Rajalakshmi Engineering College

Name: Mohamed Yahya A
Email: 240701325@rajalakshmi.edu.in
Roll no: 240701325
Phone: 9600561844
Branch: REC
Department: CSE - Section 9
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

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

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

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

Note: monthly profit is always greater than monthly expenses.

Input Format

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

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

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

Output Format

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

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

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

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10000.50
5000.75
1000.10
Output: Break-even Point: 2.50
Category: Quick Return

Answer

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

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

```
        double P = sc.nextDouble();
        double E = sc.nextDouble();

        double B = C/(P-E);
        System.out.printf("Break-even Point: %.2f\n",B);
        System.out.print("Category: ");

        if(B<=3)
        {
            System.out.println("Quick Return");
        }
        else if((B>=4) && (B<=12))
        {
            System.out.println("Average Return");
        }
        else
        {
            System.out.println("Long-term Return");
        }
    }
}
```

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

2. 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.*;

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int temp = n;
        int sum=0,digit=0;
        while(temp>0)
        {
            digit += temp%10;
            temp = temp/10;
            sum++;
        }
        if(sum==digit)
        {
            System.out.printf("The number of digits in %d matches the sum of its
digits.\n",n);
        }
        else
        {
            System.out.printf("The number of digits in %d does not match the sum of
its digits.\n",n);
        }
    }
}
```

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

3. Problem Statement

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

Write a program to compute this sum.

Example

Input:

10 12

Output:

3

Explanation:

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

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

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

Total sum = 1 + 1 + 1 = 3

Input Format

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

Output Format

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

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10 12

Output: 3

Answer

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

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

4. Problem Statement

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

height and weight.

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

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

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

Input Format

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

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

Output Format

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

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

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1.2
45.2

Output: BMI: 31.39
Classification: Obese

Answer

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

```
class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double a = sc.nextDouble();
        double b = sc.nextDouble();
        double c = b/(a*a);
        System.out.printf("BMI: %.2f\n",c);
        if(c<=18.5)
        {
            System.out.printf("Classification: Underweight");
        }
        else if((c>=18.6) && (c<=24.9))
        {
            System.out.printf("Classification: Normal Weight");
        }
        else if((c>=25.0) && (c<=29.9))
        {
            System.out.printf("Classification: Overweight");
        }
        else{
            System.out.printf("Classification: Obese");
        }
    }
}
```

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