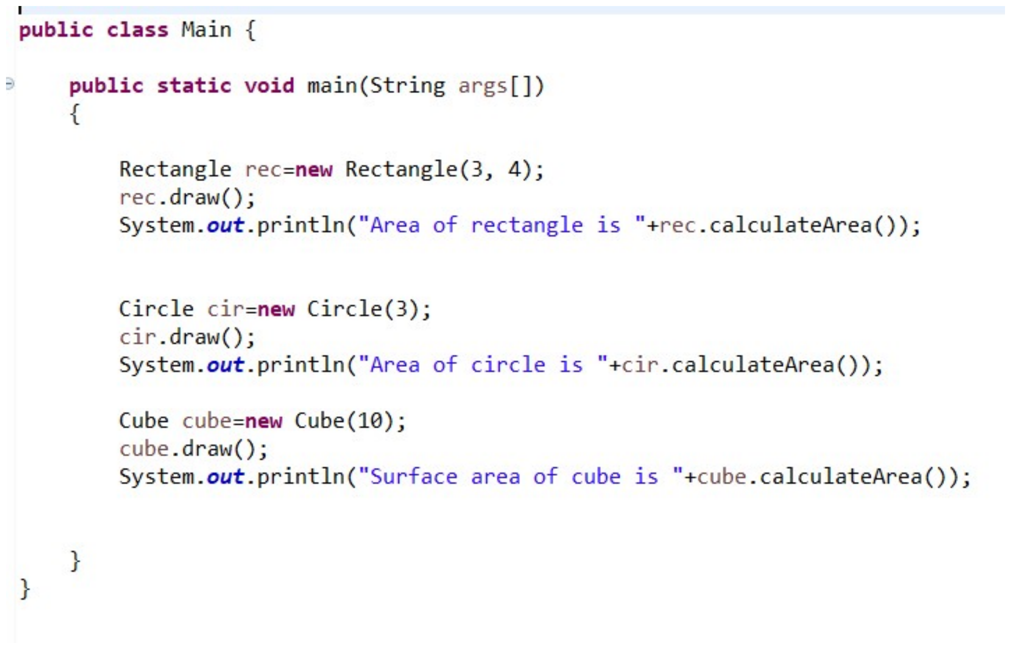
```java
class Rectangle extends Shape
{
    double length;
    double breadth;


    public Rectangle(double length, double breadth) {
        super();
        this.length = length;
        this.breadth = breadth;
    }


    public double calculateArea() {
        return length*breadth;
    }



}
```



```java
class Circle extends Shape
{
    double radius;

    public Circle(double radius) {
        super();
        this.radius = radius;
    }


    public double calculateArea() {
        return 3.14*radius*radius;
    }
}
```

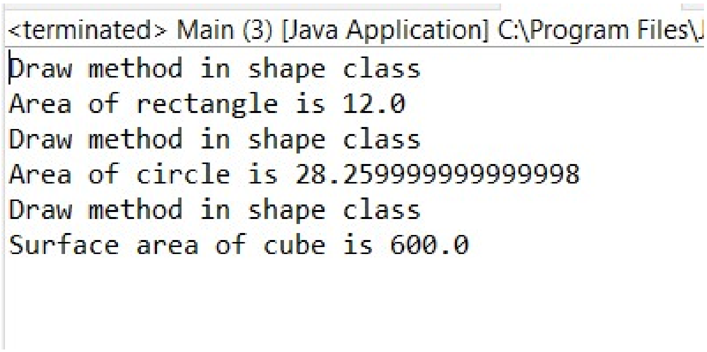


```java
class Cube extends Shape
{
    double sides;


    public Cube(double sides) {
        super();
        this.sides=sides;

    }


    public double calculateArea() {

        return 6*sides*sides;

    }


}
```



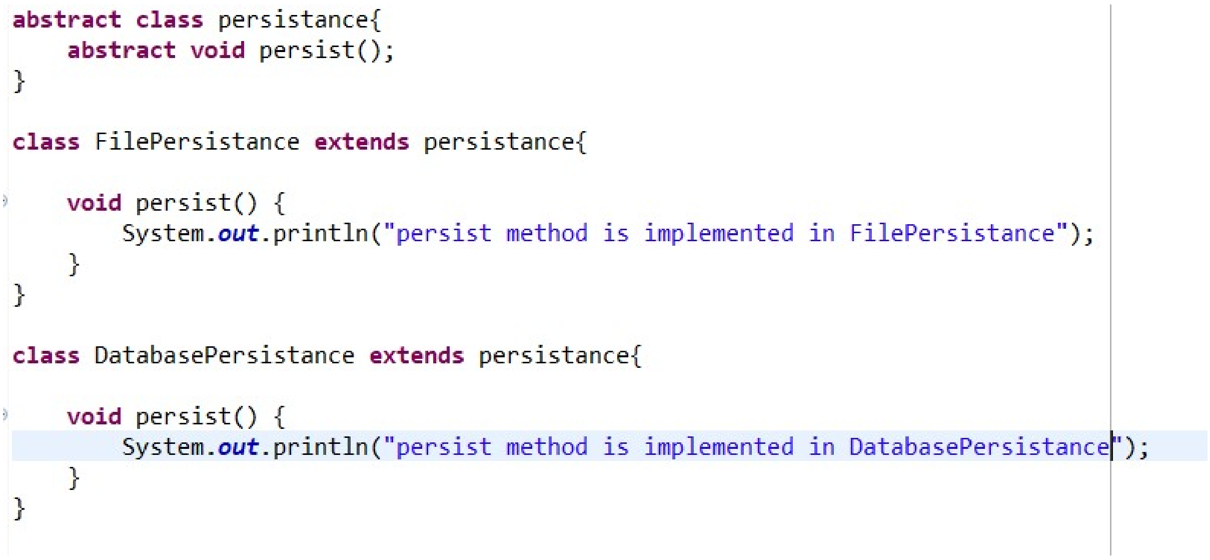
```java
public class Main {

    public static void main(String args[])
    {

        Rectangle rec=new Rectangle(3, 4);
        rec.draw();
        System.out.println("Area of rectangle is "+rec.calculateArea());


        Circle cir=new Circle(3);
        cir.draw();
        System.out.println("Area of circle is "+cir.calculateArea());

        Cube cube=new Cube(10);
        cube.draw();
        System.out.println("Surface area of cube is "+cube.calculateArea());


    }
}
```

Output:

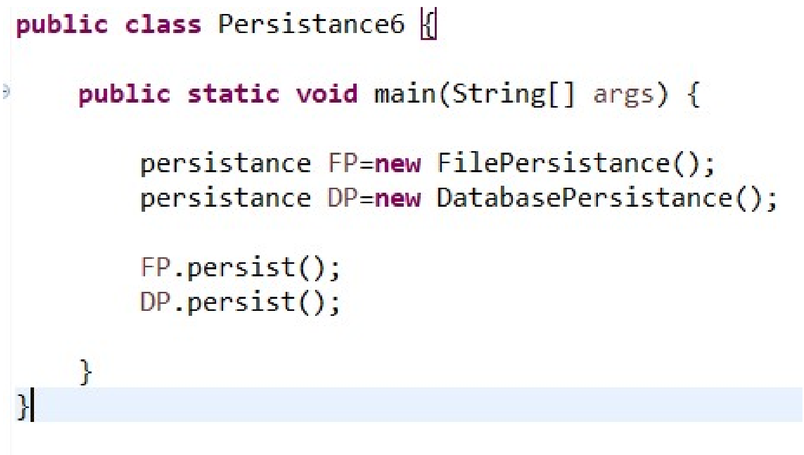


```
Draw method in shape class
Area of rectangle is 12.0
Draw method in shape class
Area of circle is 28.259999999999998
Draw method in shape class
Surface area of cube is 600.0
```

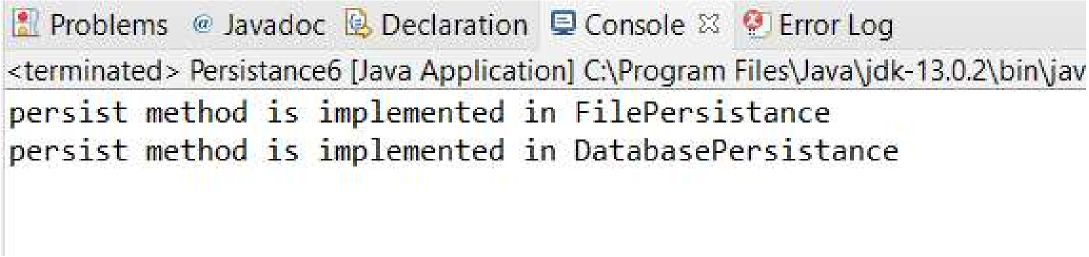
6)Persistance:



```
abstract class persistance{
    abstract void persist();
}

class FilePersistance extends persistance{

    void persist() {
        System.out.println("persist method is implemented in FilePersistance");
    }
}

class DatabasePersistance extends persistance{

    void persist() {
        System.out.println("persist method is implemented in DatabasePersistance");
    }
}
```

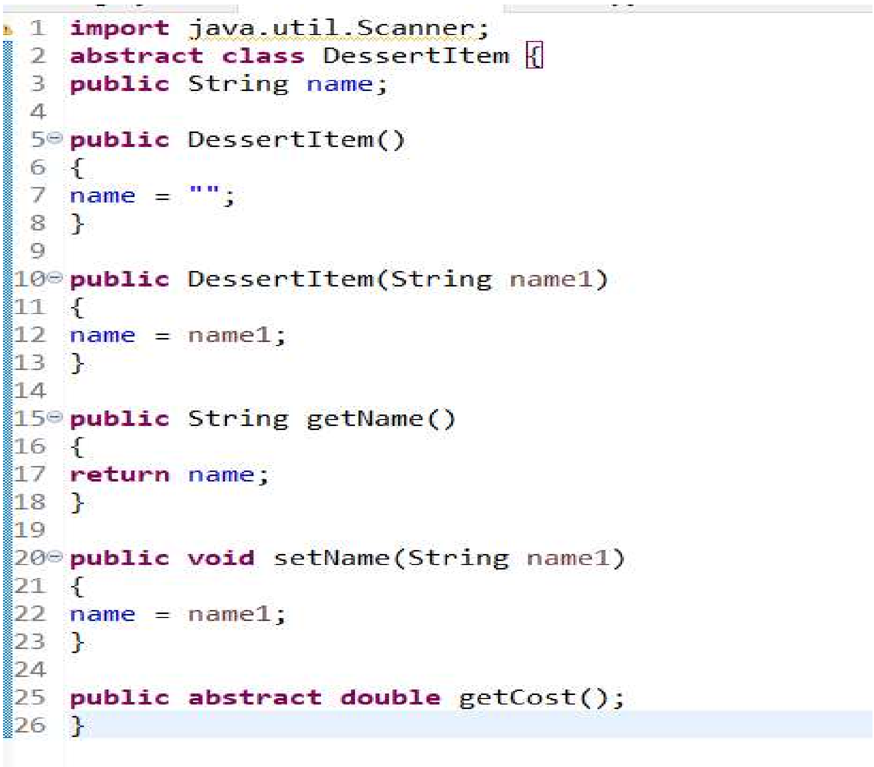


```
public class Persistance6 {

    public static void main(String[] args) {

        persistance FP=new FilePersistance();
        persistance DP=new DatabasePersistance();

        FP.persist();
        DP.persist();

    }
}
```

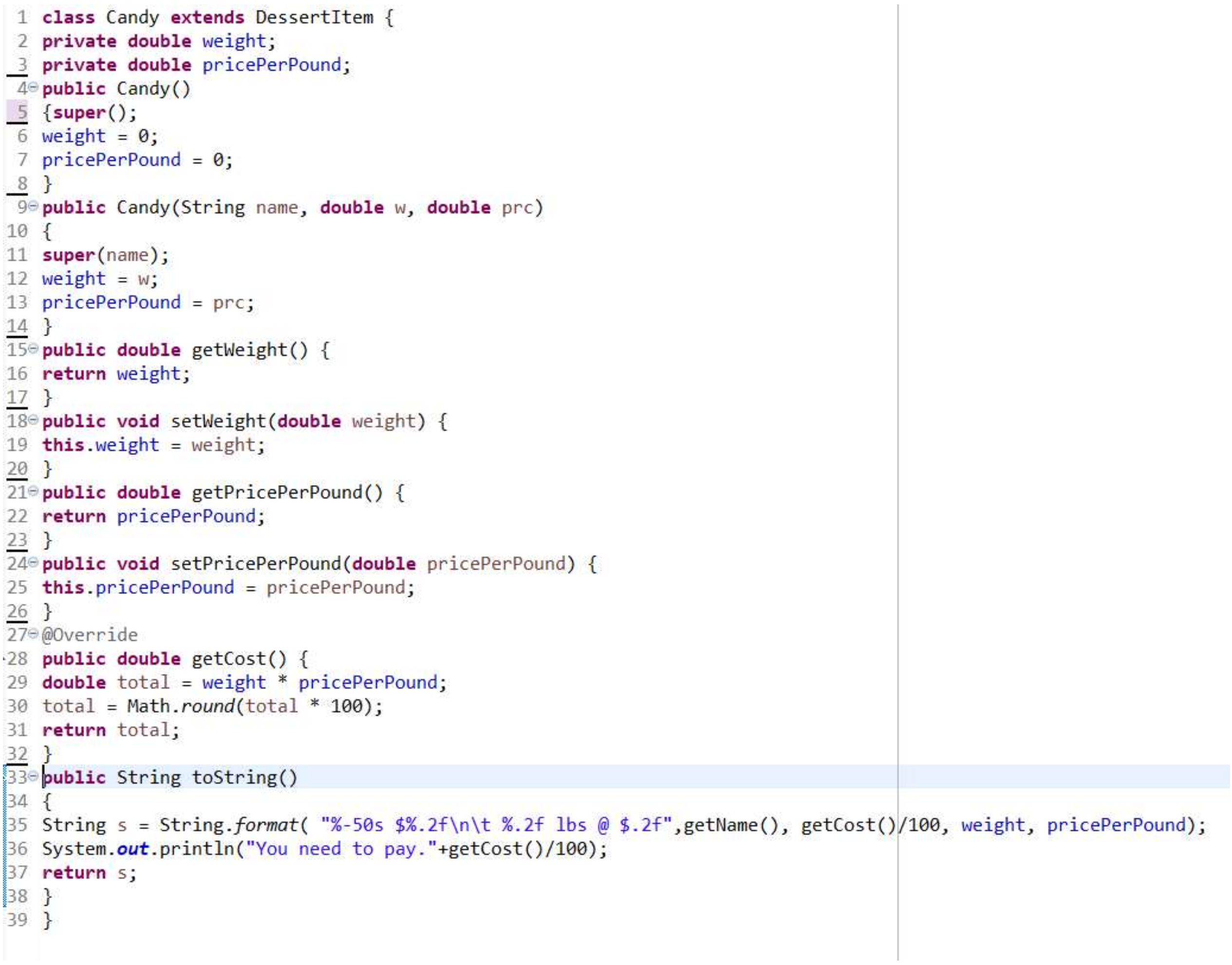


```
<terminated> Persistance6 [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\jav
persist method is implemented in FilePersistance
persist method is implemented in DatabasePersistance
```

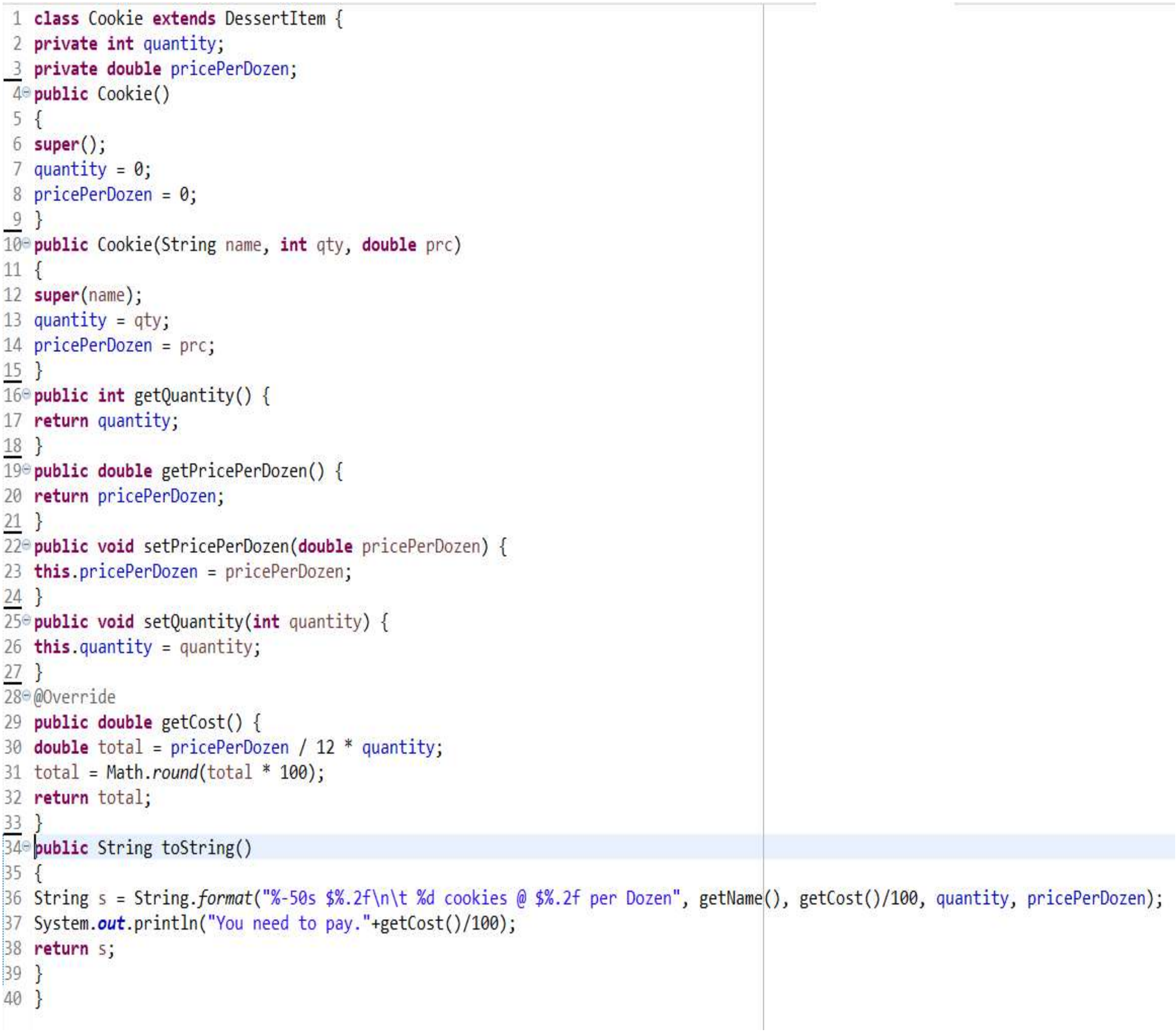
7)Dessert:



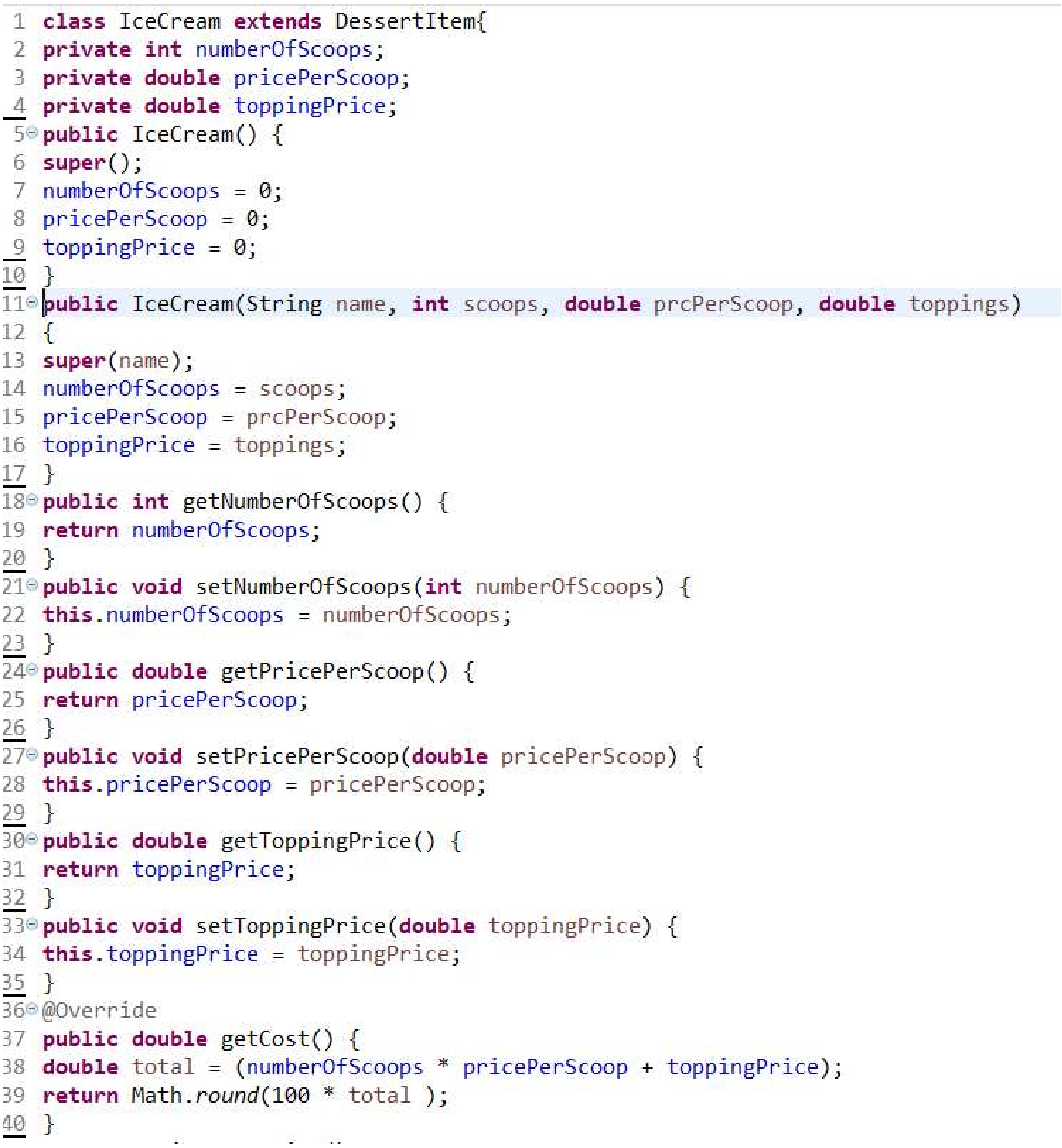
```
import java.util.Scanner;
abstract class DessertItem {
public String name;

public DessertItem()
{
name = "";
}

public DessertItem(String name1)
{
name = name1;
}

public String getName()
{
return name;
}

public void setName(String name1)
{
name = name1;
}

public abstract double getCost();
}
```



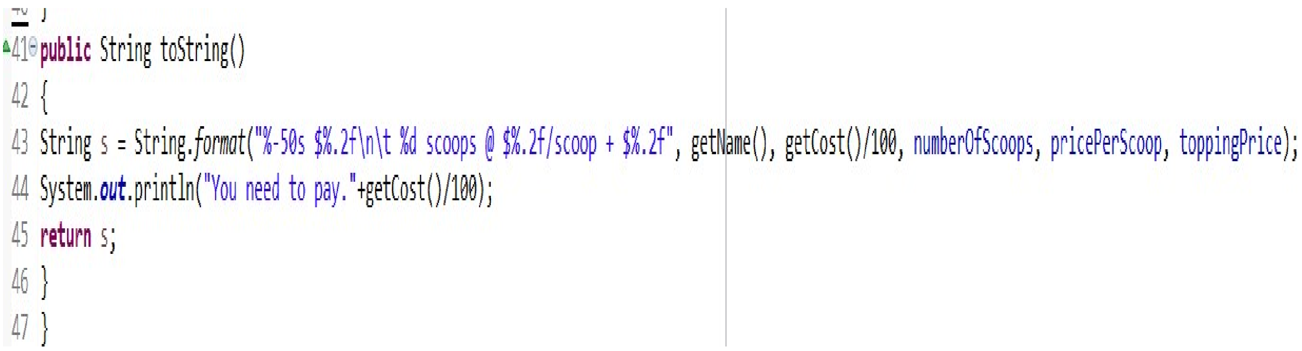
```
class Candy extends DessertItem {
private double weight;
private double pricePerPound;
public Candy()
{super();
weight = 0;
pricePerPound = 0;
}
public Candy(String name, double w, double prc)
{
super(name);
weight = w;
pricePerPound = prc;
}
public double getWeight() {
return weight;
}
public void setWeight(double weight) {
this.weight = weight;
}
public double getPricePerPound() {
return pricePerPound;
}
public void setPricePerPound(double pricePerPound) {
this.pricePerPound = pricePerPound;
}
@Override
public double getCost() {
double total = weight * pricePerPound;
total = Math.round(total * 100);
return total;
}
public String toString()
{
String s = String.format( "%-50s $%.2f\n\t %.2f lbs @ $.2f",getName(), getCost()/100, weight, pricePerPound);
System.out.println("You need to pay."+getCost()/100);
return s;
}
}
```



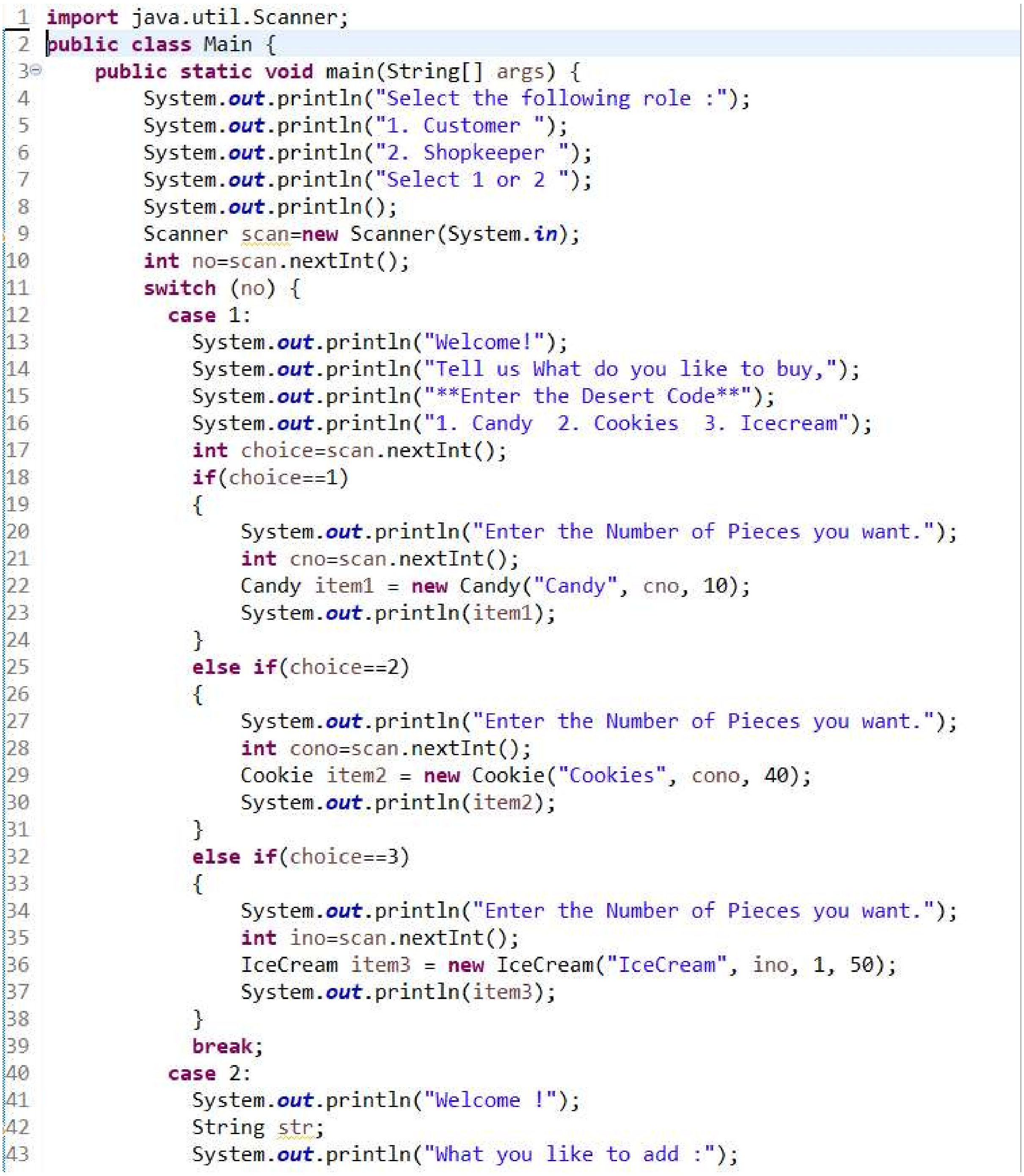
```
class Cookie extends DessertItem {
private int quantity;
private double pricePerDozen;
public Cookie()
{
super();
quantity = 0;
pricePerDozen = 0;
}
public Cookie(String name, int qty, double prc)
{
super(name);
quantity = qty;
pricePerDozen = prc;
}
public int getQuantity() {
return quantity;
}
public double getPricePerDozen() {
return pricePerDozen;
}
public void setPricePerDozen(double pricePerDozen) {
this.pricePerDozen = pricePerDozen;
}
public void setQuantity(int quantity) {
this.quantity = quantity;
}
@Override
public double getCost() {
double total = pricePerDozen / 12 * quantity;
total = Math.round(total * 100);
return total;
}
public String toString()
{
String s = String.format("%-50s $%.2f\n\t %d cookies @ $%.2f per Dozen", getName(), getCost()/100, quantity, pricePerDozen);
System.out.println("You need to pay."+getCost()/100);
return s;
}
}
```



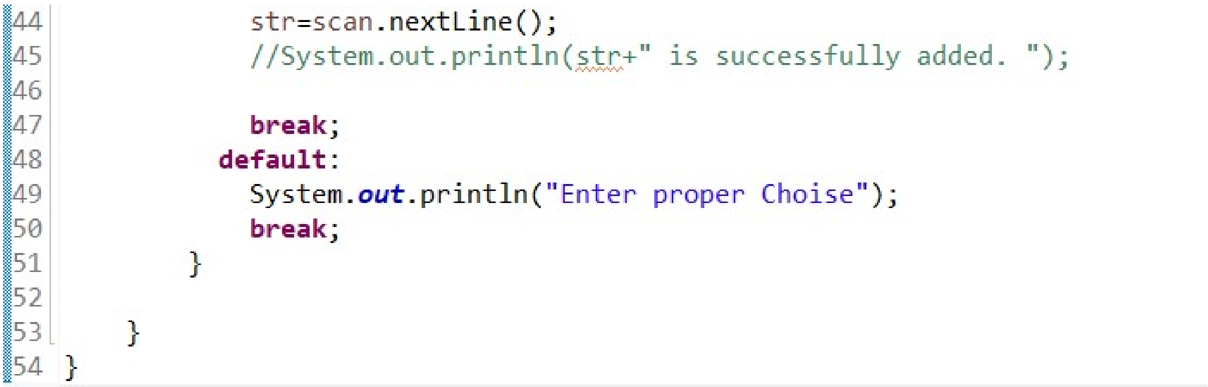
```java
class IceCream extends DessertItem{
private int numberOfScoops;
private double pricePerScoop;
private double toppingPrice;
public IceCream() {
super();
numberOfScoops = 0;
pricePerScoop = 0;
toppingPrice = 0;
}
public IceCream(String name, int scoops, double prcPerScoop, double toppings)
{
super(name);
numberOfScoops = scoops;
pricePerScoop = prcPerScoop;
toppingPrice = toppings;
}
public int getNumberOfScoops() {
return numberOfScoops;
}
public void setNumberOfScoops(int numberOfScoops) {
this.numberOfScoops = numberOfScoops;
}
public double getPricePerScoop() {
return pricePerScoop;
}
public void setPricePerScoop(double pricePerScoop) {
this.pricePerScoop = pricePerScoop;
}
public double getToppingPrice() {
return toppingPrice;
}
public void setToppingPrice(double toppingPrice) {
this.toppingPrice = toppingPrice;
}
@Override
public double getCost() {
double total = (numberOfScoops * pricePerScoop + toppingPrice);
return Math.round(100 * total );
}
```



```java
public String toString()
{
String s = String.format("%-50s $%.2f\n\t %d scoops @ $%.2f/scoop + $%.2f", getName(), getCost()/100, numberOfScoops, pricePerScoop, toppingPrice);
System.out.println("You need to pay."+getCost()/100);
return s;
}
}
```

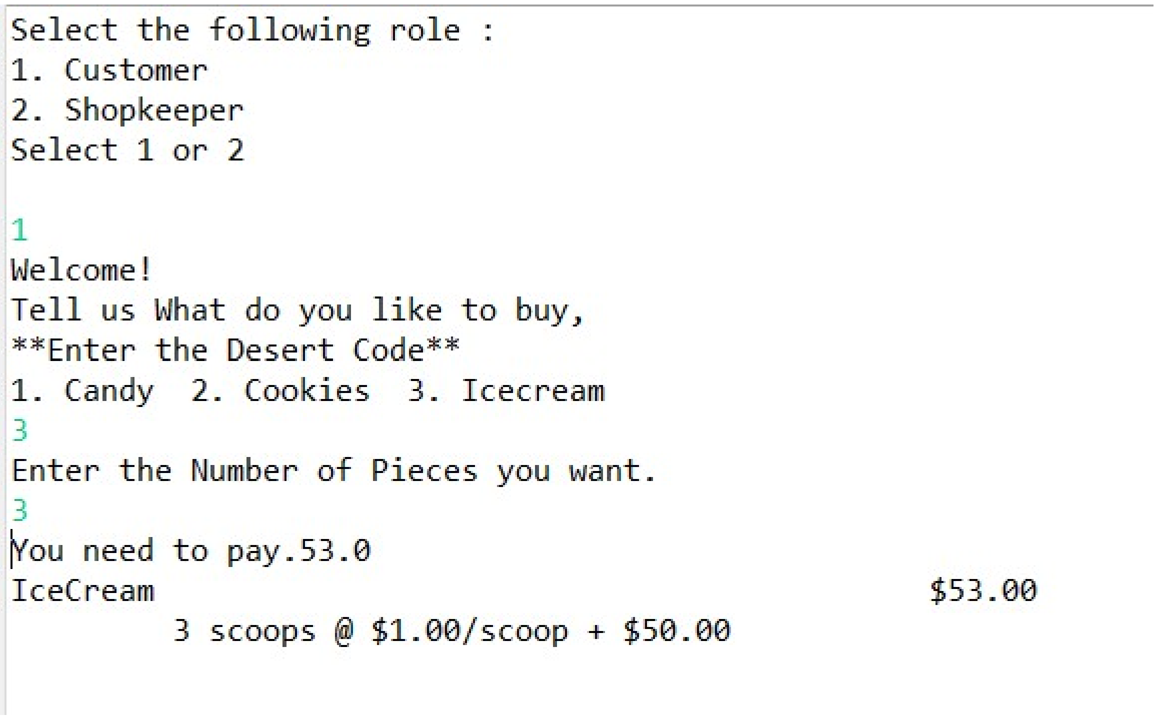


```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        System.out.println("Select the following role :");
        System.out.println("1. Customer ");
        System.out.println("2. Shopkeeper ");
        System.out.println("Select 1 or 2 ");
        System.out.println();
        Scanner scan=new Scanner(System.in);
        int no=scan.nextInt();
        switch (no) {
          case 1:
            System.out.println("Welcome!");
            System.out.println("Tell us What do you like to buy,");
            System.out.println("**Enter the Desert Code**");
            System.out.println("1. Candy  2. Cookies  3. Icecream");
            int choice=scan.nextInt();
            if(choice==1)
            {
                System.out.println("Enter the Number of Pieces you want.");
                int cno=scan.nextInt();
                Candy item1 = new Candy("Candy", cno, 10);
                System.out.println(item1);
            }
            else if(choice==2)
            {
                System.out.println("Enter the Number of Pieces you want.");
                int cono=scan.nextInt();
                Cookie item2 = new Cookie("Cookies", cono, 40);
                System.out.println(item2);
            }
            else if(choice==3)
            {
                System.out.println("Enter the Number of Pieces you want.");
                int ino=scan.nextInt();
                IceCream item3 = new IceCream("IceCream", ino, 1, 50);
                System.out.println(item3);
            }
            break;
          case 2:
            System.out.println("Welcome !");
            String str;
            System.out.println("What you like to add :");
```



```java
            str=scan.nextLine();
            //System.out.println(str+" is successfully added. ");

            break;
          default:
            System.out.println("Enter proper Choise");
            break;
        }

    }
}
```

Output:



```
Select the following role :
1. Customer
2. Shopkeeper
Select 1 or 2

1
Welcome!
Tell us What do you like to buy,
**Enter the Desert Code**
1. Candy  2. Cookies  3. Icecream
3
Enter the Number of Pieces you want.
3
You need to pay.53.0
IceCream                                           $53.00
         3 scoops @ $1.00/scoop + $50.00
```