IRC_SKCT_Java2_SB_COD_Inheritance

Test Summary

- No. of Sections: 1
- No. of Questions: 5
- Total Duration: 120 min

Section 1 - Coding

Section Summary

- No. of Questions: 5
- Duration: 120 min

Additional Instructions:

None

Q1. Write a program by creating a class Bicycle as a base class with a number of gears and speed of bicycle as integer attributes and create a class called MountainBike, a derived class that extends Bicycle class with an attribute seat height as an integer. Create a Test class to run the program and obtain the output in the console.
Note: Override toString() method to display the details of the bicycle.

Input Format

To get 3 integers from the user (Number of gears, Speed of bicycle, and Seat height).

Output Format

To display the desired output from the test class.

Constraints

integers only.

Sample Input

```
2 90 40
```

Sample Output

```
No of gears are 2
speed of bicycle is 90
seat height is 40
```

Sample Input

```
3 60 20
```

Sample Output

```
No of gears are 3
speed of bicycle is 60
seat height is 20
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. Write a program by creating a class Bicycle as a base class with a number of gears and speed of bicycle as integer attributes and create a class called MountainBike, a derived class that extends Bicycle class with an attribute seat height as an integer. Create a Test class to run the program and obtain the output in the console.
Note: Override toString() method to display the details of the bicycle.

Input Format

To get 3 integers from the user (Number of gears, Speed of bicycle, and Seat height).

Output Format

To display the desired output from the test class.

Constraints

integers only.

Sample Input

```
2 90 40
```

Sample Output

```
No of gears are 2
speed of bicycle is 90
seat height is 40
```

Sample Input

```
3 60 20
```

Sample Output

```
No of gears are 3
speed of bicycle is 60
seat height is 20
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q3. Develop a program for the banking system for account management. Each account has the following attributes: **AccountID, HolderName, and Balance**. Declare one constructor with three parameters that initialize the three attributes to some default values. Attributes must be validated.

- **AccountBalance** must be greater than or equal to zero. If not, it is set to zero.
- **AccountID** must be between 100 and 999. If not, set it to -1 to indicate that it is invalid.

Use the method **setAccountBalance (…)** to print the account balance. Write one method **Credit** to deposit money into the account. The method should return the new balance after money deposit. Then create a class **VIPAccount** that inherits from the class Account. The **VIPAccount** class overrides the method **setAccountBalance (…)** such that it prints the balance can be negative but no less than **– 10000.** The constructor of the VIPAccount class must call the constructor of the Account class.

Input Format

The first line of the input consists of the account id.
The next input is the account holder's name.
The third input is the initial balance.
The fourth input is the amount to be credited.
The last input is a negative balance (Argument to setAccountBalance in overridden method).

Output Format

The first line of the output prints the account details.
The next line prints the new balance after the amount is credited.
The next output is the result of setAccountBalance (First base class method then derived class method).

Sample Input

```
120
Alice
48200
500
```

Sample Output

```
120 Alice 48200
48700
48700
The balance can be negative but no less than -10000
```

Sample Input

```
10
Bob
120
100
```

Sample Output

```
-1 Bob 120
220
220
-1500
```

Sample Input

```
848
Charlie
-120
52040
```

Sample Output

```
848 Charlie 0
52040
52040
The balance can be negative but no less than -10000
```

Sample Input

```
1288
David
48484
84848
```

Sample Output

```
-1 David 48484
133332
133332
-5000
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q4. Create an abstract class **Shape** with length, width, radius, 3 sides as data members and two abstract methods to calculate area and perimeter. Create constructors and getter setters.

Create four classes **Square**, **Rectangle**, **Circle** and **Triangle.** Extend all the classes from **Shape** directly**.** Complete the abstract method to calculate area and perimeter in the derived classes.

Get a single character and suitable values from user to calculate area and perimeter.

Input Format

S or R or C or T in first line (S represents Square, R represents Rectangle, C represents Circle and T represents Triangle)
Enter one or two input based on Shape (1 input for Square and Circle, 2 inputs for Rectangle and 3 inputs for Triangle)

Output Format

Perimeter or Circumference
Area

Sample Input

```
S
5
```

Sample Output

```
Perimeter : 20.00
Area : 25.00
```

Sample Input

```
R
3
4
```

Sample Output

```
Perimeter : 14.00
Area : 12.00
```

Sample Input

```
C
7
```

Sample Output

```
Circumference : 43.98
Area : 153.94
```

Sample Input

```
T
3
4
5
```

Sample Output

```
Perimeter : 12.00
Area : 6.00
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q5. Write a program to implement the following logic using inheritance.
Create a parent class and implement the fun method. In the method, get the individual digits of the entered number, store it in an array, and find their sum. For example in case of 1234, the individual digits are 4,3,2,1 and the final sum → (4+3)+(4+2)+(4+1)+(3+2)+(3+1)+(2+1) = 30. Create the main class that inherits the parent class and call the fun method inside the parent function.

Input Format

The input consists of an integer.

Output Format

The output prints the final sum.

Constraints

Integers only.

Sample Input

```
1234
```

Sample Output

```
30
```

Sample Input

```
4356
```

Sample Output

```
54
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Answer Key & Solution

Section 1 - Coding

Q1

Test Case

Input

```
3 50 12
```

Output

```
No of gears are 3
speed of bicycle is 50
seat height is 12
```

Weightage - 10

Input

```
1 40 10
```

Output

```
No of gears are 1
speed of bicycle is 40
seat height is 10
```

Weightage - 10

Input

```
3 50 28
```

Output

```
No of gears are 3
speed of bicycle is 50
seat height is 28
```

Weightage - 10

Input

```
1 44 22
```

Output

```
No of gears are 1
speed of bicycle is 44
seat height is 22
```

Weightage - 10

Input

```
4 60 30
```

Output

```
No of gears are 4
speed of bicycle is 60
seat height is 30
```

Weightage - 10

Input

```
2 69 30
```

Output

```
No of gears are 2
speed of bicycle is 69
seat height is 30
```

Weightage - 10

Input

```
4 80 22
```

Output

```
No of gears are 4
speed of bicycle is 80
seat height is 22
```

Weightage - 20

Input

```
1 79 43
```

Output

```
No of gears are 1
speed of bicycle is 79
seat height is 43
```

Weightage - 20

Sample Input

```
2 90 40
```

Sample Output

```
No of gears are 2
speed of bicycle is 90
seat height is 40
```

Sample Input

```
3 60 20
```

Sample Output

```
No of gears are 3
speed of bicycle is 60
seat height is 20
```

Solution

```
import java.util.Scanner;
class Bicycle
{
    public int gear;
    public int speed;

    public Bicycle(int gear, int speed)
    {
        this.gear = gear;
        this.speed = speed;
    }

    public void applyBrake(int decrement)
    {
        speed -= decrement;
    }

    public void speedUp(int increment)
    {
        speed += increment;
    }

    // toString() method to print info of Bicycle
    public String toString()
    {
        return("No of gears are "+gear
                +"\n"
                + "speed of bicycle is "+speed);
    }
}

class MountainBike extends Bicycle
{

    public int seatHeight;

    public MountainBike(int gear,int speed,
```

```
                        int startHeight)
    {
        super(gear, speed);
        seatHeight = startHeight;
    }

    public void setHeight(int newValue)
    {
        seatHeight = newValue;
    }

    @Override
    public String toString()
    {
        return (super.toString()+
                "\nseat height is "+seatHeight);
    }

}

class Test
{
    public static void main(String args[])
    {
         int gear,speed,startHeight;
         Scanner sc=new Scanner(System.in);
         gear=sc.nextInt();
         speed=sc.nextInt();
         startHeight=sc.nextInt();
        MountainBike mb = new MountainBike(gear,speed,startHeight);
        System.out.println(mb.toString());

    }
}
```

Q2

Test Case

Input

```
3 50 12
```

Output

```
No of gears are 3
speed of bicycle is 50
seat height is 12
```

Weightage - 10

Input

```
1 40 10
```

Output

```
No of gears are 1
speed of bicycle is 40
seat height is 10
```

Weightage - 10

Input

```
3 50 28
```

Output

```
No of gears are 3
speed of bicycle is 50
seat height is 28
```

Weightage - 10

Input

```
1 44 22
```

Output

```
No of gears are 1
speed of bicycle is 44
seat height is 22
```

Weightage - 10

Input

```
4 60 30
```

Output

```
No of gears are 4
speed of bicycle is 60
seat height is 30
```

Weightage - 10

Input

```
2 69 30
```

Output

```
No of gears are 2
speed of bicycle is 69
seat height is 30
```

Weightage - 10

Input

```
4 80 22
```

Output

```
No of gears are 4
speed of bicycle is 80
seat height is 22
```

Weightage - 20

Input

```
1 79 43
```

Output

```
No of gears are 1
speed of bicycle is 79
seat height is 43
```

Weightage - 20

Sample Input

```
2 90 40
```

Sample Output

```
No of gears are 2
speed of bicycle is 90
seat height is 40
```

Sample Input

```
3 60 20
```

Sample Output

```
No of gears are 3
speed of bicycle is 60
seat height is 20
```

Solution

```
import java.util.Scanner;
class Bicycle
{
    public int gear;
    public int speed;
```

```
public Bicycle(int gear, int speed)
{
    this.gear = gear;
    this.speed = speed;
}

public void applyBrake(int decrement)
{
    speed -= decrement;
}

public void speedUp(int increment)
{
    speed += increment;
}

// toString() method to print info of Bicycle
public String toString()
{
    return("No of gears are "+gear
            + "\n"
            + "speed of bicycle is "+speed);
    }
}

class MountainBike extends Bicycle
{

    public int seatHeight;

    public MountainBike(int gear,int speed,
                        int startHeight)
    {
        super(gear, speed);
        seatHeight = startHeight;
    }

    public void setHeight(int newValue)
    {
        seatHeight = newValue;
    }

    @Override
    public String toString()
    {
        return (super.toString()+
                "\nseat height is "+seatHeight);
    }

}

class Test
{
    public static void main(String args[])
    {
         int gear,speed,startHeight;
         Scanner sc=new Scanner(System.in);
         gear=sc.nextInt();
         speed=sc.nextInt();
         startHeight=sc.nextInt();
        MountainBike mb = new MountainBike(gear,speed,startHeight);
        System.out.println(mb.toString());
```

```
    }
}
```

Q3

Test Case

Input

```
120
Alice
48200
500
```

Output

```
120 Alice 48200
48700
48700
```

Weightage - 25

Input

```
10
Bob
120
100
```

Output

```
-1 Bob 120
220
220
5200
```

Weightage - 25

Input

```
848
Charlie
-120
52040
```

Output

```
848 Charlie 0
52040
52040
```

Weightage - 25

Input

```
1288
David
48484
84848
```

Output

```
-1 David 48484
133332
133332
5200
```

Weightage - 25

Sample Input

```
120
Alice
48200
500
```

Sample Output

```
120 Alice 48200
48700
48700
```

Sample Input

```
10
Bob
120
100
```

Sample Output

```
-1 Bob 120
220
220
1500
```

Sample Input

```
848
Charlie
-120
52040
```

Sample Output

```
848 Charlie 0
52040
52040
```

Sample Input

```
1288
David
```

Sample Output

```
-1 David 48484
133332
```

48484 133332

Solution

```
import java.io.*;
import java.util.*;
class Account {
    public int AccountId;
    public String HolderName;
    public int balance;
    Account() {
        this.AccountId = 0;
        this.HolderName = null;
        this.balance = 0;
    }
    Account(int id,String name,int bal) {
        if(id>=100 && id<=999) {
            this.AccountId = id;
        }
        else {
            this.AccountId = -1;
        }
        this.HolderName = name;
        if(bal>=0) {
            this.balance = bal;
        }
        else {
            this.balance = 0;
        }
    }
    public void setAccountBalance(int s) {
        System.out.println(this.balance);
    }
    public int credit(int amount) {
        this.balance += amount;
        return this.balance;
    }
}
class VIPAccount extends Account {
    VIPAccount() {
        super();
    }
    VIPAccount(int id, String name,int bal) {
        super(id,name,bal);
    }
    public void setAccountBalance(int s) {
        super.setAccountBalance(s);
        if(s<-10000) {
        System.out.println("The balance can be negative but no less than -10000");
        }
        else {
            System.out.println(s);
        }
    }
}
class Main {
    public static void main(String [] args) {
    Scanner sc = new Scanner(System.in);
    VIPAccount va = new VIPAccount();
    va.AccountId = Integer.parseInt(sc.nextLine());
    va.HolderName = sc.nextLine();
    va.balance = Integer.parseInt(sc.nextLine());
    VIPAccount va1 = new VIPAccount(va.AccountId,va.HolderName,va.balance);
    System.out.println(va1.AccountId+" "+va1.HolderName+" "+va1.balance);
    int amount = Integer.parseInt(sc.nextLine());
```

```
    System.out.println(va1.credit(amount));
    int sal = Integer.parseInt(sc.nextLine());
    va1.setAccountBalance(sal);
    }
}
```

Q4

Test Case

Input

```
S
8
```

Output

```
Perimeter : 32.00
Area : 64.00
```

Weightage - 25

Input

```
R
10
15
```

Output

```
Perimeter : 50.00
Area : 150.00
```

Weightage - 25

Input

```
C
10
```

Output

```
Circumference : 62.83
Area : 314.16
```

Weightage - 25

Input

```
T
9
10
11
```

Output

```
Perimeter : 30.00
Area : 42.43
```

Weightage - 25

Sample Input

```
S
5
```

Sample Output

```
Perimeter : 20.00
Area : 25.00
```

Sample Input

```
R
3
4
```

Sample Output

```
Perimeter : 14.00
Area : 12.00
```

Sample Input

```
C
7
```

Sample Output

```
Circumference : 43.98
Area : 153.94
```

Sample Input

Sample Output

T 3 4 5	Perimeter : 12.00 Area : 6.00

Solution

```
import java.util.Scanner;
abstract class Shape {
    private double length, width; //square and rectangle
    private double radius; //circle
    private double side1, side2, side3; //triangle

    //set length and width
    public void setLengthAndWidth(double l, double w){
        length = l;
        width = w;
    }

    //return length
    public double getLength(){
        return length;
    }

    //return width
    public double getWidth(){
        return width;
    }

    //set radius
    public void setRadius(double r){
        radius = r;
    }

    //return radius
    public double getRadius(){
        return radius;
    }


    //set sides of triangle
    public void setSides(double s1, double s2, double s3){
        side1 = s1;
        side2 = s2;
        side3 = s3;
    }

    //return side1
    public double getSide1(){
        return side1;
    }

    //return side2
    public double getSide2(){
        return side2;
    }

    //return side3
    public double getSide3(){
        return side3;
    }


    public abstract String toString();  //return String representation of the class
    public abstract double area();  //calculate area
```

```
    public abstract double perimeter(); //calculate perimeter
}

//class Circle
 class Circle extends Shape{
    public Circle(double radius){
        setRadius(radius);
    }

    public Circle(){
        //do nothing
    }

    //calculate area of circle
    @Override
    public double area(){
        return Math.PI * Math.pow(getRadius(), 2);
    }

    //calculate circumference of circle
    @Override
    public double perimeter(){
        return 2 * Math.PI * getRadius();
    }


    //return String representation of area
    @Override
    public String toString() {
        return String.format("Circumference : %.2f\nArea : %.2f\n", perimeter(), area());
    }
}


 class Rectangle extends Shape{

    public Rectangle(double length, double width){
        setLengthAndWidth(length, width);
    }

    public Rectangle(){
        //do nothing
    }

    //calculate area
    @Override
    public double area(){
        return getLength() * getWidth();
    }

    //calculate perimeter
    @Override
    public double perimeter(){
        return 2 * (getLength() + getWidth());
    }

    //return String representation of area and perimeter
    @Override
    public String toString(){
        return String.format("Perimeter : %.2f\nArea : %.2f\n", perimeter(), area());
    }
}

 class Square extends Shape{
```

```
public Square(double l1, double l2){
    setLengthAndWidth(l1, l2);
}

public Square(){
    //do nothing
}

//calculate area
@Override
public double area(){
    return getLength() * getWidth();
}

//calculate perimeter
@Override
public double perimeter(){
    return 2 * (getLength() + getWidth());
}

//return string representation of perimeter and area
@Override
    public String toString(){
        return String.format("Perimeter : %.2f\nArea : %.2f\n",perimeter(),area());
    }
}

 class Triangle extends Shape{
    public Triangle(double side1, double side2, double side3){
        setSides(side1, side2, side3);
    }

    public Triangle(){
        //do nothing
    }

    @Override
    public double area(){
        double p = (1/2.0) * (getSide1() + getSide2() + getSide3());
        return Math.sqrt(p * (p-getSide1()) * (p-getSide2()) * (p-getSide3()));
    }

    //calculate perimeter
    @Override
    public double perimeter(){
        return getSide1() + getSide2() + getSide3();
    }

    //return String representation of area and perimeter
    @Override
    public String toString(){
        return String.format("Perimeter : %.2f\nArea : %.2f", perimeter(), area());
    }
}

 class Main {
    public static void main(String[] args){
        Scanner input = new Scanner(System.in);
        double radius, length, width;

        //System.out.println("Select a shape");
        //System.out.print("1.Square\n2.Rectangle\n3.Circle\n4.Triangle\n? ");
        char choice = input.nextLine().charAt(0);
        //System.out.println(choice);
        switch(choice){
```

```
case 'S':case 's':
    //System.out.println("Enter the length of one side of the square");
    length = Double.parseDouble(input.nextLine());
    Square square = new Square(length, length);
    System.out.println(square.toString());
    break;

case 'R': case 'r':
//  System.out.println("Enter the length and width of the rectangle");
//  System.out.print("Length : ");
    length = Double.parseDouble(input.nextLine());
//  System.out.print("Width : ");
    width = Double.parseDouble(input.nextLine());
    Rectangle rectangle = new Rectangle(length, width);
    System.out.println(rectangle.toString());
    break;

case 'C': case 'c':
//  System.out.println("Enter radius of the circle");
    radius = Double.parseDouble(input.nextLine());
    Circle circle = new Circle(radius);
    System.out.println(circle.toString());
```

```
                break;

            case 'T': case't':
            //  System.out.println("Enter the length of the sides of the triangle");
            //  System.out.print("side1: ");
                double s1 = Double.parseDouble(input.nextLine());
            //  System.out.print("side2: ");
                double s2 = Double.parseDouble(input.nextLine());
            //  System.out.print("side3: ");
                double s3 = Double.parseDouble(input.nextLine());
                Triangle triangle = new Triangle(s1, s2, s3);
                System.out.println(triangle.toString());
                break;
            default:
            //  System.out.println("Invalid choice");
        }
        input.close();

    }
}
```

Q5

Test Case

Input	Output
2468	60

Weightage - 10

Input	Output
567	36

Weightage - 10

Input

```
1190
```

Output

```
33
```

Weightage - 10

Input

```
8743
```

Output

```
66
```

Weightage - 10

Input

```
9086
```

Output

```
69
```

Weightage - 10

Input

```
685
```

Output

```
38
```

Weightage - 20

Input

```
76509
```

Output

```
108
```

Weightage - 20

Input

```
7653
```

Output

```
63
```

Weightage - 10

Sample Input

```
1234
```

Sample Output

```
30
```

Sample Input

```
4356
```

Sample Output

```
54
```

Solution

```
import java.util.Scanner;
class Parent1
{
    void fun(int n)
    {
       int i,j,k=0,sum=0;
       int a[]=new int[10];
       while(n!=0)
       {
       i=n%10;
       a[k++]=i;
       n=n/10;
       }
       if(k==1){
       System.out.print(a[0]);
       return;
       }
       for(i=0;i<k-1;i++)
       for(j=i+1;j<k;j++)
       sum=sum+a[i]+a[j];
       System.out.print(sum);
    }
    }

class Main extends Parent1
{
   public static void main(String args[])
   {
   int n;
   Scanner sc=new Scanner(System.in);
   n=sc.nextInt();
       Main t = new Main();
       t.fun(n);
   }
}
```