1-

```
Public class Calculator {
   Public static void main(String[] args) {
      // Check if the correct number of command-line arguments are provided
      If (args.length != 2) {
         System.out.println(“Usage: java Calculator <num1> <num2>”);
         System.exit(1);
      }

      // Parse the command-line arguments as numbers
      Try {
         Double num1 = Double.parseDouble(args[0]);
         Double num2 = Double.parseDouble(args[1]);

         // Calculate the desired results
         Double sumResult = num1 + num2;
         Double differenceResult = num1 – num2;
         Double productResult = num1 * num2;

         // Check for division by zero
         If (num2 != 0) {
            Double quotientResult = num1 / num2;
            Double remainderResult = num1 % num2;

            // Display the results
            System.out.println(“Sum: “ + sumResult);
            System.out.println(“Difference: “ + differenceResult);
            System.out.println(“Product: “ + productResult);
            System.out.println(“Quotient: “ + quotientResult);
```

```
            System.out.println(“Remainder: “ + remainderResult);
        } else {
            System.out.println(“Quotient and remainder are undefined (division by zero).”);
        }
    } catch (NumberFormatException e) {
        System.out.println(“Please provide valid numeric arguments.”);
    }
  }
}
```

2-

```
import java.util.Scanner;

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

    // Input the lengths of the sides
    System.out.print("Enter the length of side 1: ");
    double side1 = scanner.nextDouble();
    System.out.print("Enter the length of side 2: ");
    double side2 = scanner.nextDouble();
    System.out.print("Enter the length of side 3: ");
    double side3 = scanner.nextDouble();

    // Check if it forms a valid triangle
    if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1) {
      // Determine the type of triangle
      if (side1 == side2 && side2 == side3) {
```

```
            System.out.println("It's an equilateral triangle.");
        } else if (side1 == side2 || side1 == side3 || side2 == side3) {
            System.out.println("It's an isosceles triangle.");
        } else {
            System.out.println("It's a scalene triangle.");
        }

        // Calculate and display the area
        double s = (side1 + side2 + side3) / 2;
        double area = Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
        System.out.println("Area of the triangle: " + area);
    } else {
        System.out.println("Invalid triangle. The sum of the lengths of any two sides must be greater than the length of the third side.");
    }

    scanner.close();
  }
}
```

3-

```
import java.util.Scanner;

public class HCFAndLCMCalculator {
  public static void main(String[] args) {
    // Create a Scanner object for input
    Scanner scanner = new Scanner(System.in);
```

```
        // Prompt the user to enter two numbers
        System.out.print("Enter the first number: ");
        int num1 = scanner.nextInt();

        System.out.print("Enter the second number: ");
        int num2 = scanner.nextInt();

        // Calculate the HCF and LCM
        int hcf = findHCF(num1, num2);
        int lcm = findLCM(num1, num2);

        // Display the results
        System.out.println("The Highest Common Factor (HCF) of " + num1 + " and " + num2 + " is: " + hcf);
        System.out.println("The Lowest Common Multiple (LCM) of " + num1 + " and " + num2 + " is: " +
lcm);

        // Close the Scanner
        scanner.close();
    }

    // Function to find the Highest Common Factor (HCF) using Euclidean Algorithm
    private static int findHCF(int a, int b) {
        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
```

```
    }

    // Function to find the Lowest Common Multiple (LCM)
    private static int findLCM(int a, int b) {
        // LCM = (num1 * num2) / HCF(num1, num2)
        int hcf = findHCF(a, b);
        return (a * b) / hcf;
    }
}
```

4-

```
import java.util.Scanner;

class NumberOperations {
    private int number;

    public NumberOperations(int num) {
        this.number = num;
    }

    public int findSumOfDigits() {
        int sum = 0;
        int temp = number;

        while (temp != 0) {
            int digit = temp % 10;
            sum += digit;
            temp /= 10;
```

```
        }

        return sum;
    }

    public int reverseNumber() {
        int reversed = 0;
        int temp = number;

        while (temp != 0) {
            int digit = temp % 10;
            reversed = reversed * 10 + digit;
            temp /= 10;
        }

        return reversed;
    }
}

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

        // Input a number from the user
        System.out.print("Enter a number: ");
        int num = scanner.nextInt();

        // Create an object of NumberOperations class
        NumberOperations numOperations = new NumberOperations(num);
```

```
        // Find and display the sum of digits
        int sumOfDigits = numOperations.findSumOfDigits();
        System.out.println("Sum of digits: " + sumOfDigits);

        // Find and display the reversed number
        int reversedNumber = numOperations.reverseNumber();
        System.out.println("Reversed number: " + reversedNumber);

        scanner.close();
    }
}
```

5-

```
Import java.util.Scanner;

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

        // Input the size of the array
        System.out.print(“Enter the size of the array (10 or more): “);
        Int size = scanner.nextInt();

        If (size < 10) {
            System.out.println(“Please enter a size of 10 or more.”);
```

```
    Return;
}

// Create an array of the given size
Int[] numbers = new int[size];

// Input the elements of the array
System.out.println(“Enter the elements of the array:”);

For (int I = 0; I < size; i++) {
    System.out.print(“Element “ + (I + 1) + “: “);
    Numbers[i] = scanner.nextInt();
}

// Find the smallest element
Int smallest = numbers[0];
For (int I = 1; I < size; i++) {
    If (numbers[i] < smallest) {
        Smallest = numbers[i];
    }
}

// Find the largest element
Int largest = numbers[0];
For (int I = 1; I < size; i++) {
    If (numbers[i] > largest) {
        Largest = numbers[i];
    }
}
```

```
        // Find the second largest element
        Int secondLargest = Integer.MIN_VALUE;
        For (int I = 0; I < size; i++) {
            If (numbers[i] > secondLargest && numbers[i] != largest) {
                secondLargest = numbers[i];
            }
        }

        // Display the results
        System.out.println(“Smallest element: “ + smallest);
        System.out.println(“Largest element: “ + largest);
        System.out.println(“Second largest element: “ + secondLargest);

        Scanner.close();
    }
}
```

6 –

```
import java.util.Arrays;

public class MergeArrays {
    public static void main(String[] args) {
        // Define two arrays to merge
        int[] array1 = {1, 2, 3};
        int[] array2 = {4, 5, 6};

        // Calculate the length of the merged array
```

```
        int mergedLength = array1.length + array2.length;

        // Create a new array to store the merged elements
        int[] mergedArray = new int[mergedLength];

        // Copy elements from the first array to the merged array
        System.arraycopy(array1, 0, mergedArray, 0, array1.length);

        // Copy elements from the second array to the merged array
        System.arraycopy(array2, 0, mergedArray, array1.length, array2.length);

        // Display the merged array
        System.out.println("Merged Array: " + Arrays.toString(mergedArray));
    }
}
```

7-

```
import java.util.Scanner;

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

        // Input the dimensions of the matrix
        System.out.print("Enter the number of rows: ");
        int rows = scanner.nextInt();
```

```
    System.out.print("Enter the number of columns: ");
    int columns = scanner.nextInt();

    // Create the matrix
    int[][] matrix = new int[rows][columns];

    // Input the elements of the matrix
    System.out.println("Enter the elements of the matrix:");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
            matrix[i][j] = scanner.nextInt();
        }
    }

    // Calculate and display the trace of the matrix
    int trace = calculateTrace(matrix);
    System.out.println("Trace of the matrix: " + trace);

    // Calculate and display the transpose of the matrix
    int[][] transpose = calculateTranspose(matrix);
    System.out.println("Transpose of the matrix:");
    displayMatrix(transpose);

    scanner.close();
}

// Function to calculate the trace of a matrix
private static int calculateTrace(int[][] matrix) {
    int trace = 0;
```

```
    for (int i = 0; i < matrix.length; i++) {
        trace += matrix[i][i];
    }
    return trace;
}

// Function to calculate the transpose of a matrix
private static int[][] calculateTranspose(int[][] matrix) {
    int rows = matrix.length;
    int columns = matrix[0].length;
    int[][] transpose = new int[columns][rows];

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }

    return transpose;
}

// Function to display a matrix
private static void displayMatrix(int[][] matrix) {
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix[i].length; j++) {
            System.out.print(matrix[i][j] + " ");
        }
        System.out.println();
    }
```

```
    }
}
```

8-

```
import java.util.Scanner;

class Student {
    private String name;
    private int rollNumber;

    public void readStudentDetails() {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter Student Name: ");
        name = scanner.nextLine();
        System.out.print("Enter Roll Number: ");
        rollNumber = scanner.nextInt();
    }

    public void displayStudentDetails() {
        System.out.println("Student Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
    }
}

class Mark extends Student {
    private int[] marks = new int[5];
```

```
private int total;
private double average;

public void readMarks() {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Enter Marks for 5 Subjects:");
    for (int i = 0; i < 5; i++) {
        System.out.print("Subject " + (i + 1) + ": ");
        marks[i] = scanner.nextInt();
    }
}

public void calculateTotalAndAverage() {
    total = 0;
    for (int mark : marks) {
        total += mark;
    }
    average = (double) total / 5;
}

public void displayResult() {
    displayStudentDetails();
    System.out.println("Marks:");
    for (int i = 0; i < 5; i++) {
        System.out.println("Subject " + (i + 1) + ": " + marks[i]);
    }
    System.out.println("Total Marks: " + total);
    System.out.println("Average Marks: " + average);
```

```
    }
}

public class StudentResult {
    public static void main(String[] args) {
        Mark studentMark = new Mark();

        // Read student details
        studentMark.readStudentDetails();

        // Read marks for 5 subjects
        studentMark.readMarks();

        // Calculate total and average
        studentMark.calculateTotalAndAverage();

        // Display student's result
        studentMark.displayResult();
    }
}
```