RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM 602 105

RAJALAKSHMI ENGINEERING COLLEGE

CS23333 OOPS Using Java

Laboratory Record Note Book

Name :	Keerthisri . D
Year / Branch / Section :	IInd year {CSE (cyber security)}
University Register No. :	2116241901047
College Roll No. :	241901047
Semester :	IIIrd
Academic Year :	2025 - 2026 .

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)

CS23333 Object Oriented Programming Using Java

REG NO	211624190104[illegible]
NAME	KEERTHUSRI . D
YEAR	IInd .
SEC	A

RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution

BONAFIDE CERTIFICATE

Name: ..D. keerthisri..

Academic Year: 2025 - 2026... **Semester:** ....II nd....... **Branch:** CSE - (cyber security)

Register No. 211624901047

Certified that this is the bonafide record of work done by the above student in the......OOPS......Using......Java..................*Laboratory during the academic year 2025- 2026*

[illegible] 18/11/25

Signature of Faculty in-charge

Submitted for the Practical Examination held on................................

Internal Examiner **External Examiner**

INDEX

EX.NO	DATE	NAME OF THE EXPERIMENT	GITHUB QR
1	16/07/2025	I/O, Data Types, Operators	github.com/keerthi418/java_observation.
2	28/07/2025.	Control Structures	
3	30/07/2025	Arrays	
4	4/08/25	Strings	
5	11/8/25	Classes & Objects	
6	18/8/25	Inheritance	
7	25/8/25	Interface	
8	8/9/25.	Exceptions	
9	8/9/25	Collections	
10	29/9/25	Collections	
11	13/10/25	Project	
12	29/10/25.	Lambda	

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 15

Section 1 : MCQ

1. What will be the output of the following code?

```
import java.util.*;

class TernaryOperatorExample {
   public static void main(String[] args) {
      int a = 5, b = 10;
      int result = (a > b) ? a : b;
      System.out.println(result);
   }
}
```

Answer

10

Status : Correct *Marks : 1/1*

2. Which of the following data types is used to store single characters?

Answer

char

Status : Correct ***Marks : 1/1***

3. Which of the following is not a primitive data type?

Answer

string

Status : Correct ***Marks : 1/1***

4. What is the output of the following code?

```
import java.util.*;

class RelationalOperatorExample {
   public static void main(String[] args) {
      int x = 8, y = 4;
      boolean result = (x != y);

      System.out.println(result);
   }
}
```

Answer

true

Status : Correct ***Marks : 1/1***

5. What will be the output of the following code snippet?

```
class DivisionExample {
   public static void main(String[] args) {
      double num1 = 10.5;
      double num2 = 3;
```

```
        int result = (int)(num1 / num2);
        System.out.println(result);
    }
}
```

Answer

3

Status : Correct ***Marks : 1/1***

6. What is the output of the following code?

```
class TestClass {
    public static void main(String[] args) {
        int a = 10;
        int b = 3;
        System.out.println(a / b);
    }
}
```

Answer

3

Status : Correct ***Marks : 1/1***

7. What is the output of the following code?

```
class TestClass {
    public static void main(String[] args) {
        int x = 5;
        int X = 10;

        int sum = x + X;
        int bitwiseResult = x | X;

        System.out.println(sum);
        System.out.println(bitwiseResult);
    }
}
```

Answer

1515

Status : Correct ***Marks : 1/1***

8. What will be the output of the following code snippet?

```
import java.util.*;

class OperatorPrecedenceExample {
    public static void main(String[] args) {
        int a = 5, b = 3, c = 2;
        int result = a + b * c;

        System.out.println(result);
    }
}
```

Answer

11

Status : Correct ***Marks : 1/1***

9. What will be the output of the following program?

```
class DataTypesMCQ {
    public static void main(String[] args) {
        int a = 10;
        double b = 5;
        System.out.println(a / b);
    }
}
```

Answer

2.0

Status : Correct ***Marks : 1/1***

10. What is the output of the following code?

```
class TestClass {
   public static void main(String[] args) {
      int a = 5;
      int b = 10;

      int sum = a + b;
      int bitwiseAnd = a & b;
      int bitwiseOr = a | b;

      System.out.println(sum);
      System.out.println(bitwiseAnd);
      System.out.println(bitwiseOr);
   }
}
```

Answer

15015

Status : Correct ***Marks : 1/1***

11. What is the result of the following expression?

```
import java.util.*;

class ComplexExpressionExample {
   public static void main(String[] args) {
      int a = 5, b = 2, c = 3, d = 4;
      int result = a + b * c / d - b;

      System.out.println(result);
   }
}
```

Answer

4

Status : Correct ***Marks : 1/1***

12. What is the output of the following program?

```
class Arithmetic {
   public static void main(String[] args) {
      char ch = 'A';
      System.out.println(ch);
   }
}
```

Answer

A

Status : Correct ***Marks : 1/1***

13. What is the output of the following program?

```
class Demo {
   public static void main(String[] args) {
      String text = "Hello, World!";
      System.out.println(text);
   }
}
```

Answer

Hello, World!

Status : Correct ***Marks : 1/1***

14. What is the output of the following code?

```
class TestClass {
   public static void main(String[] args) {
      int count = 8;
      count = count ^ 1;

      System.out.println(count);
   }
}
```

Answer

9

Status : Correct *Marks : 1/1*

15. Which of the following data types is used to store floating-point numbers with greater precision?

Answer

double

Status : Correct *Marks : 1/1*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Gloria is responsible for monitoring the performance of two machines in a factory. She needs to determine which of the two machines is operating closest to the optimal temperature of 100 degrees Celsius using the relational operator.

Assist Gloria in displaying the machine's temperature, which is closer to 100, and the difference from 100.

Input Format

The first line of input consists of an integer N, representing the temperature of the first machine.

The second line consists of an integer M, representing the temperature of the second machine.

Output Format

The output prints "The integer closer to 100 is X with a difference of Y" where X is the temperature of the closer machine and Y is the difference from 100.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 90
80
Output: The integer closer to 100 is 90 with a difference of 10

Answer

```
import java.util.Scanner;

public class Main{
   public static void main(String[] args){
      Scanner sc=new Scanner(System.in);
      int N=sc.nextInt();
      int M=sc.nextInt();

      int d1=Math.abs(100-N);
      int d2=Math.abs(100-M);
      if(d1<=d2){
         System.out.println("The integer closer to 100 is"+N+"with a difference
of"+d1);
      }
      else{
         System.out.println("The integer closer to 100 is"+M+"with a difference
of"+d2);
      }
      sc.close();
   }

}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. PROBLEM STATEMENT:

Dave got two students who wants help with their doubt. Each handouts an integer and wants to find if one Integer Positive While the Other is Not Divisible by 3. Write a program to achieve this and conclude for them.

Input Format

The first line of input represents the first integer.

The second line of input represents the second integer.

Output Format

The output should display as "One of the integers is positive while the other is not divisible by 3." or "Neither of the integers meets the condition."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 4
3
Output: One of the integers is positive while the other is not divisible by 3.

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String [] args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int m=sc.nextInt();

      boolean condition=(n>0 && m%3!=0)||(m>0 && n%3!=0);
      if(condition){
         System.out.println("One of the integers is positive while the other is not
divisible by 3.");
      }
      else{
         System.out.println("Neither of the integers meets the condition.");
      }
      sc.close();
   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem statement

Manoj, a developer at MoneyMatters Inc., is working on improving the company's financial system. He needs to create a program that takes an integer input, converts it into a double, and displays both the original integer and the converted double value.

Input Format

The input consists of a single integer representing a monetary amount.

Output Format

The first line of the output displays the "Original Integer: ", followed by an integer representation of the input value.

The second line displays the "Converted Double: ", followed by a double value representing the input as a decimal value.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 20

Output: Original Integer: 20
Converted Double: 20.0

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[] args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      double m=(double)n;
      System.out.println("Original Integer:"+n);
      System.out.println("Converted Double:"+m);
      sc.close();
   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Vishal and Arun are discussing the properties of numbers. Vishal gives Arun two integers. He asks Arun to check if the sum of these two numbers is a multiple of their product.

Can you assist Arun and determine whether the sum is a multiple of the product?

Input Format

The input consists of two space-separated integers.

Output Format

The output prints:

1. "Sum is Multiple of Product" if the sum of the two numbers is divisible by their product.
2. "Sum is Not Multiple of Product" otherwise.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1 2

Output: Sum is Not Multiple of Product

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        int n1=sc.nextInt();
        int n2=sc.nextInt();
        int n=n1+n2;
        int p=n1*n2;
        if(p!=0 && n%p==0){
            System.out.println("Sum is Multiple of Product");
        }
        else{
            System.out.println("Sum is not Multiple of Product");
        }
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement:

Emily has a beautiful circular garden in her backyard. She's interested in calculating two important measurements for her garden: the circumference and the area. To do this, she needs a program that can take the radius of her circular garden as input and provide the calculated circumference and area as output. The formulas she should use are as follows:

To calculate the circumference (C) of a circle, you can use the formula:

$C = 2 * \pi * r$

$A = \pi * r^2$

Where:

C represents the circumference.

A represents the area.

π (pi) is approximately 3.14159.

r is the radius of the circle.

Emily is not a programmer, and she needs your help to create a program that will make these calculations for her garden.

Input Format

The first line of input contains a single double-point number radius, representing the radius of the circle.

Output Format

The output should consist of two lines:

The first line should print the circumference of the circle rounded to 2 decimal places, followed by the unit "meters".

The second line should print the area of the circle rounded to 2 decimal places, followed by the unit "square meters".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3.0

Output: Circumference: 18.85 meters
Area: 28.27 square meters

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      double n=sc.nextDouble();
```

```
        double c=2*3.14159*n;
        double a=3.14159*Math.pow(n,2);

        System.out.println("Circumference: "+String.format("%.2f",c)+"meters");
        System.out.println("Area: "+String.format("%.2f",a)+" square meters");

        sc.close();

    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_PAH

Attempt : 2
Total Mark : 40
Marks Obtained : 39

Section 1 : Coding

1. Problem Statement

Mickey and Miney are walking through a magical forest. The forest is full of enchanted stones, each with a unique number. There is a legend that says the magic power of the stones can be revealed by using a special operation. To determine the magic power of a given stone, you need to perform a bitwise AND operation with the number 15.

Each stone's number is represented by an integer, and Mickey needs to find the magic power of each stone by applying this operation.

Your task is to help Mickey compute the result of the bitwise AND operation of the given stone number with 15, and print the result.

Input Format

The input consists of a single integer.

Output Format

The output should display a single integer, which is the result of the bitwise AND operation between input and 15.

Refer to the sample output for format specifications.

Sample Test Case

Input: 25

Output: 9

Answer

```
// You are using Java
import java.util.*;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int k=n&15;
      System.out.println(k);
   }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

In the Kingdom of Delivery Logistics, there is a giant truck used for transporting packages across the kingdom. The truck has a maximum capacity represented by an integer, and each package also has a specific weight. The truck’s efficiency and safety depend on whether the weight of the package is below a certain threshold.

The kingdom's delivery service has a rule: if the weight of a package is less than one-third of the truck’s total capacity, the package is eligible for quick processing and dispatch. However, if the weight is too heavy, the package will require special handling.

As a logistics manager, you need to check whether the weight of the package is less than one-third of the truck's total capacity.

Write a program using a ternary operator that helps determine whether the package weight meets the requirement for quick processing or if it needs special handling.

Input Format

The first line of input consists of an integer p, representing the weight of the package.

The second line consists of an integer w, representing the total weight capacity of the truck.

Output Format

The first line of output prints "One-third of Truck: X," where X is one-third of the truck's total weight capacity as a double value with two decimal places.

The second line of output displays one of the following:

1. If p is less than one-third of the truck's total weight capacity, print "Package weight is less than one-third of the truck's capacity".
2. Otherwise, print "Package weight is not less than one-third of the truck's capacity".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 13
60

Output: One-third of Truck: 20.00
Package weight is less than one-third of truck's capacity

Answer

// You are using Java

```
import java.util.*;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int m=sc.nextInt();
      double l=m/3.0;
      System.out.printf("One-third of Truck:%.2f\n",l);
      if(n<l){
         System.out.println("Package weight is less than one-third of truck's
capacity ");
      }
      else{
         System.out.println("Package weight is not less than one-third of truck's
capacity");
      }
   }
}
```

Status : Correct ***Marks : 10/10***

3. PROBLEM STATEMENT:

Maria, a software developer, is working on a project to create a simple program to determine which of two integers is closest to zero. The integers can be either positive or negative. The program needs to take two integer inputs and calculate which one is closer to zero. If both integers are equidistant from zero, the program should return 0.

Input Format

The input contains two lines:

The first line of the input contains an integer, which can be either a positive or a negative integer.

The second line of the input contains an integer, which can be either a positive or a negative integer.

Output Format

The output displays the integer that is closest to zero in the following format:

"The integer closest to zero is: [closest_integer]"

Here, [closest_integer] should be replaced with the integer that is closer to zero based on its absolute value.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
8

Output: The integer closest to zero is: 5

Answer

```
// You are using Java
import java.util.*;
public class Main{
   public static void main(String[]ags){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int m=sc.nextInt();
      int c=n;
      if(Math.abs(m)<Math.abs(c)){
         System.out.println("The integer closest to zero is:"+m);
      }
      else{
         System.out.println("The integer closest to zero is:"+n);
      }
   }
}
```

Status : Partially correct ***Marks : 9/10***

4. PROBLEM STATEMENT:

Maria, a software developer, is working on a program to determine if two given integers which can be either positive or negative integers have the same parity (both even or both odd). She needs your help in writing this program.

Write a program that takes two integers as input and checks if both integers are either even or odd.

Input Format

The input consists of two lines:

The first line consists of an integer (input1) which can be either positive or negative.

The second line consists of an integer (input2) which can be either positive or negative.

Output Format

The output is dispalyed in the following format:

If both integers have the same parity (i.e., both even or both odd), print:

"Both integers are either even or odd"

Otherwise, print:

"The integers have different parities"

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 2
-4
Output: Both integers are either even or odd

Answer

```
// You are using Java
import java.util.*;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int m=sc.nextInt();
      boolean k=((n%2==0)&&(m%2==0))||((n%2!=0)&&(m%2!=0));
      if(k){
         System.out.println("Both integers are either even or odd ");

      }
      else{
         System.out.println("The integers have different parities");
      }
   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement:

Tom is tasked with writing a program that determines whether a given integer is the square of another integer. A perfect square is a number that can be expressed as the square of an integer. The program should take an integer as input and determine if it is a perfect square or not.

The task is to implement the logic to check if the provided integer is the square of an integer and return the result.

Input Format

The first line of the input contains an integer, "input", where |input| represents the absolute value of the integer.

Output Format

The output should display a boolean value, "result," which should be set to true if the input is a perfect square (the square of an integer), and false if it is not.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 16
Output: Is the integer a perfect square? true

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      double sqrt=Math.sqrt(n);
      if(sqrt==(int)sqrt){
         System.out.println("Is the integer a perfect square? true");
      }
      else{
         System.out.println("Is the integer a perfect square? false");
      }

   }
}
```

Status : Correct ***Marks : 10/10***

2. PROBLEM STATEMENT:

Jule a mathematician expert is given two integers to find if the second integer is above the average of the first and second integer. Write a program that achieves this using the ternary operator.

Input Format

The first line of input represents the first integer.

The second line of input represents the second integer.

Output Format

The output should be displayed as "Below Average" or "Above Average"

REFER THE SAMPLE TESTCASES FOR THE FORMAT SPECIFICATIONS.

Sample Test Case

Input: 1
1
Output: Below Average

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int m=sc.nextInt();
        double avg=(n+m)/2.0;
        if(m>avg){
            System.out.println("Above Average");
        }
        else{
            System.out.println("Below Average");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

In the faraway land of Arithmetica, there exists an ancient calculator that can only perform bitwise operations. The calculator is locked with a secret code that only works when the number is modified using a special operation called right shifting.

The ruler of Arithmetica, King Thales, needs your help to unlock the calculator. The lock on the calculator is encoded with a number, and the calculator will only open if you apply a right shift by 2 on the number. Your task is to help King Thales determine the magic number that will unlock the ancient calculator.

Input Format

The first line of input represents an integer.

Output Format

The output should display the right-shifted value by 2 bits.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 16

Output: 4

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int k=n>>2;
      System.out.println(k);
   }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Mandy is a software engineer working on a program to analyze two integers based on specific conditions using a logical operator. She needs to determine if both integers are odd or if at least one of them is divisible by 7.

Depending on the result, she wants to print different messages.

If the condition is met, the program should identify and print the first number that is divisible by 7 or indicate that both numbers are odd.If the condition is not met, the program should print a message indicating the condition was not met, along with the input numbers.

Input Format

The first line of input consists of an integer representing the first input number.

The second line consists of an integer representing the second input number.

Output Format

The output displays "Condition met: " followed by an integer representing the first number divisible by 7, or prints "Both numbers are odd" if the two inputs are odd.

If the condition is not met, it displays "Conditions not met: " followed by the two input integers, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
14

Output: Condition met: 7

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int m=sc.nextInt();
        boolean odd=(n%2!=0)&&(m%2!=0);
        boolean div7=(n%7==0)||(m%7==0);
        if(div7){
            if(n%7==0){
                System.out.println("Condition met: "+n);
            }
            else{
                System.out.println("Condition met: "+m);
            }
        }
        else if(odd){
            System.out.println("Condition met:Both numbers are odd");
        }
        else{
            System.out.println("Conditions not met: "+n+" "+m);
        }
    }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 1

Section 1 : MCQ

1. What will be the output of the following code?

```
class ConditionTest {
  public static void main(String[] args) {
    int x = 10;
    if (x > 5)
      System.out.print("High");
  }
}
```

Answer

High

Status : Correct ***Marks : 1/1***

2. What will be the output of the following code?

```
public class Main {
   public static void main(String[] args) {
      for(int i = 1; i <= 20; i = i * 2) {
         System.out.print(i + " ");
      }
   }
}
```

Answer

Status : Skipped ***Marks : 0/1***

3. What will be the output of the following code?

```
public class Main {
   public static void main(String[] args) {
      int sum = 0;
      for(int i = 1; i <= 5; i++) {
         sum += i;
      }
      System.out.println(sum);
   }
}
```

Answer

Status : Skipped ***Marks : 0/1***

4. What will be the output of the following code?

```
public class Main {
   public static void main(String[] args) {
      int i = 10;
      do {
         System.out.print(i + " ");
         i -= 3;
      } while(i > 0);
```

```
    }
}
```

Answer

-

Status : - ***Marks : 0/1***

5. What will be the output of the following code?

```
class LoopTest {
    public static void main(String[] args) {
        int i = 1;
        while (i > 0) {
            System.out.print(i + " ");
            i++;
            if (i == 5) break;
        }
    }
}
```

Answer

Status : Skipped ***Marks : 0/1***

6. What will be the output of the following code?

```
class ConditionTest {
    public static void main(String[] args) {
        int a = 7;
        if (a == 7)
            System.out.print("Match");
        else
            System.out.print("No Match");
    }
}
```

Answer

-

Status : - *Marks : 0/1*

7. What will be the output of the following code?

```
class Test {
   public static void main(String[] args) {
      int a = 4, b = 5;
      if ((a + b) % 2 == 0)
         System.out.print("Even");
      else
         System.out.print("Odd");
   }
}
```

Answer

-

Status : - *Marks : 0/1*

8. What will be the output of the following Java code snippet?

```
public class Main {
   public static void main(String[] args) {
      int day = 4;
      String result = "";
      switch(day) {
         case 1:
            result = "Monday";
            break;
         case 2:
            result = "Tuesday";
            break;
         case 3:
            result = "Wednesday";
            break;
         default:
            result = "Other Day";
      }
```

```
        System.out.println(result);
    }
}
```

Answer

-

Status : **-** ***Marks : 0/1***

9. What will be the output of the following Java code snippet?

```
public class Main {
    public static void main(String[] args) {
        int score = 75;
        if(score >= 90) {
            System.out.println("Grade: A");
        } else if(score >= 80) {
            System.out.println("Grade: B");
        } else if(score >= 70) {
            System.out.println("Grade: C");
        } else {
            System.out.println("Grade: D");
        }
    }
}
```

Answer

-

Status : **-** ***Marks : 0/1***

10. What will be the output of the following code?

```
class Test {
    public static void main(String[] args) {
        int num = 15;
        if (num > 10)
            if (num % 3 == 0)
                System.out.print("Divisible");
```

```
      else
         System.out.print("Not Divisible");
   }
}
```

Answer

-

Status : **-** ***Marks : 0/1***

11. What will be the output of the following code?

```
public class Main {
   public static void main(String[] args) {
      int i = 1;
      while(i < 10) {
         i += 2;
      }
      System.out.println(i);
   }
}
```

Answer

-

Status : **-** ***Marks : 0/1***

12. What will be the output of the following code?

```
class Main {
   public static void main(String[] args) {
      for (int i = 5; i > 0; i--) {
         System.out.print(i + " ");
      }
   }
}
```

Answer

-

Status : *-* ***Marks : 0/1***

13. What will be the output of the following code?

```
class Loop {
   public static void main(String[] args) {
      for (int i = 1; i <= 3; i++) {
         for (int j = 1; j <= 2; j++) {
            System.out.print(i + "" + j + " ");
         }
      }
   }
}
```

Answer

-

Status : *-* ***Marks : 0/1***

14. What will be the output of the following code?

```
class Test {
   public static void main(String[] args) {
      int x = 5, y = 2;
      if (x + y == 10)
         System.out.print("Ten");
      else if (x - y == 3)
         System.out.print("Three");
      else
         System.out.print("None");
   }
}
```

Answer

-

Status : *-* ***Marks : 0/1***

15. What will be the output of the following code?

```
class LoopTest {
   public static void main(String[] args) {
      int i = 1;
      do {
         System.out.print(i + " ");
         i *= 2;
      } while (i <= 8);
   }
}
```

Answer

Status : Skipped ***Marks : 0/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Arun is working on a project to automate the process of determining whether a student has passed or failed based on their subject marks.

He aims to create a simple program that takes positive integers as marks for five subjects from the user. If the average of the marks is greater than or equal to 50, the student has passed the exam. Otherwise, the student has failed.

Help Arun to implement the project.

Input Format

The input consists of five space-separated integers, representing the marks in five subjects.

Output Format

The first line of output prints "Average score: " followed by an integer representing the average score.

The second line prints one of the following:

1. If the condition is satisfied, print "The student has passed".
2. Otherwise, the output prints "The student has failed".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 50 60 70 80 90

Output: Average score: 70
The student has passed

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int a=sc.nextInt();
      int b=sc.nextInt();
      int c=sc.nextInt();
      int d=sc.nextInt();
      int e=sc.nextInt();
      int avg=(a+b+c+d+e)/5;
      System.out.println("Average score: "+avg);
      if(avg>=50){
         System.out.println("The student has passed");
      }
      else{
         System.out.println("The student has failed");
      }
   }
}
```

Status : Correct Marks : 10/10

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Samantha is a diligent math student who is exploring the world of programming. She is learning Java and has recently studied conditional statements. One day, her teacher gives her an interesting problem to solve, which takes a number as input and checks whether it is a multiple of 5 or 7.

Help her complete the task.

Input Format

The input consists of a single integer N, representing the number to be checked.

Output Format

If the number is a multiple of 5 but not 7, the output prints "N is a multiple of 5".

If the number is a multiple of 7, the output prints "N is a multiple of 7".

Otherwise the output prints "N is neither multiple of 5 nor 7" where N is an entered integer.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10
Output: 10 is a multiple of 5

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        int a=sc.nextInt();
        if(a%5==0){
            System.out.println(a+" is a multiple of 5");
        }
        else if(a%7==0){
            System.out.println(a+" is a multiple of 7");
        }
        else{
            System.out.println(a+" is neither multiple of 5 nor 7");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

John is a fitness trainer, and he wants to use the BMI calculator to assess the body mass index of his clients. He has a list of clients based on their height and weight.

John plans to write a program to quickly determine the BMI and provide a classification for each client.

If BMI is less than 18.5, the program will classify it as "Underweight"If BMI is between 18.6 and 24.9, the program will classify it as "Normal Weight"If BMI is between 25.0 and 29.9, the program will classify it as "Overweight"If BMI is 30.0 or higher, the program will classify it as "Obese"

Note: Formula to calculate BMI = weight/(height*height)

Input Format

The first line of input consists of a double value, representing the height of the person in meters.

The second line consists of a double value, representing the weight of the person in kilograms.

Output Format

The first line of output prints "BMI: " followed by a double (rounded to two decimal places) representing the calculated BMI.

The second line prints "Classification: " followed by a string indicating the BMI category (Underweight, Normal Weight, Overweight, or Obese).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1.2
45.2

Output: BMI: 31.39
Classification: Obese

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        double a=sc.nextDouble();
        double b=sc.nextDouble();
        double bmi=b/(a*a);
        System.out.println("BMI: "+String.format("%.2f",bmi));
        if(bmi<18.5){
            System.out.println("Classification: Underweight");
        }
        else if(bmi>=30.0){
            System.out.println("Classification: Obese");
        }
        else if(bmi>18.6 && bmi<24.9){
            System.out.println("Classification : Normal Weight");
        }
        else if(bmi>25.0 && bmi<29.9){
            System.out.println("Classification : Overweight");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Amit wants to evaluate the depreciation of his car over time to understand its current value and categorize it based on that value.

Write a program that helps him determine the current value of his car after a certain number of years of depreciation and classify it into one of three categories:

High: If the current value is greater than 10,000.Medium: If the current value is between 5,000 and 10,000, both inclusive.Low: If the current value is less than 5,000.

The depreciation rate of the car is 15% per year. The program should calculate the current value of the car after applying this depreciation over the given number of years and print the current value along with the category.

Input Format

The first line of input consists of an integer, representing the initial cost of the car.

The second line consists of an integer, representing the number of years the car has been depreciating.

Output Format

The first line of output prints a double value, representing the current value of the car, rounded off to two decimal places "Current Value: <value>".

The second line prints its category "Category: <categories>".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 20000
5

Output: Current Value: 8874.11
Category: Medium

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      double a=sc.nextDouble();
      int b=sc.nextInt();
      double c=a;
      for(int i=0;i<b;i++){
         c=c*0.85;
      }

      System.out.println("Current Value: "+String.format("%.2f",c));
      if(c>10000){
         System.out.println("Category: High");
      }
```

```
        else if(c>=5000 && c<=10000 ){
            System.out.println("Category: Medium");
        }
        else{
            System.out.println("Category: Low");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

You are given a number of distribution centers (rows) and are tasked with generating a zigzag shipment route pattern. Each shipment route should alternate between left-to-right and right-to-left, as described below.

The program should print the zigzag pattern with a tab (\t) separating the columns. For each row, the shipment numbers should follow a diagonal pattern where numbers are placed in a zigzag, left to right on odd rows and right to left on even rows.

Input Format

The input consists of an integer N, which represents the number of distribution centers (rows) for the zigzag pattern.

Output Format

The output prints the zigzag pattern with N rows, formatted with a tab space (\t) separating the columns.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5

Output:

```
        1
      2   6
    3   7   10
  4   8   11   13
5   9   12   14   15
```

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int t;
        for(int i=1;i<=n;i++){
            for(int j=1;j<n;j++){
                System.out.print("\t");
            }
            t=i;
            for(int k=1;k<=i;k++){
                System.out.print(t+"\t\t");
                t=t+n-k;
            }
            System.out.println();
        }
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Sampad is a high school teacher who wants to convert numeric grades into letter grades.

Write a program that accepts a numeric grade as input. The program should then convert this numeric grade into a letter grade based on specific conditions. The letter grades are A, B, C, D and F.

The conversion is determined by the following conditions:

If the numeric grade is 90 or higher, it's an "A"If the numeric grade is between 80 and 89 (inclusive), it's a "B"If the numeric grade is between 70 and 79 (inclusive), it's a "C"If the numeric grade is between 60 and 69 (inclusive), it's a "D"If the numeric grade is below 60, it's an "F"

Input Format

The input consists of an integer representing the numeric grade of the student.

Output Format

The output prints the letter grade corresponding to the input numeric grade as "Letter Grade: <grade>".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 85

Output: Letter Grade: B

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[] args){

      Scanner sc=new Scanner(System.in);
      int a=sc.nextInt();
      if(a>=90){
          System.out.println("Letter Grade: A");
      }
      else if(a>=80 && a<=89){
          System.out.println("Letter Grade: B");
      }
      else if(a>=70 && a<=79){
          System.out.println("Letter Grade: C");
      }
      else if(a>=60 && a<=69){
          System.out.println("Letter Grade: D");
      }
      else{
          System.out.println("Letter Grade: F");
      }
   }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Rohit is tasked with designing a program to analyze the digits of a given integer.

Write a program to help Rohit that takes an integer as input and identifies the minimum odd digit and the maximum even digit present in the number. If no odd or even digits are present, display appropriate messages.

Implement the solution using a 'while' loop to iterate through the digits of the given number.

Input Format

The input consists of an integer n.

Output Format

The first line of output prints the message "Minimum odd digit: " followed by an integer representing the smallest odd digit found in the input number.

If no odd digit exists, it prints "There are no odd digits in the number."

The second line of output prints the message "Maximum even digit: " followed by an integer representing the largest even digit found in the input number.

If no even digit exists, it prints "There are no even digits in the number."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3465

Output: Minimum odd digit: 3
Maximum even digit: 6

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int minOdd=10;
        int maxEven=-1;
        while(n>0){
            int d=n%10;
            if(d%2==0){
                if(d>maxEven){
                    maxEven=d;
                }
            }
            else{
                if(d<minOdd){
                    minOdd=d;
                }
            }
            n=n/10;
        }

        if(minOdd==10){
            System.out.println("There are no odd digits in the number.");
        }else{
            System.out.println("Minimum odd digit: "+minOdd);
        }
        if(maxEven==-1){
            System.out.println("There are no even digits in the number.");
        }
        else{
            System.out.println("Maximum even digit: "+maxEven);
        }

    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Ravi wants to estimate the total utility bill for a household based on the consumption of electricity, water, and gas.

Write a program to calculate the total bill using the following criteria:

The cost per unit for electricity is 0.12, for water is 0.05, and for gas is 0.08.A discount is applied to the total cost based on the following conditions:If the total cost is 100 or more, a 10% discount is applied.If the total cost is between 50 and 99.99, a 5% discount is applied.No discount is applied if the total cost is less than 50.

The program should output the total bill after applying the discount with two decimal places.

Input Format

The input consists of three double values, representing the number of units consumed for electricity, water, and gas respectively.

Output Format

The output prints a double value, representing the total bill after applying the discount, formatted to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1000.0
200.0
100.0

Output: 124.20

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      float n=sc.nextFloat();
      float m=sc.nextFloat();
      float k=sc.nextFloat();
      double o=(n*0.12)+(m*0.05)+(k*0.08);
      if(o>=100){
         double i=o-(o*0.10);
         System.out.println(String.format("%.2f",i));
      }
      else if(o>=50){
         double j=o-(o*0.05);
         System.out.println(String.format("%.2f",j));
      }
      else{
         System.out.println(String.format("%.2f",o));

      }

   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Ram wants to evaluate the time required to break even on an investment based on initial costs, monthly profits, and monthly expenses. Write a program to calculate the break-even point in months and categorize the return on investment.

Compute the break-even point by using the formula: initial cost / (monthly profit - monthly expenses)Based on the break-even point, classify the return on investment into one of the following categories:Quick Return: If the break-even point is 3 months or fewer.Average Return: If the break-even point is between 4 and 12 months, inclusive.Long-term Return: If the break-even point exceeds 12 months.

Ram is new to programming, so he seeks your assistance in creating the program.

Note: monthly profit is always greater than monthly expenses.

Input Format

The first line of input consists of a double value representing the initial cost.

The second line consists of a double value representing the monthly profit.

The third line consists of a double value representing the monthly expenses.

Output Format

The first line prints "Break-even Point:", followed by the break-even point as a decimal number (of double datatype), formatted to two decimal places.

The second line prints "Category: ",followed by the investment return as a String, which can be one of:

- "Quick Return" if break-even point ≤ 3
- "Average Return" if break-even point ≤ 12
- "Long-term Return" if break-even point > 12

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10000.50
5000.75
1000.10

Output: Break-even Point: 2.50
Category: Quick Return

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      double n=sc.nextDouble();
      double m=sc.nextDouble();
```

```
        double o=sc.nextDouble();
        double k=n/(m-o);
        System.out.println("Break-even Point: "+String.format("%.2f",k));
        if(k<=3){
            System.out.println("Category: Quick Return");
        }
        else if(k<=12){
            System.out.println("Category: Average Return");
        }
        else{
            System.out.println("Category: Long-term Return");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Noah is analyzing numbers within a given range [A, B] and wants to calculate a special sum. For each number in the range, he calculates the product of its odd digits (ignoring even digits). If the number contains no odd digits, it is skipped. The sum of these products for all numbers in the range is the result.

Write a program to compute this sum.

Example

Input:

10 12

Output:

3

Explanation:

For 10, odd digits = 1, product = 1.

For 11, odd digits = 1, 1, product = 1 * 1 = 1.

For 12, odd digits = 1, product = 1.

Total sum = 1 + 1 + 1 = 3

Input Format

The input consists of two space-separated integers A and B, representing the inclusive range boundaries.

Output Format

The output prints a single integer representing the sum of the products of odd digits for all numbers in the range.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10 12

Output: 3

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int a=sc.nextInt();
      int b=sc.nextInt();
      int sum=0;
      for(int n=a;n<=b;n++){
         int t=n;
         int p=1;
         boolean hasOdd=false;
         while(t>0){
            int d=t%10;
            if(d%2==1){
               p*=d;
               hasOdd=true;
            }
            t/=10;
```

```
        }

      if(hasOdd){
         sum+=p;
      }
      }
      System.out.println(sum);
   }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Raj is solving a physics problem involving projectile motion, where he needs to calculate the time a ball hits the ground using a quadratic equation of the form $ax^2 + bx + c = 0$. Depending on the coefficients, the ball may hit the ground once, twice, or not at all in real time.

Help Raj find all real roots of the equation, if any.

Note: discriminant = b2 - 4ac

Input Format

The input consists of three space-separated doubles a, b, and c, representing the coefficients of the quadratic equation.

Output Format

If there are two real roots, print:

- "Two real solutions:"
- "Root1 = <value>"
- "Root2 = <value>"

If there is one real root, print:

- "One real solution:"
- "Root = <value>"

If there are no real roots, print:

- "There are no real solutions."

Note: values are rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1 6 9

Output: One real solution:
Root = -3.00

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      double a=sc.nextDouble();
      double b=sc.nextDouble();
      double c=sc.nextDouble();
      double g=b*b-4*a*c;
      if(g>0){
         double root1=(-b+Math.sqrt(g))/(2*a);
         double root2=(-b-Math.sqrt(g))/(2*a);
         System.out.println("Two real solutions: ");
         System.out.println("Root1 = "+String.format("%.2f",root1));
         System.out.println("Root2 = "+String.format("%.2f",root2));
      }
      else if(g==0)
      {
         double root=-b/(2*a);
         System.out.println("One real solution: ");
         System.out.println("Root = "+String.format("%.2f",root));
      }
      else{
         System.out.println("There are no real solutions.");
      }
```

```
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

John is a fitness trainer, and he wants to use the BMI calculator to assess the body mass index of his clients. He has a list of clients based on their height and weight.

John plans to write a program to quickly determine the BMI and provide a classification for each client.

If BMI is less than 18.5, the program will classify it as "Underweight"If BMI is between 18.6 and 24.9, the program will classify it as "Normal Weight"If BMI is between 25.0 and 29.9, the program will classify it as "Overweight"If BMI is 30.0 or higher, the program will classify it as "Obese"

Note: Formula to calculate BMI = weight/(height*height)

Input Format

The first line of input consists of a double value, representing the height of the person in meters.

The second line consists of a double value, representing the weight of the person in kilograms.

Output Format

The first line of output prints "BMI: " followed by a double (rounded to two decimal places) representing the calculated BMI.

The second line prints "Classification: " followed by a string indicating the BMI category (Underweight, Normal Weight, Overweight, or Obese).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1.2
45.2

Output: BMI: 31.39
Classification: Obese

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        float a=sc.nextFloat();
        float b=sc.nextFloat();
        float c=b/(a*a);
        System.out.println("BMI: "+String.format("%.2f",c));
        if(c>=30.0){
            System.out.println("Classification: obese");
        }
        else if(c>=18.6 && c<=24.9){
            System.out.println("Classification: Normal Weight");
        }
        else if(c>=25.0 && c<=29.9){
            System.out.println("Classification: Overweight");
        }
        else if(c<18.5){
            System.out.println("Classification: Underweight");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Ram wants to evaluate the time required to break even on an investment based on initial costs, monthly profits, and monthly expenses. Write a program to calculate the break-even point in months and categorize the return on investment.

Compute the break-even point by using the formula: initial cost / (monthly profit - monthly expenses)Based on the break-even point, classify the return on investment into one of the following categories:Quick Return: If the break-even point is 3 months or fewer.Average Return: If the break-even point is between 4 and 12 months, inclusive.Long-term Return: If the break-even point exceeds 12 months.

Ram is new to programming, so he seeks your assistance in creating the program.

Note: monthly profit is always greater than monthly expenses.

Input Format

The first line of input consists of a double value representing the initial cost.

The second line consists of a double value representing the monthly profit.

The third line consists of a double value representing the monthly expenses.

Output Format

The first line prints "Break-even Point:", followed by the break-even point as a decimal number (of double datatype), formatted to two decimal places.

The second line prints "Category: ",followed by the investment return as a String, which can be one of:

- "Quick Return" if break-even point ≤ 3
- "Average Return" if break-even point ≤ 12
- "Long-term Return" if break-even point > 12

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10000.50
5000.75
1000.10

Output: Break-even Point: 2.50
Category: Quick Return

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      double n=sc.nextDouble();
      double m=sc.nextDouble();
```

```
        double o=sc.nextDouble();
        double k=n/(m-o);
        System.out.println("Break-even Point: "+String.format("%.2f",k));
        if(k<=3){
            System.out.println("Category: Quick Return");
        }
        else if(k<=12){
            System.out.println("Category: Average Return");
        }
        else{
            System.out.println("Category: Long-term Return");
        }
    }
}
```

Status : Correct **_Marks : 10/10_**

2. Problem Statement

Noah is analyzing numbers within a given range [A, B] and wants to calculate a special sum. For each number in the range, he calculates the product of its odd digits (ignoring even digits). If the number contains no odd digits, it is skipped. The sum of these products for all numbers in the range is the result.

Write a program to compute this sum.

Example

Input:

10 12

Output:

3

Explanation:

For 10, odd digits = 1, product = 1.

For 11, odd digits = 1, 1, product = 1 * 1 = 1.

For 12, odd digits = 1, product = 1.

Total sum = 1 + 1 + 1 = 3

Input Format

The input consists of two space-separated integers A and B, representing the inclusive range boundaries.

Output Format

The output prints a single integer representing the sum of the products of odd digits for all numbers in the range.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10 12

Output: 3

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int a=sc.nextInt();
      int b=sc.nextInt();
      int sum=0;
      for(int n=a;n<=b;n++){
         int t=n;
         int p=1;
         boolean hasOdd=false;
         while(t>0){
            int d=t%10;
            if(d%2==1){
               p*=d;
               hasOdd=true;
            }
            t/=10;
```

```
        }

      if(hasOdd){
         sum+=p;
      }
      }
      System.out.println(sum);
   }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Raj is solving a physics problem involving projectile motion, where he needs to calculate the time a ball hits the ground using a quadratic equation of the form $ax^2 + bx + c = 0$. Depending on the coefficients, the ball may hit the ground once, twice, or not at all in real time.

Help Raj find all real roots of the equation, if any.

Note: discriminant = b2 - 4ac

Input Format

The input consists of three space-separated doubles a, b, and c, representing the coefficients of the quadratic equation.

Output Format

If there are two real roots, print:

- "Two real solutions:"
- "Root1 = <value>"
- "Root2 = <value>"

If there is one real root, print:

- "One real solution:"
- "Root = <value>"

If there are no real roots, print:

- "There are no real solutions."

Note: values are rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1 6 9

Output: One real solution:
Root = -3.00

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        double a=sc.nextDouble();
        double b=sc.nextDouble();
        double c=sc.nextDouble();
        double g=b*b-4*a*c;
        if(g>0){
            double root1=(-b+Math.sqrt(g))/(2*a);
            double root2=(-b-Math.sqrt(g))/(2*a);
            System.out.println("Two real solutions: ");
            System.out.println("Root1 = "+String.format("%.2f",root1));
            System.out.println("Root2 = "+String.format("%.2f",root2));
        }
        else if(g==0)
        {
            double root=-b/(2*a);
            System.out.println("One real solution: ");
            System.out.println("Root = "+String.format("%.2f",root));
        }
        else{
            System.out.println("There are no real solutions.");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

John is a fitness trainer, and he wants to use the BMI calculator to assess the body mass index of his clients. He has a list of clients based on their height and weight.

John plans to write a program to quickly determine the BMI and provide a classification for each client.

If BMI is less than 18.5, the program will classify it as "Underweight"If BMI is between 18.6 and 24.9, the program will classify it as "Normal Weight"If BMI is between 25.0 and 29.9, the program will classify it as "Overweight"If BMI is 30.0 or higher, the program will classify it as "Obese"

Note: Formula to calculate BMI = weight/(height*height)

Input Format

The first line of input consists of a double value, representing the height of the person in meters.

The second line consists of a double value, representing the weight of the person in kilograms.

Output Format

The first line of output prints "BMI: " followed by a double (rounded to two decimal places) representing the calculated BMI.

The second line prints "Classification: " followed by a string indicating the BMI category (Underweight, Normal Weight, Overweight, or Obese).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1.2
45.2

Output: BMI: 31.39
Classification: Obese

Answer

```
// You are using Java
import java.util.Scanner;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        float a=sc.nextFloat();
        float b=sc.nextFloat();
        float c=b/(a*a);
        System.out.println("BMI: "+String.format("%.2f",c));
        if(c>=30.0){
            System.out.println("Classification: obese");
        }
        else if(c>=18.6 && c<=24.9){
            System.out.println("Classification: Normal Weight");
        }
        else if(c>=25.0 && c<=29.9){
            System.out.println("Classification: Overweight");
        }
        else if(c<18.5){
            System.out.println("Classification: Underweight");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 3_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 11

Section 1 : MCQ

1. What will be the output of the following code?

```
class Q {
   public static void main(String[] args) {
      int[] a = {1, 2, 3, 4};
      for (int i = 0; i < a.length; i++) {
         if (a[i] % 2 == 0)
            a[i] = 0;
      }
      System.out.println(a[1] + " " + a[3]);
   }
}
```

Answer

0 0

Status : Correct ***Marks : 1/1***

2. What will be the output of the following code?

```
 class Sample {
    public static void main(String[] args) {
        int[][] data = {
            {1, 2},
            {3, 4}
        };
        int sum = 0;

        for (int[] row : data) {
            for (int val : row) {
                sum += val;
            }
        }

        System.out.println("Sum = " + sum);
    }
}
```

Answer

Sum = 10

Status : Correct ***Marks : 1/1***

3. What will be the output of the following code?

```
class Q {
    public static void main(String[] args) {
        int[][] a = {
            {1, 2},
            {3, 4}
        };
        for (int i = 0; i < a.length; i++) {
            for (int j = 0; j < a[0].length; j++) {
                System.out.print(a[i][j] + " ");
```

```
      }
    }
  }
}
```

Answer

1 2 3 4

Status : Correct ***Marks : 1/1***

4. What will be the output of the following code?

```
class Sample {
  public static void main(String[] args) {
    int[] a = {1, 2, 3};
    int product = 1;
    for (int i = 0; i < a.length; i++) {
      product *= a[i];
    }
    System.out.println(product);
  }
}
```

Answer

6

Status : Correct ***Marks : 1/1***

5. What will be the output of the following code?

```
class ReverseArray {
  public static void main(String[] args) {
    int[] a = {1, 2, 3, 4};
    for (int i = 0; i < a.length / 2; i++) {
      int temp = a[i];
      a[i] = a[a.length - 1 - i];
      a[a.length - 1 - i] = temp;
    }
    for (int i : a)
```

```
        System.out.print(i + " ");
    }
}
```

Answer

4 3 2 1

Status : Correct ***Marks : 1/1***

6. What will be the output of the following code?

```
class Q {
    public static void main(String[] args) {
        int[] a = {1, 2, 3, 4};
        for (int i = 0; i < a.length / 2; i++) {
            int temp = a[i];
            a[i] = a[a.length - 1 - i];
            a[a.length - 1 - i] = temp;
        }
        System.out.println(a[0]);
    }
}
```

Answer

2

Status : Wrong ***Marks : 0/1***

7. What will be the output of the following code?

```
class Q {
    public static void main(String[] args) {
        int[] nums = {4, 2, 9, 5};
        int max = nums[0];
        for (int i = 1; i < nums.length; i++) {
            if (nums[i] > max)
                max = nums[i];
        }
        System.out.println(max);
    }
}
```

Answer

9

Status : Correct ***Marks : 1/1***

8. What will be the output of the following code?

```
class Sample {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6}
        };
        System.out.println(matrix[1][2]);
    }
}
```

Answer

6

Status : Correct ***Marks : 1/1***

9. What will be the output of the following code?

```
class Q {
    public static void main(String[] args) {
        int[][] arr = {
            {5, 6, 7},
            {8, 9, 10}
        };
        System.out.println(arr[0][2]);
    }
}
```

Answer

10

Status : Wrong *Marks : 0/1*

10. What will be the output of the following code?

```
public class Test {
   public static void main(String[] args) {
      int[] x = {4, 8, 12};
      int result = x[0] * x[2];
      System.out.println(result);
   }
}
```

Answer

48

Status : Correct *Marks : 1/1*

11. What will be the output of the following code?

```
class Q {
   public static void main(String[] args) {
      int[] nums = {3, 6, 7, 2, 8};
      int sum = 0;
      for (int i = 0; i < nums.length; i++) {
         if (nums[i] % 2 == 0)
            sum += nums[i];
      }
      System.out.println(sum);
   }
}
```

Answer

10

Status : Wrong *Marks : 0/1*

12. What will be the output of the following code?

```
class Q {
    public static void main(String[] args) {
        int[] a = {1, 2, 1, 3, 1, 4};
        int count = 0;
        for (int i = 0; i < a.length; i++) {
            if (a[i] == 1) count++;
        }
        System.out.println(count);
    }
}
```

Answer

2

Status : Wrong ***Marks : 0/1***

13. What will be the output of the following code?

```
class Q {
    public static void main(String[] args) {
        int[][] a = {
            {1, 2},
            {3, 4}
        };
        int sum = 0;
        for (int i = 0; i < a.length; i++)
            for (int j = 0; j < a[0].length; j++)
                sum += a[i][j];
        System.out.println(sum);
    }
}
```

Answer

10

Status : Correct ***Marks : 1/1***

14. What will be the output of the following code?

```
class M {
    public static void main(String[] args) {
        int[][] arr = {
            {1, 2},
            {3, 4},
            {5, 6}
        };

        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i][0] + " ");
        }
    }
}
```

Answer

1 3 5

Status : Correct ***Marks : 1/1***

15. What will be the output of the given code?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        int n = arr.length;
        int temp = arr[0];

        for (int i = 0; i < n - 1; i++) {
            arr[i] = arr[i + 1];
        }
        arr[n - 1] = temp;

        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

Answer

2 3 4 5 1

Status : Correct *Marks : 1/1*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Rosh is intrigued by numerical patterns. Today, she stumbled upon a puzzle while working with arrays. She wants to compute the sum of the third-largest and second-smallest elements from a list of integers. She seeks your help to implement a program that solves this for her efficiently.

Input Format

The first line of input is an integer N, representing the size of the array.

The second line of input consists of N space-separated integers, representing the elements of the array.

Output Format

The output displays a single integer representing the sum of the third-largest and second-smallest elements in the array.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10
10 20 30 40 50 60 70 80 90 100

Output: 100

Answer

```
// You are using Java
import java.util.*;
public class Main{
    public static void main(String[]args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int[] arr=new int[n];
        for(int i=0;i<n;i++){
            arr[i]=sc.nextInt();
        }
        Arrays.sort(arr);
        int s=arr[1];
        int t=arr[n-3];
        int p=s+t;
        System.out.println(p);

    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Monica is interested in finding a treasure but the key to opening is to get the sum of the main diagonal elements and secondary diagonal elements.

Write a program to help Monica find the diagonal sum of a square 2D array.

Note: The main diagonal of the array consists of the elements traversing from the top-left corner to the bottom-right corner. The secondary diagonal includes elements from the top-right corner to the bottom-left corner.

Input Format

The first line of input consists of an integer N, representing the number of rows and columns.

The following N lines consist of N space-separated integers, representing the 2D array elements.

Output Format

The first line of output prints "Sum of the main diagonal: " followed by an integer, representing the sum of the main diagonal.

The second line prints "Sum of the secondary diagonal: " followed by an integer, representing the sum of the secondary diagonal.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1 2 3
4 5 6
7 8 9

Output: Sum of the main diagonal: 15
Sum of the secondary diagonal: 15

Answer

```
// You are using Java
import java.util.*;
public class Main{
  public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int[][] arr=new int[n][n];
      for (int i=0;i<n;i++){
         for(int j=0;j<n;j++){
            arr[i][j]=sc.nextInt();
         }
      }
      int m=0;
      int s=0;
      for(int i=0;i<n;i++){
         m+=arr[i][i];
         s+=arr[i][n-1-i];
      }
```

```
        System.out.println("Sum of the main diagonal:"+m);
        System.out.println("Sum of the secondary diagonal:"+s);
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are developing a warehouse management system for a shipping company. The system uses an integer array to represent the weights of packages in a specific order. To verify that the weight capacity is not exceeded, the program needs to calculate the sum of the weights of the first and last packages in the list.

Task:

Write a code to calculate the sum of the weights of the first and last packages in the list. The program should take an integer array as input and return the total weight of the first and last packages.

Input Format

The first line of the input is an integer N representing the size of the array.

The second line of the input is N space-separated integer values.

Output Format

The output is displayed in the following format:

"Sum of the first and last elements: <<Sum>>"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 20 30 40 50
Output: Sum of the first and last elements: 60

Answer

```
// You are using Java
import java.util.*;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int[] arr=new int[n];
      for(int i=0;i<n;i++){
         arr[i]=sc.nextInt();
      }
      int a=arr[0];
      int b=arr[n-1];
      int c=a+b;
      System.out.println("Sum of the first and last elements: "+c);
   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 3_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Eminem is a billiard player who enjoys playing billiards and also likes solving mathematical puzzles. He notices that the billiard balls on the table are arranged in a grid, and he is curious to find the sum of the numbers written on each ball.

Write a program to find the sum of all the numbers written on each ball in the grid.

Input Format

The first line of input consists of an integer N, representing the number of rows.

The second line consists of an integer M, representing the number of columns.

The following lines N lines consist of M space-separated integers, representing the numbers written on each ball.

Output Format

The output prints an integer representing the sum of all the numbers written on each ball.

Refer to the sample output for the formatting specifications.

Sample Test Case

```
Input: 3
3
1 2 3
4 5 6
7 8 9
Output: 45
```

Answer

```
// You are using Java
import java.util.*;
public class Main{
   public static void main(String[]args){
      Scanner sc=new Scanner(System.in);
      int r=sc.nextInt();
      int c=sc.nextInt();
      int m=0;

      int [][] arr=new int[r][c];
      for(int i=0;i<r;i++){
         for(int j=0;j<c;j++){
            arr[i][j]=sc.nextInt();
            m+=arr[i][j];


         }
      }System.out.println(m);

   }
}
```

Status : Correct *Marks : 10/10*

2. Problem Statement

Egath is participating in a coding hackathon, and one of the challenges requires him to work with an array of integers. The task is to remove exactly one element from the array such that the sum of the remaining elements is a prime number.

Help Egath find the first possible prime sum by removing one element or determining if no such prime sum can be achieved.

Input Format

The first line of input consists of an integer N, representing the number of elements in the array.

The second line consists of N space-separated integers, representing the array elements.

Output Format

If removing one element results in a prime sum, print the sum.

If no such prime sum can be achieved by removing exactly one element, print "No valid prime sum found".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1 2 3

Output: 5

Answer

```
import java.util.*;
```

```
public class Main {

    public static boolean isPrime(int n) {
        if (n <= 1) return false;
        if (n == 2) return true;
        if (n % 2 == 0) return false;
        for (int i = 3; i <= Math.sqrt(n); i += 2) {
            if (n % i == 0) return false;
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int N = sc.nextInt();
        int[] arr = new int[N];

        int totalSum = 0;
        for (int i = 0; i < N; i++) {
            arr[i] = sc.nextInt();
            totalSum += arr[i];
        }

        boolean found = false;

        for (int i = 0; i < N; i++) {
            int newSum = totalSum - arr[i];
            if (isPrime(newSum)) {
                System.out.println(newSum);
                found = true;
                break;
            }
        }

        if (!found) {
            System.out.println("No valid prime sum found");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Priya is building a system to automate image transformations using matrix operations. To do this, she needs to multiply two matrices representing pixel data and transformation rules.

Help Priya perform matrix multiplication and print the resulting matrix if the operation is valid.

Input Format

The first line of input consists of two int values, representing the number of rows R1 and columns C1 of the first matrix.

The next R1 × C1 integers represent the elements of the first matrix.

The next line consists of two int values, representing the number of rows R2 and columns C2 of the second matrix.

The next R2 × C2 integers represent the elements of the second matrix.

Output Format

If matrix multiplication is possible, print R1 lines, each containing C2 space-separated int values representing the resulting matrix.

Otherwise, print "Matrix multiplication not possible".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2 3
1 2 3
4 5 6
3 2
7 8
9 10

11 12
Output: 58 64
139 154

Answer

```
import java.util.*;

public class Main{
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);


      int R1 = sc.nextInt();
      int C1 = sc.nextInt();
      int[][] A = new int[R1][C1];
      for (int i = 0; i < R1; i++) {
         for (int j = 0; j < C1; j++) {
            A[i][j] = sc.nextInt();
         }
      }


      int R2 = sc.nextInt();
      int C2 = sc.nextInt();
      int[][] B = new int[R2][C2];
      for (int i = 0; i < R2; i++) {
         for (int j = 0; j < C2; j++) {
            B[i][j] = sc.nextInt();
         }
      }


      if (C1 != R2) {
         System.out.println("Matrix multiplication not possible");
         return;
      }


      int[][] result = new int[R1][C2];
```

```
        for (int i = 0; i < R1; i++) {
            for (int j = 0; j < C2; j++) {
                for (int k = 0; k < C1; k++) {
                    result[i][j] += A[i][k] * B[k][j];
                }
            }
        }


        for (int i = 0; i < R1; i++) {
            for (int j = 0; j < C2; j++) {
                System.out.print(result[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

In a customer loyalty program, reward points are logged in a sorted array as customers make transactions. Occasionally, due to system errors, duplicate entries for the same transaction may appear. To ensure accurate reward calculations, it's crucial to remove these duplicates from the list.

Write a program to process the array of reward points, removing any duplicates while preserving the order of unique entries. The program should then display the cleaned list of unique reward points and the total count of these unique points.

Input Format

The first line of input consists of an integer N, representing the number of reward points.

The second line consists of N space-separated integers, representing the reward points in sorted order.

Output Format

The first line of output prints the cleaned list of unique reward points separated by a space.

The second line of output prints an integer representing the total count of unique reward points.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
100 100 200

Output: 100 200
2

Answer

```
// You are using Java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        int N = sc.nextInt();
        int[] arr = new int[N];
        for (int i = 0; i < N; i++) {
            arr[i] = sc.nextInt();
        }


        int j = 0;
        for (int i = 0; i < N - 1; i++) {
            if (arr[i] != arr[i + 1]) {
                arr[j++] = arr[i];
            }
        }
        arr[j++] = arr[N - 1];
        for (int i = 0; i < j; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();

        System.out.println(j);
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 3_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Emma is a data analyst working with a grid-based system where each cell contains important numerical data. The grid represents spatial data, inventory records, or structured reports that require periodic updates.

Due to system updates and new requirements, Emma needs to modify the grid in the following ways:

She wants to insert either a new row or a new column at a given position.Later, she needs to delete either a row or a column from the modified matrix.

Input Format

The first line contains two integers rows and cols (the dimensions of the matrix).

The next rows lines contain cols space-separated integers representing the initial matrix.

The next line contains two integers insertType and insertIndex:

- insertType = 0 for row insertion, 1 for column insertion.
- insertIndex is the position where the new row/column should be added.

If inserting a row, the next cols integers represent the new row or If inserting a column, the next rows integers represent the new column.

The next line contains two integers deleteType and deleteIndex:

- deleteType = 0 for row deletion, 1 for column deletion.
- deleteIndex is the position to be deleted.

Output Format

The first line of output prints the string "After insertion" followed by the modified matrix with the inserted row or column.

Each row of the matrix is printed on a new line with space-separated integers.

The next line prints the string "After deletion" followed by the final matrix after the specified deletion operation.

Each row of the resulting matrix is printed on a new line with space-separated integers.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3 3
1 2 3
4 5 6
7 8 9
0 1
10 11 12
1 2
Output: After insertion

```
1 2 3
10 11 12
4 5 6
7 8 9
After deletion
1 2
10 11
4 5
7 8
```

Answer

```java
// You are using Java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        int rows = sc.nextInt();
        int cols = sc.nextInt();


        int[][] matrix = new int[rows][cols];
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = sc.nextInt();
            }
        }


        int insertType = sc.nextInt();
        int insertIndex = sc.nextInt();

        if (insertType == 0) {
            int[] newRow = new int[cols];
            for (int j = 0; j < cols; j++) {
                newRow[j] = sc.nextInt();
            }
            matrix = insertRow(matrix, insertIndex, newRow);
            rows++;
        } else {
            int[] newCol = new int[rows];
            for (int i = 0; i < rows; i++) {
                newCol[i] = sc.nextInt();
            }
            matrix = insertColumn(matrix, insertIndex, newCol);
            cols++;
        }

        // Print after insertion
        System.out.println("After insertion");
        printMatrix(matrix);


        int deleteType = sc.nextInt();
        int deleteIndex = sc.nextInt();

        if (deleteType == 0) {
            matrix = deleteRow(matrix, deleteIndex);
            rows--;
        } else {
            matrix = deleteColumn(matrix, deleteIndex);
            cols--;
        }


        System.out.println("After deletion");
        printMatrix(matrix);
    }


    public static int[][] insertRow(int[][] mat, int idx, int[] newRow) {
        int r = mat.length;
        int c = mat[0].length;
        int[][] res = new int[r + 1][c];

        for (int i = 0; i < idx; i++) {
            res[i] = mat[i];
        }
        res[idx] = newRow;
        for (int i = idx; i < r; i++) {
            res[i + 1] = mat[i];
        }
        return res;
    }


    public static int[][] insertColumn(int[][] mat, int idx, int[] newCol) {
        int r = mat.length;
        int c = mat[0].length;
        int[][] res = new int[r][c + 1];

        for (int i = 0; i < r; i++) {
            for (int j = 0; j < idx; j++) {
                res[i][j] = mat[i][j];
            }
            res[i][idx] = newCol[i];
            for (int j = idx; j < c; j++) {
                res[i][j + 1] = mat[i][j];
            }
        }
        return res;
    }


    public static int[][] deleteRow(int[][] mat, int idx) {
        int r = mat.length;
        int c = mat[0].length;
        int[][] res = new int[r - 1][c];

        int k = 0;
        for (int i = 0; i < r; i++) {
            if (i == idx) continue;
            res[k++] = mat[i];
        }
        return res;
    }


    public static int[][] deleteColumn(int[][] mat, int idx) {
        int r = mat.length;
        int c = mat[0].length;
        int[][] res = new int[r][c - 1];

        for (int i = 0; i < r; i++) {
            int k = 0;
            for (int j = 0; j < c; j++) {
                if (j == idx) continue;
                res[i][k++] = mat[i][j];
            }
        }
        return res;
    }


    public static void printMatrix(int[][] mat) {
        for (int[] row : mat) {
            for (int val : row) {
                System.out.print(val + " ");
            }
            System.out.println();
        }
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement:

Emma, a budding computer vision enthusiast, is working on a challenging image processing project. She has a square image represented as a 2D matrix of integers. As part of a special filter operation, she needs to rotate the image by 90 degrees clockwise, but there's a twist — she must perform the rotation in-place, using no extra space.

This means Emma has to rotate the matrix without creating a new one. Your task is to help her implement a Java program that takes this square matrix as input and rotates it within the same structure.

Can you help Emma efficiently rotate the image so that her project can move to the next stage?

Input Format

The first line of input contains a single integer n, representing the number of rows and columns of the square matrix (i.e., the matrix is of size n x n).

The next n lines each contain n space-separated integers, representing the elements of each row of the 2D array.

Output Format

The first line of output prints "Rotated 2D Array:"

The next n lines of output print the rotated matrix.

Each line contains n space-separated integers representing a row of the rotated matrix.

Refer to the sample output for format specification.

Sample Test Case

```
Input: 3
1 2 3
4 5 6
7 8 9
Output: Rotated 2D Array:
7 4 1
8 5 2
9 6 3
```

Answer

```
// You are using Java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        int n = sc.nextInt();
        int[][] mat = new int[n][n];


        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                mat[i][j] = sc.nextInt();
            }
        }


        for (int i = 0; i < n; i++) {
            for (int j = i + 1; j < n; j++) {
                int temp = mat[i][j];
                mat[i][j] = mat[j][i];
                mat[j][i] = temp;
            }
        }


        for (int i = 0; i < n; i++) {
            int left = 0, right = n - 1;
            while (left < right) {
                int temp = mat[i][left];
                mat[i][left] = mat[i][right];
                mat[i][right] = temp;
                left++;
                right--;
            }
        }


        System.out.println("Rotated 2D Array:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(mat[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Robin is a tech-savvy teenager who is diving into programming.

He is working on a project to find special elements in an array called 'leaders.' Leaders are those exceptional elements that are greater than the sum of all the elements to their right.

Assist Robin in writing this program.

Example

Input:

6

16 28 74 19 25 11

Output:

74 25 11

Explanation:

The element 16 is not greater than the sum of elements to its right (28 + 74 + 19 + 25 + 11 = 157)

The element 28 is not greater than the sum of elements to its right (74 + 19 + 25 + 11 = 129)

The element 74 is greater than the sum of elements to its right (19 + 25 + 11 = 55)

The element 19 is not greater than the sum of elements to its right (25 + 11 = 36)

The element 25 is greater than the sum of elements to its right (11)

The last element 11 is always a leader since there are no elements to its right.

So, the output is {74, 25, 11}.

Input Format

The first line of input consists of an integer N, representing the number of elements in the array.

The second line consists of N space-separated integers, representing the elements of the array.

Output Format

The output prints the special elements in the given array, that are greater than the sum of all the elements to their right.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
3 4 2 5 1

Output: 5 1

Answer

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        int N = sc.nextInt();
        int[] arr = new int[N];
        for (int i = 0; i < N; i++) {
            arr[i] = sc.nextInt();
        }


        int totalSum = 0;
        for (int num : arr) {
            totalSum += num;
        }

        int rightSum = totalSum;
        List<Integer> leaders = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            rightSum -= arr[i];
            if (arr[i] > rightSum) {
                leaders.add(arr[i]);
            }
        }


        for (int leader : leaders) {
            System.out.print(leader + " ");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Rina is managing the inventory for a library, where each row of a 2D matrix represents the number of different genres of books available on each shelf.

She wants to perform the following operations:

Transformation: Replace each element in a row with the sum of all elements in that row.Merging: After transformation, Rina will provide one additional matrix, and specify whether to merge the transformed matrix with this new matrix row-wise or column-wise.

Input Format

The first line contains two integers R and C, representing the number of rows and columns of the initial matrix.

The next R lines contain C space-separated integers, representing the book counts in the library.

The next line contains two integers MR and MC, representing the dimensions of the second matrix (to be merged).

The next MR lines contain MC space-separated integers, representing the second matrix.

The last line contains an integer mergeType:

- 0   Row-wise merging (append the second matrix below the transformed matrix).
- 1   Column-wise merging (append the second matrix to the right of the transformed matrix).

Output Format

The output prints "Transformed matrix: "followed by the transformed 2D matrix where each element in a row is replaced with the sum of the elements in that row.

The output prints "Final merged matrix: ", followed by the merging based on mergeType.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3 4
8 2 4 9
4 5 6 1
7 8 9 3
2 4
3 5 7 2
6 1 4 9
0
Output: Transformed matrix:
23 23 23 23
16 16 16 16
27 27 27 27
Final merged matrix:
23 23 23 23
16 16 16 16
27 27 27 27
3 5 7 2
6 1 4 9

Answer

// You are using Java

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        int R = sc.nextInt();
        int C = sc.nextInt();
        int[][] mat1 = new int[R][C];
        for (int i = 0; i < R; i++) {
            for (int j = 0; j < C; j++) {
                mat1[i][j] = sc.nextInt();
            }
        }


        int[][] transformed = new int[R][C];
        for (int i = 0; i < R; i++) {
            int rowSum = 0;
            for (int j = 0; j < C; j++) {
                rowSum += mat1[i][j];
            }
            for (int j = 0; j < C; j++) {
                transformed[i][j] = rowSum;
            }
        }


        int MR = sc.nextInt();
        int MC = sc.nextInt();
        int[][] mat2 = new int[MR][MC];
        for (int i = 0; i < MR; i++) {
            for (int j = 0; j < MC; j++) {
                mat2[i][j] = sc.nextInt();
            }
        }

        int mergeType = sc.nextInt();


        System.out.println("Transformed matrix:");
        printMatrix(transformed);


        System.out.println("Final merged matrix:");
        if (mergeType == 0) {
            printMatrix(transformed);
            printMatrix(mat2);
        } else {
            if (R != MR) {
                System.out.println("Invalid column-wise merge (row counts differ).");
                return;
            }
            for (int i = 0; i < R; i++) {
                for (int j = 0; j < C; j++) {
                    System.out.print(transformed[i][j] + " ");
                }
                for (int j = 0; j < MC; j++) {
                    System.out.print(mat2[i][j] + " ");
                }
                System.out.println();
            }
        }
    }


    public static void printMatrix(int[][] mat) {
        for (int[] row : mat) {
            for (int val : row) {
                System.out.print(val + " ");
            }
            System.out.println();
        }
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 4_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 13

Section 1 : MCQ

1. What will be the output of the following program?

```
class Main {
    public static void main(String[] args) {
        String s1 = "EDUCATION";
        String s2 = new String("EDUCATION");
        String s3 = "EDUCATION";
        if (s1 == s2) {
            System.out.println("s1 and s2 equal");
        }
        else {
            System.out.println("s1 and s2 not equal");
        }
        if (s1 == s3) {
            System.out.println("s1 and s3 equal");
        }
    else {
      System.out.println("s1 and s3 not equal");
    }
  }
}
```

Answer

s1 and s2 not equals1 and s3 equal

Status : Correct ***Marks : 1/1***

2. What will be the output of the following program?

```
public class Main {
  public static void main(String[] args) {
    String str = "1234.34";
    int a = Integer.parseInt(str);
    System.out.println(a);
  }
}
```

Answer

NumberFormatException

Status : Correct ***Marks : 1/1***

3. Predict the output for the following code.

```
class Main {
  public static void main(String[] fruits) {
    String fruit1 = new String("apple");
    String fruit2 = new String("orange");
    String fruit3 = new String("pear");
    fruit3 = fruit1;
    fruit2 = fruit3;
    fruit1 = fruit2;
    System.out.println(fruit1);
    System.out.println(fruit2);
    System.out.println(fruit3);
    }
}
```

Answer

pearpearpear

Status : Wrong ***Marks : 0/1***

4. Predict the output for the following code:

```
class Main {
    public static void main(String args[]) {
        StringBuffer sb = new StringBuffer("I Java!");
        sb.insert(5, "like ");
        System.out.println(sb);
    }
}
```

Answer

I Javlike a!

Status : Correct ***Marks : 1/1***

5. Predict the output for the following code.

```
public class Main {
    public static void main(String[] args) {
      String a = "java";
      char temp = a.charAt(1);
      System.out.println(temp);
  }
}
```

Answer

a

Status : Correct ***Marks : 1/1***

6. What will be the output of the following program?

```
class Main {
   public static void main(String args[]) {
      StringBuffer sb = new StringBuffer("Hello");
      System.out.println("buffer = " + sb);
      System.out.println("length = " + sb.length());
      System.out.println("capacity = " + sb.capacity());
   }
}
```

Answer

buffer = Hellolength = 4capacity = 21

Status : Wrong ***Marks : 0/1***

7. What will be the output of the following code?

```
class Main {
   public static void main(String args[]) {
      char c[] = {'j', 'a', 'v', 'a'};
      String s1 = new String(c);
      String s2 = new String(s1);
      System.out.println(s1);
      System.out.println(s2);
   }
}
```

Answer

javajava

Status : Correct ***Marks : 1/1***

8. What will be the output of the following program?

```
class Main {
  public static void main(String[] args) {
    String greet = "Welcome\n";
    System.out.print("String: " + greet);
    int length = greet.length();
    System.out.print("Length: " + length);
  }
}
```

Answer

String: WelcomeLength: 8

Status : Correct ***Marks : 1/1***

9. Predict the output for the following code:

```
public class Main {
    public static void main(String[] args) {
        float a = 10.0f;
        String temp = Float.toString(a);
        System.out.println(temp);
    }
}
```

Answer

10.0

Status : Correct ***Marks : 1/1***

10. What will be the output of the following program?

```
class Main {
    public static void main(String[] args) {
        String s = new String("5");
        System.out.println(1 + 1111 + s + 1 + 1010);
    }
}
```

Answer

1112511010

Status : Correct ***Marks : 1/1***

11. What will be the output of the following code?

```
class Main {
   public static void main(String args[])
   {
      StringBuffer sb = new StringBuffer("Hello");
      System.out.println("buffer before = " + sb);
      System.out.println("charAt(1) before = " + sb.charAt(1));
      sb.setCharAt(1, 'i');
      sb.setLength(2);
      System.out.println("buffer after = " + sb);
      System.out.println("charAt(1) after = " + sb.charAt(1));
   }
}
```

Answer

buffer before = HellocharAt(1) before = ebuffer after = HicharAt(1) after = i

Status : Correct ***Marks : 1/1***

12. What will be the output for the following code?

```
class Main {
   public static void main(String[] args) {
   String languages[] = { "C", "C++", "Java", "Python", "Ruby"};
      for (String sample: languages) {
         System.out.println(sample);
      }
   }
}
```

Answer

CC++JavaPythonRuby

Status : Correct ***Marks : 1/1***

13. What will be the output of the following code?

```
class Main {
   public static void main(String args[]) {
        String s1 = "Hello i love java";
        String s2 = new String(s1);
        System.out.println((s1 == s2) + " " + s1.equals(s2));
    }
}
```

Answer

false true

Status : Correct ***Marks : 1/1***

14. What is the output of the following code?

```
class Main
{
    public static void main(String args[])
    {
        StringBuffer c = new StringBuffer("Hello");
        c.delete(0,2);
        System.out.println(c);
    }
}
```

Answer

llo

Status : Correct ***Marks : 1/1***

15. What will be the output of the following program?

```
class Main {
    public static void main(String args[])  {
        String name="Work Hard";
        name.concat("Success");
        System.out.println(name);
    }
}
```

Answer

```
Work Hard
```

Status : Correct *Marks : 1/1*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 4_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a publishing company, editors often need to quickly analyze passages of text to check for punctuation usage. To assist them, you are asked to write a program that counts the number of specific punctuation marks in each passage.

The punctuation marks of interest are:

Commas (,)Periods (.)Question marks (?)

Input Format

The first line of input contains an integer T, representing the number of test cases (passages).

Each of the next T lines contains a single passage of text.

Output Format

For each test case, print three integers separated by spaces, representing the number of commas, periods, and question marks in the passage.

The first line of output corresponds to the first passage, the second line to the second passage, and so on.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
Hello, world. How are you?
Output: 1 1 1

Answer

```
// You are using Java
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      int T = sc.nextInt();  // number of test cases
      sc.nextLine(); // consume leftover newline

      for (int i = 0; i < T; i++) {
         String passage = sc.nextLine();
         int commas = 0, periods = 0, questions = 0;

         // Count punctuation
         for (char c : passage.toCharArray()) {
            if (c == ',') {
               commas++;
            } else if (c == '.') {
               periods++;
            } else if (c == '?') {
               questions++;
            }
            }

            // Print result for this passage
            System.out.println(commas + " " + periods + " " + questions);
        }

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 4_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 6.5

Section 1 : Coding

1. Problem Statement

Anu is developing a tool for a conference registration system. Participants submit keywords related to their fields of interest. The organizer wants to sort these keywords alphabetically to generate tags for session grouping.

Write a program that accepts at least five keywords as input arguments and outputs them in sorted alphabetical order.

Input Format

The first line of input contains an integer n, representing the number of keywords.

The second line of input contains n space-separated keywords (string).

Output Format

The output prints n space separated strings representing the sorted keyword in alphabetical order.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
Blockchain Cloud AI Data Cybersecurity
Output: AI Blockchain Cloud Cybersecurity Data

Answer

```
// You are using Java
import java.util.*;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      int n = sc.nextInt();      // number of keywords
      sc.nextLine();             // consume leftover newline

      String[] keywords = new String[n];
      for (int i = 0; i < n; i++) {
         keywords[i] = sc.next();  // read each keyword
      }

      Arrays.sort(keywords, String.CASE_INSENSITIVE_ORDER); // sort ignoring
case

      // Print sorted keywords
      for (int i = 0; i < n; i++) {
         System.out.print(keywords[i]);
         if (i < n - 1) System.out.print(" ");
      }

      sc.close();
   }
}
```

Status : Partially correct *Marks : 6.5/10*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 4_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Bechan Chacha is seeking help to filter out valid mobile numbers from a list provided by his crush. He can only pick his crush's number if the list contains valid mobile numbers.

A mobile number is considered valid if:

It has exactly 10 digits.It consists only of numeric values (0–9).It does not begin with zero.

Your task is to determine whether each mobile number in the list is valid or not.

Input Format

The first line contains an integer T, representing the number of mobile numbers to check.

The next T lines each contain a string S, representing a mobile number.

Output Format

For each mobile number S, the output print "YES" if it is valid.

Otherwise, print "NO".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
9876543210

Output: YES

Answer

```
import java.util.*;

public class Main{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int T = sc.nextInt();
        sc.nextLine();  // consume leftover newline

        for (int i = 0; i < T; i++) {
            String mobile = sc.nextLine();

            if (isValidMobile(mobile)) {
                System.out.println("YES");
            } else {
                System.out.println("NO");
            }
        }

        sc.close();
    }
    // Function to validate mobile number
    public static boolean isValidMobile(String mobile) {
        // Rule 1: must be exactly 10 digits
        if (mobile.length() != 10) return false;

        // Rule 2: must contain only digits
        for (char c : mobile.toCharArray()) {
            if (!Character.isDigit(c)) return false;
        }

        // Rule 3: must not start with '0'
        if (mobile.charAt(0) == '0') return false;

        return true;
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 4_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

At a digital library, the system needs to analyze passages to identify the frequency of vowels, since they are key for linguistic research. You are asked to write a program that counts the number of vowels in each passage of text.

The vowels of interest are:

a, e, i, o, u (both uppercase and lowercase).

Input Format

The first line of input contains an integer T, representing the number of test cases (passages).

Each of the next T lines contains a single passage of text.

Output Format

For each test case, print a single integer representing the total number of vowels in the passage.

The first line of output corresponds to the first passage, the second line to the second passage, and so on.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
Hello World
Output: 3

Answer

```
// You are using Java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read number of test cases
        int T = sc.nextInt();
        sc.nextLine(); // consume the newline after the integer

        for (int t = 0; t < T; t++) {
            String passage = sc.nextLine();
            int count = 0;

            for (char ch : passage.toCharArray()) {
                if ("aeiouAEIOU".indexOf(ch) != -1) {
                    count++;
                }
            }

            System.out.println(count);
        }
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Riya is preparing a puzzle game for her friends. She wants to include a feature that highlights special words in a sentence — specifically, palindromic words (words that read the same forward and backward).

Your task is to help Riya by writing a program that extracts all palindrome words from the given sentence. If there are no palindromes, print "No palindromes found".

Input Format

The input contains a single string S representing a sentence.

Output Format

The output prints all palindromic words separated by a space.

If no palindrome exists, print "No palindromes found".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: madam went to school
Output: madam

Answer

```
// You are using Java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read the full sentence
        String sentence = sc.nextLine();

        // Split into words
        String[] words = sentence.split(" ");
        StringBuilder result = new StringBuilder();

        for (String word : words) {
            if (isPalindrome(word)) {
                result.append(word).append(" ");
            }
        }

        if (result.length() > 0) {
            System.out.println(result.toString());
        } else {
            System.out.println("No palindromes found");
        }

        sc.close();
    }

    // Helper method to check palindrome
    private static boolean isPalindrome(String word) {
        int left = 0, right = word.length() - 1;
        while (left < right) {
            if (word.charAt(left) != word.charAt(right)) {
                return false;
            }
            left++;
            right--;
        }
        return true;
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Sana is analyzing text for a secret code. She wants to find all words in a sentence that start and end with the same letter. These words are considered "special words" for her analysis.

Your task is to write a program that extracts and prints all words that start and end with the same letter (case-insensitive).

If no such word exists, print "No special words found".

Input Format

The input contains a single line containing a sentence with multiple words.

Output Format

The output prints all words that start and end with the same letter separated by a space.

If no word satisfies the condition, print "No special words found".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Anna went to the civic center
Output: Anna civic

Answer

```
// You are using Java
import java.util.Scanner;

public class Main{
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      // Read full input line
      String sentence = sc.nextLine();
```

```
        // Split into words
        String[] words = sentence.split(" ");
        StringBuilder result = new StringBuilder();

        for (String word : words) {
            if (word.length() > 0) {
                char first = Character.toLowerCase(word.charAt(0));
                char last = Character.toLowerCase(word.charAt(word.length() - 1));

                if (first == last) {
                    result.append(word).append(" ");
                }
            }
        }

        if (result.length() > 0) {
            System.out.println(result.toString());
        } else {
            System.out.println("No special words found");
        }

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Ravi is analyzing text messages for his research on typing patterns. He wants to count the number of uppercase letters, lowercase letters, and digits in a sentence to understand typing trends.

Your task is to help Ravi by writing a program that takes a sentence and prints the count of uppercase letters, lowercase letters, and digits.

Input Format

The input contains a single line containing a sentence (string).

Output Format

The output prints three integers separated by spaces:

- Number of uppercase letters
- Number of lowercase letters
- Number of digits

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Hello World 123
Output: 2 8 3

Answer

```
// You are using Java
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      // Read the full line input
      String sentence = sc.nextLine();

      int uppercaseCount = 0;
      int lowercaseCount = 0;
      int digitCount = 0;

      // Loop through each character
      for (char ch : sentence.toCharArray()) {
         if (Character.isUpperCase(ch)) {
            uppercaseCount++;
         } else if (Character.isLowerCase(ch)) {
            lowercaseCount++;
         } else if (Character.isDigit(ch)) {
            digitCount++;
         }
      }

      // Print result as required
```

```
        System.out.println(uppercaseCount + " " + lowercaseCount + " " +
digitCount);

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 4_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Neha is analyzing text messages to identify words that have repeated characters. A word is considered “repetitive” if any character appears more than once in that word.

Your task is to write a program that extracts all words that contain repeated characters from a given sentence.

If no such word exists, print "No repetitive words found".

Input Format

The input contains a single line containing a sentence with multiple words.

Output Format

The output prints all words that contain repeated characters separated by a space.

If no word contains repeated characters, print "No repetitive words found".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: letter balloon apple tree
Output: letter balloon apple tree

Answer

-

Status : Skipped ***Marks : 0/10***

2. Problem Statement

Riya is preparing for a vocabulary test. Her teacher told her to focus on long words in her practice sentences, specifically words that have at least 5 letters.

Riya wants to write a program that will help her identify such words quickly.

Your task is to help Riya by printing all the words in a given sentence that have a length greater than or equal to 5.

If no such word exists, display "No long words found".

Input Format

The input contains a single line containing a sentence with multiple words.

Output Format

The output prints all words having length ≥ 5, separated by a space.

If no such word is found, print "No long words found".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: The quick brown fox jumps over the lazy dog
Output: quick brown jumps

Answer

-

Status : Skipped ***Marks : 0/10***

3. Problem Statement

A library wants to analyze book titles to count the number of words that start with an uppercase letter. This helps the library track proper nouns and important words in titles.

Your task is to write a program that, for each given title, counts and prints the number of words that start with an uppercase letter.

Input Format

The first line contains an integer T, representing the number of book titles.

Each of the next T lines contains a single title (string).

Output Format

For each title, the output print a single integer representing the number of words starting with an uppercase letter.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
The Chronicles of Narnia

Output: 3

Answer

-

Status : Skipped *Marks : 0/10*

4. Problem Statement

Meera is practicing her English vocabulary. She wants to focus on words that have more vowels in them, as they help improve her pronunciation. She decides to extract only those words from a sentence that contain at least two vowels.

Your task is to help Meera by writing a program that finds such words from the given sentence.

Input Format

The input contains a string representing the sentence.

Output Format

The output prints all the words that contain at least two vowels, separated by a space.

If no such word exists, print "No words with two vowels".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: This is an example sentence
Output: example sentence

Answer

-

Status : Skipped *Marks : 0/10*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 11

Section 1 : MCQ

1. What will be the output of the following code?

```
class Ball {
   int size = 11;
}

class Game {
   public static void main(String[] args) {
      Ball b1 = new Ball();
      Ball b2 = new Ball();
      b2.size = 10;
      System.out.println(b1.size);
   }
}
```

Answer

11

Status : Correct *Marks : 1/1*

2. What will be the output of the following code?

```
class A {
   int x = 50;
}

public class Main {
   public static void main(String[] args) {
      A obj1 = new A();
      A obj2 = obj1;
      obj2.x = 100;
      System.out.println(obj1.x);
   }
}
```

Answer

Compilation error

Status : Wrong *Marks : 0/1*

3. What will be the output of the following code?

```
class Person {
   String name;
   void setName(String n) {
      name = n;
   }
   void printName() {
      System.out.println(name);
   }
}

class Test {
   public static void main(String[] args) {
      Person p = new Person();
```

```
        p.printName();
    }
}
```

Answer

Compilation error

Status : Wrong ***Marks : 0/1***

4. What will be the output of the following code?

```
class MathUtils {
    int add(int x) {
        return x + x;
    }
}

public class Main {
    public static void main(String[] args) {
        MathUtils m = new MathUtils();
        System.out.println(m.add(5));
    }
}
```

Answer

25

Status : Wrong ***Marks : 0/1***

5. What is the output of the following code?

```
class Box {
    int height;
    Box(int height) {
        this.height = height;
    }
    void modifyHeight(Box b) {
        b.height += 10;
    }
}
public class Main {
    public static void main(String[] args) {
        Box b1 = new Box(20);
        b1.modifyHeight(b1);
        System.out.println(b1.height);
    }
}
```

Answer

30

Status : Correct ***Marks : 1/1***

6. What will be the output of the following code?

```
class Test {
    private int value;
    Test(int value) {
        this.value = value;
    }
    public int getValue() {
        return value;
    }
}
public class Main {
    public static void main(String[] args) {
        Test obj = new Test(10);
        System.out.println(obj.value);
    }
}
```

Answer

Compile-time error

Status : Correct ***Marks : 1/1***

7. What will be the output of the following code?

```
class A {
   int y = 30;
}

public class Main {
   public static void main(String[] args) {
      A a1 = new A();
      A a2 = new A();
      a1.y = 50;
      System.out.println(a2.y);
   }
}
```

Answer

30

Status : Correct ***Marks : 1/1***

8. What will be the output of the following code?

```
class Sample {
   int x = 10;

   void display() {
      System.out.println("x = " + x);
   }

   public static void main(String[] args) {
      Sample s = new Sample();
      s.display();
   }
}
```

Answer

x = 10

Status : Correct ***Marks : 1/1***

9. What will be the output of the following code?

```
class A {
   int p = 5;
   int q = 2;
}

class Main {
   public static void main(String[] args) {
      A obj = new A();
      System.out.println(obj.p + obj.q);
   }
}
```

Answer

7

Status : Correct ***Marks : 1/1***

10. What will be the output of the following code?

```
class Box {
   int volume(int l, int b, int h) {
      return l * b * h;
   }
}

public class Main {
   public static void main(String[] args) {
      Box b = new Box();
      System.out.println(b.volume(2, 3, 4));
   }
}
```

Answer

Compilation error

Status : Wrong ***Marks : 0/1***

11. What will be the output of the following code?

```
class A {
   int val = 20;
}

public class Main {
   public static void main(String[] args) {
      A obj1 = new A();
      A obj2 = obj1;
      obj2.val += 5;
      System.out.println(obj1.val);
   }
}
```

Answer

25

Status : Correct ***Marks : 1/1***

12. What will be the output of the following code?

```
class Person {
   int age = 18;
}

public class Main {
   public static void main(String[] args) {
      Person p = new Person();
      p.age += 2;
      System.out.println("Age: " + p.age);
   }
}
```

Answer

Age: 20

Status : Correct ***Marks : 1/1***

13. What will be the output of the following code?

```
class Box {
   int length = 5;
   int width = 4;

   int area() {
      return length * width;
   }

   public static void main(String[] args) {
      Box b = new Box();
      System.out.println("Area = " + b.area());
   }
}
```

Answer

Area = 20

Status : Correct ***Marks : 1/1***

14. What will be the output of the following code?

```
class Demo {
   void printMessage() {
      System.out.println("Hello from Demo");
   }
}

public class Main {
   public static void main(String[] args) {
      Demo d = new Demo();
      d.printMessage();
   }
}
```

Answer

Hello from Demo

Status : Correct ***Marks : 1/1***

15. What will be the output of the following code?

```
class Alpha {
    void greet(String name) {
        System.out.println("Hello " + name);
    }
}

public class Main {
    public static void main(String[] args) {
        Alpha obj = new Alpha();
        obj.greet("Anu");
    }
}
```

Answer

Hello Anu

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are working as a developer for CityBank, which wants to build a basic account management system.

Each customer at the bank has:

An Account Number (integer)A Customer Name (string)An Initial Balance (double)

The bank allows two types of transactions:

Deposit – increases the balance.Withdrawal – decreases the balance only if enough funds are available.

If the withdrawal amount is greater than the balance, the withdrawal should not happen, and the balance should remain the same.

You are required to implement this system using:

A class with attributes for account details.A constructor to initialize account details.Setter methods to update details if needed.Getter methods to retrieve details.Objects of the class to represent customers.

Finally, display each customer's account details after all transactions.

Input Format

The first line of input contains an integer N, representing the number of customers.

For each customer:

- The next line contains the account number (integer).
- The following line contains the customer name (string).
- The next line contains the initial balance (double).
- The next line contains the deposit amount (double).
- The next line contains the withdrawal amount (double).

Output Format

For each customer, print the details in the following format:

1. Account Number: <account_number>
2. Customer Name: <customer_name>
3. Final Balance: <final_balance> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Rahul Sharma
5000
2000
3000
Output: Account Number: 1234
Customer Name: Rahul Sharma

Final Balance: 4000.0

Answer

```
// You are using Java
import java.util.*;

class BankAccount {
   private int accountNumber;
   private String customerName;
   private double balance;

   // Constructor
   public BankAccount(int accountNumber, String customerName, double
balance) {
      this.accountNumber = accountNumber;
      this.customerName = customerName;
      this.balance = balance;
   }

   // Getters
   public int getAccountNumber() { return accountNumber; }
   public String getCustomerName() { return customerName; }
   public double getBalance() { return balance; }

   // Setters
   public void setCustomerName(String customerName) { this.customerName =
customerName; }
   public void setBalance(double balance) { this.balance = balance; }

   // Deposit method
   public void deposit(double amount) {
      if (amount >= 0) {
         balance += amount;
      }
   }

   // Withdrawal method
   public void withdraw(double amount) {
      if (amount <= balance) {
         balance -= amount;
      }
   }
```

```
    // Format account details for output
    public String getDetails() {
        return "Account Number: " + accountNumber +
               " Customer Name: " + customerName +
               " Final Balance: " + String.format("%.1f", balance);
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<BankAccount> accounts = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int accNo = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            double initBalance = sc.nextDouble();
            double depositAmount = sc.nextDouble();
            double withdrawAmount = sc.nextDouble();
            if (sc.hasNextLine()) sc.nextLine(); // consume newline

            BankAccount account = new BankAccount(accNo, name, initBalance);
            account.deposit(depositAmount);
            account.withdraw(withdrawAmount);

            accounts.add(account);
        }

        // Print all account details
        StringBuilder output = new StringBuilder();
        for (BankAccount acc : accounts) {
            output.append(acc.getDetails()).append(" ");
        }

        System.out.println(output.toString().trim());
        sc.close();
    }
```

```
}
```

Status : Correct **Marks : 10/10**

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Neha is working as a developer for CityElectricity Board, which wants to build a household electricity billing system.

Each customer's electricity account has:

A Customer ID (integer)A Customer Name (string)Units Consumed (double)

The electricity bill is calculated based on these rules:

For the first 100 units   5 units charge per unitFor the next 100 units (101–200)   7 units charge per unitFor units above 200   10 units charge per unitIf the total bill exceeds 2000 units, a 5% discount is applied on the final bill.

Neha has been asked to implement this system using:

A class with attributes for customer details.A constructor to initialize customer details.Setter methods to update details if needed.Getter methods to retrieve details.Objects of the class to represent customers.

Finally, display each customer's details and final bill amount.

Input Format

The first line of input contains an integer N, representing the number of customers.

For each customer:

- The next line contains the Customer ID (integer).
- The following line contains the Customer Name (string).
- The next line contains the Units Consumed (double).

Output Format

For each customer, print the details in the following format:

Customer ID: <customer_id>

Customer Name: <customer_name>

Final Bill: <final_bill> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
80

Output: Customer ID: 1001
Customer Name: Ravi Kumar
Final Bill: 400.0

Answer

```
// You are using Java
```

```
import java.util.*;

class ElectricityCustomer {
   private int customerId;
   private String customerName;
   private double unitsConsumed;

   // Constructor
   public ElectricityCustomer(int customerId, String customerName, double
unitsConsumed) {
      this.customerId = customerId;
      this.customerName = customerName;
      this.unitsConsumed = unitsConsumed;
   }

   // Getters
   public int getCustomerId() { return customerId; }
   public String getCustomerName() { return customerName; }
   public double getUnitsConsumed() { return unitsConsumed; }

   // Setters
   public void setCustomerName(String customerName) { this.customerName =
customerName; }
   public void setUnitsConsumed(double unitsConsumed) { this.unitsConsumed
= unitsConsumed; }

   // Calculate final bill
   public double calculateBill() {
      double total = 0;
      double units = unitsConsumed;

      if (units <= 100) {
         total = units * 5;
      } else if (units <= 200) {
         total = 100 * 5 + (units - 100) * 7;
      } else {
         total = 100 * 5 + 100 * 7 + (units - 200) * 10;
      }

      // Discount if bill > 2000
      if (total > 2000) {
         total *= 0.95; // 5% discount
```

```
        }

        return total;
    }

    // Format customer details for output
    public String getDetails() {
        return "Customer ID: " + customerId +
               " Customer Name: " + customerName +
               " Final Bill: " + String.format("%.1f", calculateBill());
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<ElectricityCustomer> customers = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            double units = sc.nextDouble();
            if (sc.hasNextLine()) sc.nextLine(); // consume newline

            customers.add(new ElectricityCustomer(id, name, units));
        }

        // Print all customer details
        StringBuilder output = new StringBuilder();
        for (ElectricityCustomer c : customers) {
            output.append(c.getDetails()).append(" ");
        }

        System.out.println(output.toString().trim());
        sc.close();
    }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_PAH

Attempt : 1
Total Mark : 50
Marks Obtained : 50

Section 1 : Coding

1. Problem Statement

Neha is working as a developer for CityMovie Theatre, which wants to build a system to calculate total ticket cost for movie-goers based on the number of tickets and type of seats booked.

Each customer's booking has:

Booking ID (integer)Customer Name (string)Number of Tickets (integer)Seat Type (string: "Standard", "Premium", "VIP")

The ticket prices are:

Standard – 250 units per ticketPremium – 400 units per ticketVIP – 600 units per ticket

The calculation rules:

Total Amount = Number of Tickets × Seat Price

If a customer books more than 4 tickets, they get a 10% discount on the total amount.

If the booking is for VIP seats and the total amount exceeds 3000 units, a 5% luxury tax is added after any discount.

Neha has been asked to implement this system using:

A class with attributes for booking details.A constructor to initialize booking details.Getter and Setter methods to retrieve and update booking details if required.A method to calculate the final ticket cost.Objects of the class to represent bookings.

Finally, display each customer's details and final ticket amount.

Input Format

The first line contains an integer N, representing the number of bookings.

For each booking:

- The next line contains the Booking ID (integer).
- The next line contains the Customer Name (string).
- The next line contains Number of Tickets (integer).
- The next line contains Seat Type ("Standard", "Premium", or "VIP").

Output Format

For each booking, print:

- Booking ID: <booking_id>
- Customer Name: <customer_name>
- Final Ticket Amount: <final_amount> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1

1001
Ravi Kumar
3
Standard

Output: Booking ID: 1001
Customer Name: Ravi Kumar
Final Ticket Amount: 750.0

Answer

```
// You are using Java
import java.util.*;

class Booking {
   private int bookingId;
   private String customerName;
   private int numTickets;
   private String seatType;

   // Constructor
   public Booking(int bookingId, String customerName, int numTickets, String
seatType) {
      this.bookingId = bookingId;
      this.customerName = customerName;
      this.numTickets = numTickets;
      this.seatType = seatType;
   }

   // Getters
   public int getBookingId() { return bookingId; }
   public String getCustomerName() { return customerName; }
   public int getNumTickets() { return numTickets; }
   public String getSeatType() { return seatType; }

   // Setters
   public void setCustomerName(String customerName) { this.customerName =
customerName; }
   public void setNumTickets(int numTickets) { this.numTickets = numTickets; }
   public void setSeatType(String seatType) { this.seatType = seatType; }

   // Calculate final ticket amount
   public double calculateFinalAmount() {
      double seatPrice = 0;
```

```
        switch (seatType) {
            case "Standard": seatPrice = 250; break;
            case "Premium": seatPrice = 400; break;
            case "VIP": seatPrice = 600; break;
        }

        double total = numTickets * seatPrice;

        // Discount for more than 4 tickets
        if (numTickets > 4) {
            total *= 0.9; // 10% discount
        }

        // Luxury tax for VIP seats if total > 3000
        if (seatType.equals("VIP") && total > 3000) {
            total *= 1.05; // 5% luxury tax
        }

        return total;
    }

    // Format booking details for output
    public String getDetails() {
        return "Booking ID: " + bookingId +
               " Customer Name: " + customerName +
               " Final Ticket Amount: " + String.format("%.1f", calculateFinalAmount());
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<Booking> bookings = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            int tickets = sc.nextInt();
            sc.nextLine(); // consume newline
            String seat = sc.nextLine();
            bookings.add(new Booking(id, name, tickets, seat));
        }

        // Print all booking details
        StringBuilder output = new StringBuilder();
        for (Booking b : bookings) {
            output.append(b.getDetails()).append(" ");
        }

        System.out.println(output.toString().trim());
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Anjali is working as a developer for CityFitness Gym, which wants to build a system to calculate monthly membership fees for gym members based on the type of membership and the number of personal training sessions booked.

Each member's record has:

Member ID (integer)Member Name (string)Membership Type (string: "Basic", "Premium", "Elite")Number of Personal Training Sessions (integer)

The monthly fees are:

Basic – 1000 unitsPremium – 1500 unitsElite – 2000 units

The cost of personal training sessions is 500 units per session.

The calculation rules:

Total Amount = Membership Fee + (Number of Personal Training Sessions × 500)If the number of sessions is more than 5, a 10% discount is applied on the total amount.If the member has Elite membership and the total amount exceeds 4000, an additional 5% service tax is added after discount.

Anjali has been asked to implement this system using:

A class with attributes for member details.A constructor to initialize member details.Getter and Setter methods to retrieve and update member details if required.A method to calculate the final monthly fee.Objects of the class to represent members.

Finally, display each member's details and the final monthly fee.

Input Format

The first line contains an integer N, representing the number of members.

For each member:

- Next line contains Member ID (integer)
- Next line contains Member Name (string)
- Next line contains Membership Type ("Basic", "Premium", "Elite")
- Next line contains Number of Personal Training Sessions (integer)

Output Format

For each member, print:

- Member ID: <member_id>
- Member Name: <member_name>
- Final Monthly Fee: <final_fee> (The final fee must be rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
Basic
3
Output: Member ID: 1001

```
Member Name: Ravi Kumar
Final Monthly Fee: 2500.0
```

Answer

```
// You are using Java
import java.util.*;

class GymMember {
   private int memberId;
   private String memberName;
   private String membershipType;
   private int personalTrainingSessions;

   // Constructor
   public GymMember(int memberId, String memberName, String
membershipType, int personalTrainingSessions) {
      this.memberId = memberId;
      this.memberName = memberName;
      this.membershipType = membershipType;
      this.personalTrainingSessions = personalTrainingSessions;
   }

   // Getters
   public int getMemberId() { return memberId; }
   public String getMemberName() { return memberName; }
   public String getMembershipType() { return membershipType; }
   public int getPersonalTrainingSessions() { return personalTrainingSessions; }

   // Setters
   public void setMemberName(String memberName) { this.memberName =
memberName; }
   public void setMembershipType(String membershipType)
{ this.membershipType = membershipType; }
   public void setPersonalTrainingSessions(int sessions)
{ this.personalTrainingSessions = sessions; }

   // Calculate final monthly fee
   public double calculateFee() {
      double membershipFee = 0;
      switch (membershipType) {
         case "Basic": membershipFee = 1000; break;
         case "Premium": membershipFee = 1500; break;
```

```
            case "Elite": membershipFee = 2000; break;
        }

        double total = membershipFee + personalTrainingSessions * 500;

        // Discount for more than 5 sessions
        if (personalTrainingSessions > 5) {
            total *= 0.9; // 10% discount
        }

        // Elite membership extra service tax
        if (membershipType.equals("Elite") && total > 4000) {
            total *= 1.05; // 5% service tax
        }

        return total;
    }

    // Format member details for output
    public String getDetails() {
        return "Member ID: " + memberId +
               " Member Name: " + memberName +
               " Final Monthly Fee: " + String.format("%.1f", calculateFee());
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<GymMember> members = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            String type = sc.nextLine();
            int sessions = sc.nextInt();
            if (sc.hasNextLine()) sc.nextLine(); // consume newline
            members.add(new GymMember(id, name, type, sessions));
        }

        // Print all member details
        StringBuilder output = new StringBuilder();
        for (GymMember m : members) {
            output.append(m.getDetails()).append(" ");
        }

        System.out.println(output.toString().trim());
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Ravi is working as a developer for SecureLogin Systems, which wants to build a system to evaluate the strength of user passwords.

Each user record has:

User ID (integer)User Name (string)Password (string)

The system must calculate whether a password is strong or weak.

A password is considered strong if it meets all of the following conditions:

At least 8 characters long.Contains at least one uppercase letter.Contains at least one lowercase letter.Contains at least one digit.Contains at least one special character (from !@#$%^&amp;*).

Ravi has been asked to implement this system using:

A class with attributes for user details.A constructor to initialize user details.Getter and setter methods to retrieve or update user details.A method to check whether the password is strong.Objects of the class to represent users.

Finally, display each user's details and indicate whether their password is

Strong or Weak.

Input Format

The first line contains an integer N, representing the number of users.

For each user:

The next line contains the User ID (integer).

The next line contains the User Name (string).

The next line contains the Password (string).

Output Format

For each user, print the details in the following format:

User ID: <user_id>

User Name: <user_name>

Password: <password>

Password Strength: <Strong/Weak>

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
Abc@1234

Output: User ID: 1001
User Name: Ravi Kumar
Password: Abc@1234
Password Strength: Strong

Answer

// You are using Java

```
import java.util.*;

class User {
   private int userId;
   private String userName;
   private String password;

   // Constructor
   public User(int userId, String userName, String password) {
      this.userId = userId;
      this.userName = userName;
      this.password = password;
   }

   // Getters
   public int getUserId() { return userId; }
   public String getUserName() { return userName; }
   public String getPassword() { return password; }

   // Setters
   public void setUserName(String userName) { this.userName = userName; }
   public void setPassword(String password) { this.password = password; }

   // Method to check password strength
   public String checkPasswordStrength() {
      if (password.length() < 8) return "Weak";

      boolean hasUpper = false, hasLower = false, hasDigit = false, hasSpecial = 
false;
      String specialChars = "!@#$%^&*";

      for (char c : password.toCharArray()) {
         if (Character.isUpperCase(c)) hasUpper = true;
         else if (Character.isLowerCase(c)) hasLower = true;
         else if (Character.isDigit(c)) hasDigit = true;
         else if (specialChars.indexOf(c) >= 0) hasSpecial = true;
      }

      if (hasUpper && hasLower && hasDigit && hasSpecial) return "Strong";
      return "Weak";
   }
```

```
    // Format user details for output
    public String getDetails() {
        return "User ID: " + userId +
               " User Name: " + userName +
               " Password: " + password +
               " Password Strength: " + checkPasswordStrength();
    }
}

public class Main{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<User> users = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            String password = sc.nextLine();
            users.add(new User(id, name, password));
        }

        // Print user details and password strength
        StringBuilder output = new StringBuilder();
        for (User u : users) {
            output.append(u.getDetails()).append(" ");
        }

        System.out.println(output.toString().trim());
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Each customer at the bank has an Account Number, Customer Name, and an Initial Balance. The bank allows two types of transactions:

Deposit – Increases the balance.Withdrawal – Decreases the balance, but only if enough funds are available. If the withdrawal amount exceeds the available balance, the transaction should be skipped, and the balance should remain unchanged.

You are required to implement this banking system by:

Creating a class with the necessary attributes to store account details.

Using a constructor to initialize the account details when a new account is created.Providing setter methods to update the details if required.Providing getter methods to retrieve account details.Creating objects of this class to represent different customers, where each customer can perform deposits and withdrawals.

Instructions:

Implement the class to store account details.Implement the logic for performing deposit and withdrawal transactions.Ensure that withdrawals don't exceed the available balance.After performing the transactions, print the account number, customer name, and final balance.

Input Format

The first line of input contains an integer N, representing the number of customers.

For each customer:

- The next line contains the account number (integer).
- The following line contains the customer name (string).
- The next line contains the initial balance (double).
- The next line contains the deposit amount (double).
- The next line contains the withdrawal amount (double).

Output Format

For each customer, print the details in the following format:

1. Account Number: <account_number>
2. Customer Name: <customer_name>
3. Final Balance: <final_balance> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Rahul Sharma
5000
2000
3000

Output: Account Number: 1234
Customer Name: Rahul Sharma
Final Balance: 4000.0

Answer

```
import java.util.*;

class BankAccount {
   private int accountNumber;
   private String customerName;
   private double balance;

   // Constructor
   public BankAccount(int accountNumber, String customerName, double
balance) {
      this.accountNumber = accountNumber;
      this.customerName = customerName;
      this.balance = balance;
   }

   // Getters
   public int getAccountNumber() { return accountNumber; }
   public String getCustomerName() { return customerName; }
   public double getBalance() { return balance; }

   // Setters
   public void setCustomerName(String customerName) { this.customerName =
customerName; }
   public void setBalance(double balance) { this.balance = balance; }
```

```
    // Deposit method
    public void deposit(double amount) {
        if (amount > 0) balance += amount;
    }

    // Withdrawal method
    public void withdraw(double amount) {
        if (amount <= balance) balance -= amount;
    }

    // Format details for output
    public String getDetails() {
        return "Account Number: " + accountNumber +
               " Customer Name: " + customerName +
               " Final Balance: " + String.format("%.1f", balance);
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<BankAccount> accounts = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int accNum = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            double initialBalance = sc.nextDouble();
            double depositAmount = sc.nextDouble();
            double withdrawalAmount = sc.nextDouble();
            if (sc.hasNextLine()) sc.nextLine(); // consume newline

            BankAccount account = new BankAccount(accNum, name,
initialBalance);
            account.deposit(depositAmount);
            account.withdraw(withdrawalAmount);
            accounts.add(account);
        }
        // Print all account details
        StringBuilder output = new StringBuilder();
        for (BankAccount acc : accounts) {
            output.append(acc.getDetails()).append(" ");
        }

        System.out.println(output.toString().trim());
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

5. Problem Statement

Neha is working as a developer for CityQuiz Platform, which wants to build a system to calculate quiz scores and identify top scorers among participants.

Each participant's record has:

Participant ID (integer)Participant Name (string)An array of scores in 5 quiz rounds (integers, each between 0 and 100)

The system must calculate:

Total Score = sum of scores in all 5 rounds.Average Score = Total Score ÷ 5.If a participant scores above 80 in all rounds, a bonus of 10 points is added to the total score.Identify the Top Scorer among all participants. If two participants have the same total score, the one with the lower Participant ID is considered the top scorer.

Neha has been asked to implement this system using:

A class with attributes for participant details.A constructor to initialize participant details.Getter and setter methods to retrieve or update participant details.A method to calculate total score and average score (including bonus if applicable).Objects of the class to represent participants.

Finally, display each participant’s details and announce the Top Scorer.

Input Format

The first line of input contains an integer N, representing the number of participants.

For each participant:

- Next line: Participant ID (integer)
- Next line: Participant Name (string)
- Next line: 5 integers separated by spaces (scores for 5 quiz rounds)

Output Format

For each participant:

- Participant ID: <participant_id>
- Participant Name: <participant_name>
- Total Score: <total_score>
- Average Score: <average_score>

Finally, print "Top Scorer: <participant_name> with <total_score> points"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
85 90 88 92 87

Output: Participant ID: 1001
Participant Name: Ravi Kumar
Total Score: 452
Average Score: 90
Top Scorer: Ravi Kumar with 452 points

Answer

```
import java.util.*;

class Participant {
   private int participantId;
   private String participantName;
   private int[] scores;
   private int totalScore;
   private int averageScore;

   // Constructor
   public Participant(int participantId, String participantName, int[] scores) {
      this.participantId = participantId;
      this.participantName = participantName;
      this.scores = scores;
      calculateScores();
   }

   // Getters
   public int getParticipantId() { return participantId; }
   public String getParticipantName() { return participantName; }
   public int getTotalScore() { return totalScore; }
   public int getAverageScore() { return averageScore; }

   // Setters
   public void setScores(int[] scores) {
      this.scores = scores;
      calculateScores();
   }

   // Calculate total and average scores, apply bonus if all scores > 80
   private void calculateScores() {
      totalScore = 0;
      boolean allAbove80 = true;
      for (int s : scores) {
         totalScore += s;
         if (s <= 80) allAbove80 = false;
      }
      if (allAbove80) totalScore += 10; // bonus
      averageScore = totalScore / scores.length;
   }
```

```
    // Format participant details
    public String getDetails() {
        return "Participant ID: " + participantId +
               " Participant Name: " + participantName +
               " Total Score: " + totalScore +
               " Average Score: " + averageScore;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<Participant> participants = new ArrayList<>();

        // Read participant details
        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            int[] scores = new int[5];
            for (int j = 0; j < 5; j++) {
                scores[j] = sc.nextInt();
            }
            if (sc.hasNextLine()) sc.nextLine(); // consume newline

            participants.add(new Participant(id, name, scores));
        }

        // Track top scorer
        Participant topScorer = participants.get(0);

        StringBuilder output = new StringBuilder();

        for (Participant p : participants) {
            output.append(p.getDetails()).append(" ");
            if (p.getTotalScore() > topScorer.getTotalScore() ||
                (p.getTotalScore() == topScorer.getTotalScore() && p.getParticipantId()
< topScorer.getParticipantId())) {
                topScorer = p;
            }
        }

        output.append("Top Scorer: ").append(topScorer.getParticipantName())
              .append(" with ").append(topScorer.getTotalScore()).append(" points");

        System.out.println(output.toString().trim());
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 5_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Arjun is working as a developer for CityWater Supply Board, which wants to build a household water billing system.

Each household's water account has:

A Customer ID (integer)A Customer Name (string)Liters Consumed (double)

The water bill is calculated based on these rules:

For the first 500 liters   2 per literFor the next 500 liters (501–1000)   3 per literFor liters above 1000   5 per literIf the total bill exceeds 3000, a 10% discount is applied on the final bill.

Arjun has been asked to implement this system using:

A class with attributes for customer details.A constructor to initialize customer details.Setter methods to update details if needed.Getter methods to retrieve details.Objects of the class to represent customers.

Finally, display each customer's details and final bill amount.

Input Format

The first line of input contains an integer N, representing the number of customers.

For each customer:

- The next line contains the Customer ID (integer).
- The following line contains the Customer Name (string).
- The next line contains the Liters Consumed (double).

Output Format

For each customer, print the details in the following format:

Customer ID: <customer_id>

Customer Name: <customer_name>

Final Bill: <final_bill> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
300

Output: Customer ID: 1001
Customer Name: Ravi Kumar
Final Bill: 600.0

Answer

```
import java.util.*;
```

```
// Class representing a water customer
class WaterCustomer {
   private int customerId;
   private String customerName;
   private double litersConsumed;

   // Constructor
   public WaterCustomer(int customerId, String customerName, double
litersConsumed) {
      this.customerId = customerId;
      this.customerName = customerName;
      this.litersConsumed = litersConsumed;
   }

   // Getters
   public int getCustomerId() {
      return customerId;
   }

   public String getCustomerName() {
      return customerName;
   }

   public double getLitersConsumed() {
      return litersConsumed;
   }

   // Setters
   public void setCustomerName(String customerName) {
      this.customerName = customerName;
   }

   public void setLitersConsumed(double litersConsumed) {
      this.litersConsumed = litersConsumed;
   }

   // Method to calculate water bill
   public double calculateBill() {
      double bill = 0.0;
      double remaining = litersConsumed;
```

```
        // First 500 liters
        if (remaining > 0) {
            double firstSlab = Math.min(remaining, 500);
            bill += firstSlab * 2;
            remaining -= firstSlab;
        }

        // Next 500 liters (501–1000)
        if (remaining > 0) {
            double secondSlab = Math.min(remaining, 500);
            bill += secondSlab * 3;
            remaining -= secondSlab;
        }

        // Above 1000 liters
        if (remaining > 0) {
            bill += remaining * 5;
        }

        // Apply discount if bill > 3000
        if (bill > 3000) {
            bill *= 0.9; // 10% discount
        }

        return bill;
    }
}
public class Main
{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<WaterCustomer> customers = new ArrayList<>();

        // Read customer details
        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            double liters = sc.nextDouble();
            if (sc.hasNextLine()) sc.nextLine(); // consume newline
            customers.add(new WaterCustomer(id, name, liters));
        }

        // Display details and bill
        for (WaterCustomer c : customers) {
            double bill = c.calculateBill();
            System.out.println("Customer ID: " + c.getCustomerId());
            System.out.println("Customer Name: " + c.getCustomerName());
            System.out.printf("Final Bill: %.1f\n", bill);
        }

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Anjali is now working as a developer for the City Marathon Association, which wants to build a system to track and find the fastest runner among marathon participants.

Each runner's record has:

Runner ID (integer)Runner Name (string)An array of times (in minutes) taken in 5 marathon events (integers)

The system must calculate:

The average time of each runner (sum of all times / 5).Identify the fastest runner (the one with the lowest average time).If two or more runners have the same average time, the one with the lower Runner ID is considered the fastest runner.

Anjali has been asked to implement this system using:

A class with attributes for runner details.A constructor to initialize runner details.Getter and Setter methods to retrieve and update runner details if required.A method to calculate the average time.Objects of the class to represent runners.

Finally, display each runner's details and announce the Fastest Runner.

Input Format

The first line of input contains an integer N (number of runners).

For each runner:

- The next line contains the Runner ID (integer).
- The following line contains the Runner Name (string).
- The next line contains 5 integers separated by spaces (times in minutes for 5 marathon events).

Output Format

For each runner the output prints the following details:

- Runner ID: <runner_id>
- Runner Name: <runner_name>
- Average Time: <average_time>

Finally, print "Fastest Runner: <runner_name> with <average_time> minutes"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
```
1001
Ravi Kumar
240 250 245 255 260
```
Output: Runner ID: 1001
```
Runner Name: Ravi Kumar
Average Time: 250
Fastest Runner: Ravi Kumar with 250 minutes
```

Answer

```
// You are using Java
import java.util.*;

class Runner {
    private int runnerId;
    private String runnerName;
    private int[] times;  // times for 5 marathon events

    // Constructor
    public Runner(int runnerId, String runnerName, int[] times) {
        this.runnerId = runnerId;
        this.runnerName = runnerName;
        this.times = times;
    }

    // Getters
    public int getRunnerId() {
        return runnerId;
    }

    public String getRunnerName() {
        return runnerName;
    }

    // Calculate average time
    public int getAverageTime() {
        int sum = 0;
        for (int t : times) {
            sum += t;
        }
        return sum / times.length;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int N = sc.nextInt();
        sc.nextLine(); // consume newline
        List<Runner> runners = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();

            int[] times = new int[5];
            for (int j = 0; j < 5; j++) {
                times[j] = sc.nextInt();
            }
            if (sc.hasNextLine()) sc.nextLine(); // consume newline

            runners.add(new Runner(id, name, times));
        }

        // Track fastest runner
        Runner fastestRunner = null;
        int fastestAvg = Integer.MAX_VALUE;

        for (Runner r : runners) {
            int avg = r.getAverageTime();
            System.out.println("Runner ID: " + r.getRunnerId());
            System.out.println("Runner Name: " + r.getRunnerName());
            System.out.println("Average Time: " + avg);

            if (avg < fastestAvg || (avg == fastestAvg && r.getRunnerId() <
fastestRunner.getRunnerId())) {
                fastestRunner = r;
                fastestAvg = avg;
            }
        }

        System.out.println("Fastest Runner: " + fastestRunner.getRunnerName() + "
with " + fastestAvg + " minutes");
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Anjali is working as a developer for the City Basketball Association, which wants to build a system to track and find the top scorer among basketball players.

Each player's record has:

Player ID (integer)Player Name (string)An array of points scored in 5 matches (integers)

The system must calculate:

The total score of each player (sum of all match points).Identify the highest scorer among all players.If two or more players have the same total score, the one with the lower Player ID is considered the top scorer.

Anjali has been asked to implement this system using:

A class with attributes for player details.A constructor to initialize player details.Getter and Setter methods to retrieve and update player details if required.A method to calculate the total score.Objects of the class to represent players.

Finally, display each player's details and announce the Top Scorer.

Input Format

The first line of input contains an integer N (number of players).

For each player:

- The next line contains the Player ID (integer).
- The following line contains the Player Name (string).
- The next line contains 5 integers separated by spaces (points scored in 5 matches).

Output Format

For each player the output prints the following details:

- Player ID: <player_id>
- Player Name: <player_name>
- Total Score: <total_score>

Finally, print "Top Scorer: <player_name> with <total_score> points"

Refer to the sample output for formatting specifications.

Sample Test Case

```
Input: 1
1001
Ravi Kumar
10 20 30 40 50
Output: Player ID: 1001
Player Name: Ravi Kumar
Total Score: 150
Top Scorer: Ravi Kumar with 150 points
```

Answer

```
import java.util.*;

class Player {
   private int playerId;
   private String playerName;
   private int[] points;
   private int totalScore;

   // Constructor
   public Player(int playerId, String playerName, int[] points) {
      this.playerId = playerId;
      this.playerName = playerName;
      this.points = points;
      calculateTotal();
   }

   // Getters
   public int getPlayerId() { return playerId; }
   public String getPlayerName() { return playerName; }
   public int getTotalScore() { return totalScore; }
```

```
    // Calculate total score
    private void calculateTotal() {
        totalScore = 0;
        for (int p : points) totalScore += p;
    }

    // Format player details for output
    public String getDetails() {
        return "Player ID: " + playerId + " Player Name: " + playerName + " Total
Score: " + totalScore;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<Player> players = new ArrayList<>();

        // Read player details
        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            int[] points = new int[5];
            for (int j = 0; j < 5; j++) points[j] = sc.nextInt();
            if (sc.hasNextLine()) sc.nextLine(); // consume newline

            players.add(new Player(id, name, points));
        }

        // Track top scorer
        Player topScorer = players.get(0);

        // Print all players in single line
        StringBuilder output = new StringBuilder();
        for (Player p : players) {
            output.append(p.getDetails()).append(" ");
            if (p.getTotalScore() > topScorer.getTotalScore() ||
               (p.getTotalScore() == topScorer.getTotalScore() && p.getPlayerId() <
topScorer.getPlayerId())) {
            topScorer = p;
         }
      }

      // Append top scorer info
      output.append("Top Scorer: ").append(topScorer.getPlayerName())
          .append(" with ").append(topScorer.getTotalScore()).append(" points");

      System.out.println(output.toString().trim());

      sc.close();
   }
}
```

Status : Correct **Marks : 10/10**

4. Problem Statement

You are working as a developer for CityMobile, which wants to build a basic mobile data usage management system.

Each customer has:

A Customer ID (integer)A Customer Name (string)An Initial Data Balance (in GB, double)

The company allows two types of operations:

Recharge – increases the data balance.Usage – decreases the data balance only if enough data is available.

If the usage amount is greater than the available data balance, the usage should not happen, and the balance should remain the same.

You are required to implement this system using:

A class with attributes for customer details.A constructor to initialize customer details.Setter methods to update details if needed.Getter methods to retrieve details.Objects of the class to represent customers.

Finally, display each customer's details after all operations.

Input Format

The first line of input contains an integer N, representing the number of customers.

For each customer:

- The next line contains the Customer ID (integer).
- The following line contains the Customer Name (string).
- The next line contains the Initial Data Balance (double).
- The next line contains the Recharge Amount in GB (double).
- The next line contains the Usage Amount in GB (double).

Output Format

For each customer, print the details in the following format:

Customer ID: <customer_id>

Customer Name: <customer_name>

Final Data Balance: <final_data_balance> GB (The final balance must be rounded to one decimal place.)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Ravi Kumar
5.0
2.0
3.0

Output: Customer ID: 1234
Customer Name: Ravi Kumar
Final Data Balance: 4.0 GB

Answer

```
// You are using Java
import java.util.*;

class Customer {
    private int customerId;
    private String customerName;
    private double dataBalance;

    // Constructor
    public Customer(int customerId, String customerName, double dataBalance) {
        this.customerId = customerId;
        this.customerName = customerName;
        this.dataBalance = dataBalance;
    }

    // Getters
    public int getCustomerId() { return customerId; }
    public String getCustomerName() { return customerName; }
    public double getDataBalance() { return dataBalance; }

    // Recharge method
    public void recharge(double amount) {
        if (amount > 0) dataBalance += amount;
    }

    // Usage method
    public void useData(double amount) {
        if (amount <= dataBalance) dataBalance -= amount;
    }

    // Format customer details for output
    public String getDetails() {
        return "Customer ID: " + customerId +
               " Customer Name: " + customerName +
               " Final Data Balance: " + String.format("%.1f", dataBalance) + " GB";
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine(); // consume newline

        List<Customer> customers = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            String name = sc.nextLine();
            double initialBalance = sc.nextDouble();
            double rechargeAmount = sc.nextDouble();
            double usageAmount = sc.nextDouble();
            if (sc.hasNextLine()) sc.nextLine(); // consume newline

            Customer c = new Customer(id, name, initialBalance);
            c.recharge(rechargeAmount);
            c.useData(usageAmount);
            customers.add(c);
        }

        // Print all customers in single line
        StringBuilder output = new StringBuilder();
        for (Customer c : customers) {
            output.append(c.getDetails()).append(" ");
        }

        System.out.println(output.toString().trim());
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 6_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 15

Section 1 : MCQ

1. What will be the output of the following Java program?

```
class Vehicle {
   void startEngine() {
      System.out.println("Vehicle engine started");
   }
}

class Car extends Vehicle {
   void startEngine() {
      System.out.println("Car engine started");
   }
}

class Main {
   public static void main(String[] args) {
        Vehicle myVehicle = new Car();
        myVehicle.startEngine();
    }
}
```

Answer

Car engine started

Status : Correct ***Marks : 1/1***

2. What will be the output of the following code?

```
class A {
    void display() {
        System.out.println("Display A");
    }
}

class B extends A {
    void display() {
        System.out.println("Display B");
    }
}

class C extends B {
    void display() {
        super.display();
    }
}

class Test {
    public static void main(String[] args) {
        C obj = new C();
        obj.display();
    }
}
```

Answer

Display B

Status : Correct ***Marks : 1/1***

3. What will be the output of the following Java program?

```
class A {
   void display() {
      System.out.println("Class A");
   }
}

class B extends A {
   void show() {
      System.out.println("Class B");
   }
}

class C extends B {
   void print() {
      System.out.println("Class C");
   }
}

class Test {
   public static void main(String[] args) {
      C obj = new C();
      obj.display();
      obj.show();
      obj.print();
   }
}
```

Answer

Class AClass BClass C

Status : Correct ***Marks : 1/1***

4. What will be the output of the following program?

```
class A {
  public int i;
  private int j;
}
class B extends A {
   void display() {
      super.j = super.i + 1;
      System.out.println(super.i + " " + super.j);
   }
}
class inheritance {
  public static void main(String args[]) {
     B obj = new B();
     obj.i=1;
     obj.j=2;
     obj.display();
  }
}
```

Answer

Compile Time Error

Status : Correct ***Marks : 1/1***

5. Select the correct keyword for implementing inheritance through the class.

Answer

extends

Status : Correct ***Marks : 1/1***

6. What will be the output of the following Java program?

```
class Vehicle {
   void start() {
     System.out.println("Vehicle starts");
   }
}
class Car extends Vehicle {

   void start() {
      System.out.println("Car starts");
   }
}
class ElectricCar extends Car {
   void start() {
      System.out.println("Electric Car starts silently");
   }
}
class Test {
   public static void main(String[] args) {
      Vehicle v = new ElectricCar();
      v.start();
   }
}
```

Answer

Electric Car starts silently

Status : Correct ***Marks : 1/1***

7. What will be the output of the following Java program?

```
class Test {
   void show(int a) {
      System.out.println("Integer method");
   }
   void show(String s) {
      System.out.println("String method");
   }
   public static void main(String[] args) {
      Test obj = new Test();
      obj.show(null);
   }
}
```

Answer

String method

Status : Correct ***Marks : 1/1***

8. What will be the output of the following code?

```
class A {
    int sum(int x) {
        return x + 2;
    }
}

class B extends A {
    int sum(int x) {
        return super.sum(x) * 2;
    }
}

class C extends B {
    int sum(int x) {
        return super.sum(x) - 3;
    }
}

class Test {
    public static void main(String[] args) {
        C obj = new C();
        System.out.println(obj.sum(4));
    }
}
```

Answer

9

Status : Correct ***Marks : 1/1***

9. What will be the output of the following program?

```
class A {
   int x = 10;
}

class B extends A {
   int x = 20;
}

class C extends B {
   int x = 30;

   void display() {
      System.out.println(x);
      System.out.println(super.x);
   }
}

class Test {
   public static void main(String[] args) {
      C obj = new C();
      obj.display();
   }
}
```

Answer

3020

Status : Correct ***Marks : 1/1***

10. What will be the output of the following Java program?

```
class A {
   int value = 10;
   void display() {
      System.out.println("A's display: " + value);
   }
}
class B extends A {
    int value = 20;
    void display() {
        System.out.println("B's display: " + value);
    }
}
class Test {
    public static void main(String[] args) {
        A obj = new B();
        obj.display();
        System.out.println("Value: " + obj.value);
    }
}
```

Answer

B's display: 20 Value: 10

Status : Correct ***Marks : 1/1***

11. What will be the output of the following Java program?

```
class Parent {
    void show() {
        System.out.println("Parent class");
    }
}
class Child extends Parent {
    void show() {
        System.out.println("Child class");
    }
}
class Test {
    public static void main(String[] args) {
        Parent obj = new Child();
        obj.show();
    }
}
```

Answer

Child class

Status : Correct ***Marks : 1/1***

12. Which of the following is the correct way for class B to inherit from class A?

Answer

class B extends A {}

Status : Correct ***Marks : 1/1***

13. What will be the output of the following program?

```
class Vehicle {
   String type = "Vehicle";
}

class Car extends Vehicle {
   String type = "Car";
}

class Test {
   public static void main(String[] args) {
      Car c = new Car();
      System.out.println(c.type);
   }
}
```

Answer

Car

Status : Correct ***Marks : 1/1***

14. Which of the following is true about method overriding in Java?

Answer

The method must have the same name, same parameters, and must be in different classes with an inheritance relationship

Status : Correct ***Marks : 1/1***

15. What will be the output of the following Java program?

```
class Test {
   void display(int a, int b) {
      System.out.println("Method 1");
   }
   void display(double a, double b) {
      System.out.println("Method 2");
   }
   public static void main(String[] args) {
      Test obj = new Test();
      obj.display(10, 10.0);
   }
}
```

Answer

Method 2

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 6_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Elsa subscribes to a premium service with a base monthly cost, a service tax and an extra feature cost. Assist her in writing an inheritance program that takes input for these values and calculates the total monthly cost.

Refer to the below class diagram:

Input Format

The first line of input consists of a double value, representing the base monthly cost.

The second line consists of a double value, representing the service tax.

The third line consists of a double value, representing the extra feature cost.

Output Format

The output prints "Rs. X" where X is a double value, rounded off to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10.0
2.5
5.0
Output: Rs. 17.50

Answer

```
import java.util.Scanner;

class Subscription {
   public double monthlyCost;
   public double serviceTax;
   public double extraFeatureCost;
}

class PremiumSubscription extends Subscription {
   public PremiumSubscription(double monthlyCost, double serviceTax, double
extraFeatureCost) {
      this.monthlyCost = monthlyCost;
      this.serviceTax = serviceTax;
      this.extraFeatureCost = extraFeatureCost;
   }

   public double calculateMonthlyCost() {
      double totalCost = monthlyCost + serviceTax + extraFeatureCost;
      return totalCost;
   }
}

public class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
```

```
        double baseMonthlyCost = scanner.nextDouble();
        double serviceTax = scanner.nextDouble();
        double extraFeatureCost = scanner.nextDouble();

        PremiumSubscription premiumSubscription = new
PremiumSubscription(baseMonthlyCost, serviceTax, extraFeatureCost);

        double totalMonthlyCost = premiumSubscription.calculateMonthlyCost();

        System.out.printf("Rs. %.2f%n", totalMonthlyCost);

        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 6_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

Alice is managing an online store and wants to implement a program using inheritance to calculate the selling price of products after applying discounts.

Guide her by following the instructions:

Create a base class called Product with a public double attribute price.Create a subclass called DiscountedProduct, which extends Product and includes a private double attribute discount rate. This subclass has a method called calculateSellingPrice() to determine the final selling price after applying the discount.

Formula: Discounted selling price = price * (1 - discount rate)

Input Format

The first line of input consists of a double value p, the initial price of the product.

The second line consists of a double value d, the discount rate.

Output Format

The output prints "Rs. X", where X is a double value, representing the calculated discounted selling price, rounded off to two decimal places.

If the discount rate is greater than 1, print "Not applicable".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 50.00
0.20
Output: Rs. 40.00

Answer

```
import java.util.Scanner;

// You are using Java

class ProductPricing {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      double initialPrice = scanner.nextDouble();
      double discountRate = scanner.nextDouble();
      DiscountedProduct discountedProduct = new
DiscountedProduct(initialPrice, discountRate);
      double sellingPrice = discountedProduct.calculateSellingPrice();

      if (sellingPrice >= 0) {
         System.out.printf("Rs. %.2f%n", sellingPrice);
      } else {
         System.out.println("Not applicable");
      }
      scanner.close();
   }
}
```

Status : Wrong   ***Marks : 0/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 6_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Ram is designing a program to calculate the Body Mass Index (BMI). Your task is to assist him by following the given specifications.

Create a base class BMIcalculator with a method calculateBMI() to compute BMI using the formula weight / (height * height).

Extend the class with a subclass CustomBMIcalculator that overrides the method calculateBMI() to calculate BMI based on custom criteria, assigning categories such as "Underweight," "Normal Weight," "Overweight," or "Obese."

BMI < 18.5, category = "Underweight"BMI >= 18.5 &amp; < 24.9, category = "Normal Weight"BMI >= 25 &amp; < 29.9, category = "Overweight"else category = "Obese"

Implement user input for weight and height and display both the standard and custom BMI calculations.

Input Format

The first line of input consists of a double value, representing the weight in kgs.

The second line consists of a double value, representing the height in meters.

Output Format

The first line of output prints: "Standard BMI Calculation:"

The second line of output prints: "BMI: " followed by the calculated BMI value (to two decimal places).

The third line of output prints: "Custom BMI Calculation:"

The fourth line of output prints: "Category: " followed by the BMI category.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 69.7
2.6
Output: Standard BMI Calculation:
BMI: 10.31
Custom BMI Calculation:
Category: Underweight

Answer

```
import java.util.Scanner;

class BMIcalculator {
    protected double weight;
    protected double height;

    public BMIcalculator(double weight, double height) {
        this.weight = weight;
        this.height = height;
    }

    public double calculateBMI() {
        return weight / (height * height);
    }

    public void displayBMI() {
        double bmi = calculateBMI();
        System.out.printf("BMI: %.2f\n", bmi);
    }
}

class CustomBMIcalculator extends BMIcalculator {
    private String category;

    public CustomBMIcalculator(double weight, double height) {
        super(weight, height);
    }

    public double calculateBMI() {
        double bmi = super.calculateBMI();

        if (bmi < 18.5) {
            category = "Underweight";
        } else if (bmi >= 18.5 && bmi < 24.9) {
            category = "Normal Weight";
        } else if (bmi >= 25 && bmi < 29.9) {
            category = "Overweight";
        } else {
            category = "Obese";
        }

        return bmi;
    }

    public void displayCustomBMI() {
        double bmi = calculateBMI();
        System.out.printf("Category: %s", category);
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        double weight = scanner.nextDouble();
        double height = scanner.nextDouble();

        BMIcalculator bmiCalculator = new BMIcalculator(weight, height);
        System.out.println("Standard BMI Calculation:");
        bmiCalculator.displayBMI();

        CustomBMIcalculator customBMIcalculator = new
CustomBMIcalculator(weight, height);
        System.out.println("Custom BMI Calculation:");
        customBMIcalculator.displayCustomBMI();

        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

John is planning a long road trip and wants to calculate the distance his car can travel based on its speed and fuel capacity. As John knows that different cars have different fuel efficiencies, he wants a program that can help him estimate the travel distance for any given car.

To do this, you are tasked with creating a program that calculates the travel distance of a car based on its speed and fuel capacity. The calculation is simple and follows the formula:

Travel Distance = Speed * Fuel Capacity

You need to model this system using a Vehicle class and a Car class. The Vehicle class will have attributes for the speed and fuel capacity, while the Car class will inherit from the Vehicle class and include a method to calculate the travel distance.

Input Format

The first line of input consists of a double value representing the speed of the car in km/h.

The second line of input consists of a double value representing the fuel capacity of the car in liters.

Output Format

The first line should print "Speed: X km/h", where X is the speed of the car, rounded to two decimal places.

The second line should print "Fuel Capacity: Y liters", where Y is the fuel capacity of the car, rounded to two decimal places.

The third line should print "Travel Distance: Z km", where Z is the total travel distance the car can cover, rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10.0
1.0

Output: Speed: 10.00 km/h
Fuel Capacity: 1.00 liters
Travel Distance: 10.00 km

Answer

```
import java.util.Scanner;

class Vehicle {
   public double speed;
   public double fuelCapacity;
}

class Car extends Vehicle {
   public Car(double speed, double fuelCapacity) {
      this.speed = speed;
      this.fuelCapacity = fuelCapacity;
   }
```

```
    public double calculateTravelDistance() {
        return speed * fuelCapacity;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        double speed = scanner.nextDouble();
        double fuelCapacity = scanner.nextDouble();

        Car car = new Car(speed, fuelCapacity);

        System.out.println("Speed: " + String.format("%.2f", car.speed) + " km/h");
        System.out.println("Fuel Capacity: " + String.format("%.2f", car.fuelCapacity)
+ " liters");
        System.out.println("Travel Distance: " + String.format("%.2f",
car.calculateTravelDistance()) + " km");

        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

In a company, each manager has a unique employee ID and a monthly salary. You are required to design a program that will calculate and display the annual(12 months) salary of a manager based on the input details provided by the user.

Implement the solution using a single inheritance approach.

Employee: The base class with attributes name and employeeID.

Manager: The derived class inheriting from Employee, with an additional attribute salary.

Input Format

The first line of input consists of a string name, representing the manager's name.

The second line of input consists of an integer employeeID, representing the manager's employee ID.

The third line of input consists of a double salary, representing the manager's monthly salary.

Output Format

The first line of output prints: Name: <name>

The second line of output prints: Annual Salary: Rs. <annual_salary> (rounded to two decimal places).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Davis
234
28750.75

Output: Name: Davis
Annual Salary: Rs. 345009.00

Answer

```
import java.util.Scanner;
import java.text.DecimalFormat;

class Employee {
   protected String name;
   protected int employeeID;

   public Employee(String name, int employeeID) {
      this.name = name;
      this.employeeID = employeeID;
   }
}

class Manager extends Employee {
    private double salary;

    public Manager(String name, int employeeID, double salary) {
        super(name, employeeID);
        this.salary = salary;
    }

    public double calculateAnnualSalary() {
        return salary * 12;
    }
}

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        DecimalFormat df = new DecimalFormat("0.00");

        String name = scanner.nextLine();
        int employeeID = scanner.nextInt();
        double salary = scanner.nextDouble();

        Manager manager = new Manager(name, employeeID, salary);

        System.out.println("Name: " + manager.name);
        System.out.println("Annual Salary: Rs. " +
df.format(manager.calculateAnnualSalary()));

        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Sharon, a software developer, is working on a project to automate velocity calculations for various objects. She wants to create a class named VelocityCalculator with overloaded methods calculateVelocity to calculate the velocity. One method will accept distance in meters and time in seconds as integers, while another will accept distance and time as doubles.

Help her in completing the project.

Formula: Velocity = distance / time

Input Format

The first line of input consists of an integer, representing the distance in meters (for the integer method).

The second line consists of an integer, representing the time in seconds (for the integer method).

The third line consists of a double value, representing the distance in meters (for the double method).

The fourth line consists of a double value, representing the time in seconds (for the double method).

Output Format

The first line prints the velocity calculated using the integer inputs in the format:

Velocity with integer inputs: <velocity> m/s

The second line prints the velocity calculated using the double inputs in the format:

Velocity with double inputs: <velocity> m/s

Note:

The velocity for the double inputs should be printed with two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 100
10
100.5
10.2
Output: Velocity with integer inputs: 10 m/s
Velocity with double inputs: 9.85 m/s

Answer

```
import java.util.Scanner;

class VelocityCalculator {

   public static int calculateVelocity(int distance, int time) {
      return distance / time;
   }

   public static double calculateVelocity(double distance, double time) {
      return distance / time;
   }
}

public class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      int distanceInt = scanner.nextInt();
      int timeInt = scanner.nextInt();

      double distanceDouble = scanner.nextDouble();
      double timeDouble = scanner.nextDouble();

      int velocityInt = VelocityCalculator.calculateVelocity(distanceInt, timeInt);
      double velocityDouble =
VelocityCalculator.calculateVelocity(distanceDouble, timeDouble);

      System.out.println("Velocity with integer inputs: " + velocityInt + " m/s");
      System.out.printf("Velocity with double inputs: %.2f m/s", velocityDouble);

      scanner.close();
   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 6_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 20

Section 1 : Coding

1. Problem Statement

Teena is launching a new airline, Boeing747, and needs to calculate the total revenue generated from ticket sales based on the ticket cost and seat availability. Teena's airline offers two types of seats: regular and premium. The ticket cost and seat availability for both types of seats need to be considered for revenue calculation.

To help with this, Teena wants to implement a system using multilevel inheritance with three classes:

Airline: This class will have the ticket cost as an attribute and defines the method setCost(double cost) and double getCost().Indigo: This class will extend Airline and add the seat availability attribute and defines the method getSeatAvailability() and setSeatAvailability(int seatAvailability) .Boeing747: This class will extend Indigo and include a method calculateTotalRevenue() based on the ticket cost and seat availability .

Teena needs to calculate the total revenue using the formula:

Total Revenue = ticket cost * seat availability

Help Teena implement this system for calculating the revenue of her airline.

Input Format

The first line of input consists of a double value, representing the flight's ticket cost.

The second line consists of an integer, representing seat availability.

Output Format

The first line of output prints "Ticket Cost: Rs. " followed by a double value representing the ticket cost rounded to one decimal place.

The second line of output prints "Seat Availability: X seats" where X is an integer value representing the seat availability.

The third line of output prints "Total Revenue: Rs. " followed by a double value representing the total revenue rounded to one decimal place.

Refer to the sample output for the exact text and format.

Sample Test Case

Input: 1000.0
100

Output: Ticket Cost: Rs. 1000.0
Seat Availability: 100 seats
Total Revenue: Rs. 100000.0

Answer

```
import java.util.Scanner;

class Airline {
   private double cost;
```

```
    public void setCost(double cost) {
        this.cost = cost;
    }

    public double getCost() {
        return cost;
    }
}

// Derived class 1
class Indigo extends Airline {
    private int seatAvailability;

    public void setSeatAvailability(int seatAvailability) {
        this.seatAvailability = seatAvailability;
    }

    public int getSeatAvailability() {
        return seatAvailability;
    }
}

// Derived class 2 (multilevel)
class Boeing747 extends Indigo {

    public double calculateTotalRevenue() {
        return getCost() * getSeatAvailability();
    }
}


public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Boeing747 plane = new Boeing747();

        double ticketCost = scanner.nextDouble();
        plane.setCost(ticketCost);
        int seatAvailability = scanner.nextInt();
        plane.setSeatAvailability(seatAvailability);

        System.out.printf("Ticket Cost: Rs. %.1f\n", plane.getCost());
        System.out.println("Seat Availability: " + plane.getSeatAvailability() + " 
seats");
        System.out.printf("Total Revenue: Rs. %.1f\n", 
plane.calculateTotalRevenue());
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Adams has a reputation company with a great number of employees. He must calculate the salary weekly according to the hourly rate and working hours. Create a program to define a class Employee with attributes name and hourly rate. Create a subclass HourlyEmployee that calculates the weekly salary based on the number of hours worked.

(The first 40 hours are based on the regular hour rate. If the work hours are greater than 40 then the work wage is 1.5 times the hourly rate)

Note: Use Math(Math.max, Math.min) functions .

Example

Input:

Chris

10

45

Output:

Weekly Salary: Rs.475.00

Explanation:

Calculation:

The first 40 hours are paid normally: 40 × 10 = 400.00The extra 5 hours are paid at 1.5 times the hourly rate: 5 × (10×1.5) = 5 × 15 = 75.00Total salary: 400.00 + 75.00 = 475.00

Input Format

The first line of input consists of a string that represents the name of the employee.

The second line consists of a double value that represents the rate for an hour.

The last line consists of an integer that represents the total hours worked.

Output Format

The output displays the total salary of the employee, where salary is rounded to two decimal places in the format: "Weekly Salary: Rs.<double value>".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Dave
10.0
40
Output: Weekly Salary: Rs.400.00

Answer

```
import java.util.Scanner;
import java.text.DecimalFormat;

import java.util.Scanner;

class Employee {
   protected String name;
   protected double hourlyRate;

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

class HourlyEmployee extends Employee {
   private double hoursWorked;
```

```
    public HourlyEmployee(String name, double hourlyRate, double hoursWorked)
{
        super(name, hourlyRate);
        this.hoursWorked = hoursWorked;
    }

    public double calculateWeeklySalary() {
        double regularHours = Math.min(hoursWorked, 40);
        double overtimeHours = Math.max(0, hoursWorked - 40);

        double regularPay = regularHours * hourlyRate;
        double overtimePay = overtimeHours * (hourlyRate * 1.5);

        return regularPay + overtimePay;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        String name = scanner.nextLine();
        double hourlyRate = scanner.nextDouble();
        int hoursWorked = scanner.nextInt();

        HourlyEmployee employee = new HourlyEmployee(name, hourlyRate,
hoursWorked);

        double weeklySalary = employee.calculateWeeklySalary();
        DecimalFormat df = new DecimalFormat("#.00");
        String formattedSalary = df.format(weeklySalary);
        System.out.println("Weekly Salary: Rs." + formattedSalary);
        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

A bank provides two types of deposit schemes: Fixed Deposits (FD) and Recurring Deposits (RD). Customers want to calculate the interest they can earn based on their selected scheme.

Develop a Java program using inheritance to compute the interest for FD and RD. The program should include:

A base class Account with attributes accountHolder and principalAmount, along with a method for interest calculation.A subclass FixedDeposit that calculates interest for FD.A subclass RecurringDeposit that calculates interest for RD.

Formulas Used:

Interest for FD: (principal amount * duration in years * rate of interest) / 100

Interest for RD: (maturity amount * duration in months * rate of interest) / (12 * 100), where maturity amount = monthly deposit * duration in months.

Input Format

The first line of input consists of the choice (1 for FD, 2 for RD).

If the choice is 1, the following lines consist of account holder (string), principal amount (double), duration in years (int), and rate of interest (double).

If the choice is 2, the following lines consist of account holder (string), monthly deposit (int), duration in months (int), and rate of interest (double).

Output Format

The output prints the calculated interest with one decimal place in the following format.

For choice 1: "Interest for FD: <calculated interest >"

For choice 2: "Interest for FD: <calculated interest >"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
Alice
50000.56
5
6.5

Output: Interest for FD: 16250.2

Answer

```
import java.util.Scanner;

import java.util.Scanner;

class Account {
   String accountHolder;
   double principalAmount;
   double rateOfInterest;

   public Account(String accountHolder, double principalAmount, double
rateOfInterest) {
      this.accountHolder = accountHolder;
      this.principalAmount = principalAmount;
      this.rateOfInterest = rateOfInterest;
   }

   public double calculateInterest() {
      return 0.0; // Base class does not calculate interest directly
   }
}

class FixedDeposit extends Account {
   int durationInYears;

   public FixedDeposit(String accountHolder, double principalAmount, double
rateOfInterest, int durationInYears) {
      super(accountHolder, principalAmount, rateOfInterest);
      this.durationInYears = durationInYears;
   }

   @Override
   public double calculateInterest() {
      return (principalAmount * durationInYears * rateOfInterest) / 100;
   }
```

```
}

class RecurringDeposit extends Account {
   int durationInMonths;
   double monthlyDeposit;

   public RecurringDeposit(String accountHolder, double monthlyDeposit, double
rateOfInterest, int durationInMonths) {
      super(accountHolder, 0, rateOfInterest);
      this.monthlyDeposit = monthlyDeposit;
      this.durationInMonths = durationInMonths;
   }

   @Override
   public double calculateInterest() {
      double maturityAmount = monthlyDeposit * durationInMonths;
      return (maturityAmount * durationInMonths * rateOfInterest) / (12 * 100);
   }
}


public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      int choice = sc.nextInt();

      switch (choice) {
         case 1:
            sc.nextLine();
            String fdName = sc.nextLine();
            double fdPrincipal = sc.nextDouble();
            int fdDuration = sc.nextInt();
            double fdRate = sc.nextDouble();

            FixedDeposit fd = new FixedDeposit(fdName, fdPrincipal, fdDuration,
fdRate);
            System.out.printf("Interest for FD: %.1f", fd.calculateInterest());
            break;

         case 2:
            sc.nextLine();
            String rdName = sc.nextLine();
```

```
                int rdDeposit = sc.nextInt();
                int rdDuration = sc.nextInt();
                double rdRate = sc.nextDouble();

                RecurringDeposit rd = new RecurringDeposit(rdName, rdDeposit,
rdDuration, rdRate);
                System.out.printf("Interest for RD: %.1f", rd.calculateInterest());
                break;

            default:
                System.out.println("Invalid Choice");
        }
    }
}
```

Status : Wrong ***Marks : 0/10***

4. Problem Statement

Teena's retail store has implemented a Loyalty Points System to reward customers based on their spending. The program calculates and displays the loyalty points based on whether the customer is a regular or a premium customer.

For regular customers (class Customer), the loyalty points are calculated as:

Loyalty points = amount spent / 10

For premium customers (class PremiumCustomer, which inherits from Customer), the loyalty points are calculated as:

Loyalty points = 2 * (amount spent / 10)

The program should use method overriding for premium customers to calculate their loyalty points. The method that needs to be overridden is calculateLoyaltyPoints in the Customer class.

Input Format

The first line of input consists of an integer representing the amount spent by the customer.

The second line consists of a string representing the premium customer status:

- "yes" if the customer is a premium customer.
- "no" if the customer is not a premium customer.

Output Format

The output should display the loyalty points earned based on the amount spent and the customer type.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 50
yes

Output: 10

Answer

```
import java.util.Scanner;

import java.util.Scanner;

class Customer {
   protected double amountSpent;

   public Customer(double amountSpent) {
      this.amountSpent = amountSpent;
   }

   public double calculateLoyaltyPoints() {
      return amountSpent / 10.0;
   }
}

class PremiumCustomer extends Customer {
   public PremiumCustomer(double amountSpent) {
      super(amountSpent);
   }
```

```
    @Override
    public double calculateLoyaltyPoints() {
        return 2 * (amountSpent / 10.0);
    }
}

public class LoyaltyPoints {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter amount spent:");
        double amountSpent = scanner.nextDouble();

        System.out.println("Are you a premium customer? (yes/no):");
        String customerType = scanner.next().toLowerCase();

        if (customerType.equals("yes")) {
            PremiumCustomer premiumCustomer = new
PremiumCustomer(amountSpent);
            double points = premiumCustomer.calculateLoyaltyPoints();
            System.out.println("Loyalty Points: " + points);
        } else {
            Customer regularCustomer = new Customer(amountSpent);
            double points = regularCustomer.calculateLoyaltyPoints();
            System.out.println("Loyalty Points: " + points);
        }

        scanner.close();
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        int amountSpent = scanner.nextInt();

        String isPremium = scanner.next().toLowerCase();

        Customer customer;

        if (isPremium.equals("yes")) {
            customer = new PremiumCustomer();
        } else {
            customer = new Customer();
        }

        int loyaltyPoints = customer.calculateLoyaltyPoints(amountSpent);

        System.out.println(loyaltyPoints);
    }
}
```

Status : Wrong ***Marks : 0/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 7_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 15

Section 1 : MCQ

1. What happens when an implementing class does not override a default method from an interface?

Answer

The default method's implementation from the interface will be used.

Status : Correct ***Marks : 1/1***

2. Which of the following statements about Java interfaces is true?

Answer

A class can implement multiple interfaces.

Status : Correct ***Marks : 1/1***

3. Which of the following is the correct way to declare an interface in Java?

Answer

interface Vehicle {   void start();}

Status : Correct ***Marks : 1/1***

4. How can a class explicitly call a default method from an interface if there is a naming conflict?

Answer

Using InterfaceName.super.methodName();

Status : Correct ***Marks : 1/1***

5. What is the output of the following code?

```
interface A {
   default void show() {
      System.out.println("A's Default Method");
   }
}

class B {
   public void show() {
      System.out.println("B's Method");
   }
}

class C extends B implements A {
}

public class Main {
   public static void main(String[] args) {
      C obj = new C();
      obj.show();
   }
}
```

Answer

B's Method

Status : Correct ***Marks : 1/1***

6. What is the output of the following code?

```
interface A {
   default void show() {
      System.out.println("A's Default Method");
   }
}

interface B {
   default void show() {
      System.out.println("B's Default Method");
   }
}

class C implements A, B {
   public void show() {
      A.super.show();
   }
}

public class Main {
   public static void main(String[] args) {
      C obj = new C();
      obj.show();
   }
}
```

Answer

A's Default Method

Status : Correct ***Marks : 1/1***

7. If a class implements two interfaces that have the same default method, what must the class do?

Answer

The class must override the method to resolve ambiguity.

Status : Correct ***Marks : 1/1***

8. Can a Java interface contain both default and static methods?

Answer

Yes, an interface can have both default and static methods.

Status : Correct ***Marks : 1/1***

9. What is the output of the following code?

```
interface A {
   static void display() {
      System.out.println("Static method in A");
   }
}

class B implements A {
   static void display() {
      System.out.println("Static method in B");
   }
}

public class Main {
   public static void main(String[] args) {
      B.display();
   }
}
```

Answer

Static method in B

Status : Correct ***Marks : 1/1***

10. What is the primary purpose of static methods in Java interfaces?

Answer

They allow an interface to provide helper methods without requiring an implementing class.

Status : Correct ***Marks : 1/1***

11. What is the output of the following code?

```
interface MathOperations {
   static int square(int x) {
      return x * x;
   }
}

public class Main {
   public static void main(String[] args) {
      System.out.println(MathOperations.square(5));
   }
}
```

Answer

25

Status : Correct ***Marks : 1/1***

12. Consider a class implementing an interface and extending a class, both having a method with the same name. Which method gets called?

Answer

The method from the superclass

Status : Correct ***Marks : 1/1***

13. How do you call a static method from an interface MyInterface?

Answer

```
MyInterface.staticMethod();
```

Status : Correct ***Marks : 1/1***

14. What is the output of the following code?

```
interface X {
   default void show() {
      System.out.println("X's Default Method");
   }
}

interface Y {
   default void show() {
      System.out.println("Y's Default Method");
   }
}

class Z implements X, Y {
   public void show() {
      System.out.println("Z's Method");
   }
}

public class Main {
   public static void main(String[] args) {
      Z obj = new Z();
      obj.show();
   }
}
```

Answer

Z's Method

Status : Correct ***Marks : 1/1***

15. Which of the following statements is true regarding default methods in Java interfaces?

Answer

A default method can be overridden in a class implementing the interface.

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 7_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement:

Rajiv is analyzing the energy consumption in his household and wants to calculate the total cost based on the daily energy usage. He is given the rate per unit of electricity and the energy consumed for multiple days. To structure this calculation efficiently, he decides to use an interface-based approach.

Implement an interface CostCalculator with the necessary methods to retrieve energy details and compute the cost. The calculations should be handled in the EnergyConsumptionTracker class, while the EnergyConsumptionApp class should only handle input and output.

Formula

Energy Cost for one day = Energy Consumed per day * Rate Per Unit

Input Format

The first line of input consists of the rate per unit as an 'R' (a double value).

The second line of input consists of the number of days 'N' (an integer).

The third line of input consists of the daily energy consumption values for each day 'D" (double values), separated by space.

Output Format

The first line of the output prints: "Day-wise Energy Cost:"

The next N lines of the output print the day-wise energy costs(double type) and the total energy cost (double type) in Indian Rupees in the following format: "Day [day_number]: Rs. [energy_cost]"

The last line of the output prints: "Total Energy Cost: Rs. [total_cost]"

Note: energy_cost and total_cost are rounded off to two decimal points

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 0.01
3
10.0 20.0 30.0

Output: Day-wise Energy Cost:
Day 1: Rs. 0.10
Day 2: Rs. 0.20
Day 3: Rs. 0.30
Total Energy Cost: Rs. 0.60

Answer

```
import java.util.Scanner;

interface  CostCalculator{
    void getEnergyDetails(double rate,double[]dailyConsumption);
    void computeCost();
}

class EnergyConsumptionTracker implements CostCalculator{
    double rate;
    double[] dailyConsumption;
    double totalCost;
    public void getEnergyDetails(double rate,double[] dailyConsumption){
        this.rate=rate;
        this.dailyConsumption=dailyConsumption;
    }
    public void computeCost(){
        System.out.println("Day-wise Energy Cost:");
        for(int i=0;i<dailyConsumption.length;i++){
            double cost=dailyConsumption[i]*rate;
            totalCost+=cost;
            System.out.printf("Day %d: Rs.%.2f%n",(i+1),cost);
        }
        System.out.printf("Total Energy Cost : Rs.%.2f%n",totalCost);
    }
}

class EnergyConsumptionApp {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        double ratePerUnit = scanner.nextDouble();
        int numDays = scanner.nextInt();

        CostCalculator tracker = new EnergyConsumptionTracker(ratePerUnit,
numDays);

        tracker.getEnergyDetails(scanner);
        tracker.calculateAndDisplayCost();

        scanner.close();
    }
}
```

Status : Skipped ***Marks : 0/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 7_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Jaheer is working on a health monitoring system to help individuals calculate their Body Mass Index (BMI). He has implemented a basic BMI calculator and an interface called HealthCalculator. It should have a method called calculateBMI.

You are tasked with creating a program that takes weight and height as input, calculates the BMI using the BMICalculator class, and displays the result. If the height or weight is less than or equal to zero, then return -1.

Formula: BMI = weight / (height * height)

Input Format

The first line of input consists of a double value W, the person's weight in kilograms.

The second line consists of a double value H, the height of the person in meters.

Output Format

The output displays "BMI: " followed by a double value, representing the calculated BMI, rounded off to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 70.0
1.75
Output: BMI: 22.86

Answer

```
import java.util.Scanner;

// You are using Java
interface HealthCalculator{
  double calculateBMI(double weight,double height);
}
class BMICalculator implements HealthCalculator{
@Override
   public double calculateBMI(double weight,double height){
     return weight/(height*height);
   }

}

class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      double weight = scanner.nextDouble();
      double height = scanner.nextDouble();

      BMICalculator bmiCalculator = new BMICalculator();

      double bmi = bmiCalculator.calculateBMI(weight, height);
```

```
        System.out.printf("BMI: %.2f\n", bmi);


        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 7_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement:

Alice has been tasked with implementing a simple calculator interface and a corresponding class for performing basic addition and subtraction operations. The task is to create an interface called Calculator with two methods: add and subtract. The add method should take two numbers as input and return their sum, while the subtract method should take two numbers as input and return their difference.

Implement a class called SimpleCalculator that implements the Calculator interface. This class should provide the functionality for adding and subtracting numbers. Write a code that satisfies the above requirements.

Input Format

The first line of input consists of a single integer, representing the choice

If the choice is 1 or 2, the next two lines consist of 2 double values, representing the numbers to do addition or subtraction.

Output Format

The output prints a float-value with one decimal value representing the sum of two number or difference of two number.

Refer to the sample output for format specification.

Sample Test Case

Input: 1
5.5
3.5

Output: Result: 9.0

Answer

```
import java.util.Scanner;

interface Calculator {
   double add(double a, double b);
   double subtract(double a, double b);
}

class SimpleCalculator implements Calculator {
   @Override
   public double add(double a, double b) {
      return a + b;
   }

   @Override
   public double subtract(double a, double b) {
      return a - b;
   }
}

class MathOperationsProgram {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
```

```
        SimpleCalculator calculator = new SimpleCalculator();

        int choice = scanner.nextInt();

        if (choice == 1) {
            double num1 = scanner.nextDouble();
            double num2 = scanner.nextDouble();
            double result = calculator.add(num1, num2);
            System.out.println("Result: " + result);
        } else if (choice == 2) {
            double num1 = scanner.nextDouble();
            double num2 = scanner.nextDouble();
            double result = calculator.subtract(num1, num2);
            System.out.println("Result: " + result);
        } else {
            System.out.println("Invalid choice. Please choose 1 for addition or 2 for
subtraction.");
        }

        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Oviya is fascinated by automorphic numbers and wants to create a program to determine whether a given number is an automorphic number or not.

An automorphic number is a number whose square ends with the same digits as the number itself. For example, 25 = (25)2 = 625

Oviya has defined two interfaces: NumberInput for taking user input and AutomorphicChecker for checking if a given number is automorphic. The class AutomorphicNumber implements both interfaces.

Help her complete the task.

Input Format

The input consists of a single integer n.

Output Format

If the input number is an automorphic number, print "n is an automorphic number". Otherwise, print "n is not an automorphic number".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 25
Output: 25 is an automorphic number

Answer

```
import java.util.Scanner;


interface NumberInput {
   int getInput();
}

interface AutomorphicChecker {
   boolean checkAutomorphic(int n);
}

class AutomorphicNumber implements NumberInput, AutomorphicChecker {
   public int getInput() {
      Scanner sc = new Scanner(System.in);
      return sc.nextInt();
   }

   public boolean checkAutomorphic(int n) {
      int square = n * n;
      String numStr = String.valueOf(n);
      String squareStr = String.valueOf(square);
      return squareStr.endsWith(numStr);
   }
```

```
  public static void main(String[] args) {
    AutomorphicNumber automorphicNumber = new AutomorphicNumber();
    int inputNumber = automorphicNumber.getInput();
    if (automorphicNumber.checkAutomorphic(inputNumber))
      System.out.print(inputNumber + " is an automorphic number");
    else
      System.out.print(inputNumber + " is not an automorphic number");
  }
}


public class Main {
  public static void main(String[] args) {
    AutomorphicNumber automorphicNumber = new AutomorphicNumber();
    int inputNumber = automorphicNumber.getInput();

    boolean isAutomorphic = 
automorphicNumber.checkAutomorphic(inputNumber);

    if (isAutomorphic) {
      System.out.println(inputNumber+" is an automorphic number");
    } else {
      System.out.println(inputNumber+" is not an automorphic number");
    }
  }
}
```

Status : Correct **_Marks : 10/10_**

3. Problem Statement

Develop a program for managing employee information that caters to both full-time and part-time employees. The program should be capable of computing the salary for each category of employee and presenting their particulars. To achieve this, create two classes, FullTimeEmployee and PartTimeEmployee, that adhere to the Employee interface.

The program is expected to accept input data, including the name and monthly salary for full-time employees, as well as the name, hourly rate, and hours worked for part-time employees. Subsequently, it should calculate and exhibit the employee details and their respective salaries.

For Full-Time employees, the annual salary should be calculated as 12 times the monthly salary.

For Part-Time employees, the salary calculation should be based on the formula: hourly rate * hours worked.

Input Format

The first line of input should be a string representing the name of a full-time employee.

The second line of input should be an integer representing the monthly salary of the full-time employee.

The third line of input should be a string representing the name of a part-time employee.

The fourth line of input should be an integer representing the hourly rate of the part-time employee.

The fifth line of input should be an integer representing the number of hours worked by the part-time employee.

Output Format

The output displays the following details:

Full-Time Employee Details:

Name: [Full-Time Employee Name] (string)

Monthly Salary: $[Monthly Salary] (integer)

Annual Salary: $[12 times Monthly Salary] (integer)

Part-Time Employee Details:

Name: [Part-Time Employee Name] (string)

Hourly Rate: $[Hourly Rate] (integer)

Hours Worked: [Hours Worked] hours (integer)

Monthly Salary: $[Calculated Monthly Salary] (integer)

Refer to the sample output for the formatting specifications.

Sample Test Case

```
Input: John Smith
15000
Mary Johnson
100
100
Output: Full-Time Employee Details:
Name: John Smith
Monthly Salary: $15000
Annual Salary: $180000

Part-Time Employee Details:
Name: Mary Johnson
Hourly Rate: $100
Hours Worked: 100 hours
Monthly Salary: $10000
```

Answer

```
import java.util.Scanner;

// You are using Java


interface Employee {
    void displayDetails();
}
```

```
class FullTimeEmployee implements Employee {
   String name;
   int monthlySalary;

   FullTimeEmployee(String name, int monthlySalary) {
      this.name = name;
      this.monthlySalary = monthlySalary;
   }

   public void displayDetails() {
      int annualSalary = monthlySalary * 12;
      System.out.println("Full-Time Employee Details: ");
      System.out.println("Name: " + name + " ");
      System.out.println("Monthly Salary: $" + monthlySalary + " ");
      System.out.println("Annual Salary: $" + annualSalary + "  ");
   }
}

class PartTimeEmployee implements Employee {
   String name;
   int hourlyRate;
   int hoursWorked;

   PartTimeEmployee(String name, int hourlyRate, int hoursWorked) {
      this.name = name;
      this.hourlyRate = hourlyRate;
      this.hoursWorked = hoursWorked;
   }

   public void displayDetails() {
      int salary = hourlyRate * hoursWorked;
      System.out.println("Part-Time Employee Details: ");
      System.out.println("Name: " + name + " ");
      System.out.println("Hourly Rate: $" + hourlyRate + " ");
      System.out.println("Hours Worked: " + hoursWorked + " hours ");
      System.out.println("Monthly Salary: $" + salary + " ");
   }
}

class EmployeeDetails {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
```

```
        String fullTimeName = sc.nextLine();
        int monthlySalary = sc.nextInt();
        sc.nextLine();
        String partTimeName = sc.nextLine();
        int hourlyRate = sc.nextInt();
        int hoursWorked = sc.nextInt();

        FullTimeEmployee fte = new FullTimeEmployee(fullTimeName,
monthlySalary);
        PartTimeEmployee pte = new PartTimeEmployee(partTimeName,
hourlyRate, hoursWorked);

        fte.displayDetails();
        pte.displayDetails();
    }
}
class EmployeeInheritanceDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String fullName = scanner.nextLine();
        int fullTimeSalary = scanner.nextInt();
        scanner.nextLine();
        String partTimeName = scanner.nextLine();
        int hourlyRate = scanner.nextInt();
        int hoursWorked = scanner.nextInt();
        FullTimeEmployee fullTimeEmployee = new FullTimeEmployee(fullName,
fullTimeSalary);
        PartTimeEmployee partTimeEmployee = new
PartTimeEmployee(partTimeName, hourlyRate, hoursWorked);
        fullTimeEmployee.displayDetails();
        System.out.println();
        partTimeEmployee.displayDetails();
        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Sophia is developing a matrix analysis tool for a data analytics company. The tool needs to analyze square matrices and extract insights from the matrix diagonals.

To organize the code properly, Sophia creates an interface named Matrix that declares a method for finding the smallest and largest elements along the principal and secondary diagonals of the matrix.

Sophia then creates a class named MatrixAnalyzer that implements the Matrix interface. This class provides the logic to process a given square matrix and print:

The smallest and largest elements in the principal diagonal (from top-left to bottom-right).The smallest and largest elements in the secondary diagonal (from top-right to bottom-left).

Your task is to implement the Matrix interface and the MatrixAnalyzer class. The main driver program (in the class Main) will read the input matrix, create an instance of MatrixAnalyzer, and invoke its method to display the results.

Input Format

The first line contains an integer n, representing the size of the square matrix.

The next n lines each contain n integers separated by spaces, representing the elements of the matrix.

Output Format

The output prints the four lines:

"Smallest Element - 1: <smallest element in the principal diagonal>" (integer)

"Largest Element - 1: <largest element in the principal diagonal>" (integer)

"Smallest Element - 2: <smallest element in the secondary diagonal>" (integer)

"Largest Element - 2: <largest element in the secondary diagonal>" (integer)

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
7 8 9 0 1
2 3 4 5 6
5 4 2 0 8
23 5 6 8 9
12 5 6 7 32
Output: Smallest Element - 1: 2
Largest Element - 1: 32
Smallest Element - 2: 1
Largest Element - 2: 12

Answer

```
import java.util.Scanner;


interface Matrix {
   void diagonalsMinMax(int[][] matrix);
}

class MatrixAnalyzer implements Matrix {
   public void diagonalsMinMax(int[][] matrix) {
      int n = matrix.length;
      int smallest1 = matrix[0][0];
      int largest1 = matrix[0][0];
      int smallest2 = matrix[0][n - 1];
      int largest2 = matrix[0][n - 1];

      for (int i = 0; i < n; i++) {
         if (matrix[i][i] < smallest1)
            smallest1 = matrix[i][i];
         if (matrix[i][i] > largest1)
            largest1 = matrix[i][i];

         if (matrix[i][n - i - 1] < smallest2)
            smallest2 = matrix[i][n - i - 1];
         if (matrix[i][n - i - 1] > largest2)
            largest2 = matrix[i][n - i - 1];
      }
```

```
        System.out.println("Smallest Element - 1: " + smallest1 + " ");
        System.out.println("Largest Element - 1: " + largest1 + " ");
        System.out.println("Smallest Element - 2: " + smallest2 + " ");
        System.out.println("Largest Element - 2: " + largest2 + " ");
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[][] matrix = new int[n][n];
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = sc.nextInt();
            }
        }
        MatrixAnalyzer analyzer = new MatrixAnalyzer();
        analyzer.diagonalsMinMax(matrix);
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 7_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Alex and Bob are designing a control system for household appliances, and one of the appliances is a washing machine. You want to create a program to help them that models the washing machine as a motor and calculates its electricity consumption based on its capacity.

Define an interface named Motor with the following methods:

void run() double consume(double capacity)

Create a class called WashingMachine that implements the Motor interface.

In the WashingMachine class:

Implement the run() method to print "Washing machine is running."Implement a consume() method to print "Washing machine is consuming electricity."Implement the consume(double capacity) method to calculate the electricity consumption (in kWh) of the washing machine based on its capacity. The formula for electricity consumption is (capacity * 0.05).

Input Format

The input consists of a double value representing the capacity of the washing machine in kW.

Output Format

The first line of output prints "Washing machine is running."

The second line prints "Washing machine is consuming electricity."

The third line prints "Electricity consumption: X kWh" where X is a double value, rounded off to two decimal places, representing the electricity consumption.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2.5

Output: Washing machine is running.
Washing machine is consuming electricity.
Electricity consumption: 0.13 kWh

Answer

```
import java.util.Scanner;



interface Motor {
   void run();
   double consume(double capacity);
}

class WashingMachine implements Motor {
   public void run() {
       System.out.println("Washing machine is running.");
   }

   public void consume() {
       System.out.println("Washing machine is consuming electricity.");
   }

   public double consume(double capacity) {
       return capacity * 0.05;
   }
}

public class Main {
   public static void main(String[] args) {
       Scanner scanner = new Scanner(System.in);

       WashingMachine washingMachine = new WashingMachine();

       double capacity = scanner.nextDouble();

       washingMachine.run();
       washingMachine.consume();


       double consumption = washingMachine.consume(capacity);
       System.out.printf("Electricity consumption: %.2f kWh" ,consumption);

       scanner.close();
   }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

John is developing a car loan calculator and has structured his program using two interfaces, Principal and InterestRate, defining methods for principal and interest rate retrieval.

The Loan class implements these interfaces, taking principal and annual interest rates as parameters. User input is solicited for these values, and the program ensures their validity before performing calculations. If input values are invalid (less than or equal to zero), an error message is displayed.

Note: Total interest = principal * interest rate * years

Input Format

The first line of input consists of a double value P, representing the principal.

The second line consists of a double value R, representing the annual interest rate.

The third line consists of an integer value N, representing the loan duration in years.

Output Format

If the input values are valid, print "Total interest paid: Rs. " followed by a double value, representing the total interest paid, rounded off to two decimal places.

If the input values are invalid (negative or zero values for principal, annual interest rate, or loan duration), print "Invalid input values!".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 20000.00
0.05
5

Output: Total interest paid: Rs.5000.00

Answer

```
import java.util.Scanner;

interface Principal {
   double getPrincipal();
}
```

```
interface InterestRate {
   double getInterestRate();
}

class Loan implements Principal, InterestRate {
   private double principal;
   private double interestRate;

   public Loan(double p, double rate) {
      this.principal = p;
      this.interestRate = rate;
   }

   public double getPrincipal() {
      return principal;
   }

   public double getInterestRate() {
      return interestRate;
   }

   public double calculateTotalInterest(int years) {
      return principal * interestRate * years;
   }
}
public class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      double carPrice = scanner.nextDouble();

      double annualInterestRate = scanner.nextDouble();

      int loanDuration = scanner.nextInt();

      if (carPrice <= 0 || annualInterestRate <= 0 || loanDuration <= 0) {
         System.out.println("Invalid input values!");
         return;
      }

      Loan carLoan = new Loan(carPrice, annualInterestRate);
      double totalInterest = carLoan.calculateTotalInterest(loanDuration);
```

```
        System.out.printf("Total interest paid: Rs.%.2f%n", totalInterest);
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Jeevan is developing a fitness-tracking application to monitor daily physical activity.

The application incorporates a FitnessTracker class that implements two interfaces: StepCounter for tracking the number of steps taken and CalorieCalculator for estimating total calories burned based on total steps.

Jeevan needs your help creating a program.

Note

The calorie calculation formula is: Total caloriesBurned = (total steps / 100.0) * 20.0.

Input Format

The first line of input is an integer n, representing the number of days Jeevan wants to input data.

The second line consists of space-separated integers, representing the number of steps Jeevan took on each day.

Output Format

The first line of output prints: "Total Steps: <totalSteps>", where '<totalSteps>' is the sum of steps (integer) taken over 'n' days.

The second line prints: "Calories Burned: <caloriesBurned>", where '<caloriesBurned>' is the estimated total calories (double-point number) burned based on the total steps taken rounded off to two decimal places.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
340 234 987

Output: Total Steps: 1561
Calories Burned: 312.20

Answer

```
import java.util.Scanner;


interface StepCounter {
   void countSteps(int steps);
}

interface CalorieCalculator {
   double calculateCaloriesBurned(int steps);
}

class FitnessTracker implements StepCounter, CalorieCalculator {
   private int totalSteps;

   public void countSteps(int steps) {
      totalSteps += steps;
   }

   public double calculateCaloriesBurned(int steps) {
      double caloriesBurned = (steps / 100.0) * 20.0;
      return caloriesBurned;
   }

   public int getTotalSteps() {
      return totalSteps;
   }
}

class Main
{

   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
```

```
        FitnessTracker tracker = new FitnessTracker();

        int n = scanner.nextInt();

        for (int i = 0; i < n; i++) {
            int steps = scanner.nextInt();
            tracker.countSteps(steps);
        }

        int totalSteps = tracker.getTotalSteps();
        System.out.println("Total Steps: " + totalSteps);

        double caloriesBurned = tracker.calculateCaloriesBurned(totalSteps);
        System.out.printf("Calories Burned: %.2f%n", caloriesBurned);

        scanner.close();
    }
}
```

Status : Correct **Marks : 10/10**

4. Problem Statement

Maria, an online store owner, is looking to implement a pricing system that calculates the final price of products after applying discounts. She needs a program that takes the original price of a product and the discount percentage as input and computes the final discounted price. The discount is applied as a percentage of the original price. Maria wants to ensure that the final price is formatted to display exactly two decimal places.

Implement this functionality using the PriceCalculator interface and the DiscountCalculator class.

Input Format

The first line of input consists of the original price (a double value).

The second line of input consists of a discount percentage (a double value).

Output Format

The output displays the final price after the discount, adhering to the following format: "Final Price after discount: $[final_price]".

Here, [final_price] should be replaced with the calculated final price, formatted as a currency value with two decimal places.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 100.0
10.0
Output: Final Price after discount: $90.00

Answer

```
import java.util.Scanner;



interface PriceCalculator {
   double calculatePrice(double originalPrice, double discount);
}

class DiscountCalculator implements PriceCalculator {
   public double calculatePrice(double originalPrice, double discount) {
      double discountedPrice = originalPrice - (originalPrice * discount / 100);
      return discountedPrice;
   }
}

class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      double originalPrice = scanner.nextDouble();
      double discount = scanner.nextDouble();
      PriceCalculator calculator = new DiscountCalculator();
      double finalPrice = calculator.calculatePrice(originalPrice, discount);
      System.out.printf("Final Price after discount: $%.2f%n", finalPrice); //
Formats output to 2 decimal places
   }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 8_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 15

Section 1 : MCQ

1. What will be the output for the following code?

```
class NegativeBalanceException extends Exception {
   public NegativeBalanceException(String message) {
      super(message);
   }
}

class Test {
   public static void main(String[] args) {
      try {
         double balance = -500;
         if (balance < 0) {
            throw new NegativeBalanceException("Balance cannot be
negative");
         }
```

```
        } catch (NegativeBalanceException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Answer

Error: Balance cannot be negative

Status : Correct ***Marks : 1/1***

2. Which keyword is used to explicitly throw a custom exception?

Answer

throw

Status : Correct ***Marks : 1/1***

3. what is the output of the following code?

```
class MyException extends Exception {
    public MyException(String message) {
        super(message);
    }
}

class Test {
    public static void main(String[] args) {
        try {
            throw new MyException("Error occurred");
        } catch (MyException e) {
            System.out.println(e);
        }
    }
}
```

Answer

MyException: Error occurred

Status : Correct ***Marks : 1/1***

4. Which of the following is true about custom exceptions?

Answer

Custom exceptions must extend either Exception or RuntimeException

Status : Correct ***Marks : 1/1***

5. How do you create an unchecked custom exception?

Answer

By extending RuntimeException

Status : Correct ***Marks : 1/1***

6. What will be the output for the following code?

```
import java.io.*;

class TemperatureTooHighException extends Exception {
   public TemperatureTooHighException(String message) {
      super(message);
   }
}

class Test {
   public static void main(String[] args) {
      try {
         int temperature = 110;
         if (temperature > 100) {
            throw new TemperatureTooHighException("Temperature too
high");
         }
      } catch (TemperatureTooHighException e) {
         System.out.println(e.getMessage());
      }
```

```
    }
}
```

Answer

Temperature too high

Status : Correct ***Marks : 1/1***

7. What will be the output for the following code?

```
import java.io.*;

class OutOfStockException extends Exception {
    public OutOfStockException(String message) {
        super(message);
    }
}

class Test {
    public static void main(String[] args) {
        try {
            int stock = 0;
            if (stock == 0) {
                throw new OutOfStockException("Item is out of stock");
            }
        } catch (OutOfStockException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Item is out of stock

Status : Correct ***Marks : 1/1***

8. what is the output of the following code?

```
class MyException extends Exception {
    public MyException(String message) {
        super(message);
    }
}

class Test {
    static void check() throws MyException {
        throw new MyException("Custom Exception Occurred");
    }

    public static void main(String[] args) {
        try {
            check();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Custom Exception Occurred

Status : Correct ***Marks : 1/1***

9. What will be the output for the following code?

```
class InvalidUsernameException extends Exception {
    public InvalidUsernameException(String message) {
        super(message);
    }
}

class Test {
    public static void main(String[] args) {
        try {
            String username = "abc";
            if (username.length() < 5) {
                throw new InvalidUsernameException("Username must be at
least 5 characters long");
```

```
            }
        } catch (InvalidUsernameException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Username must be at least 5 characters long

Status : Correct ***Marks : 1/1***

10. What will happen if a checked custom exception is thrown inside a method without being caught or declared?

Answer

Compilation Error

Status : Correct ***Marks : 1/1***

11. What will be the output of the following code?

```
class MyException extends Exception {
    public MyException() {
        super("Default Exception Message");
    }
}

class Test {
    public static void main(String[] args) {
        try {
            throw new MyException();
        } catch (MyException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Default Exception Message

Status : Correct ***Marks : 1/1***

12. What will be the output for the following code?

```
class InvalidVotingAgeException extends Exception {
   public InvalidVotingAgeException(String message) {
      super(message);
   }
}

class Test {
   public static void main(String[] args) {
      try {
         int age = 15;
         if (age < 18) {
            throw new InvalidVotingAgeException("You are not eligible to
vote");
         }
         System.out.println("Eligible to vote");
      } catch (InvalidVotingAgeException e) {
         System.out.println(e.getMessage());
      }
   }
}
```

Answer

You are not eligible to vote

Status : Correct ***Marks : 1/1***

13. What will be the output for the following code?

```
import java.io.*;

class NegativeAgeException extends Exception {
   public NegativeAgeException(String message) {
      super(message);
```

```
    }
}

class Test {
    public static void main(String[] args) {
        try {
            int age = -5;
            if (age < 0) {
                throw new NegativeAgeException("Age cannot be negative");
            }
        } catch (NegativeAgeException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Age cannot be negative

Status : Correct ***Marks : 1/1***

14. What is the purpose of a custom exception in Java?

Answer

To create user-defined exceptions for specific scenarios

Status : Correct ***Marks : 1/1***

15. What will be the output for the following code?

```
import java.io.*;

class UnderageException extends Exception {
    public UnderageException(String message) {
        super(message);
    }
}
```

```
class Test {
  public static void main(String[] args) {
    try {
      int age = 17;
      if (age < 18) {
        throw new UnderageException("Underage, cannot proceed");
      }
    } catch (UnderageException e) {
      System.out.println(e.getMessage());
    }
  }
}
```

Answer

Underage, cannot proceed

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Write a program to validate the email address and display suitable exceptions if there is any mistake.

Create 3 custom exception classes as below

DotExceptionAtTheRateExceptionDomainException

A typical email address should have a " . " character, and a "@" character, and also the domain name should be valid. Valid domain names for practice be 'in', 'com', 'net', or 'biz'.

Display Invalid Dot usage, Invalid @ usage, or Invalid Domain message based on email id.

Get the email address from the user, validate the email by checking the above-mentioned criteria, and print the validity status of the input email address.

Input Format

The first line of input contains the email to be validated.

Output Format

The output prints a Valid email address or an Invalid email address along with the suitable exception

If email ends with . or contains not exactly one . after @, it throws:

DotException: Invalid Dot usage

Invalid email address

If @ appears not exactly once, it throws:

AtTheRateException: Invalid @ usage

Invalid email address

If the part after the last dot is not among accepted domains:

DomainException: Invalid Domain

Invalid email address

If all conditions satisfied then print:

Valid email address

Refer to the sample input and output for format specifications.

Sample Test Case

Input: sample@gmail.com
Output: Valid email address

Answer

```
import java.util.Scanner;

class DomainException extends Exception {
    String expDescription;
    DomainException(String expDescription) {
        super(expDescription);
     }
 }

class DotException extends Exception {
    String expDescription;
    DotException(String expDescription) {
        super(expDescription);
     }
 }

class AtTheRateException extends Exception {
    String expDescription;
    AtTheRateException(String expDescription) {
        super(expDescription);
     }
 }


class EmailValidationMain {
   public static void main(String[] args) {
      Scanner myObj = new Scanner(System.in);
      String email = myObj.next();
      boolean checkEndDot  = false;
      checkEndDot = email.endsWith(".");
      int indexOfAt = email.indexOf('@');
      int lastIndexOfAt = email.lastIndexOf('.');
```

```
        int countOfAt = 0;
        for (int i = 0; i < email.length(); i++)
        {
            if(email.charAt(i)=='@')
                countOfAt++;
        }
        String buffering = email.substring(email.indexOf('@')+1, email.length());
        int len = buffering.length();
        int countOfDotAfterAt = 0;
        for (int i=0; i < len; i++) {
            if(buffering.charAt(i)=='.')
                countOfDotAfterAt++; }
        String userName = email.substring(0, email.indexOf('@'));
        String domainName = email.substring(email.indexOf('.')+1, email.length());
        int domainCheck=0;
        if((domainName.equals("in")) || (domainName.equals("com")) ||
(domainName.equals("net")) || (domainName.equals("biz")))
          domainCheck=1;

        try {
          if((checkEndDot) || (countOfDotAfterAt!=1)) {
            throw new DotException("Invalid Dot usage");
          }

          if(countOfAt!=1) {
            throw new AtTheRateException("Invalid @ usage");
          }

          if(domainCheck!=1) {
            throw new DomainException("Invalid Domain");
          }

        }catch(DotException e) {
          System.out.println(e);
        }catch(AtTheRateException e) {
          System.out.println(e);
        }catch(DomainException e) {
          System.out.println(e);
        }

        if ((countOfAt==1) && (userName.endsWith(".")==false)  &&
(domainCheck==1) && (countOfDotAfterAt ==1) &&((indexOfAt+3) <=
(lastIndexOfAt) && !checkEndDot)) {
          System.out.println("Valid email address");
        }

         else {
           System.out.println("Invalid email address");
         }
        myObj.close();
    }
}
```

Status : Correct   ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Elsa, a busy professional, is using a scheduling application to plan her meetings efficiently. The application requires users to input meeting durations in minutes, ensuring that the duration is a positive integer and does not exceed 240 minutes (4 hours). Elsa needs a program to assist her in scheduling meetings securely with proper exception handling.

Create a Java class named ElsaMeetingScheduler.Implement a custom exception:InvalidDurationException for invalid meeting duration entries.Implement the main method to interactively take user input for a meeting duration.Implement the validateMeetingDuration method to validate the meeting duration based on the specified rules and throw a custom exception if the validation fails.Print appropriate success or error messages based on the meeting duration.

Implement a custom exception, InvalidDurationException, to handle cases where the entered meeting duration does not meet the specified criteria.

Input Format

The input consists of an integer value 'n', representing the meeting duration.

Output Format

The output is displayed in the following format:

If the entered meeting duration meets the specified criteria, the program outputs

"Meeting scheduled successfully!"

If the entered meeting duration is invalid, the program outputs an error message indicating the issue.

"Error: Invalid meeting duration. Please enter a positive integer not exceeding 240 minutes (4 hours)."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 120
Output: Meeting scheduled successfully!

Answer

```
import java.util.Scanner;

class InvalidDurationException extends Exception {
   public InvalidDurationException(String message) {
      super(message);
   }
}

class MeetingValidator {
   public void validateMeetingDuration(int duration) throws
InvalidDurationException {
      if (duration <= 0 || duration > 240) {
            throw new InvalidDurationException(
                "Invalid meeting duration. Please enter a positive integer not exceeding
240 minutes (4 hours)."
            );
        }
    }
}

class MeetingScheduler {
    private MeetingValidator validator = new MeetingValidator();

    public void scheduleMeeting() {
        Scanner scanner = new Scanner(System.in);

        try {
            int meetingDuration = scanner.nextInt();

            validator.validateMeetingDuration(meetingDuration);

            System.out.println("Meeting scheduled successfully!");
        } catch (InvalidDurationException | java.util.InputMismatchException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
public class Main {
    public static void main(String[] args) {
        MeetingScheduler scheduler = new MeetingScheduler();
        scheduler.scheduleMeeting();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 8_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Enigma is developing a simple web application that takes a user-input URL, validates it, and throws a custom exception InvalidURLFormatException if the URL does not start with "http://" or "https://".

The main method prompts the user for input, validates the URL, and prints whether it is valid or not.

Input Format

The input consists of a string, representing the URL entered by the user.

Output Format

The output displays one of the following results:

If the entered URL is valid according to the specified format, the program prints:

"[URL] is a valid URL"

If the entered URL is not valid according to the specified format, the program prints:

"Invalid URL format: [URL]"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: http://www.example.com

Output: http://www.example.com is a valid URL

Answer

```
import java.util.Scanner;
class WebApplication {
    public static void main(String[] args) {
        String userInputURL = getUserInputURL();
        try {
            validateURL(userInputURL);
            System.out.println(userInputURL+" is a valid URL");
        } catch (InvalidURLFormatException e) {
            System.out.println(e.getMessage()+userInputURL);
        }
    }

    private static String getUserInputURL() {
        Scanner scanner = new Scanner(System.in);
        return scanner.nextLine();
    }

    private static void validateURL(String url) throws InvalidURLFormatException {
        if (!(url.startsWith("http://") || url.startsWith("https://"))) {
            throw new InvalidURLFormatException("Invalid URL format: ");
        }
    }
}
class InvalidURLFormatException extends Exception {
    public InvalidURLFormatException(String message) {
        super(message);
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

You are tasked to create a program that defines a custom exception GradeException. The program should include a Student class with fields for the student's name, age, and grade. Implement a method in the Student class that checks the grade, and if the grade is below 40, it should throw a GradeException. Otherwise, it should display the student's details.

Input Format

The input consists of three parameters in separate lines:

1. A string representing the student's name.
2. An integer representing the student's age.
3. An integer representing the student's grade.

Output Format

The output will display the student's details if the grade is valid.

If the grade is below 40, the program will display an error message "Grade is below 40".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Alice
20

85
Output: Name: Alice
Age: 20
Grade: 85

Answer

```
import java.util.Scanner;
class GradeException extends Exception {
   GradeException(String message) {
      System.out.println(message);
   }
}

class Student {
   String name;
   int age;
   int grade;

   Student(String studentName, int studentAge, int studentGrade) {
      name = studentName;
      age = studentAge;
      grade = studentGrade;
   }

   void validateGrade() throws GradeException {
      if (grade < 40) {
         throw new GradeException("Grade is below 40");
      }
   }

   void displayDetails() {
      System.out.println("Name: " + name);
      System.out.println("Age: " + age);
      System.out.println("Grade: " + grade);
   }
}
public class Main {
   public static void main(String[] args) {
      try {
         java.util.Scanner scanner = new java.util.Scanner(System.in);
         String studentName = scanner.nextLine();
         int studentAge = scanner.nextInt();
```

```
            int studentGrade = scanner.nextInt();
            Student student = new Student(studentName, studentAge, studentGrade);
            student.validateGrade();
            student.displayDetails();
            scanner.close();
        } catch (GradeException e) {

        }
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

An HR software system is being developed to process employee payrolls. During payroll processing, the system must ensure that no employee has a negative salary and that no employee's salary exceeds   2,00,000. If either condition occurs, the system should throw a custom exception.

Create a custom exception InvalidSalaryException and a class Employee that processes salary according to the following rules:

If salary < 0, throw InvalidSalaryException with the message: "Salary cannot be negative". If salary > 200000, throw InvalidSalaryException with the message: "Salary exceeds threshold limit". Otherwise, display: "Salary processed successfully for <empName>: <salary>".

The payroll processing should always display: "Payroll process completed" at the end, regardless of whether an exception occurs.

Input Format

The first line of input contains an integer representing the employee ID.

The second line contains a string representing the employee's name.

The third line contains a floating-point number representing the salary of the employee.

Output Format

If the salary is valid: "Salary processed successfully for <empName>: <salary>"

"Payroll process completed"

If the salary is invalid: "<Exception Message>"

"Payroll process completed"

Refer to the sample output for formatting specifications.

Sample Test Case

```
Input: 101
Rahul
150000.0
Output: Salary processed successfully for Rahul: 150000.0
Payroll process completed
```

Answer

```
import java.util.Scanner;

class InvalidSalaryException extends Exception {
   public InvalidSalaryException(String message) {
      super(message);
   }
}

class Employee {
   int empId;
   String empName;
   double salary;

   public Employee(int empId, String empName, double salary) {
      this.empId = empId;
      this.empName = empName;
      this.salary = salary;
   }

   public void processSalary() throws InvalidSalaryException {
      if (salary < 0) {
            throw new InvalidSalaryException("Salary cannot be negative");
        }
        if (salary > 200000) {
            throw new InvalidSalaryException("Salary exceeds threshold limit");
        }
        System.out.println("Salary processed successfully for " + empName + ": " +
salary);
    }
}

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            int empId = scanner.nextInt();
            scanner.nextLine();
            String empName = scanner.nextLine();
            double salary = scanner.nextDouble();
            Employee employee = new Employee(empId, empName, salary);
            try {
                employee.processSalary();
            } catch (InvalidSalaryException e) {
                System.out.println(e.getMessage());
            }
        } finally {
            System.out.println("Payroll process completed");
            scanner.close();
        }
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Daniel is developing a program to verify the age of users. He wants to ensure that the entered age is within a valid range. Write a program to help Daniel implement this age-checking feature using custom exceptions.

Daniel needs a program that takes an integer input representing a person's age.If the age is between 0 and 150 (inclusive), the program should print "Age is valid!".If the age is less than 0 or greater than 150, the program should throw a custom exception (InvalidAgeException) with the message "Invalid age. Please enter an age between 0 and 150."

Implement a custom exception, InvalidAgeException, to handle cases where the entered age does not meet the specified criteria.

Input Format

The input consists of an integer value 'n', representing the age.

Output Format

The output is displayed in the following format:

If the age is valid (between 0 and 150, inclusive), print

"Age is valid!".

If the age is invalid, print

"Error: Invalid age. Please enter an age between 0 and 150."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 45
Output: Age is valid!

Answer

```
import java.util.Scanner;
class AgeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            int age = scanner.nextInt();
            checkAge(age);
            System.out.println("Age is valid!");
        } catch (InvalidAgeException | java.util.InputMismatchException e) {
            System.out.println("Error: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }

    private static void checkAge(int age) throws InvalidAgeException {
        if (age < 0 || age > 150) {
            throw new InvalidAgeException("Invalid age. Please enter an age between
0 and 150.");
        }
    }
}
class InvalidAgeException extends Exception {
    public InvalidAgeException(String message) {
        super(message);
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 8_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Theo is trying to update his payment information on a subscription-based streaming service. To proceed, the system requires Theo to provide a valid credit card number consisting of 16 digits. However, Theo wants to make sure that the credit card number he enters meets the specified criteria with proper exception handling.

The credit card number must consist of exactly 16 digits.If the entered credit card number does not meet the specified criteria, the program should throw a custom exception, InvalidCreditCardException, and provide Theo with specific error messages:If the length of the credit card number is not 16 digits, the exception message should be: "Invalid credit card number length."If the credit card number contains non-numeric characters, the exception message should be: "Invalid credit card number format."

Implement a custom exception, InvalidCreditCardException, to fulfill Theo's requirements and keep his payment information secure.

Input Format

The input consists of a string value 's', consisting of the 16-digit credit card number.

Output Format

The output is displayed in the following format:

If the entered credit card number is valid, the program should output a success message:

"Payment information updated successfully!"

If the entered credit card has more than 16 digits or less than 16 digits it displays

"Error: Invalid credit card number length."

If the entered 16-digit credit card has non-integers it displays

"Error: Invalid credit card number format."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1234567890123456
Output: Payment information updated successfully!

Answer

```
import java.util.Scanner;

class InvalidCreditCardException extends Exception {
   public InvalidCreditCardException(String message) {
      super(message);
   }
}
```

```
class CreditCardValidator {
   public void validateCreditCardNumber(String creditCardNumber) throws
InvalidCreditCardException {
      if (!creditCardNumber.matches("^\\d{16}$")) {
         if (creditCardNumber.length() != 16) {
            throw new InvalidCreditCardException("Invalid credit card number
length.");
         } else {
            throw new InvalidCreditCardException("Invalid credit card number
format.");
         }
      }
   }
}

class CreditCardUpdater {
   private CreditCardValidator validator = new CreditCardValidator();

   public void updateCreditCard() {
      Scanner scanner = new Scanner(System.in);
      try {
         String creditCardNumber = scanner.nextLine();

         validator.validateCreditCardNumber(creditCardNumber);

         System.out.println("Payment information updated successfully!");
      } catch (InvalidCreditCardException | java.util.InputMismatchException e) {
         System.out.println("Error: " + e.getMessage());
      } finally {
         scanner.close();
      }
   }
}
public class Main {
   public static void main(String[] args) {
      CreditCardUpdater updater = new CreditCardUpdater();
      updater.updateCreditCard();
   }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Faustus is managing his bank account and wants to create a program to update his account balance based on certain conditions. However, he needs to handle specific scenarios related to invalid inputs and insufficient balances. Faustus wants to update his account balance. He inputs the current balance and the amount to be updated.

The initial account balance should be positive. If Faustus enters a negative initial balance, the program should throw an InvalidAmountException with the message "Invalid amount. Please enter a positive initial balance."If the amount to be updated is negative, the program should check if the subtraction results in a negative balance. If so, it should throw an InsufficientBalanceException with the message "Insufficient balance."If the amount to be updated is positive, it should be added to the current balance, and the new balance should be printed.

Implement a custom exception, InvalidAmountException, and InsufficientBalanceException, to manage his bank account.

Input Format

The first line of input consists of a double value 'd', representing the initial account balance.

The second line of input consists of a double value 'd1', representing the amount to be updated.

Output Format

The output is displayed in the following format:

If the validation passes, print

"Account balance updated successfully! New balance: {new_balance}"

where {new_balance} is the updated account balance.

If the initial bank amount is negative it displays

"Error: Invalid amount. Please enter a positive initial balance."

If the updated amount exceeds the initial account balance in withdrawal it displays

"Error: Insufficient balance."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1000
500
Output: Account balance updated successfully! New balance: 1500.0

Answer

```
import java.util.Scanner;
class BalanceUpdater {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      try {
         double currentBalance = scanner.nextDouble();
         if (currentBalance < 0) {
            throw new InvalidAmountException("Invalid amount. Please enter a
positive initial balance.");
         }
         double updateAmount = scanner.nextDouble();
         updateBalance(currentBalance, updateAmount);
         System.out.println("Account balance updated successfully! New balance:
" + (currentBalance + updateAmount));
      } catch (InvalidAmountException | InsufficientBalanceException |
java.util.InputMismatchException e) {
         System.out.println("Error: " + e.getMessage());
      } finally {
         scanner.close();
      }
   }

   private static void updateBalance(double currentBalance, double
updateAmount)
         throws InvalidAmountException, InsufficientBalanceException {
      if (updateAmount < 0) {
            if (currentBalance + updateAmount < 0) {
                throw new InsufficientBalanceException("Insufficient balance.");
            }
        } else {
            currentBalance += updateAmount;
        }
    }
}
class InvalidAmountException extends Exception {
    public InvalidAmountException(String message) {
        super(message);
    }
}

class InsufficientBalanceException extends Exception {
    public InsufficientBalanceException(String message) {
        super(message);
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Alice is designing a program that requires users to enter positive numbers. She wants to implement a solution that validates whether the entered number is positive. In case the input is not a positive number, she wants to throw a custom exception.

The number should be a positive integer.If this condition is violated, the program should throw a custom exception:InvalidPositiveNumberException with the message "Invalid input. Please enter a positive integer."

Implement a custom exception, InvalidPositiveNumberException , to handle cases where the entered number does not meet the specified criteria.

Input Format

The input consists of an integer value 'n', representing the entered number.

Output Format

The output is displayed in the following format:

If the validation passes, print

"Number {number} is positive."

The {number} represents the entered positive integer.

If the entered number is negative then it displays

"Error: Invalid input. Please enter a positive integer."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 100
Output: Number 100 is positive.

Answer

```
import java.util.Scanner;
class PositiveNumberValidator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            int number = scanner.nextInt();
            validatePositiveNumber(number);
            System.out.println("Number " + number + " is positive.");
        } catch (InvalidPositiveNumberException | java.util.InputMismatchException
e) {
            System.out.println("Error: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }

    private static void validatePositiveNumber(int number) throws
InvalidPositiveNumberException {
        if (number <= 0) {
            throw new InvalidPositiveNumberException("Invalid input. Please enter a 
positive integer.");
        }
    }
}
class InvalidPositiveNumberException extends Exception {
    public InvalidPositiveNumberException(String message) {
        super(message);
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Hemanth is designing a banking system for XYZ Bank. The system should allow customers to perform deposit, withdrawal, and balance inquiry operations. Implement exception handling for scenarios involving invalid transaction amounts or insufficient funds.

Create two custom exception classes, InvalidAmountException and InsufficientFundsException, both extending the Exception class.Throw an InvalidAmountException with a message if the deposit amount is less than or equal to zero.Throw an InsufficientFundsException if the withdrawal amount is greater than the available balance.Deduct the withdrawal amount from the balance if the withdrawal is successful.

Assist Hemanth in designing the program.

Input Format

The first line of input consists of a double value B, representing the initial balance.

The second line consists of a double value D, representing the deposit amount.

The third line consists of a double value W, representing the withdrawal amount.

Output Format

If the withdrawal is successful, print the amount withdrawn and the current balance, rounded off to one decimal place.

If an InvalidAmountException occurs, print "Error: [D] is not valid".

If an InsufficientFundsException occurs, print "Error: Insufficient funds".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1050.1
270.2
150.3
Output: Amount Withdrawn: 150.3
Current Balance: 1170.0

Answer

```
import java.util.Scanner;

class InvalidAmountException extends Exception {
   public InvalidAmountException(String message) {
      super(message);
   }
}

class InsufficientFundsException extends Exception {
   public InsufficientFundsException(String message) {
      super(message);
   }
}

class HDFCBank {
   private double balance;

   public static void main(String[] args) {
      HDFCBank hdfcBank = new HDFCBank();
      hdfcBank.processTransactions();
   }

   public void processTransactions() {
      Scanner scanner = new Scanner(System.in);

      try {
          balance = scanner.nextDouble();
          double depositAmount = scanner.nextDouble();
          deposit(depositAmount);

          double withdrawAmount = scanner.nextDouble();
          double withdrawnAmount = withdraw(withdrawAmount);

          System.out.printf("Amount Withdrawn: %.1f\n" ,withdrawnAmount);
          balanceEnquiry();
      } catch (InvalidAmountException | InsufficientFundsException e) {
          System.out.println("Error: " + e.getMessage());
      } finally {
          scanner.close();
      }
   }

   public void deposit(double amount) throws InvalidAmountException {
      if (amount <= 0) {
          throw new InvalidAmountException(amount + " is not valid");
      }
      balance = balance + amount;
   }

   public double withdraw(double amount) throws InsufficientFundsException {
      if (balance < amount) {
          throw new InsufficientFundsException("Insufficient funds");
      }
      balance = balance - amount;
      return amount;
   }

   public void balanceEnquiry() {
      System.out.printf("Current Balance: %.1f\n" ,balance);
   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 9_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 14

Section 1 : MCQ

1. Which method is used to add an element to the top of the stack?

Answer

push()

Status : Correct ***Marks : 1/1***

2. What will be the output of the following code?

```
import java.util.ArrayList;

public class Main {
   public static void main(String[] args) {
      ArrayList<Integer> list = new ArrayList<>();
      list.add(10);
```

```
        list.add(20);
        list.add(30);
        System.out.println("Size of the list: " + list.size());
    }
}
```

Answer

Size of the list: 3

Status : Correct ***Marks : 1/1***

3. What is the correct way to create an ArrayList in Java?

Answer

ArrayList&lt;String&gt; list = new ArrayList&lt;&gt;();

Status : Correct ***Marks : 1/1***

4. What will be the output of the following code?

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>();
        list.add(10);
        list.add(20);
        list.add(30);
        list.remove(1);
        System.out.println(list);
    }
}
```

Answer

[10, 30]

Status : Correct ***Marks : 1/1***

5. What will be the output of the following code?

```
import java.util.*;
class Main {
   public static void main(String[] args) {
      ArrayList<String> list = new ArrayList<>();
      list.add("apple");
      list.add("banana");
      list.add("cherry");
      list.add("banana");
      System.out.println(list.lastIndexOf("banana"));
   }
}
```

Answer

3

Status : Correct ***Marks : 1/1***

6. What will be the output of the following code?

```
import java.util.ArrayList;

public class Main {
   public static void main(String[] args) {
      ArrayList<String> list = new ArrayList<>();
      list.add("Apple");
      list.add("Banana");
      list.remove("Apple");
      System.out.println(list);

   }
}
```

Answer

[Banana]

Status : Correct ***Marks : 1/1***

7. How can you access the first element of an ArrayList named as list?

Answer

list.get(0);

Status : Correct ***Marks : 1/1***

8. What is Collection in Java?

Answer

A group of interfaces

Status : Wrong ***Marks : 0/1***

9. What will be the output of the following code?

```
import java.util.*;
class Main {
   public static void main(String[] args) {
      ArrayList<String> list = new ArrayList<>();
      list.add("Java");
      list.add("Python");
      list.add("Java");
      list.add("C++");
      System.out.println(list.indexOf("Java"));
   }
}
```

Answer

0

Status : Correct ***Marks : 1/1***

10. What will be the output of the following code?

```
import java.util.*;
class Main {
   public static void main(String[] args) {
      ArrayList<Integer> list = new ArrayList<>();
```

```
        list.add(1);
        list.add(2);
        list.add(3);
        list.add(4);
        list.add(5);
        System.out.println(list.get(3));
    }
}
```

Answer

4

Status : Correct ***Marks : 1/1***

11. Which of the following methods removes and returns the last element from a LinkedList?

Answer

removeLast()

Status : Correct ***Marks : 1/1***

12. What does the addFirst() method of LinkedList do?

Answer

Adds an element to the beginning of the list

Status : Correct ***Marks : 1/1***

13. What will be the output of the following code?

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Stack<Integer> s = new Stack<>();
        s.push(10);
        s.push(20);
        s.push(30);
        System.out.println(s.peek());
    }
}
```

Answer

30

Status : Correct ***Marks : 1/1***

14. What will be the output of the following code?

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Stack<Integer> stack = new Stack<>();
        for (int i = 1; i <= 3; i++)
            stack.push(i * 2);
        stack.pop();
        stack.push(10);
        System.out.println(stack.peek());
    }
}
```

Answer

10

Status : Correct ***Marks : 1/1***

15. What will be the output of the following code?

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
        list.add(4);
        list.set(2, 10);
        System.out.println(list);
    }
}
```

Answer

[1, 2, 10, 4]

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 9_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Bobby is tasked with processing a sequence of numbers from a monitoring system. He needs to extract a strictly increasing subsequence using an ArrayList. The program should dynamically add numbers to the ArrayList only if they are greater than the last number currently stored in the list. Bobby aims to efficiently utilize the dynamic resizing and indexing features of the ArrayList to solve this problem.

Help Bobby implement this solution.

Input Format

The first line of input consists of an integer N, representing the number of elements.

The second line consists of N space-separated integers, representing the elements.

Output Format

The output prints the list of integers in increasing sequence, ignoring out-of-order elements.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 7
3 5 9 1 11 7 13
Output: [3, 5, 9, 11, 13]

Answer

```
import java.util.ArrayList;
import java.util.Scanner;

class NumberProcessor {
   private ArrayList<Integer> numList;

   public NumberProcessor(ArrayList<Integer> numList) {
      this.numList = numList;
   }

   public void processNumbers() {
      ArrayList<Integer> filteredList = new ArrayList<>();
      for (int num : numList) {
         if (filteredList.isEmpty() || num > filteredList.get(filteredList.size() - 1)) {
            filteredList.add(num);
         }
      }
      System.out.println(filteredList);
   }
}
public class Main {
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      int number_of_elements = input.nextInt();
```

```
        if (number_of_elements <= 0) {
            return;
        }

        ArrayList<Integer> numList = new ArrayList<>();
        for (int ctr = 0; ctr < number_of_elements; ctr++) {
            numList.add(input.nextInt());
        }

        NumberProcessor processor = new NumberProcessor(numList);
        processor.processNumbers();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 9_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Vikram loves listening to music and wants to create a simple playlist manager using Java Collections. The playlist supports the following operations:

"ADD <song>"   Adds the song to the end of the playlist."REMOVE <song>"   Removes the first occurrence of the song from the playlist. If the song is not found, do nothing."SHOW"   Displays all songs in the playlist in order. If the playlist is empty, print "EMPTY"."NEXT"   Moves to the next song in the playlist and prints its name. If the playlist is empty, print "EMPTY".

The playlist maintains a "current song" position that starts at the first song when it's added. The NEXT command moves to the next song and prints it, wrapping around to the first song after reaching the last song. When removing songs, the current position adjusts accordingly to maintain proper navigation.

Help Vikram implement this playlist manager.

Input Format

The first line of the input consists of an integer n, the number of operations.

The next n lines, each containing a command:

- "ADD <song>"
- "REMOVE <song>"
- "SHOW"
- "NEXT"

Output Format

For each "SHOW" command, print the songs in order, separated by spaces.

For each "NEXT" command, print the next song in the playlist.

If no song exists, print "EMPTY".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
ADD song1
ADD song2
SHOW
NEXT
REMOVE song2
SHOW
NEXT

Output: song1 song2
song2
song1
song1

Answer

```
import java.util.*;
class Playlist {
    private LinkedList<String> playlist;
    private int currentIndex;

    public Playlist() {
        playlist = new LinkedList<>();
        currentIndex = -1;
    }

    public void addSong(String song) {
        playlist.add(song);
        if (currentIndex == -1) {
            currentIndex = 0;
        }
    }

    public void removeSong(String song) {
        int idx = playlist.indexOf(song);
        if (idx != -1) {
            playlist.remove(idx);
            if (playlist.isEmpty()) {
                currentIndex = -1;
            } else if (idx <= currentIndex && currentIndex > 0) {
                currentIndex--;
            }
        }
    }

    public void showPlaylist() {
        if (playlist.isEmpty()) {
            System.out.println("EMPTY");
        } else {
            for (String s : playlist) {
                System.out.print(s + " ");
            }
            System.out.println();
        }
    }

    public void nextSong() {
        if (playlist.isEmpty()) {
            System.out.println("EMPTY");
        } else {
            currentIndex++;
            if (currentIndex >= playlist.size()) {
                currentIndex = 0;
            }
            System.out.println(playlist.get(currentIndex));
        }
    }
}
class PlaylistManager {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());

        Playlist playlist = new Playlist();

        for (int i = 0; i < n; i++) {
            String command = sc.nextLine();

            if (command.startsWith("ADD ")) {
                String song = command.substring(4);
                playlist.addSong(song);
            } else if (command.startsWith("REMOVE ")) {
                String song = command.substring(7);
                playlist.removeSong(song);
            } else if (command.equals("SHOW")) {
                playlist.showPlaylist();
            } else if (command.equals("NEXT")) {
                playlist.nextSong();
            }
        }

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 9_PAH

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Rekha is a teacher who wants to calculate the average of marks scored by her students in a test. She needs to store all the marks dynamically because the number of students may vary each time. Using an ArrayList allows her to easily add any number of marks without worrying about the initial size.

Help her implement the task.

Input Format

The first line of input is an integer n, representing the number of students..

The second line of input consists of n double values, representing the marks of each student, separated by a space.

Output Format

The output prints: "Average of the list: " followed by the average value formatted to two decimal places.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
1.0 2.0 3.0 4.0 5.0

Output: Average of the list: 3.00

Answer

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class AverageCalculator {
   private List<Double> numbers;

   public AverageCalculator() {
      numbers = new ArrayList<>();
   }

   public void addNumber(double num) {
      numbers.add(num);
   }

   public double calculateAverage() {
      double sum = 0;
      for (double num : numbers) {
         sum += num;
      }
      return sum / numbers.size();
   }
}
class Main {
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
```

```
        int n = input.nextInt();

        AverageCalculator calculator = new AverageCalculator();

        for (int i = 0; i < n; i++) {
            double num = input.nextDouble();
            calculator.addNumber(num);
        }

        double average = calculator.calculateAverage();
        System.out.println("Average of the list: " + String.format("%.2f", average));

        input.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Arun is building a task manager to keep track of tasks using a LinkedList. The task manager supports the following operations:

"ADD <task>" Adds the given task to the end of the list."REMOVE" Removes the first task from the list."SHOW" Displays all tasks in the list in order. If the list is empty, print "EMPTY".

Help Arun implement this functionality using a LinkedList.

Input Format

The first line of the input consists of an integer n, the number of operations.

The next n lines, each containing a command:

- "ADD <task>"
- "REMOVE"
- "SHOW"

Output Format

For each "SHOW" command, the output prints the tasks in order, separated by spaces.

If no tasks exist, print "EMPTY".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
ADD homework
ADD project
SHOW
REMOVE
SHOW

Output: homework project
project

Answer

```
import java.util.*;

class TaskManager {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        LinkedList<String> tasks;
        sc.nextLine();
         tasks = new LinkedList<>();

        for (int i = 0; i < n; i++) {
            String command = sc.nextLine();

            if (command.startsWith("ADD")) {
                String task = command.substring(4);
                tasks.add(task);
            } else if (command.equals("REMOVE")) {
                if (!tasks.isEmpty()) {
                    tasks.removeFirst();
                }
            } else if (command.equals("SHOW")) {
                if (tasks.isEmpty()) {
```

```
                System.out.println("EMPTY");
            } else {
                System.out.println(String.join(" ", tasks));
            }
        }
    }
    sc.close();
  }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Aditi is analyzing stock market trends and wants to find the Next Greater Element (NGE) for each stock price in a list. The Next Greater Element for an element x in an array is the first element to the right that is greater than x. If no greater element exists, return -1 for that position.

Your task is to help Aditi by efficiently computing the Next Greater Element for each element in the given array using a Stack.

Example:

Input:

6

4 5 2 10 8 6

Output:

5 10 10 -1 -1 -1

Explanation:

For each element:

4 5 (next greater element)5 102 1010 -1 (No greater element)8 -16 -1

Input Format

The first line contains an integer n, representing the number of elements.

The second line contains n space-separated integers arr[i], where arr[i] is the stock price on the i-th day.

Output Format

The output prints n space-separated integers representing the Next Greater Element for each element in the array.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
4 5 2 10 8 6

Output: 5 10 10 -1 -1 -1

Answer

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        sc.close();

        int[] nge = new int[n];
        Stack<Integer> stack = new Stack<>();

        for (int i = n - 1; i >= 0; i--) {
            while (!stack.isEmpty() && stack.peek() <= arr[i]) {
                stack.pop();
            }
            nge[i] = stack.isEmpty() ? -1 : stack.peek();
            stack.push(arr[i]);
        }
        for (int i = 0; i < n; i++) {
            System.out.print(nge[i] + " ");
        }
    }
}
```

Status : Correct **_Marks : 10/10_**

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 9_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

A teacher is filtering a list of words provided by students. Some words contain too many vowels, making them difficult for a spelling competition. The teacher decides to remove all words that contain more than two vowels.

Help the teacher to implement it using ArrayList.

Input Format

The first line contains an integer N, representing the number of words in the list.

The next N lines contain a string representing the words (one per line).

Output Format

The output consists of words that contain two or less than two vowels, printed in the same order they appeared in the input. Each word is printed on a new line.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1
sri

Output: sri

Answer

```
import java.util.ArrayList;
import java.util.Scanner;

// You are using Java
class VowelFilter {
   //Type your code here
}import java.util.ArrayList;
import java.util.Scanner;

public class WordFilter {

   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      ArrayList<String> words = new ArrayList<>();

      System.out.println("Enter words (type 'done' to finish):");
      String input;
      while (!(input = scanner.nextLine()).equalsIgnoreCase("done")) {
         words.add(input);
      }

      ArrayList<String> filteredWords = new ArrayList<>();
      for (String word : words) {
         int vowelCount = 0;
         for (char c : word.toLowerCase().toCharArray()) {
            if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {
               vowelCount++;
            }
            }
            if (vowelCount <= 2) {
                filteredWords.add(word);
            }
        }

        System.out.println("Filtered words (words with at most two vowels):");
        for (String word : filteredWords) {
            System.out.println(word);
        }

        scanner.close();
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        VowelFilter.filterWords(n, sc);
        sc.close();
    }
}
```

Status : Wrong ***Marks : 0/10***

2. Problem Statement

Aarav is developing a music playlist application where users can manage their favorite songs. He wants to implement a feature that allows users to reorder the playlist by moving a song from one position to another.

You need to implement a function that performs the following operations using a LinkedList:

Add songs to the playlist in the given order.Move a song from a specified position to another position in the playlist.Print the final playlist after all operations.

Input Format

The first line of the input consists of an integer n representing the number of songs.

The next n lines, each containing a string representing a song name.

After the songs are given the next line contains an integer m, the number of move operations.

The next m lines, each containing two integers x and y representing the move operation where the song at position x (0-based index) should be moved to position y.

Output Format

The output prints the final playlist, each song on a new line.

Refer to the sample output for formatting specifications.

Sample Test Case

```
Input: 5
SongA
SongB
SongC
SongD
SongE
2
2 4
0 3
```

```
Output: SongB
SongD
SongE
SongA
SongC
```

Answer

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        LinkedList<String> playlist = new LinkedList<>();
        for (int i = 0; i < n; i++) playlist.add(sc.nextLine());
        int m = sc.nextInt();
        for (int i = 0; i < m; i++) {
            int x = sc.nextInt();
            int y = sc.nextInt();
            String song = playlist.remove(x);
            playlist.add(y, song);
        }
        for (String song : playlist) System.out.println(song);
    }
}
```

Status : Correct **Marks : 10/10**

3. Problem Statement

Rahul is working on a list manipulation problem where he needs to reverse a specific subarray using a stack. Given an array and two indices l and r, he wants to reverse only the portion of the array from index l to r (both inclusive) while keeping the rest of the array unchanged.

Since Rahul wants to solve this problem efficiently, he decides to use a stack to reverse the subarray in O(r - l) time.

Your task is to help Rahul by implementing this functionality.

Input Format

The first line contains an integer n, the size of the array.

The second line contains n space-separated integers arr[i].

The third line contains two integers l and r, denoting the start and end indices of the subarray to reverse.

Note: The array follows 0-based indexing.

Output Format

The output prints the modified array after reversing the subarray between indices l and r.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
1 2 3 4 5 6
1 4
Output: 1 5 4 3 2 6

Answer

```
import java.util.*;

public class Main {


    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];

        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        int l = sc.nextInt();
        int r = sc.nextInt();
        sc.close();

        Stack<Integer> stack = new Stack<>();

        for (int i = l; i <= r; i++) {
            stack.push(arr[i]);
        }

        for (int i = l; i <= r; i++) {
            arr[i] = stack.pop();
        }
        for (int i = 0; i < n; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Rahul, a stock trader, wants to analyze the stock prices of a company over several days. For each day, he wants to determine the stock span, which is the number of consecutive days (including the current day) where the stock price is less than or equal to the price on that day.

The stock span helps him understand how long a stock has been continuously increasing or staying the same. You need to help Rahul by computing the stock span for each day using a Stack data structure efficiently.

Example:

Input:

7

100 80 60 70 60 75 85

Output:

1 1 1 2 1 4 6

Explanation:

For each day:

Day 1: Price = 100   Span = 1 (Only this day)Day 2: Price = 80   Span = 1 (Only this day)Day 3: Price = 60   Span = 1 (Only this day)Day 4: Price = 70   Span = 2 (Includes today and previous day)Day 5: Price = 60   Span = 1 (Only this day)Day 6: Price = 75   Span = 4 (Includes today and previous three days)Day 7: Price = 85   Span = 6 (Includes today and previous five days)

Input Format

The first line contains an integer n, the number of days.

The second line contains n space-separated integers prices[i], where prices[i] represents the stock price on the i-th day.

Output Format

The output prints n space-separated integers representing the stock span for each day.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
100 80 60 70 60 75 85

Output: 1 1 1 2 1 4 6

Answer

```
import java.util.*;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int n = sc.nextInt();
      int[] prices = new int[n];
      for (int i = 0; i < n; i++) {
         prices[i] = sc.nextInt();
      }
      sc.close();

      int[] span = new int[n];
      Stack<Integer> stack = new Stack<>();

      for (int i = 0; i < n; i++) {
         while (!stack.isEmpty() && prices[stack.peek()] <= prices[i]) {
            stack.pop();
         }
         span[i] = stack.isEmpty() ? (i + 1) : (i - stack.peek());
```

```
            stack.push(i);
        }

        for (int i = 0; i < n; i++) {
            System.out.print(span[i] + " ");
        }
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 10_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 15

Section 1 : MCQ

1. What is the time complexity of retrieving an element from a HashSet?

Answer

O(1)

Status : Correct *Marks : 1/1*

2. What will be the output of the following code?

```
import java.util.*;
class Main {
   public static void main(String[] args) {
      HashMap<String, Integer> map = new HashMap<>();
      map.put("A", 1);
      map.put("B", 2);
```

```
        map.put("C", 3);
        System.out.println(map.containsKey("B"));
    }
}
```

Answer

true

Status : Correct ***Marks : 1/1***

3. What will happen if you add elements in descending order in a TreeSet?

Answer

They are sorted in ascending order

Status : Correct ***Marks : 1/1***

4. Which method retrieves the lowest key in a TreeMap?

Answer

firstKey()

Status : Correct ***Marks : 1/1***

5. How does HashSet check for duplicate elements?

Answer

Using equals() and hashCode()

Status : Correct ***Marks : 1/1***

6. What will be the output of the following code?

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        HashMap<String, Integer> map = new HashMap<>();
        map.put("X", 10);
        map.put("Y", 20);
        map.put("Z", 30);
        map.remove("Y");
        System.out.println(map);
    }
}
```

Answer

{X=10, Z=30}

Status : Correct ***Marks : 1/1***

7. What happens when you add duplicate elements to a HashSet?

Answer

The duplicate is ignored

Status : Correct ***Marks : 1/1***

8. What will be the output of the following code?

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        HashMap<String, String> map = new HashMap<>();
        map.put("A", "Apple");
        map.put("B", "Banana");
        map.put("C", "Cherry");
        map.replace("B", "Blueberry");
        System.out.println(map);
    }
}
```

Answer

{A=Apple, B=Blueberry, C=Cherry}

Status : Correct ***Marks : 1/1***

9. Which method removes all elements from a Set?

Answer

clear()

Status : Correct ***Marks : 1/1***

10. What will happen if you add a null element to a TreeSet?

Answer

An exception occurs

Status : Correct ***Marks : 1/1***

11. Which statement is true about HashSet and TreeSet?

Answer

TreeSet provides sorted elements

Status : Correct ***Marks : 1/1***

12. Which of the following is true about TreeMap?

Answer

It maintains natural ordering

Status : Correct ***Marks : 1/1***

13. Which of the following allows null keys in Java?

Answer

HashMap

Status : Correct ***Marks : 1/1***

14. Which of the following is true about HashMap?

Answer

It is not synchronized

Status : Correct ***Marks : 1/1***

15. What happens if two keys have the same hash code in a HashMap?

Answer

A linked list is used to store values with the same hash

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

A city traffic management system needs to track vehicles entering a toll booth. Each vehicle is uniquely identified by its registration number. The system should allow adding vehicles to a record, ensuring that no duplicate registration numbers exist. The vehicles should be stored in a HashSet, which does not guarantee any specific order.

Your task is to implement a program using a HashSet that allows adding vehicle details and displaying the records.

Input Format

The first line of input contains an integer N - the number of vehicles.

The next N lines contain details of each vehicle in the format: "RegNumber

OwnerName VehicleType"

1. RegNumber (String) - A unique registration number (Alphanumeric).
2. OwnerName (String) - The name of the vehicle owner.
3. VehicleType (String, Car, Bike, or Truck) - The type of vehicle.

If a vehicle with the same registration number is already present, ignore the duplicate entry.

Output Format

The output prints the unique vehicle records in any order (since HashSet does not maintain order).

Output format: "RegNumber OwnerName VehicleType"

Refer to the sample output for formatting specifications.

Sample Test Case

```
Input: 5
KA01AB1234 John Car
MH02CD5678 Alice Bike
DL03EF9012 Bob Truck
TN04GH3456 Mike Car
KA01AB1234 John Car
Output: TN04GH3456 Mike Car
KA01AB1234 John Car
MH02CD5678 Alice Bike
DL03EF9012 Bob Truck
```

Answer

```
import java.util.*;
class Vehicle {
   String regNumber;
   String ownerName;
   String vehicleType;
   private static HashSet<Vehicle> vehicleSet = new HashSet<>();
   public Vehicle(String regNumber, String ownerName, String vehicleType) {
      this.regNumber = regNumber;
      this.ownerName = ownerName;
```

```
        this.vehicleType = vehicleType;
    }
    public static void addVehicle(String regNumber, String ownerName, String
vehicleType) {
        vehicleSet.add(new Vehicle(regNumber, ownerName, vehicleType));
    }
    public static void displayVehicles() {
        for (Vehicle v : vehicleSet) {
            System.out.println(v);
        }
    }
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Vehicle vehicle = (Vehicle) obj;
        return regNumber.equals(vehicle.regNumber);
    }
    public int hashCode() {
        return Objects.hash(regNumber);
    }
    public String toString() {
        return regNumber + " " + ownerName + " " + vehicleType;
    }
}class TollBoothSystem {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        for (int i = 0; i < n; i++) {
            String regNumber = sc.next();
            String ownerName = sc.next();
            String vehicleType = sc.next();
            Vehicle.addVehicle(regNumber, ownerName, vehicleType);
        }
        Vehicle.displayVehicles();
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

John is organizing a fruit festival, and the quantities of various fruits are stored in a HashMap where fruit names are keys and quantities are values.

Help him develop a program to find the total quantity of fruits for the festival by summing up the values in the HashMap.

Input Format

The input consists of fruit quantities in the format 'fruitName:quantity', where fruitName is the name of the fruit(a string), and quantity is a double value representing the quantity.

The input is terminated by entering "done".

Output Format

The output prints a double value, representing the sum of values in the HashMap, rounded off to two decimal places.

If the value is not numeric, print "Invalid input".

If any special characters other than ':' are entered, print "Invalid format".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Banana:15.2
Orange:56.3
Mango:47.3
done

Output: 118.80

Answer

```
import java.util.Scanner;
import java.util.Map;
import java.util.HashMap;
class ValueProcessor {
   public static Map<String, Double> readValues(Scanner scanner) {
      Map<String, Double> valueMap = new HashMap<>();
      while (true) {
         String input = scanner.nextLine();
         if (input.toLowerCase().equals("done")) {
            break;
         }
         String[] pair = input.split(":");
         if (pair.length == 2) {
            String key = pair[0].trim();
            try {
               double value = Double.parseDouble(pair[1].trim());
               valueMap.put(key, value);
            } catch (NumberFormatException e) {
               System.out.println("Invalid input");
               return null;
            }
         } else {
                System.out.println("Invalid format");
                return null;
            }
        }
        return valueMap;
    }

    public static double calculateSum(Map<String, Double> valueMap) {
        double sum = 0;
        for (double value : valueMap.values()) {
            sum += value;
        }
        return sum;
    }
}
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Map<String, Double> valueMap = ValueProcessor.readValues(scanner);
        if (valueMap != null) {
            double sum = ValueProcessor.calculateSum(valueMap);
            System.out.printf("%.2f\n", sum);
        }
        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 10_PAH

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

A university maintains a list of student records and wants to store them in a sorted manner based on their GPA. If two students have the same GPA, they should be further sorted by their name in lexicographical order. Implement a program that uses a TreeSet to store student records and ensures unique student IDs.

Input Format

The first line contains an integer N - the number of students.

The next N lines contain details of each student in the format: "StudentID Name GPA"

- StudentID (Integer) - A unique identifier.
- Name (String) - The student's name (can contain spaces).

- GPA (Double) - The Grade Point Average.

Output Format

The output prints the list of students in ascending order of GPA.

If two students have the same GPA, sort them by name.

Print details in the format: "StudentID Name GPA" in the output, GPA is rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

```
Input: 5
101 John 8.5
102 Alice 9.1
103 Bob 8.5
104 Zoe 7.3
105 Charlie 9.1
Output: 104 Zoe 7.30
103 Bob 8.50
101 John 8.50
102 Alice 9.10
105 Charlie 9.10
```

Answer

```
import java.util.*;
class Student implements Comparable<Student> {
    int studentID;
    String name;
    double gpa;

    public Student(int studentID, String name, double gpa) {
        this.studentID = studentID;
        this.name = name;
        this.gpa = gpa;
    }

    public int compareTo(Student other) {
        if (this.gpa != other.gpa) {
            return Double.compare(this.gpa, other.gpa);
        }
        return this.name.compareTo(other.name);
    }

    public String toString() {
        return studentID + " " + name + " " + String.format("%.2f", gpa);
    }
}
class UniversityRecords {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        TreeSet<Student> studentSet = new TreeSet<>();
        for (int i = 0; i < n; i++) {
            int id = sc.nextInt();
            String name = sc.next();
            double gpa = sc.nextDouble();
            studentSet.add(new Student(id, name, gpa));
        }
        for (Student s : studentSet) {
            System.out.println(s);
        }
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Riya is building a calendar event scheduler where each event is stored in chronological order using a TreeMap. The key represents the event time in 24-hour format (HH:MM), and the value is the event description.

She wants the system to:

Automatically sort events by time.Avoid duplicate time entries — if a duplicate time is entered, ignore the new entry.Print all scheduled events in order.

Implement this logic using a class named EventManager.

Input Format

The first line of the input contains an integer n, representing the number of events.

The next n lines each contain a string in the format: "HH:MM Description"

(Example: 09:00 TeamMeeting).

Output Format

The first line of the output prints "Scheduled Events:"

The next k lines print each event in the format: "HH:MM - Description"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
09:00 TeamMeeting
13:30 LunchBreak
11:00 ProjectUpdate
09:00 Standup
15:00 ClientCall
Output: Scheduled Events:
09:00 - TeamMeeting
11:00 - ProjectUpdate
13:30 - LunchBreak
15:00 - ClientCall

Answer

```
import java.util.*;
class EventManager {
   TreeMap<String, String> schedule;

   public EventManager() {
       schedule = new TreeMap<>();
   }

   public void addEvent(String time, String description) {
       if (!schedule.containsKey(time)) {
           schedule.put(time, description);
       }
   }

   public void printSchedule() {
       System.out.println("Scheduled Events:");
       for (Map.Entry<String, String> entry : schedule.entrySet()) {
           System.out.println(entry.getKey() + " - " + entry.getValue());
       }
   }
}
public class Main {
   public static void main(String[] args) {
       Scanner sc = new Scanner(System.in);
       int n = Integer.parseInt(sc.nextLine());

       EventManager manager = new EventManager();

       for (int i = 0; i < n; i++) {
           String line = sc.nextLine();
           int spaceIndex = line.indexOf(' ');
           String time = line.substring(0, spaceIndex);
           String desc = line.substring(spaceIndex + 1);
           manager.addEvent(time, desc);
       }

       manager.printSchedule();
   }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Sarah is working on a spam detection system that analyzes incoming messages for unique patterns. Spammers often use repetitive character sequences, making it important to identify the first non-repeating character in a message.

Given a string, Sarah needs to determine the first character that appears only once. If all characters repeat, the system should return -1.

She decides to use a HashMap to efficiently track character frequencies and find the solution.

Input Format

The first line contains an integer N representing , the length of the string.

The second line contains a string of N lowercase English letters (a-z).

Output Format

The output prints a character representing the first non-repeating character. If none exist, print -1.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10
abacabadac

Output: d

Answer

```
import java.util.*;
class NonRepeatingCharacterFinder {

  public char findFirstNonRepeatingCharacter(String str) {
     HashMap<Character, Integer> charCount = new HashMap<>();

     for (char ch : str.toCharArray()) {
        charCount.put(ch, charCount.getOrDefault(ch, 0) + 1);
     }

     for (char ch : str.toCharArray()) {
            if (charCount.get(ch) == 1) {
                return ch;
            }
        }

        return '\0';
    }
}
class FirstNonRepeatingCharacter {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int N = sc.nextInt();
        String str = sc.next();

        NonRepeatingCharacterFinder finder = new NonRepeatingCharacterFinder();
        char result = finder.findFirstNonRepeatingCharacter(str);

        if (result == '\0') {
            System.out.println(-1);
        } else {
            System.out.println(result);
        }

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 10_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : COD

1. Problem Statement

Bob wants to develop a score-tracking application for a gaming tournament. Each player's score is stored in a HashMap with the player's name as the key and the score as the value.

Write a program to assist Bob that takes user input to enter player scores, calculates the maximum score from the HashMap, and prints the player with the highest score.

Input Format

The input consists of strings representing player details in the format "playerName:score".

The input is terminated by entering "done".

Output Format

The output displays a string, representing the player's name who scored the maximum.

If the value is not numeric, print "Invalid input".

If any special characters other than ':' are given, print "Invalid format".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Alice:15
Bob:56
done

Output: Bob

Answer

```
import java.util.*;

class ScoreTracker {
   Map<String, Integer> scoreMap = new HashMap<>();

   boolean processInput(String input) {
      if (input.split(":").length != 2) {
         System.out.println("Invalid format");
         return false;
      }

      String[] parts = input.split(":");
      String playerName = parts[0].trim();
      String scoreStr = parts[1].trim();

      try {
         int score = Integer.parseInt(scoreStr);

         if (score < 1 || score > 100) {
            System.out.println("Invalid input");
            return false;
         }
```

```
            scoreMap.put(playerName, score);
            return true;
        } catch (NumberFormatException e) {
            System.out.println("Invalid input");
            return false;
        }
    }

    String findTopPlayer() {
        int maxScore = Integer.MIN_VALUE;
        String topPlayer = "";

        for (Map.Entry<String, Integer> entry : scoreMap.entrySet()) {
            if (entry.getValue() > maxScore) {
                maxScore = entry.getValue();
                topPlayer = entry.getKey();
            }
        }

        return topPlayer;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ScoreTracker tracker = new ScoreTracker();
        boolean validInput = true;

        while (true) {
            String input = scanner.nextLine();

            if (input.toLowerCase().equals("done")) {
                break;
            }

            if (!tracker.processInput(input)) {
                validInput = false;
                break;
            }
        }
        if (validInput && !tracker.scoreMap.isEmpty()) {
            System.out.println(tracker.findTopPlayer());
        }

        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

A college professor wants to keep track of students who attend classes. Each student has a unique roll number and their attendance count increases every time they attend a class. The system should allow adding a student, marking their attendance, and displaying all students with their total attendance.

Your task is to implement a Java program using TreeSet to maintain students in sorted order of roll numbers and track their attendance count.

Operations:

A roll_no name   Add a student with roll number and name (if not already added).M roll_no   Mark attendance for the student with the given roll number (increase their count by 1).D   Display all students in ascending order of roll number along with their attendance count.

Input Format

The first line contains an integer N - the number of students.

The next N lines contain one of the following commands:

A roll_no name

M roll_no

D

- A (Add)   Adds a new student with a unique roll number and name.
- M (Mark)   Increases attendance count for the given roll number.

- D (Display)   Prints all students in ascending order of roll number.

Output Format

For D, output prints each student's roll number, name, and attendance count in ascending order of roll number.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
A 101 Alice
A 102 Bob
M 101
M 101
D

Output: 101 Alice 2
102 Bob 0

Answer

```
import java.util.*;
class Student implements Comparable<Student> {
   int rollNo;
   String name;
   int attendance;

   public Student(int rollNo, String name) {
      this.rollNo = rollNo;
      this.name = name;
      this.attendance = 0;
   }

   public void markAttendance() {
      this.attendance++;
   }

   public int compareTo(Student s) {
      return Integer.compare(this.rollNo, s.rollNo);
   }
```

```
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Student student = (Student) obj;
        return rollNo == student.rollNo;
    }

    public int hashCode() {
        return Objects.hash(rollNo);
    }

    public String toString() {
        return rollNo + " " + name + " " + attendance;
    }
}

class AttendanceTracker {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        TreeSet<Student> students = new TreeSet<>();
        for (int i = 0; i < n; i++) {
            String[] command = sc.nextLine().split(" ");
            String operation = command[0];

            if (operation.equals("A")) {
                int rollNo = Integer.parseInt(command[1]);
                String name = command[2];
                students.add(new Student(rollNo, name));
            }
            else if (operation.equals("M")) {
                int rollNo = Integer.parseInt(command[1]);
                for (Student s : students) {
                    if (s.rollNo == rollNo) {
                        s.markAttendance();
                        break;
                    }
                }
            }
            else if (operation.equals("D")) {
                for (Student s : students) {
```

```
                System.out.println(s);
            }
        }
    }
    sc.close();
  }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

A linguist named Meera is classifying a list of words based on their first character. She wants to store words grouped by their starting letter using a TreeMap so that the groups appear in sorted order of characters (i.e., 'a' to 'z'). For each letter, all words starting with that letter should be stored in the order they appear.

Implement the logic inside a class named WordClassifier using the TreeMap<Character, List<String>> collection.

Input Format

The first line of the input contains an integer n, representing the number of words.

The next n lines each contain a word.

Output Format

The first line of the output prints: "Grouped Words by Starting Letter:"

The next lines print each character key and its list of words in the format:

"letter: word1 word2 word3...

..."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
dog
deer
cat
cow
camel

Output: Grouped Words by Starting Letter:
c: cat cow camel
d: dog deer

Answer

```
import java.util.*;

class WordClassifier {
   public void classifyWords(List<String> words) {
      TreeMap<Character, List<String>> map = new TreeMap<>();

      for (String word : words) {
         char initial = word.charAt(0);
         if (!map.containsKey(initial)) {
            map.put(initial, new ArrayList<>());
         }
         map.get(initial).add(word);
      }

      System.out.println("Grouped Words by Starting Letter:");
      for (Map.Entry<Character, List<String>> entry : map.entrySet()) {
         System.out.print(entry.getKey() + ": ");
         for (String word : entry.getValue()) {
            System.out.print(word + " ");
         }
         System.out.println();
      }
   }
}

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int n = Integer.parseInt(sc.nextLine());
```

```
        List<String> words = new ArrayList<>();
        for (int i = 0; i < n; i++) {
            words.add(sc.nextLine());
        }

        WordClassifier classifier = new WordClassifier();
        classifier.classifyWords(words);
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

The city library maintains a record of books available for lending. Each book is uniquely identified by its ISBN number, along with its title and author. The librarian wants to efficiently store and manage these records, ensuring books can be listed in the order they were added.

Your task is to implement a Library Management System using HashSet where:

The librarian adds books with ISBN, title, and author.The librarian can remove books by providing an ISBN.Finally, the librarian displays the available books in the order they were added.

Implement a class Library that will handle these operations. The main function should manage user input and interact with the Library class accordingly.

Input Format

The first line contains an integer n – the number of books to be added.

The next n lines contain three values: ISBN (integer), Title (string without spaces), and Author (string without spaces).

1. An integer employee_id
2. A string title
3. A string author name

The next line contains an integer m – the number of books to be removed.

The next m lines follow, each contains an ISBN number to remove.

Output Format

The output prints a list of books available in the library after performing all operations in the format:

"ISBN: <isbn>, Title: <title>, Author: <author>"

If no books remain, print: "No books available"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1234 JavaCompleteGuide JohnDoe
5678 PythonBasics JaneDoe
9012 DataStructures AliceSmith
1
5679

Output: ISBN: 1234, Title: JavaCompleteGuide, Author: JohnDoe
ISBN: 9012, Title: DataStructures, Author: AliceSmith
ISBN: 5678, Title: PythonBasics, Author: JaneDoe

Answer

```
import java.util.*;

class Book {
   int isbn;
   String title, author;

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

   public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Book book = (Book) obj;
        return isbn == book.isbn;
    }

    public int hashCode() {
        return Objects.hash(isbn);
    }
}

class Library {
    HashSet<Book> books = new HashSet<>();

    void addBook(int isbn, String title, String author) {
        books.add(new Book(isbn, title, author));
    }

    void removeBook(int isbn) {
        books.removeIf(book -> book.isbn == isbn);
    }

    void displayBooks() {
        if (books.isEmpty()) {
            System.out.println("No books available");
        } else {
            for (Book book : books) {
                System.out.println("ISBN: " + book.isbn + ", Title: " + book.title + ",
Author: " + book.author);
            }
        }
    }
}

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Library library = new Library();
        int n = sc.nextInt();
        for (int i = 0; i < n; i++) {
            int isbn = sc.nextInt();
            String title = sc.next();
            String author = sc.next();
            library.addBook(isbn, title, author);
        }
        int m = sc.nextInt();
        for (int i = 0; i < m; i++) {
            int isbn = sc.nextInt();
            library.removeBook(isbn);
        }
        library.displayBooks();
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 11

Attempt : 1
Total Mark : 20
Marks Obtained : 20

Section 1 : Project

1. Problem Statement

In Café Central, the menu is cataloged and stored in a database.

To efficiently manage the restaurant's menu using Java and JDBC, you must build a Restaurant Management System that supports:

Adding new menu items

Updating menu item prices

Viewing details of a menu item

Displaying all menu items in sorted order

You are given two files:

File 1: MenuItem.java (POJO Class)

This class represents the MenuItem entity.

A MenuItem contains the following details:

Field	Description
itemId	Unique Menu Item ID (Integer)
name	Item Name (String)
category	Item Category (String)
price	Item Price (Double)

Students must write code in the marked area:

```
class MenuItem {
   private int itemId;
   private String name;
   private String category;
   private double price;

   public MenuItem() {}

   public MenuItem(int itemId, String name, String category, double price) {
      // write your code here
   }

   // Include getters and setters
}
```

Expected in this part:

Assign parameter values to instance variables inside the constructor.

Add getters and setters for all attributes.

File 2: MenuItemDAO.java (Data Access Layer)

This class handles all database operations using JDBC.

Students must complete the missing JDBC logic in the following methods:

```
class MenuItemDAO {

    public void addMenuItem(Connection conn, MenuItem menuItem)
throws SQLException {

        // write your code here

    }

    public void updateItemPrice(Connection conn, int itemId, double
newPrice) throws SQLException {

        // write your code here

    }

    public void deleteMenuItem(Connection conn, int itemId) throws
SQLException {

        // write your code here

    }

    public MenuItem viewItemDetails(Connection conn, int itemId) throws
SQLException {

        // write your code here

    }

    public List<MenuItem> displayAllMenuItems(Connection conn) throws
SQLException {

        // write your code here

    }

    private MenuItem mapToMenuItem(ResultSet rs) throws SQLException {

        return new MenuItem(
            // write your code here
        );
    }
}
```

Expected in this part:

Write SQL queries for INSERT, UPDATE, DELETE, SELECT.

Execute queries using PreparedStatement or Statement.

Map ResultSet rows to MenuItem objects using mapToMenuItem().

Return a List<MenuItem> where required.

The system should connect to a MySQL database using the following default credentials:

DB URL: jdbc:mysql://localhost/ri_db

USER: test

PWD: test123

The menu table has already been created with the following structure:

Table Name: menu

Input Format

The first line of input consists of an integer choice, representing the operation to be performed (1 for Add Item, 2 for Restock item, 3 for reduce item, 4 for Display, 5 for Exit).

For choice 1 (Add Menu Item):

- The second line consists of an integer item_id.
- The third line consists of a string name.
- The fourth line consists of a string category.
- The fifth line consists of a double price.

For choice 2 (Update Item Price):

- The second line consists of an integer item_id.
- The third line consists of a double new_price.

For choice 3 (View Item Details):

- The second line consists of an integer item_id.

For choice 4 (Display All Menu Items):

- No additional inputs are required.

For choice 5 (Exit):

- No additional inputs are required.

Output Format

For choice 1 (Add Menu Item):

- Print "Menu item added successfully" if the item was added.
- Print "Failed to add item." if the insertion failed.

For choice 2 (Update Item Price):

- Print "Item price updated successfully" if the price update was successful.
- Print "Item not found." if the specified item ID does not exist.

For choice 3 (View Item Details):

- Display the item details in the format:
- ID: [item_id] | Name: [name] | Category: [category] | Price: [price]
- Print "Item not found." if the specified item ID does not exist.

For choice 4 (Display All Menu Items):

- Display each item on a new line in the format:
- ID | Name | Category | Price
- If no items are available, print nothing (or handle with an appropriate message if desired).

For choice 5 (Exit):

- Print "Exiting Restaurant Management System."

For invalid input:

- Print "Invalid choice. Please try again."

Sample Test Case

```
Input: 1
11
Margherita Pizza
Main Course
12.99
4
5
Output: Menu item added successfully
ID | Name | Category      | Price
11 | Margherita Pizza | Main Course | 12.99
Exiting Restaurant Management System.
```

Answer

```
import java.sql.*;
import java.util.Scanner;

class RestaurantManagementSystem {
   public static void main(String[] args) {
      try (Connection conn = DriverManager.getConnection("jdbc:mysql://
localhost/ri_db", "test", "test123");
          Scanner scanner = new Scanner(System.in)) {

         boolean running = true;

         while (running) {
            int choice = scanner.nextInt();

            switch (choice) {
               case 1:
                  addMenuItem(conn, scanner);
                  break;
               case 2:
                  updateItemPrice(conn, scanner);
                  break;
               case 3:
                  viewItemDetails(conn, scanner);
                  break;
               case 4:
                  displayAllMenuItems(conn);
                  break;
               case 5:
                  System.out.println("Exiting Restaurant Management System.");
                  running = false;
                  break;
               default:
                  System.out.println("Invalid choice. Please try again.");
            }
         }
      } catch (SQLException e) {
         e.printStackTrace();
      }
   }

   public static void addMenuItem(Connection conn, Scanner scanner) {
      int itemId = scanner.nextInt();
      scanner.nextLine();  // Consume newline

      String name = scanner.nextLine();
      String category = scanner.nextLine();
      double price = scanner.nextDouble();

      MenuItem menuItem = new MenuItem(itemId, name, category, price); //
Using POJO

      String insertQuery = "INSERT INTO menu (item_id, name, category, price)
VALUES (?, ?, ?, ?)";
      try (PreparedStatement stmt = conn.prepareStatement(insertQuery)) {
         stmt.setInt(1, menuItem.getItemId());
         stmt.setString(2, menuItem.getName());
         stmt.setString(3, menuItem.getCategory());
         stmt.setDouble(4, menuItem.getPrice());

         int rowsInserted = stmt.executeUpdate();
         System.out.println(rowsInserted > 0 ? "Menu item added successfully" :
"Failed to add item.");
      } catch (SQLException e) {
         System.out.println("Error adding item: " + e.getMessage());
```

```
        }
    }

    public static void updateItemPrice(Connection conn, Scanner scanner) {
        int itemId = scanner.nextInt();
        double newPrice = scanner.nextDouble();

        String updateQuery = "UPDATE menu SET price = ? WHERE item_id = ?";
        try (PreparedStatement stmt = conn.prepareStatement(updateQuery)) {
            stmt.setDouble(1, newPrice);
            stmt.setInt(2, itemId);

            int rowsUpdated = stmt.executeUpdate();
            System.out.println(rowsUpdated > 0 ? "Item price updated successfully" :
"Item not found.");
        } catch (SQLException e) {
            System.out.println("Error updating price: " + e.getMessage());
        }
    }

    public static void viewItemDetails(Connection conn, Scanner scanner) {
        int itemId = scanner.nextInt();

        String selectQuery = "SELECT * FROM menu WHERE item_id = ?";
        try (PreparedStatement stmt = conn.prepareStatement(selectQuery)) {
            stmt.setInt(1, itemId);

            ResultSet rs = stmt.executeQuery();
            if (rs.next()) {
                MenuItem menuItem = new MenuItem(
                        rs.getInt("item_id"),
                        rs.getString("name"),
                        rs.getString("category"),
                        rs.getDouble("price")
                );

                System.out.printf("ID: %d | Name: %s | Category: %s | Price: %.2f%n",
                        menuItem.getItemId(),
                        menuItem.getName(),
                        menuItem.getCategory(),
                        menuItem.getPrice());
            } else {
                System.out.println("Item not found.");
            }
        } catch (SQLException e) {
            System.out.println("Error retrieving item details: " + e.getMessage());
        }
    }

    public static void displayAllMenuItems(Connection conn) {
        String displayQuery = "SELECT * FROM menu ORDER BY item_id";
        try (Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(displayQuery)) {

            System.out.println("ID | Name | Category       | Price");
            while (rs.next()) {
                MenuItem menuItem = new MenuItem(
                        rs.getInt("item_id"),
                        rs.getString("name"),
                        rs.getString("category"),
                        rs.getDouble("price")
                );

                System.out.printf("%d | %s | %s | %.2f%n",
                        menuItem.getItemId(),
                        menuItem.getName(),
                        menuItem.getCategory(),
                        menuItem.getPrice());
            }
        } catch (SQLException e) {
            System.out.println("Error displaying menu items: " + e.getMessage());
        }
    }
}

 class MenuItem {
    private int itemId;
    private String name;
    private String category;
    private double price;

    // Constructor
    public MenuItem(int itemId, String name, String category, double price) {
        this.itemId = itemId;
        this.name = name;
        this.category = category;
        this.price = price;
    }

    // Getters and Setters
    public int getItemId() { return itemId; }
    public void setItemId(int itemId) { this.itemId = itemId; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public String getCategory() { return category; }
    public void setCategory(String category) { this.category = category; }

    public double getPrice() { return price; }
    public void setPrice(double price) { this.price = price; }
}
//
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Create a JDBC-based Hospital Management System that handles runtime input to manage patient records. The system should allow users to:

Add a new patient (patient ID, name, age, status).

Update a patient's status.

View a specific patient's record by patient ID.

Display all patient records in the database.

Exit the application.

The system should connect to a MySQL database using the following default credentials:

DB URL: jdbc:mysql://localhost/ri_db

USER: test

PWD: test123

The patients table has already been created with the following structure:

Table Name: patients

Input Format

The first line of input consists of an integer choice, representing the operation to be performed:

(1 for Add Patient, 2 for Update Patient Status, 3 for View Patient Record, 4 for Display All Patients, 5 for Exit)

For choice 1 (Add Patient):

- The second line consists of an integer patient_id.
- The third line consists of a string name.
- The fourth line consists of an integer age.
- The fifth line consists of a string status.

For choice 2 (Update Patient Status):

- The second line consists of an integer patient_id.
- The third line consists of a string new_status.

For choice 3 (View Patient Record):

- The second line consists of an integer patient_id.

For choice 4 (Display All Patients):

- No additional inputs are required.

For choice 5 (Exit):

- No additional inputs are required.

Output Format

For choice 1 (Add Patient):

- Print "Patient added successfully" if the patient was added.
- Print "Failed to add patient." if the insertion failed.

For choice 2 (Update Patient Status):

- Print "Patient status updated successfully" if the update was successful.
- Print "Patient not found." if the specified patient ID does not exist.

For choice 3 (View Patient Record):

- Display the patient details in the format:
- ID: [patient_id] | Name: [name] | Age: [age] | Status: [status]
- Print "Patient not found." if the specified patient ID does not exist.

For choice 4 (Display All Patients):

- Display each patient on a new line in the format:
- ID | Name | Age | Status
- If no records are available, print nothing (or handle it with an appropriate message if desired).

For choice 5 (Exit):

- Print "Exiting Hospital Management System."

For invalid input:

- Print "Invalid choice. Please try again."

Sample Test Case

Input: 1
101
John Doe
45
Admitted
4
5
Output: Patient added successfully

```
ID | Name | Age | Status
101 | John Doe | 45 | Admitted
Exiting Hospital Management System.
```

Answer

```
import java.sql.*;
import java.util.Scanner;

class HospitalManagementSystem {
   public static void main(String[] args) {
      try (Connection conn = DriverManager.getConnection("jdbc:mysql://
localhost/ri_db", "test", "test123");
           Scanner scanner = new Scanner(System.in)) {

          boolean running = true;

          while (running) {

             int choice = scanner.nextInt();

             switch (choice) {
                case 1:
                   addPatient(conn, scanner);
                   break;
                case 2:
                   updatePatientStatus(conn, scanner);
                   break;
                case 3:
                   viewPatientRecord(conn, scanner);
                   break;
                case 4:
                   displayAllPatients(conn);
                   break;
                case 5:
                   System.out.println("Exiting Hospital Management System.");
                   running = false;
                   break;
                default:
                   System.out.println("Invalid choice. Please try again.");
             }
          }
      } catch (SQLException e) {
          e.printStackTrace();
      }
   }


   public static void addPatient(Connection conn, Scanner scanner) {
      int patientId = scanner.nextInt();
      scanner.nextLine();  // Consume newline

      String name = scanner.nextLine();

      int age = scanner.nextInt();

      scanner.nextLine();  // Consume newline
      String status = scanner.nextLine();

      String insertQuery = "INSERT INTO patients (patient_id, name, age, status)
VALUES (?, ?, ?, ?)";
      try (PreparedStatement stmt = conn.prepareStatement(insertQuery)) {
          stmt.setInt(1, patientId);
          stmt.setString(2, name);
          stmt.setInt(3, age);
          stmt.setString(4, status);

          int rowsInserted = stmt.executeUpdate();
          System.out.println(rowsInserted > 0 ? "Patient added successfully" :
"Failed to add patient.");
      } catch (SQLException e) {
          System.out.println("Error adding patient: " + e.getMessage());
      }
   }

   public static void updatePatientStatus(Connection conn, Scanner scanner) {
      int patientId = scanner.nextInt();
      scanner.nextLine();  // Consume newline

      String newStatus = scanner.nextLine();

      String updateQuery = "UPDATE patients SET status = ? WHERE patient_id
= ?";
      try (PreparedStatement stmt = conn.prepareStatement(updateQuery)) {
          stmt.setString(1, newStatus);
          stmt.setInt(2, patientId);

          int rowsUpdated = stmt.executeUpdate();
          System.out.println(rowsUpdated > 0 ? "Patient status updated
successfully" : "Patient not found.");
      } catch (SQLException e) {
          System.out.println("Error updating patient status: " + e.getMessage());
      }
   }

   public static void viewPatientRecord(Connection conn, Scanner scanner) {
      int patientId = scanner.nextInt();

      String selectQuery = "SELECT * FROM patients WHERE patient_id = ?";
      try (PreparedStatement stmt = conn.prepareStatement(selectQuery)) {
          stmt.setInt(1, patientId);

          ResultSet rs = stmt.executeQuery();
          if (rs.next()) {
              System.out.printf("ID: %d | Name: %s | Age: %d | Status: %s%n",
                      rs.getInt("patient_id"),
                      rs.getString("name"),
                      rs.getInt("age"),
                      rs.getString("status"));
          } else {
              System.out.println("Patient not found.");
          }
      } catch (SQLException e) {
          System.out.println("Error retrieving patient record: " + e.getMessage());
      }
   }

   public static void displayAllPatients(Connection conn) {
      String displayQuery = "SELECT * FROM patients ORDER BY patient_id";
      try (Statement stmt = conn.createStatement();
           ResultSet rs = stmt.executeQuery(displayQuery)) {

          System.out.println("ID | Name | Age | Status");
          while (rs.next()) {
              System.out.printf("%d | %s | %d | %s%n",
                      rs.getInt("patient_id"),
                      rs.getString("name"),
                  rs.getInt("age"),
                  rs.getString("status"));
          }
      } catch (SQLException e) {
          System.out.println("Error displaying patients: " + e.getMessage());
      }
   }


}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_Week 12_Java_Lamba Expressions_MCQ

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : MCQ

1. What is the return type of a lambda expression in Java?

Answer

The return type is inferred from the context

Status : Correct ***Marks : 1/1***

2. What is the syntax for a basic lambda expression in Java?

Answer

(parameters) -&gt; expression

Status : Correct ***Marks : 1/1***

3. Which of the following interfaces is NOT a functional interface in Java?

Answer

Iterable

Status : Correct ***Marks : 1/1***

4. Can a lambda expression have more than one parameter?

Answer

Yes, it can have multiple parameters

Status : Correct ***Marks : 1/1***

5. Which of the following is a valid lambda expression in Java?

Answer

All of the mentioned options

Status : Correct ***Marks : 1/1***

6. What is a lambda expression in Java?

Answer

A way to define anonymous methods

Status : Correct ***Marks : 1/1***

7. Which functional interface is commonly used with lambda expressions in Java?

Answer

Runnable

Status : Correct ***Marks : 1/1***

8. Can a lambda expression in Java have a body with multiple statements?

Answer

Yes, if the statements are enclosed in curly braces

Status : Correct ***Marks : 1/1***

9. Can a lambda expression in Java have a body with multiple statements?

Answer

Yes, if the statements are enclosed in curly braces

Status : Correct ***Marks : 1/1***

10. Which functional interface in Java takes two arguments and returns a result?

Answer

BiFunction

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Sabrina is working on a project that involves analyzing a set of numbers. In her exploration, she encounters scenarios where extracting even numbers and finding their sum is essential.

Create a program that calculates the sum of even numbers from a given array of integers using a lambda expression.

Input Format

The first line of input consists of an integer N, representing the size of the array.

The second line consists of N space-separated integers, representing the elements of the array.

Output Format

The output prints the sum of the even integers from the array.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
29 37 45

Output: 0

Answer

```
import java.util.Scanner;
class EvenSumCalculator {
   public static int calculateEvenSum(int[] numbers) {
      return java.util.Arrays.stream(numbers)
            .filter(n -> n % 2 == 0)
            .sum();
   }
}
class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      int count = scanner.nextInt();
      int[] numbers = new int[count];

      for (int i = 0; i < count; i++) {
         numbers[i] = scanner.nextInt();
      }
      int sum = EvenSumCalculator.calculateEvenSum(numbers);
      System.out.println(sum);

      scanner.close();
   }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_Week 12_Java_Lamba Expressions_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 37.5

Section 1 : COD

1. Problem Statement

Sneha is developing a feature for an e-commerce application that helps display product details after applying a seasonal discount.

She decides to use lambda expressions with the Consumer functional interface to print each product's name, original price, and discounted price neatly.

The program should:

Accept a list of product names and their prices.Apply a 15% discount on all products.Use a Consumer lambda expression to display the details in a formatted manner.

Input Format

The first line of input consists of an integer n, representing the number of products.

The next n lines each contain a String (product name) and a double (price) separated by a space.

Output Format

For each product, print the details in the format:

```
 Product: <name>, Original Price: <price>, Discounted Price: <discounted price>
```

If there are no products, print:

```
 No products available
```

Sample Test Case

Input: 1
Phone 60000

Output: Product: Phone, Original Price: 60000.0, Discounted Price: 51000.0

Answer

```
import java.util.*;
import java.util.function.Consumer;

class Product {
   String name;
   double price;

   Product(String name, double price) {
      this.name = name;
      this.price = price;
   }
}

class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      int n = sc.nextInt();
      sc.nextLine();

      if (n == 0) {
          System.out.println("No products available");
          return;
      }

      List<Product> products = new ArrayList<>();

      for (int i = 0; i < n; i++) {
          String[] input = sc.nextLine().split(" ");
          String name = input[0];
          double price = Double.parseDouble(input[1]);
          products.add(new Product(name, price));
      }
      Consumer<Product> displayProduct = product -> {
          double discountedPrice = product.price * 0.85;
          System.out.println("Product: " + product.name +
                           ", Original Price: " + product.price +
                           ", Discounted Price: " + discountedPrice);
      };
      products.forEach(displayProduct);

      sc.close();
   }
}
```

Status : Partially correct ***Marks : 7.5/10***

2. Problem Statement

Rishi is working as an HR analyst in a software company. He wants to filter a list of employees based on their salary using modern Java techniques. He has a list of employee names and salaries and wants to use lambda expressions to filter those who earn more than a specific threshold.

Implement a program using lambda expressions and functional interfaces to print the names of employees whose salary is greater than or equal to 50,000.

Input Format

The first line of input consists of an integer n, representing the number of employees.

The next n lines. Each line contains a String (employee name) and an int (salary).

Output Format

The output prints the names of employees whose salary is greater than or equal to 50000, each on a new line.

If no employee found with salary greater than 50000, print: No employee found with salary >= 50000

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
Amit 45000
Sneha 50000
Ravi 60000
Priya 30000
Output: Sneha
Ravi

Answer

```
import java.util.*;
import java.util.function.Predicate;
import java.util.stream.Collectors;

class Employee {
   String name;
   int salary;

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

class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
```

```
        int n = sc.nextInt();
        List<Employee> employees = new ArrayList<>();

        for (int i = 0; i < n; i++) {
            String name = sc.next();
            int salary = sc.nextInt();
            employees.add(new Employee(name, salary));
        }

        Predicate<Employee> highSalary = e -> e.salary >= 50000;

        List<String> result = employees.stream()
                .filter(highSalary)
                .map(e -> e.name)
                .collect(Collectors.toList());

        if (result.isEmpty()) {
            System.out.println("No employee found with salary >= 50000");
        } else {
            result.forEach(System.out::println);
        }
    }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Aditya is developing a reading app that recommends books to users based on a predefined list.

Each time a user opens the app, it should supply the next book title in the list, one at a time, using a lambda expression and the Supplier functional interface.

When all books have been recommended, the list should start again from the beginning.

Input Format

The first line contains an integer n — the total number of available book titles.

The next n lines each contain a book title (a string).

The next line contains an integer m — the number of times users open the app (i.e., the number of recommendations to be made).

Output Format

Print the supplied book title for each recommendation, one per line.

If m > n, repeat the list from the start.

Sample Test Case

Input: 3
The Alchemist
Atomic Habits
Ikigai
5
Output: The Alchemist
Atomic Habits
Ikigai
The Alchemist
Atomic Habits

Answer

```
import java.util.*;
import java.util.function.Supplier;

class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      // Read number of books
      int n = sc.nextInt();
      sc.nextLine(); // consume newline

      List<String> books = new ArrayList<>();
      for (int i = 0; i < n; i++) {
         books.add(sc.nextLine());
      }

      // Read number of recommendations (times user opens app)
      int m = sc.nextInt();
```

```
        // Index tracker for current book
        final int[] index = {0};

        // Supplier to provide next book
        Supplier<String> bookSupplier = () -> {
            String book = books.get(index[0]);
            index[0] = (index[0] + 1) % books.size(); // Loop back to start
            return book;
        };

        // Print book recommendations
        for (int i = 0; i < m; i++) {
            System.out.println(bookSupplier.get());
        }

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Emily, an analyst at a data processing firm, is tasked with cleaning up datasets to remove duplicate values from lists of integers.

Create a Java program that allows Emily to input a series of integers, with the program then utilizing a lambda expression to efficiently remove any duplicates.

Input Format

The first line of input consists of an integer N, representing the size of the array.

The second line consists of N space-separated integers, each denoting an array element.

Output Format

The output prints the array elements after removing the duplicates inside the square bracket separated by a comma and space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 15
1 2 3 4 3 2 1 2 3 4 4 4 5 5 6

Output: [1, 2, 3, 4, 5, 6]

Answer

```
import java.util.*;
import java.util.function.Function;
import java.util.stream.Collectors;

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();
        List<Integer> numbers = new ArrayList<>();
        for (int i = 0; i < n; i++) {
            numbers.add(sc.nextInt());
        }

        Function<List<Integer>, List<Integer>> removeDuplicates =
            list -> list.stream().distinct().collect(Collectors.toList());

        List<Integer> uniqueNumbers = removeDuplicates.apply(numbers);

        System.out.println(uniqueNumbers);

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Keerthisri D
Email: 241901047@rajalakshmi.edu.in
Roll no: 241901047
Phone: 9342453075
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_Week 12_Java_Lamba Expressions_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Nethra is a researcher working on a project that involves analyzing experimental data. As part of her analysis, she needs to determine whether a given word is a palindrome or not.

Create a Java program that allows Nethra to input a word, and then check and display whether the entered word is a palindrome. Use lambda expressions to perform the palindrome check.

Input Format

The first line of input consists of a word.

Output Format

The output prints whether the given word is a palindrome or not in the following format:

"<input> is palindrome" or "<input> is not palindrome".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: malayalam
Output: malayalam is palindrome

Answer

```
import java.util.Scanner;
import java.util.function.Predicate;

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        String inputString = scanner.nextLine();

        Predicate<String> isPalindrome = str -> {
            String reversed = new StringBuilder(str).reverse().toString();
            return str.equalsIgnoreCase(reversed);
        };

        if (isPalindrome.test(inputString)) {
            System.out.println(inputString + " is palindrome");
        } else {
            System.out.println(inputString + " is not palindrome");
        }

        scanner.close();
    }
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Riya is developing a college admission system that assigns unique roll numbers to each newly admitted student.

Each roll number should follow this fixed format:

<DEPT>-<YEAR>-<4-digit-sequence>

where:

<DEPT> is the department code (in uppercase, e.g., CSE, ECE, MECH).<YEAR> is the admission year (e.g., 2025).<4-digit-sequence> starts from a given number and increases sequentially for each student.Write a Java program using a Supplier<String> lambda to generate and print the roll numbers for n students.

Input Format

First line: integer n — number of roll numbers to generate

Second line: string DEPT — department code (uppercase letters only)

Third line: integer YEAR — admission year

Fourth line: integer start — starting sequence number (0 ≤ start ≤ 9999)

Output Format

Print n roll numbers, one per line, in the required format

Sequence must be zero-padded to 4 digits

If sequence exceeds 9999, wrap around to 0000

Sample Test Case

Input: 5
CSE
2025
98

Output: CSE-2025-0098
CSE-2025-0099
CSE-2025-0100
CSE-2025-0101
CSE-2025-0102

Answer

```
import java.util.*;
import java.util.function.Supplier;

class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      int n = sc.nextInt();
      String dept = sc.next();
      int year = sc.nextInt();
      int start = sc.nextInt();

      final int[] current = { start };

      Supplier<String> rollNumberSupplier = () -> {
         String roll = String.format("%s-%d-%04d", dept, year, current[0]);
         current[0] = (current[0] + 1) % 10000;
         return roll;
      };

      for (int i = 0; i < n; i++) {
         System.out.println(rollNumberSupplier.get());
      }

      sc.close();
   }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

A company named TechNova is collecting feedback from its customers. Each customer gives a feedback score (an integer between 1 and 10) along with their name.

The company wants to:

Display each customer's name along with their feedback in a formatted way using a lambda expression and a Consumer functional interface.After displaying all feedbacks, calculate and display the average feedback score.You need to implement this functionality using Java lambda expressions and streams, emphasizing the Consumer interface for displaying formatted output.

Input Format

The first line of input contains an integer n, representing the number of customers.

The next n lines each contain a String (customer name) followed by an int (feedback score).

Output Format

- Each line prints a customer's name and feedback in the format:
- Customer: <name>, Feedback Score: <score>

- After all customers are displayed, print the average feedback as:
- Average Feedback: <average_value>

(Average should be displayed up to two decimal places.)

Sample Test Case

Input: 3
Ravi 7
Ananya 9
Kiran 8

Output: Customer: Ravi, Feedback Score: 7
Customer: Ananya, Feedback Score: 9
Customer: Kiran, Feedback Score: 8
Average Feedback: 8.00

Answer

```
import java.util.*;
import java.util.function.Consumer;
```

```
class Customer {
   String name;
   int feedback;

   Customer(String name, int feedback) {
      this.name = name;
      this.feedback = feedback;
   }
}

class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      int n = sc.nextInt();
      List<Customer> customers = new ArrayList<>();

      for (int i = 0; i < n; i++) {
         String name = sc.next();
         int feedback = sc.nextInt();
         customers.add(new Customer(name, feedback));
      }

      Consumer<Customer> displayFeedback = c ->
            System.out.println("Customer: " + c.name + ", Feedback Score: " +
c.feedback);

      customers.forEach(displayFeedback);

      double avgFeedback = customers.stream()
            .mapToInt(c -> c.feedback)
            .average()
            .orElse(0.0);

      System.out.printf("Average Feedback: %.2f", avgFeedback);
      sc.close();
   }
}
```

Status : Correct

Marks : 10/10

4. Problem Statement

Problem Statement

Sophia, a data analyst, is studying experimental results collected from various lab sensors. Each sensor provides a list of numeric readings, and Sophia wants to calculate the average of these readings to analyze consistency.

She decides to use lambda expressions and the Function functional interface to compute the average of all the recorded values efficiently.

Your Task

Write a Java program that:

Reads the total number of measurements.Reads all the measurement values as doubles.Uses a Function<double[], Double> lambda expression to calculate the average value.Displays the final average, formatted to two decimal places.

Input Format

The first line of input consists of an integer N, representing the number of measurements.

The second line contains N space-separated double values.

Output Format

Print the average of the entered values, rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
2.2 1.2 5.4 4.6 2.9 55.7

Output: 12.00

Answer

import java.util.*;

```
import java.util.function.Function;

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();
        double[] values = new double[n];

        for (int i = 0; i < n; i++) {
            values[i] = sc.nextDouble();
        }

        // Function to calculate average using lambda expression
        Function<double[], Double> calculateAverage = arr -> {
            double sum = 0;
            for (double val : arr) {
                sum += val;
            }
            return sum / arr.length;
        };

        double average = calculateAverage.apply(values);
        System.out.printf("%.2f", average);

        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***