

# Rajalakshmi Engineering College

Name: KAVYA SRIRAM

Email: 241901045@rajalakshmi.edu.in

Roll no: 241901045

Phone: 8939657782

Branch: REC

Department: CSE (CS) - Section 1

Batch: 2028

Degree: B.E - CSE (CS)

Scan to verify results



## 2024\_28\_III\_OOPS Using Java Lab

### 2028\_REC\_OOPS using Java\_Week 2\_CY

Attempt : 1

Total Mark : 40

Marks Obtained : 40

### Section 1 : Coding

#### 1. Problem Statement

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**

```
import java.util.Scanner;

class Samantha{
    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.print(N + " is a multiple of 5");
        } else if (N % 7 == 0) {
            System.out.print(N + " is a multiple of 7");
        } else {
            System.out.print(N + " is neither multiple of 5 nor 7");
        }
    }
}
```

**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***

```
import java.util.Scanner;

class Ted{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        int num = N;
        int sum = 0;
        int digits = 0;

        do {
            int digit = num % 10;
            sum += digit;
            digits++;
            num /= 10;
        } while (num > 0);
```

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

**Status :** Correct

**Marks :** 10/10

### 3. 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 =  $\text{weight}/(\text{height} \times \text{height})$

#### **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;

class john{
    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);

        if (bmi < 18.5) {
            System.out.print("Classification: Underweight");
        } else if (bmi >= 18.6 && bmi <= 24.9) {
            System.out.print("Classification: Normal Weight");
        } else if (bmi >= 25.0 && bmi <= 29.9) {
            System.out.print("Classification: Overweight");
        } else {
            System.out.print("Classification: Obese");
        }
    }
}
```

**Status :** Correct

**Marks :** 10/10

## **4. 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:  $\text{initial cost} / (\text{monthly profit} - \text{monthly expenses})$ . 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  $\leq 3$
- "Average Return" if break-even point  $\leq 12$
- "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**

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

```
class ram{
```

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

```
        double initialCost = sc.nextDouble();  
        double monthlyProfit = sc.nextDouble();  
        double monthlyExpenses = sc.nextDouble();
```

```
        double breakEven = initialCost / (monthlyProfit - monthlyExpenses);
```

```
        System.out.printf("Break-even Point: %.2f \n", breakEven);
```

```
        if (breakEven <= 3) {  
            System.out.print("Category: Quick Return");  
        } else if (breakEven <= 12) {  
            System.out.print("Category: Average Return");  
        } else {  
            System.out.print("Category: Long-term Return");  
        }  
    }  
}
```

**Status :** Correct

**Marks :** 10/10