

Rajalakshmi Engineering College

Name: Haresh R
Email: 241901031@rajalakshmi.edu.in
Roll no:
Phone: null
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

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

```
import java.util.Scanner;

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

```

int rows = sc.nextInt();

sc.close();

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

2. Problem Statement

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

Write a program to compute this sum.

Example

Input:

10 12

Output:

3

Explanation:

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

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

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

Total sum = $1 + 1 + 1 = 3$

Input Format

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

Output Format

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

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10 12

Output: 3

Answer

```
import java.util.Scanner;

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

        int A = sc.nextInt();
        int B = sc.nextInt();

        sc.close();

        long totalSum = 0;
```

```

for (int i = A; i <= B; i++) {
    int currentNumber = i;
    long oddDigitsProduct = 1;
    boolean foundOddDigit = false;

    while (currentNumber > 0) {
        int digit = currentNumber % 10;

        if (digit % 2 != 0) {
            oddDigitsProduct *= digit;
            foundOddDigit = true;
        }

        currentNumber /= 10;
    }

    if (foundOddDigit) {
        totalSum += oddDigitsProduct;
    }
}

System.out.println(totalSum);
}
}

```

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;
```

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

```

sc.close();

double discriminant = b * b - 4 * a * c;

if (discriminant > 0) {
    double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
    double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);

    System.out.println("Two real solutions:");
    System.out.printf("Root1 = %.2f\n", root1);
    System.out.printf("Root2 = %.2f\n", root2);
} else if (discriminant == 0) {
    double root = -b / (2 * a);

    System.out.println("One real solution:");
    System.out.printf("Root = %.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;
```

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



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

sc.close();

double monthlyNetProfit = monthlyProfit - monthlyExpenses;

double breakEvenPoint = initialCost / monthlyNetProfit;

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

String category;
if (breakEvenPoint <= 3) {
    category = "Quick Return";
} else if (breakEvenPoint <= 12) {
    category = "Average Return";
} else {
    category = "Long-term Return";
}

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

Status : Correct

Marks : 10/10