RAJALAKSHMI ENGINEERING COLLEGE

SUBJECT:	OBJECT ORIENTED PROGRAMMING
CODE:	CS23333
NAME:	DIVYASH H
ROLL NO:	240701130
YEAR:	2024-2028
SEC:	8

QR CODE :

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

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

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


        int N = scanner.nextInt();
        int M = scanner.nextInt();


        int diffN = Math.abs(100 - N);
        int diffM = Math.abs(100 - M);


        if (diffN < diffM) {
            System.out.println("The integer closer to 100 is " + N + " with a difference
of " + diffN);
        } else {
            System.out.println("The integer closer to 100 is " + M + " with a difference
of " + diffM);
        }

        scanner.close();
    }
```

}

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;

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

        int num1 = scanner.nextInt();
        int num2 = scanner.nextInt();


        boolean condition1 = (num1 > 0 && num2 % 3 != 0);
        boolean condition2 = (num2 > 0 && num1 % 3 != 0);

        if (condition1 || condition2) {
            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.");
        }

        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;

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


        int monetaryAmount = scanner.nextInt();


        double convertedAmount = (double) monetaryAmount;


        System.out.println("Original Integer: " + monetaryAmount);
        System.out.println("Converted Double: " + convertedAmount);

        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;

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

        int a = sc.nextInt();
        int b = sc.nextInt();


        int sum = a + b;
        int product = a * b;


        if (sum % product == 0) {
            System.out.println("Sum is Multiple of Product");
        } else {
            System.out.println("Sum is Not Multiple of Product");
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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 * π * r

A = π * 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

```
import java.util.Scanner;

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

```
        double radius = sc.nextDouble();


        final double PI = 3.14159;


        double circumference = 2 * PI * radius;
        double area = PI * radius * radius;

        System.out.printf("Circumference: %.2f meters%n", circumference);
        System.out.printf("Area: %.2f square meters%n", area);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q6

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Joey is learning about bitwise operations and is working on a project that involves extracting specific bits from integers. He needs to write a program that takes an integer and the number of bits N as input and outputs the value of the lowest N bits of the integer.

Help Joey in his project to understand and visualize how bitwise operations work in practical scenarios.

Input Format

The first line of input consists of an integer X, representing the given integer.

The second line consists of an integer N, representing the number of bits to extract.

Output Format

The output displays "Result: " followed by an integer representing the value of the lowest N bits of the given integer.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 85
2
Output: Result: 1

Answer

```
import java.util.Scanner;

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


        int X = sc.nextInt();
        int N = sc.nextInt();

        int mask = (1 << N) - 1;


        int result = X & mask;
        System.out.println("Result: " + result);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q7

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement:

Miles is working on a program that involves analyzing two integers. He wants to check if either one of the integers is both:

Less than or equal to zero, andOdd.Can you help him create a program that identifies whether either of the integers meets these conditions?

Input Format

The input consists of two integers on separate lines, denoted as 'input1' and 'input2'.

Output Format

A single line with a boolean result (either 'true' or 'false') indicating whether either 'input1' or 'input2' is both less than or equal to zero and odd.

Refer to the sample output for format specifications

Sample Test Case

Input: -45
10

Output: true

Answer

```
// You are using Java
import java.util.Scanner;
class Main{
    public static void main(String args[]){
        Scanner scan=new Scanner(System.in);
        int a=scan.nextInt();
        int b=scan.nextInt();
        if((a<=0 && a%2!=0)||(b<=0&&b%2!=0)){
            System.out.print("true");

        }
        else{
            System.out.print("false");
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q8

Attempt : 1
Total Mark : 10
Marks Obtained : 5

Section 1 : Coding

1. Problem Statement

In the Kingdom of Finance, the royal treasury is managed by the treasurer, Sir Cedric. Sir Cedric tracks the daily expenses of the kingdom using an expense report that lists three major categories: food, clothing, and utilities. However, the King wants to know if the average daily expense is greater than at least two of these categories to ensure the kingdom is spending wisely.

Your task is to help Sir Cedric determine if the average daily expense is greater than two of the categories. Specifically, you need to calculate the average of the three expenses and check if it is greater than any two categories.

Note: Use the ternary operator

Input Format

Three integers a, b, and c represent the daily expenses for food, clothing, and utilities. Each integer is provided on a single line.

Output Format

The average of the three expenses, rounded to two decimal places.

A message indicating whether the average is greater than at least two of the expense categories.

1. If the average is greater than the two smallest monthly expenses, print "Average is greater than both X and Y," where X and Y are the two smallest expenses.
2. Otherwise, display "Average is not greater than two smallest expenses".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
6
10

Output: 6.67
Average is greater than both 4 and 6

Answer

```
import java.util.*;

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

      int a = sc.nextInt();
      int b = sc.nextInt();
      int c = sc.nextInt();

      double average = (a + b + c) / 3.0;
```

```
        int[] expenses = {a, b, c};
        Arrays.sort(expenses);
        int x = expenses[0];
        int y = expenses[1];

        String message = (average > x && average > y)
            ? "Average is greater than both " + x + " and " + y
            : "Average is not greater than two smallest expenses";

        System.out.printf("%.2f\n", average);
        System.out.println(message);
    }
}
```

Status : Partially correct ***Marks : 5/10***

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q9

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Phill is a quality control manager at a manufacturing plant. He needs to verify if a sensor reading at a midpoint station (S2) falls exactly halfway between the readings of the previous station (S1) and the next station (S3). Help him by developing a program that checks if the second sensor reading is the average (midpoint) of the first and third sensor readings.

Use the relational operator to solve the program.

Input Format

The first line of input consists of an integer S1, representing the sensor reading of the first station.

The second line consists of an integer S2, representing the sensor reading of the midpoint station.

The third line consists of an integer S3, representing the sensor reading of the next station.

Output Format

The first line of output displays a boolean value representing whether the sensor reading at the midpoint station is halfway between the readings of the first and the next stations.

The second line displays one of the following:

1. If the result is true, print "The second integer is halfway between the first and third integers."
2. Otherwise, print "The second integer is not halfway between the first and third integers."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
7
10
Output: false
The second integer is not halfway between the first and third integers.

Answer

```
import java.util.*;

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

      int S1 = sc.nextInt();
      int S2 = sc.nextInt();
      int S3 = sc.nextInt();

      boolean isMidpoint = S2 == (S1 + S3) / 2;

      System.out.println(isMidpoint);
```

```
        System.out.println(isMidpoint
            ? "The second integer is halfway between the first and third integers."
            : "The second integer is not halfway between the first and third integers.");
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 1_Q10

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Aishu is supervising a construction project that needs to be completed with the help of three workers: A, B, and C.

She knows how many days each of them would take to complete the entire project individually:

A can complete it in x days,B in y days,C in z days.

Initially, all three workers (A, B, and C) work together for d1 days.

After that, C leaves, and only A and B continue for another d2 days.

Then B also leaves, and A works alone to finish the remaining work.

Your tasks is to help aishu to implement this functionality using the class WorkDistribution and Method calculateWork(int x, int y, int z, int d1, int d2)

Calculate the total work completed in the first d1 days by A, B, and C.Calculate the work completed in the next d2 days by A and B.Determine the remaining work after these d1 + d2 days.

Input Format

The first line of input contains five space-separated integers: x y z d1 d2

where:

x represents the Days A takes to complete the work alone

y represents the Days B takes to complete the work alone

z represents the Days C takes to complete the work alone

d1 represents the Days A, B, and C work together

d2 represents the Days A and B work together (after C leaves)

Output Format

The first line of output prints "Work done in first d1 days (A+B+C): " followed by a double value rounded to 2 decimal places.

The second line of output prints "Work done in next d2 days (A+B): " followed by a double value rounded to 2 decimal places.

The third line prints "Remaining work: " followed by a double value rounded to 2 decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10 20 30 2 2

Output: Work done in first d1 days (A+B+C): 0.37
Work done in next d2 days (A+B): 0.30
Remaining work: 0.33

Answer

```
import java.util.*;

class WorkDistribution {
    public void calculateWork(int x, int y, int z, int d1, int d2) {
        double rateA = 1.0 / x;
        double rateB = 1.0 / y;
        double rateC = 1.0 / z;

        double work1 = d1 * (rateA + rateB + rateC);

        double work2 = d2 * (rateA + rateB);

        double remainingWork = 1.0 - (work1 + work2);

        System.out.printf("Work done in first d1 days (A+B+C): %.2f\n", work1);
        System.out.printf("Work done in next d2 days (A+B): %.2f\n", work2);
        System.out.printf("Remaining work: %.2f\n", remainingWork);
    }
}

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

        int x = sc.nextInt();
        int y = sc.nextInt();
        int z = sc.nextInt();
        int d1 = sc.nextInt();
        int d2 = sc.nextInt();

        WorkDistribution wd = new WorkDistribution();
        wd.calculateWork(x, y, z, d1, d2);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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;
class StudentsResult {
   public static void main(String[] args){
      Scanner sc = new Scanner(System.in);
      int m1 = sc.nextInt();
      int m2 = sc.nextInt();
      int m3 = sc.nextInt();
      int m4 = sc.nextInt();
      int m5 = sc.nextInt();
      int average = (m1+m2+m3+m4+m5)/5;
      System.out.println("Average score: " + average);
      if(average>=50){
         System.out.println("The student has passed");

      }
      else{
         System.out.println("The student has failed");
      }
      sc.close();
```

```
    }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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;
class Main{
   public static void main(String[] args){
      Scanner sc = new Scanner(System.in);
      int N = sc.nextInt();
      if (N % 5 == 0 && N % 7 == 0)
      {
         System.out.println(N + "is a multiple of both 5 and 7");
      }
      else if (N%5==0)
      {
         System.out.println(N + "is a multiple of 5");
      }
      else if (N%7==0)
      {
         System.out.println(N + "is a multiple of 7");
      }
      else
      {
         System.out.println(N + "is neither multiple of 5 nor 7");
      }
   }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;

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

        double height = sc.nextDouble();
        double weight = sc.nextDouble();

        double bmi = weight / (height * height);
        System.out.printf("BMI: %.2f\n", bmi);

        String classification;
        if (bmi < 18.5) {
            classification = "Underweight";
        } else if (bmi >= 18.6 && bmi <= 24.9) {
```

```
            classification = "Normal Weight";
        } else if (bmi >= 25.0 && bmi <= 29.9) {
            classification = "Overweight";
        } else {
            classification = "Obese";
        }

        System.out.println("Classification: " + classification);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;

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

        int initialCost = sc.nextInt();
        int years = sc.nextInt();

        double depreciationRate = 0.15;
        double currentValue = initialCost;

        for (int i = 0; i < years; i++) {
            currentValue *= (1 - depreciationRate);
        }
```

```
        System.out.printf("Current Value: %.2f\n", currentValue);

        String category;
        if (currentValue > 10000) {
            category = "High";
        } else if (currentValue >= 5000) {
            category = "Medium";
        } else {
            category = "Low";
        }

        System.out.println("Category: " + category);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Ted, the computer science enthusiast, has accepted the challenge of writing a program that checks if the number of digits in an integer matches the sum of its digits.

Guide Ted in designing and writing the code to solve this problem using a 'do-while' loop.

Input Format

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

Output Format

If the sum is equal to the number of digits, print "The number of digits in N matches the sum of its digits."

Else, print "The number of digits in N does not match the sum of its digits."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 20

Output: The number of digits in 20 matches the sum of its digits.

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 temp = N;
        int sum = 0;
        int digitCount = 0;

        do {
            int digit = temp % 10;
            sum += digit;
            digitCount++;
            temp /= 10;
        } while (temp > 0);

        if (sum == digitCount) {
            System.out.println("The number of digits in " + N + " matches the sum of
its digits.");
        } else {
            System.out.println("The number of digits in " + N + " does not match the
sum of its digits.");
        }
    }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q6

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Maya, a student in an arts and crafts class, wants to create a pattern using stars (*) in a specific format. She plans to use a program to help her construct the pattern.

Write a program that takes an integer as input and constructs the following pattern using nested for loops.

Input: 5

Output:

*

* *

* * *

* * * *

* * * * *

* * * *

* * *

* *

*

Input Format

The input consists of a number (integer) representing the number of rows.

Output Format

The output displays the required pattern.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
Output: *
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*

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 rows = sc.nextInt();


        for (int i = 1; i <= rows; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print("* ");
            }
            System.out.println();
        }


        for (int i = rows - 1; i >= 1; i--) {
            for (int j = 1; j <= i; j++) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q7

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are taking part in a coding challenge where your task is to design a program that conjures a mesmerizing numerical pyramid pattern. The enchanting pattern is fashioned using a for loop and is customized based on user input.

Participants are prompted to unveil the pyramid's magic by specifying its height - essentially dictating the number of rows in this spellbinding creation.

Write a program that employs to weave this captivating numerical pyramid as shown below.

Example

Input:
4

Output:

Input Format

The input consists of a positive integer n representing the number of rows in the pattern.

Output Format

The output prints the required pyramid pattern, as shown in the sample output.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 4
Output: 1
 123
 12345
1234567

Answer

```
import java.util.Scanner;

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

        for (int i = 1; i <= n; i++) {
            // Print leading spaces
            for (int j = 1; j <= n - i; j++) {
                System.out.print(" ");
            }

            // Print numbers from 1 to (2*i - 1)
```

```
            for (int k = 1; k <= (2 * i - 1); k++) {
                System.out.print(k);
            }

            // Move to next line
            System.out.println();
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_Q8

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

A bank generates secure codes using 3-digit numbers where each digit is unique, and the code must be divisible by 3. You are tasked with generating the first N such codes based on user input, ensuring the digits are unique and the number is divisible by 3.

Note: Use nested for loops to solve.

Input Format

The first line contains an integer N representing the number of valid codes to generate.

Output Format

The output prints N line,s each line contains a valid 3-digit code.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5

Output: 102
105
108
120
123

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 count = 0;

        for (int i = 1; i <= 9 && count < N; i++) {
            for (int j = 0; j <= 9 && count < N; j++) {
                for (int k = 0; k <= 9 && count < N; k++) {
                    // Ensure digits are unique
                    if (i != j && j != k && i != k) {
                        int num = i * 100 + j * 10 + k;
                        if (num % 3 == 0) {
                            System.out.println(num);
                            count++;
                        }
                    }
                }
            }
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.*;

class ArrayPuzzle {
    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 secondSmallest = arr[1];
        int thirdLargest = arr[N - 3];

        System.out.println(secondSmallest + thirdLargest);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;

 class DiagonalTreasure {
    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();
            }
        }

        int mainDiagonalSum = 0;
        int secondaryDiagonalSum = 0;
```

```
        for (int i = 0; i < N; i++) {
            mainDiagonalSum += matrix[i][i];
            secondaryDiagonalSum += matrix[i][N - 1 - i];
        }

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

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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.Scanner;
 class PackageWeightSum{
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        int N= scanner.nextInt();
        int[]weights=new int[N];
        for(int i=0;i<N;i++){
            weights[i]=scanner.nextInt();

        }
        int sum=weights[0]+weights[N-1];
        System.out.println("Sumof the first and last elements:"+sum);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Sesha is developing a weather monitoring system for a region with multiple weather stations. Each weather station collects temperature data hourly and stores it in a 2D array.

Write a program that can add the temperature data from two different weather stations to create a combined temperature record for the region.

Input Format

The first line of input consists of two space-separated integers N and M, representing the number of rows and columns of the matrices, respectively.

The next N lines consist of M space-separated integers, representing the values of the first matrix.

The following N lines consist of M space-separated integers, representing the values of the second matrix.

Output Format

The output prints the addition of the two matrices in N rows and M columns, representing the combined temperature record.

Refer to the sample output for formatting specifications.

Sample Test Case

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

Output: 2 3 4
6 7 8
10 11 12

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[][] matrix1 = new int[N][M];
        int[][] matrix2 = new int[N][M];
        int[][] result = new int[N][M];


        for (int i = 0; i < N; i++) {
            for (int j = 0; j < M; j++) {
```

```
            matrix1[i][j] = sc.nextInt();
        }
    }

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < M; j++) {
            matrix2[i][j] = sc.nextInt();
        }
    }

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < M; j++) {
            result[i][j] = matrix1[i][j] + matrix2[i][j];
            System.out.print(result[i][j]);
            if (j < M - 1) System.out.print(" ");
        }
        System.out.println();
    }

    sc.close();
  }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Sharon is creating a program that finds the first repeated element in an integer array. The program should efficiently identify the first element that appears more than once in the given array. If no such element is found, it should appropriately display a message.

Help Sharon to complete the program.

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 a repeated element is found, print the first element that appears more than once.

If no repeated element is found, print "No repeated element found in the array".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 8
12 21 13 14 21 36 47 21
Output: 21

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];
      Set<Integer> seen = new HashSet<>();
      int repeated = -1;

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

      for (int i = 0; i < n; i++) {
         if (seen.contains(arr[i])) {
            repeated = arr[i];
            break;
         } else {
            seen.add(arr[i]);
         }
      }
```

```
        if (repeated != -1) {
            System.out.print(repeated);
        } else {
            System.out.print("No repeated element found in the array");
        }

        sc.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;

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

        int T = Integer.parseInt(scanner.nextLine());

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

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

            System.out.println(commas + " " + periods + " " + questions);
        }

        scanner.close();
```

```
    }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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 : 10

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

```
import java.util.*;

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

        int n = Integer.parseInt(scanner.nextLine());

        String[] keywords = scanner.nextLine().split(" ");

        Arrays.sort(keywords);


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

        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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.Scanner;

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

        for (int i = 0; i < T; i++) {
            String number = sc.nextLine();
            if (isValidMobileNumber(number)) {
                System.out.println("YES");
            } else {
                System.out.println("NO");
            }
        }
    }

    public static boolean isValidMobileNumber(String number) {

        if (number.length() != 10) return false;

        for (char ch : number.toCharArray()) {
```

```
            if (!Character.isDigit(ch)) return false;
        }

        return number.charAt(0) != '0';
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 4_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Arjun is learning how to filter words from a sentence based on grammar rules. He wants to identify the valid words in a sentence.

A word is considered valid if it satisfies all these conditions:

The word contains only alphabets (a–z, A–Z).The word length is at least 2 characters.The word should not contain digits or special characters.

Your task is to read a sentence and print all the valid words in it.

Input Format

The input contains a single line containing a sentence S.

Output Format

The output prints all the valid words separated by spaces.

If no valid word exists, print "No valid words."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Hello world1 123 ab" @#$ Hi
Output: Hello Hi

Answer

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String sentence = sc.nextLine();
        String[] words = sentence.split(" ");
        List<String> validWords = new ArrayList<>();

        for (String word : words) {
            if (isValid(word)) {
                validWords.add(word);
            }
        }

        if (validWords.isEmpty()) {
            System.out.println("No valid words.");
        } else {
            System.out.println(String.join(" ", validWords));
        }
    }

    public static boolean isValid(String word) {
        if (word.length() < 2) return false;
        for (char ch : word.toCharArray()) {
            if (!Character.isLetter(ch)) return false;
        }
        return true;
```

```
    }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 4_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a secure banking system, customers are required to create PIN codes for accessing their accounts. The bank wants to validate these PIN codes before accepting them.

A PIN code is considered valid if:

It consists of exactly 4 digits.All characters must be numeric (0–9).It cannot contain all identical digits (e.g., 1111 is invalid).

Your task is to determine whether each PIN code in the list is valid or not.

Input Format

The first line of input contains an integer T, representing the number of PIN codes to check.

The next T lines each contain a string S, representing a PIN code.

Output Format

For each PIN code S, the output print "YES" if it is valid.

Otherwise, the output print "NO".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Output: YES

Answer

```
import java.util.Scanner;

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

        for (int i = 0; i < T; i++) {
            String pin = sc.nextLine();
            if (isValidPIN(pin)) {
                System.out.println("YES");
            } else {
                System.out.println("NO");
            }
        }
    }

    public static boolean isValidPIN(String pin) {
        if (pin.length() != 4) return false;

        for (char ch : pin.toCharArray()) {
            if (!Character.isDigit(ch)) return false;
        }
```

```
        char first = pin.charAt(0);
        for (int i = 1; i < 4; i++) {
            if (pin.charAt(i) != first) {
                return true;
            }
        }

        return false;
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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.Scanner;
class Account {
    private int accountNumber;
    private String customerName;
    private double balance;
    public Account(int accountNumber, String customerName, double
initialBalance) {
        this.accountNumber = accountNumber;
        this.customerName = customerName;
        this.balance = initialBalance;
    }
    public void setAccountNumber(int accountNumber) {
        this.accountNumber = accountNumber;
    }

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }
    public int getAccountNumber() {
        return accountNumber;
    }

    public String getCustomerName() {
        return customerName;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount >= 0) {
            balance += amount;
        }
```

```
    }
    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
        }
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        scanner.nextLine();

        for (int i = 0; i < n; i++) {
            int accountNumber = scanner.nextInt();
            scanner.nextLine();
            String customerName = scanner.nextLine();
            double initialBalance = scanner.nextDouble();
            double depositAmount = scanner.nextDouble();
            double withdrawalAmount = scanner.nextDouble();

            Account customer = new Account(accountNumber, customerName,
initialBalance);
            customer.deposit(depositAmount);
            customer.withdraw(withdrawalAmount);

            System.out.println("Account Number: " + customer.getAccountNumber());
            System.out.println("Customer Name: " + customer.getCustomerName());
            System.out.printf("Final Balance: %.1f\n", customer.getBalance());
        }

        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;
```

```
class Customer {
    private int customerID;
    private String customerName;
    private double unitsConsumed;
    public Customer(int customerID, String customerName, double
unitsConsumed) {
        this.customerID = customerID;
        this.customerName = customerName;
        this.unitsConsumed = unitsConsumed;
    }
    public int getCustomerID() {
        return customerID;
    }
    public String getCustomerName() {
        return customerName;
    }
    public double calculateBill() {
        double bill = 0.0;
        double units = unitsConsumed;
        if (units <= 100) {
            bill = units * 5;
        } else if (units <= 200) {
            bill = 100 * 5 + (units - 100) * 7;
        } else {
            bill = 100 * 5 + 100 * 7 + (units - 200) * 10;
        }
        if (bill > 2000) {
            bill *= 0.95;
        }
        return bill;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = Integer.parseInt(scanner.nextLine());
        for (int i = 0; i < n; i++) {
            int customerID = Integer.parseInt(scanner.nextLine());
            String customerName = scanner.nextLine();
            double unitsConsumed = Double.parseDouble(scanner.nextLine());
            Customer customer = new Customer(customerID, customerName,
unitsConsumed);
```

```
            double finalBill = customer.calculateBill();
            System.out.println("Customer ID: " + customer.getCustomerID());
            System.out.println("Customer Name: " + customer.getCustomerName());
            System.out.printf("Final Bill: %.1f\n", finalBill);
        }
        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
import java.util.Scanner;
```

```
class Customer {
    private int customerID;
    private String customerName;
    private double unitsConsumed;
    public Customer(int customerID, String customerName, double
unitsConsumed) {
        this.customerID = customerID;
        this.customerName = customerName;
        this.unitsConsumed = unitsConsumed;
    }
    public int getCustomerID() {
        return customerID;
    }
    public String getCustomerName() {
        return customerName;
    }
    public double calculateBill() {
        double bill = 0.0;
        double units = unitsConsumed;
        if (units <= 100) {
            bill = units * 5;
        } else if (units <= 200) {
            bill = 100 * 5 + (units - 100) * 7;
        } else {
            bill = 100 * 5 + 100 * 7 + (units - 200) * 10;
        }
        if (bill > 2000) {
            bill *= 0.95;
        }
        return bill;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = Integer.parseInt(scanner.nextLine());
        for (int i = 0; i < n; i++) {
            int customerID = Integer.parseInt(scanner.nextLine());
            String customerName = scanner.nextLine();
            double unitsConsumed = Double.parseDouble(scanner.nextLine());
            Customer customer = new Customer(customerID, customerName,
unitsConsumed);
```

```
            double finalBill = customer.calculateBill();
            System.out.println("Customer ID: " + customer.getCustomerID());
            System.out.println("Customer Name: " + customer.getCustomerName());
            System.out.printf("Final Bill: %.1f\n", finalBill);
        }
        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are working as a developer for CityCab, a taxi service company that wants to build a ride fare management system.

Each customer booking has:

A Booking ID (integer)A Customer Name (string)A Distance Travelled in km (double)

The fare calculation rules are:

Base Fare = 50 units (flat charge for every ride).Per km charge = 10 units/km.If the distance is greater than 20 km, a 10% discount is applied on the total fare.

You are required to implement this system using:

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

Finally, display each booking's details and final fare.

Input Format

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

For each booking:

- The next line contains the booking ID (integer).
- The following line contains the customer's name (string).
- The next line contains the distance travelled (double).

Output Format

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

1. Booking ID: <booking_id>
2. Customer Name: <customer_name>
3. Final Fare: <final_fare> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Rahul Sharma
15

Output: Booking ID: 1234
Customer Name: Rahul Sharma
Final Fare: 200.0

Answer

```
import java.util.Scanner;
class Booking {
    private int bookingID;
```

```
    private String customerName;
    private double distance;
    public Booking(int bookingID, String customerName, double distance) {
        this.bookingID = bookingID;
        this.customerName = customerName;
        this.distance = distance;
    }
    public int getBookingID() {
        return bookingID;
    }
    public String getCustomerName() {
        return customerName;
    }
    public double getDistance() {
        return distance;
    }
    public double calculateFare() {
        double baseFare = 50;
        double perKmCharge = 10;
        double totalFare = baseFare + (distance * perKmCharge);
        if (distance > 20) {
            totalFare *= 0.9;
        }
        return totalFare;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = Integer.parseInt(scanner.nextLine());
        for (int i = 0; i < n; i++) {
            int bookingID = Integer.parseInt(scanner.nextLine());
            String customerName = scanner.nextLine();
            double distance = Double.parseDouble(scanner.nextLine());
            Booking booking = new Booking(bookingID, customerName, distance);
            double finalFare = booking.calculateFare();
            System.out.println("Booking ID: " + booking.getBookingID());
            System.out.println("Customer Name: " + booking.getCustomerName());
            System.out.printf("Final Fare: %.1f\n", finalFare);
        }
        scanner.close();
    }
```

}

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Ram is working as a developer for BrightEdu Coaching Center, which wants to build a student fee management system.

Each student's enrollment has:

An Enrollment ID (integer)A Student Name (string)The Number of Subjects (integer)

The fee calculation rules are:

Registration Fee = 1000 units (flat for every student).Per Subject Fee = 800 units.If the student enrolls in more than 5 subjects, a 20% scholarship (discount) is applied on the total fee.

Ram has been asked to implement this system using:

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

Finally, display each student's details and final fee.

Input Format

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

For each student:

- The next line contains the Enrollment ID (integer).
- The following line contains the student's name (string).
- The next line contains the Number of subjects (integer).

Output Format

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

- Enrollment ID: <enrollment_id>
- Student Name: <student_name>
- Final Fee: <final_fee> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Ravi Kumar
3
Output: Enrollment ID: 1234
Student Name: Ravi Kumar
Final Fee: 3400.0

Answer

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

class Student {
```

```
    private int enrollmentID;
    private String studentName;
    private int numberOfSubjects;

    public Student(int enrollmentID, String studentName, int numberOfSubjects) {
        this.enrollmentID = enrollmentID;
        this.studentName = studentName;
        this.numberOfSubjects = numberOfSubjects;
    }

    public int getEnrollmentID() {
        return enrollmentID;
    }

    public String getStudentName() {
        return studentName;
    }

    public int getNumberOfSubjects() {
        return numberOfSubjects;
    }

    public double calculateFee() {
        double registrationFee = 1000;
        double subjectFee = numberOfSubjects * 800;
        double totalFee = registrationFee + subjectFee;

        if (numberOfSubjects > 5) {
            totalFee *= 0.8;
        }

        return totalFee;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = Integer.parseInt(scanner.nextLine());

        for (int i = 0; i < n; i++) {
            int enrollmentID = Integer.parseInt(scanner.nextLine());
```

```
        String studentName = scanner.nextLine();
        int numberOfSubjects = Integer.parseInt(scanner.nextLine());

        Student student = new Student(enrollmentID, studentName,
numberOfSubjects);
        double finalFee = student.calculateFee();

        System.out.println("Enrollment ID: " + student.getEnrollmentID());
        System.out.println("Student Name: " + student.getStudentName());
        System.out.printf("Final Fee: %.1f\n", finalFee);
    }

    scanner.close();
  }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 6_Q1

Attempt : 2
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;

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

class Subscription {
    protected double baseMonthlyCost;
    protected double serviceTax;

    public Subscription(double baseMonthlyCost, double serviceTax) {
        this.baseMonthlyCost = baseMonthlyCost;
        this.serviceTax = serviceTax;
    }

    public double getBaseAndTax() {
        return baseMonthlyCost + serviceTax;
    }
}

class PremiumSubscription extends Subscription {
    private double extraFeatureCost;

    public PremiumSubscription(double baseMonthlyCost, double serviceTax,
double extraFeatureCost) {
        super(baseMonthlyCost, serviceTax);
        this.extraFeatureCost = extraFeatureCost;
```

```
    }

    public double calculateMonthlyCost() {
        return getBaseAndTax() + extraFeatureCost;
    }
}


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: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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 : 10

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
import java.util.Scanner;

class Product {
    public double price;

    public Product(double price) {
        this.price = price;
    }
}

class DiscountedProduct extends Product {
    private double discountRate;

    public DiscountedProduct(double price, double discountRate) {
        super(price);
        this.discountRate = discountRate;
    }
```

```
    public double calculateSellingPrice() {
        if (discountRate > 1.0) {
            return -1;
        }
        return price * (1 - discountRate);
    }
}

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 6_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Preethi is working on a project to automate sales tax calculations for items in a store. She wants to create a program that takes the price of an item and the sales tax rate as input and calculates the final price of the item after applying the sales tax.

Write a program using the class SalesTaxCalculator, which contains an overloaded method named calculateFinalPrice to handle both integer and double inputs. The program should also include a Main class that takes user input, calls the appropriate method from SalesTaxCalculator, and prints the final price of the item.

Formula Used: Final price = price + ((price * sales tax rate) / 100)

Input Format

The first line of input consists of an integer price (the price of the item for integer inputs).

The second line of input consists of an integer taxRate (the sales tax rate for integer inputs).

The third line of input consists of a double price (the price of the item for double inputs).

The fourth line of input consists of a double taxRate (the sales tax rate for double inputs).

Output Format

The first line of output prints an integer, representing the final price of the item after applying the sales tax for integer inputs (a and b).

The second line prints a double value, representing the final price of the item after applying the sales tax for double-value inputs (m and n), rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 100
10
100.0
5.0

Output: 110
105.00

Answer

```
import java.util.Scanner;

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

class SalesTaxCalculator {
    public static int calculateFinalPrice(int price, int taxRate) {
        return price + (price * taxRate / 100);
```

```
    }
    public static double calculateFinalPrice(double price, double taxRate) {
        return price + ((price * taxRate) / 100);
    }
}

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int intPrice = scanner.nextInt();
        int intTaxRate = scanner.nextInt();
        double doublePrice = scanner.nextDouble();
        double doubleTaxRate = scanner.nextDouble();

        int finalPriceInt = SalesTaxCalculator.calculateFinalPrice(intPrice,
intTaxRate);
        double finalPriceDouble =
SalesTaxCalculator.calculateFinalPrice(doublePrice, doubleTaxRate);

        System.out.println(finalPriceInt);
        System.out.format("%.2f", finalPriceDouble);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 6_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Mr.Kapoor wants to create a program to calculate the volume of a Cuboid and a Cube using method overriding.

Implements a base class Cuboid with attributes for length, width, and height. Include a method calculateVolume() that computes the volume of the cuboid.

Extends the base class with a subclass Cube representing a cube, where all sides are equal. Override the calculateVolume() method in the Cube class to compute the volume of the cube.

The program should take user input for the dimensions of the cuboid and the side length of the cube and display the calculated volumes with two decimal places.

Input Format

The first line of input consists of 3 space-separated double values, representing the cuboid length, width, and height, respectively.

The second line consists of a double value, representing the side length of the cube.

Output Format

The first line of output prints the volume of the cuboid, rounded off to two decimal places.

The second line prints the volume of the cube, rounded off to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 60.0 60.0 60.0
50.0

Output: Volume of Cuboid: 216000.00
Volume of Cube: 125000.00

Answer

```
import java.util.Scanner;

// You are using Java
import java.util.Scanner;
class Cuboid {
    protected double length;
    protected double width;
    protected double height;
    public Cuboid(double length, double width, double height) {
        this.length = length;
        this.width = width;
        this.height = height;
    }
    public double calculateVolume() {
        return length * width * height;
```

```
    }
}
class Cube extends Cuboid {
    public Cube(double side) {
        super(side, side, side);
    }
    public double calculateVolume() {
        return length * length * length;
    }
}

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

        double cuboidLength = scanner.nextDouble();
        double cuboidWidth = scanner.nextDouble();
        double cuboidHeight = scanner.nextDouble();

        // Regular object instantiation for Cuboid
        Cuboid cuboid = new Cuboid(cuboidLength, cuboidWidth, cuboidHeight);
        System.out.printf("Volume of Cuboid: %.2f\n", cuboid.calculateVolume());

        double cubeSide = scanner.nextDouble();

        // Upcasting - Using superclass reference for subclass object (DMD)
        Cuboid cube = new Cube(cubeSide); // Upcasting
        System.out.printf("Volume of Cube: %.2f", cube.calculateVolume()); // Calls
Cube's method dynamically

        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 6_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem statement:

Tim was tasked with developing a grocery shopping app. You have a class hierarchy that includes Item, Produce, and OrganicProduce. Your goal is to calculate the total cost of a shopping list, which may contain a mix of regular produce and organic produce items. Additionally, you need to apply discounts to organic items. Apply a 10% discount on organic produce items

Class Hierarchy:

Item: Base class for all items.

Produce: Subclass of Item for regular produce items.

OrganicProduce: Subclass of Produce for organic produce items.

Input Format

The first line of input consists of an integer, 'n'.

For each 'n' item, the user will provide:

- A string 'type' representing the item type ('Regular' or 'Organic').
- A string 'name' represents the item name.
- A double 'price' represents the item price.

Output Format

The output will display the total cost of the shopping list, including discounts on organic items.

Refer to the sample output for format specifications.

Sample Test Case

Input: 1
Regular Banana 1.99
Output: 1.99

Answer

```
import java.util.Scanner;
```

```
import java.util.Scanner;
class Item {
    protected String name;
    protected double price;

    public Item(String name, double price) {
        this.name = name;
        this.price = price;
    }

    public double calculateCost() {
        return price;
    }
}
class Produce extends Item {
```

```
    public Produce(String name, double price) {
        super(name, price);
    }
}
class OrganicProduce extends Produce {
    public OrganicProduce(String name, double price) {
        super(name, price);
    }
    public double calculateCost() {
        return price * 0.90;
    }
}

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

        int n = sc.nextInt();
        sc.nextLine(); // Consume newline

        double totalCost = 0.0;

        for (int i = 0; i < n; i++) {
            String type = sc.next();
            String name = sc.next();
            double price = sc.nextDouble();

            if (type.equals("Regular")) {
                Item item = new Produce(name, price);
                totalCost += item.calculateCost();
            } else if (type.equals("Organic")) {
                Item item = new OrganicProduce(name, price);
                totalCost += item.calculateCost();
            }
        }

        System.out.printf("%.2f%n", totalCost);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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 : 10

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;
// You are using Java
```

```
import java.util.*;

interface CostCalculator {
    void getEnergyDetails(Scanner sc);
    void calculateAndDisplayCost();
}

class EnergyConsumptionTracker implements CostCalculator {
    private double ratePerUnit;
    private int numDays;
    private double[] energyConsumed;

    EnergyConsumptionTracker(double ratePerUnit, int numDays) {
        this.ratePerUnit = ratePerUnit;
        this.numDays = numDays;
        this.energyConsumed = new double[numDays];
    }

    public void getEnergyDetails(Scanner sc) {
        for (int i = 0; i < numDays; i++) {
            energyConsumed[i] = sc.nextDouble();
        }
    }

    public void calculateAndDisplayCost() {
        System.out.println("Day-wise Energy Cost:");
        double totalCost = 0.0;

        for (int i = 0; i < numDays; i++) {
            double dayCost = energyConsumed[i] * ratePerUnit;
            totalCost += dayCost;
            System.out.printf("Day %d: Rs. %.2f%n", (i + 1), dayCost);
        }

        System.out.printf("Total Energy Cost: Rs. %.2f", 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 : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

interface HealthCalculator {
    double calculateBMI(double weight, double height);
}

class BMICalculator implements HealthCalculator {
    public double calculateBMI(double weight, double height) {
        if (weight <= 0 || height <= 0) {
            return -1;
        }
        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: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 7_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

A financial analyst, Alex, needs a program to calculate simple interest for various financial transactions. He requires a straightforward tool that takes in the principal amount, interest rate, and time in years and computes the interest.

The formula to be used is: Interest = Principal × Rate × Time / 100

Implement this functionality using the InterestCalculator interface and the SimpleInterestCalculator class.

Input Format

The first line of input consists of the principal amount P as a double value.

The second line of input consists of the annual interest rate r as a double value.

The third line of input consists of the number of years t as a positive integer, which is an integer value.

Output Format

The output displays the calculated simple interest in the following format: "Simple Interest: [interest_value]", Here, [interest_value] should be replaced with the actual interest value calculated by the program.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1000.00
5.00
2
Output: Simple Interest: 100.0

Answer

```
import java.util.Scanner;

// You are using Java
interface InterestCalculator {
    double simpleInterest(double principal, double rate, int time);
}
class SimpleInterestCalculator implements InterestCalculator {
    public double simpleInterest(double principal, double rate, int time) {
        return (principal * rate * time) / 100;
    }
}

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

        double principal = scanner.nextDouble();

        double rate = scanner.nextDouble();

        int time = scanner.nextInt();
```

```
        InterestCalculator calculator = new SimpleInterestCalculator();
        double interest = calculator.simpleInterest(principal, rate, time);
        System.out.println("Simple Interest: " + interest);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 7_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Maria, a software developer, is working on an inventory management system project using Java that utilizes an inventory interface to manage a store's products.

The interface should define two methods: addProduct, which adds a product by accepting its name, price, and quantity, and calculateTotalValue, which computes the total value of all products in the inventory. Implement the interface in a class called SimpleInventory, which internally manages a list of Product objects.

Each Product object should encapsulate the product's name, price, and quantity and include a method to calculate its value as price × quantity. The system should allow users to dynamically add products to the inventory and calculate the total value of all products stored.

Help Maria achieve the task.

Input Format

The first line of input consists of an integer to choose one of the following options:

1 - to add a product to the inventory.

2 - to calculate and view the total inventory value.

3 - to exit the program.

For Choice 1 (Add Product):

The next input line is the string representing the product name as a string (single or multi-word, without quotes).

The next line is a double value representing the price as a decimal value

The next line is an integer value representing the quantity as an integer

For Choices 2 and 3, no additional input is required

Output Format

The output displays the results of the commands as follows:

- For the addProduct command, the program should display "Product added to inventory."
- For choice 2, the program should display "Total inventory value [totalvalue]. "The total value should be displayed with one decimal place. If there is no product in the inventory, print the total as 0.0.
- For choice 3, the program should exit

If the choice is not 1, 2, or 3, then print "Invalid choice. Please select a valid option (1/2/3).".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1
Laptop
800.0
3
2
5
3

Output: Product added to inventory.
Total inventory value: $2400.0
Invalid choice. Please select a valid option (1/2/3).

Answer

```
import java.util.Scanner;
import java.util.*;
interface Inventory {
    void addProduct(String name, double price, int quantity);
    double calculateTotalValue();
}
class Product {
    private String name;
    private double price;
    private int quantity;

    public Product(String name, double price, int quantity) {
        this.name = name;
        this.price = price;
        this.quantity = quantity;
    }

    public double getValue() {
        return price * quantity;
    }
}
class SimpleInventory implements Inventory {
    private List<Product> products;

    public SimpleInventory(int capacity) {
        products = new ArrayList<>(capacity);
    }
    public void addProduct(String name, double price, int quantity) {
```

```
        products.add(new Product(name, price, quantity));
        System.out.println("Product added to inventory.");
    }
    public double calculateTotalValue() {
        double total = 0.0;
        for (Product p : products) {
            total += p.getValue();
        }
        return total;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Inventory inventory = new SimpleInventory(10);
        while (true) {
            int choice = scanner.nextInt();
            if (choice == 1) {
                scanner.nextLine();
                String productName = scanner.nextLine();
                double price = scanner.nextDouble();
                int quantity = scanner.nextInt();
                inventory.addProduct(productName, price, quantity);
            } else if (choice == 2) {
                double totalValue = inventory.calculateTotalValue();
                System.out.println("Total inventory value: $" + totalValue);
            } else if (choice == 3) {
                break;
            } else {
                System.out.println("Invalid choice. Please select a valid option
(1/2/3).");
            }
        }
        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 7_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Raj is curious about how old he is in the current year.

He has asked you to create a simple program that calculates a person's age based on their birth year. You decide to implement this functionality using the AgeCalculator interface and the HumanAgeCalculator class.

Note: The current year is 2024. Calculate the current age by using the formula: current year - birth year.

Input Format

The input consists of an integer representing the birth year.

Output Format

The output displays "You are X years old." where X is an integer representing the calculated age based on the entered birth year.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1934

Output: You are 90 years old.

Answer

```
import java.util.Scanner;


interface AgeCalculator {
    int calculateAge(int birthYear);
}


class HumanAgeCalculator implements AgeCalculator {
    private static final int CURRENT_YEAR = 2024;

    @Override
    public int calculateAge(int birthYear) {
        return CURRENT_YEAR - birthYear;
    }
}


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


        if (birthYear < 1900 || birthYear > 2022) {
            System.out.println("Invalid birth year. Please enter a year between 1900
and 2022.");
```

```
            return;
        }

        AgeCalculator calculator = new HumanAgeCalculator();
        int age = calculator.calculateAge(birthYear);

        System.out.println("You are " + age + " years old.");
    }
}

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

        AgeCalculator ageCalculator = new HumanAgeCalculator();

        int birthYear = scanner.nextInt();
        int age = ageCalculator.calculateAge(birthYear);

        System.out.println("You are " + age + " years old.");
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
// You are using Java
import java.util.*;
class DotException extends Exception {
    public DotException(String msg) {
        super(msg);
    }
}
class AtTheRateException extends Exception {
    public AtTheRateException(String msg) {
        super(msg);
    }
}
class DomainException extends Exception {
    public DomainException(String msg) {
        super(msg);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String email = sc.nextLine().trim();
        sc.close();
        try {
            validateEmail(email);
            System.out.println("Valid email address");
        }
        catch (DotException e) {
            System.out.println("DotException: " + e.getMessage());
            System.out.println("Invalid email address");
        }
        catch (AtTheRateException e) {
            System.out.println("AtTheRateException: " + e.getMessage());
```

```
            System.out.println("Invalid email address");
        }
        catch (DomainException e) {
            System.out.println("DomainException: " + e.getMessage());
            System.out.println("Invalid email address");
        }
    }
    public static void validateEmail(String email) throws DotException,
AtTheRateException, DomainException {        int atCount = email.length() -
email.replace("@", "").length();
        if (atCount != 1 || email.startsWith("@") || email.endsWith("@") ||
email.contains("@@")) {
            throw new AtTheRateException("Invalid @ usage");
        }
        int dotCount = email.length() - email.replace(".", "").length();
        if (dotCount < 1 || email.endsWith(".") || email.startsWith(".") ||
email.contains("..")) {
            throw new DotException("Invalid Dot usage");
        }
        int atIndex = email.indexOf('@');
        int lastDotIndex = email.lastIndexOf('.');
        if (lastDotIndex < atIndex + 2) {
            throw new DotException("Invalid Dot usage");
        }
        String domain = email.substring(lastDotIndex + 1);
        List<String> validDomains = Arrays.asList("in", "com", "net", "biz");
        if (!validDomains.contains(domain)) {
            throw new DomainException("Invalid Domain");
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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.*;

class InvalidDurationException extends Exception {
    InvalidDurationException(String msg) {
        super(msg);
    }
}

class ElsaMeetingScheduler {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
```

```
        sc.close();
        try {
            if (n <= 0 || n > 240)
                throw new InvalidDurationException("Invalid meeting duration. Please
enter a positive integer not exceeding 240 minutes (4 hours).");
            System.out.println("Meeting scheduled successfully!");
        } catch (InvalidDurationException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

A local municipality is implementing an online voting system for a community event and wants to ensure that only eligible voters (those aged 18 or older) can participate.

Your task is to develop a program that validates the age of individuals attempting to vote online. If the user's age is below 18, the program should throw a custom exception, InvalidAgeException, preventing them from casting their vote. If the input is invalid, catch the appropriate InputMismatchException and print the in-built exception message.

Input Format

The input consists of an integer representing the age.

Output Format

If the age is 18 or older, print "Eligible to vote"

If the age is below 18, print "Exception occurred: InvalidAgeException: Age is not valid to vote"

If there is any other type of exception, print "An error occurred: " followed by the in-built exception message.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 20
Output: Eligible to vote

Answer

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

class InvalidAgeException extends Exception {
    InvalidAgeException(String msg) {
        super(msg);
    }
}

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        try {
            int age = sc.nextInt();
            sc.close();
            validateAge(age);
            System.out.println("Eligible to vote");
        }
        catch (InvalidAgeException e) {
            System.out.println("Exception occurred: InvalidAgeException: " +
e.getMessage());
        }
        catch (InputMismatchException e) {
            System.out.println("An error occurred: " + e);
```

```
        }
        catch (Exception e) {
            System.out.println("An error occurred: " + e);
        }
    }

    public static void validateAge(int age) throws InvalidAgeException {
        if (age < 18) {
            throw new InvalidAgeException("Age is not valid to vote");
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a file management system, users are required to provide a valid file name when creating new files. The system enforces specific rules for file names to maintain consistency and avoid potential issues. Your task is to implement a Java program named FileNameValidator that takes user input for a file name and validates it according to the specified rules.

Rules for Valid File Name:

The file name must consist of alphanumeric characters (letters and digits) only.The file name must have a minimum length of 3 characters.

Implement a custom exception, FileNameValidator, to handle cases where the entered filename does not meet the specified criteria.

Input Format

The input consists of a string S, representing the desired filename.

Output Format

The output is displayed in the following format:

If the entered file name meets the specified criteria, the program outputs

"Valid file name"

If the entered file name does not meet the criteria and triggers the InvalidFileNameException, the program outputs

"Error: Invalid file name. It must be alphanumeric and have a minimum length of 3 characters."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: myfile123
Output: Valid file name

Answer

```
import java.util.Scanner;

class InvalidFileNameException extends Exception {
    public InvalidFileNameException(String message) {
        super(message);
    }
}

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

        try {
            validateFileName(fileName);
            System.out.println("Valid file name");
```

```
        } catch (InvalidFileNameException e) {
            System.out.println(e.getMessage());
        }
    }

    public static void validateFileName(String fileName) throws
InvalidFileNameException {
        if (fileName.length() < 3 || !fileName.matches("[A-Za-z0-9]+")) {
            throw new InvalidFileNameException(
                "Error: Invalid file name. It must be alphanumeric and have a minimum
length of 3 characters."
            );
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

```
// 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();
        ArrayList<Integer> list = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int num = sc.nextInt();
            if (list.isEmpty() || num > list.get(list.size() - 1)) {
                list.add(num);
            }
        }

        System.out.print(list);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

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

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());
        LinkedList<String> playlist = new LinkedList<>();
        int current = 0;

        while (n-- > 0) {
            String input = sc.nextLine();
            String[] parts = input.split(" ", 2);
            String cmd = parts[0];

            if (cmd.equals("ADD")) {
                playlist.add(parts[1]);
            }
            else if (cmd.equals("REMOVE")) {
                String song = parts[1];
                int idx = playlist.indexOf(song);
                if (idx != -1) {
                    playlist.remove(idx);
                    if (playlist.isEmpty()) current = 0;
                    else if (idx < current) current--;
                    else if (current >= playlist.size()) current = 0;
                }
            }
            else if (cmd.equals("SHOW")) {
                if (playlist.isEmpty()) {
                    System.out.println("EMPTY");
                } else {
                    for (String s : playlist) {
                        System.out.print(s + " ");
                    }
                    System.out.println();
                }
            }
            else if (cmd.equals("NEXT")) {
                if (playlist.isEmpty()) {
                    System.out.println("EMPTY");
                } else {
```

```
                current = (current + 1) % playlist.size();
                System.out.println(playlist.get(current));
            }
        }
    }
  }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 9_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Assist Pranitha in developing a program that takes an integer N as input, representing the number of names to be read. Then read N names and store them in an ArrayList. Finally, input a search string and output the frequency of that string in the list of names.

Note: Some parts of the code are provided as snippets, and you need to complete the remaining sections by writing the necessary code.

Input Format

The first line of input consists of an integer N, representing the number of names to be read.

The following N lines consist of N names, as a string.

The last line consists of a string, representing the name to be searched.

Output Format

The output prints a single integer, representing the frequency of the specified name in the given list.

If the specified name is not found, print 0.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
Alice
Bob
Ankit
Alice
Pranitha
Alice

Output: 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();
        sc.nextLine(); // consume the newline after integer input

        ArrayList<String> names = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            names.add(sc.nextLine());
        }

        String searchName = sc.nextLine();
        int count = 0;
```

```
        for (String name : names) {
            if (name.equals(searchName)) {
                count++;
            }
        }

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

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

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

class Vehicle {
    String r, o, t;
    Vehicle(String r, String o, String t) { this.r = r; this.o = o; this.t = t; }
    public int hashCode() { return r.hashCode(); }
    public boolean equals(Object x) { return x instanceof Vehicle &&
r.equals(((Vehicle)x).r); }
```

```
    public String toString() { return r + " " + o + " " + t; }
}

public class Main {
    public static void main(String[] a) {
        Scanner s = new Scanner(System.in);
        int n = s.nextInt();
        HashSet<Vehicle> h = new HashSet<>();
        for (int i = 0; i < n; i++) h.add(new Vehicle(s.next(), s.next(), s.next()));
        for (Vehicle v : h) System.out.println(v);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

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

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        HashMap<String, Double> map = new HashMap<>();
        boolean invalidInput = false, invalidFormat = false;

        while (true) {
            String line = sc.nextLine();
            if (line.equals("done")) break;
            if (!line.contains(":")) { invalidFormat = true; break; }
            if (!line.matches("[A-Za-z:0-9.]+")) { invalidFormat = true; break; }

            String[] parts = line.split(":");
            if (parts.length != 2) { invalidFormat = true; break; }

            try {
                double val = Double.parseDouble(parts[1]);
                map.put(parts[0], val);
            } catch (Exception e) {
```

```
                invalidInput = true; break;
            }
        }

        if (invalidFormat) System.out.print("Invalid format");
        else if (invalidInput) System.out.print("Invalid input");
        else {
            double sum = 0;
            for (double v : map.values()) sum += v;
            System.out.printf("%.2f", sum);
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

Priya is analyzing encrypted messages in a research project. She wants to analyze the frequency of each character in a given paragraph. The characters should be stored in a TreeMap so that the output is sorted in ascending order of characters automatically.

You are required to build a Java program that:

Uses a TreeMap<Character, Integer> to count how many times each character appears in the message.Ignores spaces and considers only alphabets (case-sensitive).Outputs the frequencies of characters in sorted order.

You must use a TreeMap in the class named MessageAnalyzer.

Input Format

The first line of input contains an integer n, the number of lines in the message.

The next n lines each contain a string (the encrypted message line).

Output Format

The first line of output prints: "Character Frequency:"

Then print each character and its frequency in the format: "<character>: <count>"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2
Hello World
Java
Output: Character Frequency:
H: 1
J: 1
W: 1
a: 2
d: 1
e: 1
l: 3
o: 2
r: 1
v: 1

Answer

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

class MessageAnalyzer {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());
        TreeMap<Character, Integer> map = new TreeMap<>();

        for (int i = 0; i < n; i++) {
```

```
        String line = sc.nextLine();
        for (char c : line.toCharArray()) {
            if (Character.isLetter(c)) {
                map.put(c, map.getOrDefault(c, 0) + 1);
            }
        }
    }

    System.out.println("Character Frequency:");
    for (Map.Entry<Character, Integer> e : map.entrySet()) {
        System.out.println(e.getKey() + ": " + e.getValue());
    }
  }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

In a ticket reservation system, you store the available seat numbers in a TreeSet. Users input their desired seat number, and the program checks whether the chosen seat is available.

Using a TreeSet ensures quick and efficient verification of seat availability, ensuring a smooth and organized ticket booking process.

Input Format

The first line of input contains a single integer n, representing the number of available seats.

The second line contains n space-separated integers, representing the available seat numbers.

The third line contains an integer m, representing the seat number that needs to be searched.

Output Format

The output displays "[m] is present!" if the given seat is available. Otherwise, it displays "[m] is not present!"

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 4
2 4 5 6
5
Output: 5 is present!

Answer

```
// You are using Java
import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        TreeSet<Integer> seats = new TreeSet<>();
        for (int i = 0; i < n; i++) {
            seats.add(sc.nextInt());
        }
        int m = sc.nextInt();
        if (seats.contains(m)) {
            System.out.println(m + " is present!");
        } else {
            System.out.println(m + " is not present!");
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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 : 10

Section 1 : Project

1. Problem Statement

In ABC Corporation, employee records are stored in a database.

To efficiently manage employee details using Java and JDBC, you are tasked with building an Employee Management System that supports the following functionalities:

Adding a new employee

Updating an employee's salary

Viewing an employee's details

Displaying all employees

You are given two files:

File 1: Employee.java (POJO Class)

This class represents the Employee entity.

An Employee contains the following details:

Field Description

employeeId Unique Employee ID (Integer)

name Employee Name (String)

department Employee Department (String)

salary Employee Salary (Double)

Students must write code in the marked area:

```
class Employee {
    private int employeeId;
    private String name;
    private String department;
    private double salary;

    public Employee() {}

    public Employee(int employeeId, String name, String department, double salary) {
        // 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: EmployeeDAO.java (Data Access Layer)

This class handles all database operations using JDBC.

Students must complete the missing JDBC logic in the following methods:

```
class EmployeeDAO {

    public void addEmployee(Connection conn, Employee employee) throws
SQLException {
        // write your code here
    }

    public void updateSalary(Connection conn, int employeeId, double
newSalary) throws SQLException {
        // write your code here
    }

    public void deleteEmployee(Connection conn, int employeeId) throws
SQLException {
        // write your code here
    }

    public Employee viewEmployeeRecord(Connection conn, int employeeId)
throws SQLException {
        // write your code here
    }

    public List<Employee> displayAllEmployees(Connection conn) throws
SQLException {
        // write your code here
    }
```

```
    private Employee mapToEmployee(ResultSet rs) throws SQLException {
        return new Employee(
            // 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 Employee objects using mapToEmployee().

Return a List<Employee> where required.

The system should connect to a MySQL database using the following default credentials:

DB URL: jdbc:mysql://localhost/ri_dbUsername: testPassword: test123

The employees table has already been created with the following structure:

Input Format

The first line of input consists of an integer choice, representing the operation to be performed:

(1 for Add Employee, 2 for Update Salary, 3 for View Employee Record, 4 for Display All Employees, 5 for Exit)

For choice 1 (Add Employee):

1. The second line consists of an integer employee_id.
2. The third line consists of a string name.
3. The fourth line consists of a string department.
4. The fifth line consists of a double salary (must be at least 30000).

For choice 2 (Update Salary):

1. The second line consists of an integer employee_id.
2. The third line consists of a double new_salary (must be at least 30000).

For choice 3 (View Employee Record):

1. The second line consists of an integer employee_id.

For choice 4 (Display All Employees).

For choice 5 (Exit).

Output Format

For choice 1 (Add Employee),

1. Print "Employee added successfully" if the employee was added.

For choice 2 (Update Salary),

1. Print "Salary updated successfully" if the salary update was successful.
2. Print "Employee not found." if the specified employee ID does not exist.
3. Print "Salary must be at least 30000." if the provided salary is below the minimum.

For choice 3 (View Employee Record),

1. Display the employee details in the format:
2. ID: [employee_id] | Name: [name] | Department: [department] | Salary: [salary]
3. Print "Employee not found." if the specified employee ID does not exist.

For choice 4 (Display All Employees),

1. Display each employee on a new line in the format:
2. ID | Name | Department | Salary

For choice 5 (Exit),

1. Print "Exiting Employee Management System."

For invalid input:

1. Print "Invalid choice. Please try again."

Sample Test Case

Input: 1
101
Alice Johnson
Engineering
31000.75
4
6
5
Output: Employee added successfully
ID | Name | Department | Salary
101 | Alice Johnson | Engineering | 31000.75
Invalid choice. Please try again.
Exiting Employee Management System.

Answer

```
import java.sql.*;
import java.util.Scanner;
```

```
import java.sql.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

// Employee POJO Class
```

```
class Employee {
    private int employeeId;
    private String name;
    private String department;
    private double salary;

    // Constructor
    public Employee(int employeeId, String name, String department, double
salary) {
        this.employeeId = employeeId;
        this.name = name;
        this.department = department;
        this.salary = salary;
    }

    // Getters and Setters
    public int getEmployeeId() { return employeeId; }
    public void setEmployeeId(int employeeId) { this.employeeId = employeeId; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public String getDepartment() { return department; }
    public void setDepartment(String department) { this.department =
department; }

    public double getSalary() { return salary; }
    public void setSalary(double salary) { this.salary = salary; }
}

// Employee Management System
class EmployeeManagementSystem {

    public static void addEmployee(Connection conn, Scanner scanner) {
        try {
            int id = scanner.nextInt();
            scanner.nextLine(); // consume newline
            String name = scanner.nextLine();
            String department = scanner.nextLine();
            double salary = scanner.nextDouble();

            if (salary < 30000) {
```

```
            System.out.println("Salary must be at least 30000.");
            return;
        }

        String sql = "INSERT INTO employees(employeeId, name, department,
salary) VALUES (?, ?, ?, ?)";
        try (PreparedStatement ps = conn.prepareStatement(sql)) {
            ps.setInt(1, id);
            ps.setString(2, name);
            ps.setString(3, department);
            ps.setDouble(4, salary);
            ps.executeUpdate();
            System.out.println("Employee added successfully");
        }

    } catch (SQLException e) {
        System.out.println("Employee not found.");
    }
  }

  public static void updateSalary(Connection conn, Scanner scanner) {
    try {
        int id = scanner.nextInt();
        double newSalary = scanner.nextDouble();

        if (newSalary < 30000) {
            System.out.println("Salary must be at least 30000.");
            return;
        }

        String checkSql = "SELECT * FROM employees WHERE employeeId=?";
        try (PreparedStatement checkStmt = conn.prepareStatement(checkSql)) {
            checkStmt.setInt(1, id);
            ResultSet rs = checkStmt.executeQuery();
            if (!rs.next()) {
                System.out.println("Employee not found.");
                return;
            }
        }

        String updateSql = "UPDATE employees SET salary=? WHERE
employeeId=?";
```

```
            try (PreparedStatement ps = conn.prepareStatement(updateSql)) {
                ps.setDouble(1, newSalary);
                ps.setInt(2, id);
                ps.executeUpdate();
                System.out.println("Salary updated successfully");
            }

        } catch (SQLException e) {
            System.out.println("Employee not found.");
        }
    }

    public static void viewEmployeeRecord(Connection conn, Scanner scanner) {
        try {
            int id = scanner.nextInt();

            String sql = "SELECT * FROM employees WHERE employeeId=?";
            try (PreparedStatement ps = conn.prepareStatement(sql)) {
                ps.setInt(1, id);
                ResultSet rs = ps.executeQuery();
                if (rs.next()) {
                    System.out.printf("ID: %d | Name: %s | Department: %s | Salary: %.2f
%n",
                            rs.getInt("employeeId"),
                            rs.getString("name"),
                            rs.getString("department"),
                            rs.getDouble("salary"));
                } else {
                    System.out.println("Employee not found.");
                }
            }

        } catch (SQLException e) {
            System.out.println("Employee not found.");
        }
    }

    public static void displayAllEmployees(Connection conn) {
        String sql = "SELECT * FROM employees";
        try (Statement stmt = conn.createStatement()) {
            ResultSet rs = stmt.executeQuery(sql);
            System.out.println("ID | Name | Department | Salary");
```

```
            while (rs.next()) {
                System.out.printf("%d | %s | %s | %.2f%n",
                        rs.getInt("employeeId"),
                        rs.getString("name"),
                        rs.getString("department"),
                        rs.getDouble("salary"));
            }
        } catch (SQLException e) {
            System.out.println("Error fetching employee records.");
        }
    }

    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost/ri_db";
        String username = "test";
        String password = "test123";

        try (Connection conn = DriverManager.getConnection(url, username,
password);
             Scanner scanner = new Scanner(System.in)) {

            int choice;
            do {
                choice = scanner.nextInt();

                switch (choice) {
                    case 1 -> addEmployee(conn, scanner);
                    case 2 -> updateSalary(conn, scanner);
                    case 3 -> viewEmployeeRecord(conn, scanner);
                    case 4 -> displayAllEmployees(conn);
                    case 5 -> System.out.println("Exiting Employee Management
System.");
                    default -> System.out.println("Invalid choice. Please try again.");
                }

            } while (choice != 5);

        } catch (SQLException e) {
            System.out.println("Database Error: " + e.getMessage());
        }
    }
}
```

```
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost/ri_db";
        String username = "test";
        String password = "test123";

        try (Connection conn = DriverManager.getConnection(url, username,
password);
             Scanner scanner = new Scanner(System.in)) {

            int choice;
            do {
                choice = scanner.nextInt();

                switch (choice) {
                    case 1 -> addEmployee(conn, scanner);
                    case 2 -> updateSalary(conn, scanner);
                    case 3 -> viewEmployeeRecord(conn, scanner);
                    case 4 -> displayAllEmployees(conn);
                    case 5 -> System.out.println("Exiting Employee Management
System.");
                    default -> System.out.println("Invalid choice. Please try again.");
                }

            } while (choice != 5);

        } catch (SQLException e) {
            System.out.println("Database Error: " + e.getMessage());
        }
    }
}
```

Status : Wrong **Marks : 0/10**

2. Problem Statement

Create a JDBC-based Inventory Management System that handles runtime input to manage items in an inventory. The system should allow users to:

Add a new item (item ID, name, quantity, price).

Restock an item by increasing its quantity.

Reduce the stock of an item, ensuring sufficient quantity.

Display all items in the inventory in a sorted order by item ID.

Exit the application.

Half of the code is given here; Only the remaining part should be completed.

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 items table has already been created with the following structure:

Table Name: items

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 Item):

- The second line consists of an integer item_id.
- The third line consists of a string name.
- The fourth line consists of an integer quantity.
- The fifth line consists of a double price.

For choice 2 (Restock Item):

- The second line consists of an integer item_id.
- The third line consists of an integer quantity_to_add (must be positive).

For choice 3 (Reduce Stock):

- The second line consists of an integer item_id.
- The third line consists of an integer quantity_to_remove (must be positive).

For choice 4 (Display Inventory):

- No additional inputs are required.

For choice 5 (Exit):

- No additional inputs are required.

Output Format

For choice 1 (Add Item):

- Print "Item added successfully" if the item was added.
- Print "Failed to add item." if the insertion failed.

For choice 2 (Restock Item):

- Print "Item restocked successfully" if the restock was successful.
- Print "Item not found." if the specified item ID does not exist.

For choice 3 (Reduce Stock):

- Print "Stock reduced successfully" if the stock reduction was successful.
- Print "Not enough stock to remove." if there is insufficient quantity.
- Print "Item not found." if the specified item ID does not exist.

For choice 4 (Display Inventory):

- Display each item on a new line in the format:
- ID | Name | Quantity | Price
- If no items are available, print nothing (or handle with an appropriate message if desired).

For choice 5 (Exit):

- Print "Exiting Inventory Management System."

For invalid input:

- Print "Invalid choice. Please try again."

Sample Test Case

Input: 1
101
Laptop
50
1200.00
4
5
Output: Item added successfully
ID | Name | Quantity | Price
101 | Laptop | 50 | 1200.00
Exiting Inventory Management System.

Answer

```
import java.sql.*;
import java.util.Scanner;

class InventoryManagementSystem {
   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:
                  addItem(conn, scanner);
                  break;
               case 2:
                  restockItem(conn, scanner);
                  break;
               case 3:
                  reduceStock(conn, scanner);
```

```
                    break;
                case 4:
                    displayInventory(conn);
                    break;
                case 5:
                    System.out.println("Exiting Inventory Management System.");
                    running = false;
                    break;
                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public static void addItem(Connection conn, Scanner scanner) {
    int itemId = scanner.nextInt();
    scanner.nextLine();

    String name = scanner.nextLine();

    int quantity = scanner.nextInt();

    double price = scanner.nextDouble();

    String insertQuery = "INSERT INTO items (item_id, name, quantity, price)
VALUES (?, ?, ?, ?)";
    try (PreparedStatement stmt = conn.prepareStatement(insertQuery)) {
        stmt.setInt(1, itemId);
        stmt.setString(2, name);
        stmt.setInt(3, quantity);
        stmt.setDouble(4, price);

        int rowsInserted = stmt.executeUpdate();
        System.out.println(rowsInserted > 0 ? "Item added successfully" : "Failed
to add item.");
    } catch (SQLException e) {
        System.out.println("Error adding item: " + e.getMessage());
    }
}
```

```
    public static void restockItem(Connection conn, Scanner scanner) {
        int itemId = scanner.nextInt();

        int quantityToAdd = scanner.nextInt();

        // Check if the quantity is positive
        if (quantityToAdd <= 0) {
            System.out.println("Quantity to add must be positive.");
            return;
        }

        String updateQuery = "UPDATE items SET quantity = quantity + ? WHERE
item_id = ?";
        try (PreparedStatement stmt = conn.prepareStatement(updateQuery)) {
            stmt.setInt(1, quantityToAdd);
            stmt.setInt(2, itemId);

            int rowsUpdated = stmt.executeUpdate();
            System.out.println(rowsUpdated > 0 ? "Item restocked successfully" :
"Item not found.");
        } catch (SQLException e) {
            System.out.println("Error during restock: " + e.getMessage());
        }
    }

    public static void reduceStock(Connection conn, Scanner scanner) {
        int itemId = scanner.nextInt();

        int quantityToRemove = scanner.nextInt();

        // Check if the quantity is positive
        if (quantityToRemove <= 0) {
            System.out.println("Quantity to remove must be positive.");
            return;
        }

        String checkQuantityQuery = "SELECT quantity FROM items WHERE item_id
= ?";
        String updateQuery = "UPDATE items SET quantity = quantity - ? WHERE
item_id = ?";

        try (PreparedStatement checkStmt =
```

```
conn.prepareStatement(checkQuantityQuery)) {
            checkStmt.setInt(1, itemId);
            ResultSet rs = checkStmt.executeQuery();

            if (rs.next()) {
                int currentQuantity = rs.getInt("quantity");

                if (currentQuantity >= quantityToRemove) {
                    try (PreparedStatement stmt =
conn.prepareStatement(updateQuery)) {
                        stmt.setInt(1, quantityToRemove);
                        stmt.setInt(2, itemId);

                        int rowsUpdated = stmt.executeUpdate();
                        System.out.println(rowsUpdated > 0 ? "Stock reduced
successfully" : "Failed to reduce stock.");
                    }
                } else {
                    System.out.println("Not enough stock to remove.");
                }
            } else {
                System.out.println("Item not found.");
            }
        } catch (SQLException e) {
            System.out.println("Error during stock reduction: " + e.getMessage());
        }
    }

    public static void displayInventory(Connection conn) {
        String displayQuery = "SELECT * FROM items ORDER BY item_id";
        try (Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(displayQuery)) {

            System.out.println("ID | Name | Quantity | Price");
            while (rs.next()) {
                System.out.printf("%d | %s | %d | %.2f%n",
                        rs.getInt("item_id"),
                        rs.getString("name"),
                        rs.getInt("quantity"),
                        rs.getDouble("price"));
            }
        } catch (SQLException e) {
```

```
            System.out.println("Error displaying inventory: " + e.getMessage());
        }
    }

}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

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

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

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 sum = Arrays.stream(arr)
                        .filter(x -> x % 2 == 0)
                        .sum();

        System.out.print(sum);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Alex is learning about Java's functional interfaces and lambda expressions.

He wants to write a simple program that prints the square of each number in an array using a predefined functional interface.

Help Alex complete this task using the Consumer functional interface.

Input Format

- The first line contains an integer N, the number of elements in the array.
- The second line contains N space-separated integers.

Output Format

- Print the squares of all elements in the array, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
1 2 3 4
Output: 1 4 9 16

Answer

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

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();
        }

        Consumer<Integer> square = x -> System.out.print((x * x) + " ");

        for (int num : arr) {
            square.accept(num);
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In the mystical realm of programming, there exists a magical incantation to reveal hidden words.

Elara, the skilled enchantress, wishes to summon a word using her spell and then reverse its characters to uncover its enchanted reflection.

Write a program that uses the predefined functional interface Supplier<String> and a lambda expression to:

Supply (generate) a string, and

Display its reversed form.

Input Format

No input is required from the user.

The string must be supplied internally using a Supplier<String>.

Output Format

Print the reversed version of the supplied string.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Wizard!!
Output: !!draziW

Answer

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

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Supplier<String> supplier = () -> sc.nextLine();
        String str = supplier.get();
        String reversed = new StringBuilder(str).reverse().toString();
        System.out.print(reversed);
    }
}
```

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

Rajalakshmi Engineering College

Name: Divyash H
Email: 240701130@rajalakshmi.edu.in
Roll no: 240701130
Phone: 9345353215
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Abi is working on a text analysis project where she needs to categorize words based on their length.

Words that have three or fewer characters are considered “Short”, while

words with more than three characters are classified as “Long.”

Write a Java program that takes a sentence as input, analyzes each word, and prints a list showing whether each word is “Short” or “Long.”

Use the predefined functional interface Function<String, String> along with a lambda expression for categorization.

Input Format

A single line containing a sentence (words separated by spaces).

Output Format

- A single line with each word categorized as "Short" or "Long", separated by spaces.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: I love my cat
Output: Short Long Short Short

Answer

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

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

        Function<String, String> categorize = word -> word.length() <= 3 ? "Short" :
"Long";

        for (String word : words) {
            System.out.print(categorize.apply(word) + " ");
        }
    }
}
```

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