

Rajalakshmi Engineering College

Name: Adithya B
Email: 240701018@rajalakshmi.edu.in
Roll no: 240701018
Phone: 9444117405
Branch: REC
Department: CSE - Section 10
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

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;

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

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;

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

        int digitCount = 0;
        int digitSum = 0;
        int tempNumber = Math.abs(number);

        do {
            int digit = tempNumber % 10;
            digitCount++;
            digitSum += digit;
            tempNumber /= 10;
        } while (tempNumber != 0);
```

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

Status : Correct

Marks : 10/10

3. Problem Statement

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

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

Note: discriminant = $b^2 - 4ac$

Input Format

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

Output Format

If there are two real roots, print:

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

If there is one real root, print:

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

If there are no real roots, print:

- "There are no real solutions."

Note: values are rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1 6 9

Output: One real solution:

Root = -3.00

Answer

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

```
class QuadraticEquationSolver {  
    public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        double a = scanner.nextDouble();  
        double b = scanner.nextDouble();  
        double c = scanner.nextDouble();  
        scanner.close();
```

```
        double discriminant = (b * b) - (4 * a * c);
```

```
        if (discriminant > 0) {  
            double sqrtD = Math.sqrt(discriminant);
```

```
            double root1 = (-b + sqrtD) / (2 * a);  
            double root2 = (-b - sqrtD) / (2 * a);
```

```
            System.out.printf("Two real solutions:%nRoot1 = %.2f%nRoot2 = %.2f%n",  
root1, root2);
```

```
        }  
        else if (discriminant == 0) {  
            double root = -b / (2 * a);
```

```

        System.out.printf("One real solution:%nRoot = %.2f%n", root);
    }
    else {
        System.out.println("There are no real solutions.");
    }
}
}
}

```

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 ≤ 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

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

```
class BreakEvenPoint {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        double initialCost = scanner.nextDouble();
        double monthlyProfit = scanner.nextDouble();
        double monthlyExpenses = scanner.nextDouble();

        double breakEvenPoint = initialCost / (monthlyProfit - monthlyExpenses);

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

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

Status : Correct

Marks : 10/10