# ASSIGNMENT-3

Array coding question :

1. Find the Largest and Smallest Element

○ Given an array, find the smallest and largest elements in it.

```
---- class MinMaxNo{
public static void main( String Args[]){
int arr[]={54,23,56,5,80,30,15};
 int Max= arr[0];
 for(int i=1;
i < arr.length ; i++) if(Max < arr[i]){ Max=arr[i];
}
System.out.println("Maximum number of an array is = "+Max);
int Min =arr[0];
 for(int i=1; i < arr.length ; i++) if(Min> arr[i]){ Min=arr[i];
 }
System.out.println("Minimum number of an array is =" +Min);
 }


}
```

2. Reverse an Array

○ Reverse the given array in place.

```
---- public class ReverseArray {
public static void reverseArray(int[] arr) {
 int left = 0, right = arr.length - 1;
 while (left < right) {

 // Swap elements int temp = arr[left];

 arr[left] = arr[right]; arr[right] = temp;

 // Move pointers left++; right--;

}

 }

 public static void main(String[] args) {

 int[] arr = {1, 2, 3, 4, 5};

 System.out.println("Original Array: " + Arrays.toString(arr)); reverseArray(arr);
System.out.println("Reversed Array: " + Arrays.toString(arr));

 }

}
```

## 3. Find the Second Largest Element

○ Find the second-largest element in the given array.

```
----  public class SecondLargest {

public static int findSecondLargest(int[] arr) {

 int n = arr.length;

int largest = 0;

 for (int i = 1; i < n; i++) {

if (arr[i] > arr[largest]) largest = i;

 } int secondLargest = -1;

for (int i = 0; i < n; i++)

{

 if (arr[i] != arr[largest])

 {

if (secondLargest == -1 || arr[i] > arr[secondLargest]) secondLargest = i;

}
```

```
 }
if (secondLargest == -1) return -1;
// No second largest element return arr[secondLargest];
 }
public static void main(String[] args) {
 int[] arr = {10, 5, 8, 20, 15};
int secondLargest = findSecondLargest(arr);
 if (secondLargest == -1) System.out.println("No second largest element");
 else
System.out.println("Second largest element: " + secondLargest);
 }
 }
```

## 4. Count Even and Odd Numbers

○ Count the number of even and odd numbers in an array.

```
---- // Class to find sum of even and odd numbers class Numbers {
public static void AddEven(int arr[]) {
 int SumEven = 0;
 // Variable to store sum of even numbers
 int SumOdd = 0;
 // Variable to store sum of odd numbers for
 (int i = 0; i < arr.length; i++){
 // Corrected loop syntax
if (arr[i] % 2 == 0) {
// Check if even
 SumEven += arr[i];
 }
Else
```

```
 {
// If odd SumOdd += arr[i];
 }
 }
System.out.println("Sum of Even Numbers is: " + SumEven);
 System.out.println("Sum of Odd Numbers is: " + SumOdd);
}
}
// Main class to test the function public class EvenOddAddition {
public static void main(String args[]) {
 int arr[] = {5, 8, 7, 10, 9, 52};
// Array input
Numbers.AddEven(arr);
 // Corrected function call
 }
 }
```

## 5. Find Sum and Average

○ Compute the sum and average of all elements in the array.

```
----   class Numbers {
public static void AddAverage(int arr[]) {
 int Sum = 0;
 int Average = 0;
for (int i = 0; i < arr.length; i++)
 {
Sum += arr[i];
 Average = Sum / arr.length ;
 }
System.out.println("Sum of all Numbers is: " + Sum);
System.out.println("Average of all Numbers is : " + Average);
}
}
 // Main class to test the function public class SumAverage {
public static void main(String args[]) {
 int arr[] = {5, 8, 7, 10, 9, 52};
 Numbers.AddAverage(arr);
 }
}
```

## 6. Remove Duplicates from a Sorted Array

○ Remove duplicate elements from a sorted array without using extra space.

```
---- import java.util.Arrays;
 class Duplicate {
 public static void Demo (int arr[]){
Arrays.sort(arr);
System.out.println("Updated Array without duplicates =");
int a =0; for(int i = 1; i < arr.length ; i++){
if(arr[i] != arr[i-1]){
 a = arr[i];
 System.out.print(a + ", ");
}
a++;
 }
}
}
// Main class to test the function
```

```
 public class DuplicateArray1 {
public static void main(String args[]) {
 int arr[] = {5, 8, 7,6,7,5,5 ,10, 9, 52};
 Duplicate.Demo(arr);
}
 }
```

## 7. Rotate an Array

○ Rotate the array to the right by k positions.

```
---  import java.util.Arrays;
class Rotate{
 public static void done(int arr[], int k){
int a = arr.length; k = k % a;
 int temp[] = new int[a];
for(int i = 0; i < arr.length ; i++){
 temp[(i = k) % a] = arr[i];
}
 for(int i = 0; i < arr.length; i++){
 arr[i] = temp[i];
 }
System.out.println("Updated Array: " +Arrays.toString(arr));
}
 }
public class RotateDemo{ public static void main(String args[]){
 int arr[] = {1,2,3,4,5,6,7,8};
 int k = 2;
 Rotate.done(arr,2);
}
```

}

8. Merge Two Sorted Arrays

○ Merge two sorted arrays into a single sorted array without using extra space.

```
---- import java.util.Arrays;
class MergeDemo {
public static void main(String args[]) {
int arr1[] = {4, 5, 8, 9, 10, 20};
int arr2[] = {1, 3, 7, 15, 12, 6};
// Step 1: Create merged array of size arr1.length + arr2.length
int mergedLength = arr1.length + arr2.length;
int Sorted[] = new int[mergedLength];
// Step 2: Copy arr1 and arr2 into Sorted array
System.arraycopy(arr1, 0, Sorted, 0, arr1.length);
System.arraycopy(arr2, 0, Sorted, arr1.length, arr2.length);
// Step 3: Sort the merged array
Arrays.sort(Sorted);
// Step 4: Print the sorted merged array
System.out.println("Sorted merged array: " + Arrays.toString(Sorted));
}
}
```

9. Find Missing Number in an Array

○ Given an array of size n-1 containing numbers from 1 to n, find the missing number.

```
----- class Missing{
 public static void Array(int arr[]){
int n = arr.length;
 for(int i = 0; i < n ; i++)
{
if(arr[i] != (i + 1)){
 System.out.println("Missing number is "+(i+1));
 break;
}
}
}
 }
 public class MissingNumber{
 public static void main(String args[]){
 int arr[] = {1,2,3,5,6,7,8};
 Missing.Array(arr);
}
 }
```

## 10. Find Intersection and Union of Two Arrays

○ Find the intersection and union of two unsorted arrays.

```
----- import java.util.Arrays;
 class UnionIntersection {
 // Function to find the Union of two sorted arrays
 public static void findUnion(int arr1[], int arr2[]) {
int i = 0, j = 0;
 System.out.print("Union: ");
 while (i < arr1.length && j < arr2.length) {
 if (arr1[i] < arr2[j])
 {
 System.out.print(arr1[i++] + " ");
}
 else if (arr1[i] > arr2[j]) {
 System.out.print(arr2[j++] + " ");
 }
 else
{
// Both elements are equal, take only one and move both pointers
 System.out.print(arr1[i] + " ");
 i++;
 j++;
 }
 }
// Print remaining elements of arr1
while (i < arr1.length) {
 System.out.print(arr1[i++] + " ");
}
// Print remaining elements of arr2
 while (j < arr2.length) {
 System.out.print(arr2[j++] + " ");
 }
 System.out.println();
 }
// Function to find the Intersection of two sorted arrays
 public static void findIntersection(int arr1[], int arr2[]) {
 int i = 0, j = 0;
 System.out.print("Intersection: ");
 while (i < arr1.length && j < arr2.length) {
```

```
 if (arr1[i] < arr2[j]) {
i++;
 }
 else if (
arr1[i] > arr2[j]) { j++;
 }
 else {
// Both are equal, add to intersection
 System.out.print(arr1[i] + " ");
 i++;
 j++;
}
}
 System.out.println();
 }
 public static void main(String args[]) {
 int arr1[] = {1, 2, 3, 5, 6, 7, 8};
 int arr2[] = {1, 2, 3, 4, 7, 8, 11};
 // Sort arrays first (if they are not already sorted)
 Arrays.sort(arr1); Arrays.sort(arr2);
 // Find Union and Intersection
 findUnion(arr1, arr2);
findIntersection(arr1, arr2);
}
 }
```

## 11. Find a Subarray with Given Sum

○ Given an array of integers, find the subarray that sums to a given value S.

```
---- import java.util.*;

public class SubarrayWithGivenSum {
    static int[] findSubarray(int[] arr, int S) {
        int start = 0, currentSum = 0;

        for (int end = 0; end < arr.length; end++) {
            currentSum += arr[end]; // Expand the window

            while (currentSum > S && start <= end) {
                currentSum -= arr[start]; // Shrink the window
                start++;
            }

            if (currentSum == S) {
                return Arrays.copyOfRange(arr, start, end + 1);
            }
        }
        return new int[]{}; // Return an empty array if no subarray is found
    }

    public static void main(String[] args) {
        int[] arr = {1, 4, 20, 3, 10, 5};
        int S = 33;
        int[] result = findSubarray(arr, S);

        if (result.length > 0) {
            System.out.println("Subarray with sum " + S + ": " + Arrays.toString(result));
        } else {
            System.out.println("No subarray found.");
```

12. Write a program to accept 20 integer numbers in a single Dimensional Array. Find and Display the following:

○ Number of even numbers.

○ Number of odd numbers.

○ Number of multiples of 3

---  import java.util.Scanner;

```
public class ArrayAnalysis {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] numbers = new int[20]; // Array to store 20 integers
        int evenCount = 0, oddCount = 0, multipleOfThreeCount = 0;

        // Accept 20 numbers from the user
        System.out.println("Enter 20 integers:");
        for (int i = 0; i < 20; i++) {
            numbers[i] = scanner.nextInt();
            // Check if the number is even or odd
            if (numbers[i] % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }

            // Check if the number is a multiple of 3
            if (numbers[i] % 3 == 0) {
                multipleOfThreeCount++;
// Display the results
        System.out.println("Number of even numbers: " + evenCount);
        System.out.println("Number of odd numbers: " + oddCount);
        System.out.println("Number of multiples of 3: " + multipleOfThreeCount);

        scanner.close();
    }
}
```

13. Write a program to accept the marks in Physics, Chemistry and Maths secured by 20 class students in a single Dimensional Array. Find and display the following:

○ Number of students securing 75% and above in aggregate.

○ Number of students securing 40% and below in aggregate.

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

```
public class StudentMarksAnalysis {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] marks = new int[60]; // Array to store marks (20 students * 3 subjects)
        int above75 = 0, below40 = 0;
        int totalStudents = 20;

        // Accept marks for 20 students (3 subjects each)
        System.out.println("Enter marks for 20 students (Physics, Chemistry, Maths
for each student):");
        for (int i = 0; i < totalStudents; i++) {
            int physics = scanner.nextInt();
            int chemistry = scanner.nextInt();
            int maths = scanner.nextInt();

            // Calculate total and percentage
            int totalMarks = physics + chemistry + maths;
            double percentage = (totalMarks / 300.0) * 100;

            // Check conditions
            if (percentage >= 75) {
                above75++;
            }
            if (percentage <= 40) {
```

```
            below40++;
          }
        }

        // Display results
        System.out.println("Number of students securing 75% and above: " +
above75);
        System.out.println("Number of students securing 40% and below: " +
below40);

        scanner.close();
    }
}
```

14. Write a program in Java to accept 20 numbers in a single dimensional array arr[20]. Transfer and store all the even numbers in an array even[ ] and all the odd numbers in another array odd[ ]. Finally, print the elements of the even & the odd array.

```
---- import java.util.*;

public class EvenOddArray {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] arr = new int[20]; // Main array for 20 numbers
        List<Integer> evenList = new ArrayList<>(); // Dynamic list for even numbers
        List<Integer> oddList = new ArrayList<>(); // Dynamic list for odd numbers
```

```
// Accept 20 numbers from the user
System.out.println("Enter 20 integers:");
for (int i = 0; i < 20; i++) {
    arr[i] = scanner.nextInt();

    // Check even or odd
    if (arr[i] % 2 == 0) {
        evenList.add(arr[i]); // Store in even list
    } else {
        oddList.add(arr[i]); // Store in odd list
    }
}

// Convert lists to arrays
int[] even = evenList.stream().mapToInt(i -> i).toArray();
int[] odd = oddList.stream().mapToInt(i -> i).toArray();

// Print even numbers
System.out.println("Even numbers: " + Arrays.toString(even));

// Print odd numbers
System.out.println("Odd numbers: " + Arrays.toString(odd));
```

```
        scanner.close();
    }
}
```

15. Write a Java program to print all sub-arrays with 0 sum present in a given array of integers.

Example:

Input : nums1 = { 1, 3, -7, 3, 2, 3, 1, -3, -2, -2 }

nums2 = { 1, 2, -3, 4, 5, 6 }

nums3= { 1, 2, -2, 3, 4, 5, 6 }

Output: Sub-arrays with 0 sum : [1, 3, -7, 3]

Sub-arrays with 0 sum : [3, -7, 3, 2, 3, 1, -3, -2]

Sub-arrays with 0 sum : [1, 2, -3]

Sub-arrays with 0 sum : [2, -2]

```
--- Test Case 1: [1, 3, -7, 3, 2, 3, 1, -3, -2, -2]
Sub-arrays with 0 sum:
[1, 3, -7, 3]
[3, -7, 3, 2, 3, 1, -3, -2]

Test Case 2: [1, 2, -3, 4, 5, 6]
Sub-arrays with 0 sum:
[1, 2, -3]

Test Case 3: [1, 2, -2, 3, 4, 5, 6]
Sub-arrays with 0 sum:
[2, -2]
```

16. Given two sorted arrays A and B of size p and q, write a Java program to merge elements of A with B by maintaining the sorted order i.e. fill A with first p smallest elements and fill B with remaining elements.

Example:

Input : int[] A = { 1, 5, 6, 7, 8, 10 }

int[] B = { 2, 4, 9 }

Output:

Sorted Arrays: A: [1, 2, 4, 5, 6, 7] B: [8, 9, 10]

```
--- import java.util.Arrays;


public class MergeSortedArrays {
   static void mergeAndSort(int[] A, int[] B) {
       int p = A.length, q = B.length;
       int[] merged = new int[p + q];


       // Copy elements of A and B into merged array
       System.arraycopy(A, 0, merged, 0, p);
       System.arraycopy(B, 0, merged, p, q);


       // Sort the merged array
       Arrays.sort(merged);


       // Refill A with the first p smallest elements
       System.arraycopy(merged, 0, A, 0, p);
```

```
        // Refill B with the remaining q elements
        System.arraycopy(merged, p, B, 0, q);

        // Print the updated arrays
        System.out.println("Sorted Arrays:");
        System.out.println("A: " + Arrays.toString(A));
        System.out.println("B: " + Arrays.toString(B));
    }

    public static void main(String[] args) {
        int[] A = {1, 5, 6, 7, 8, 10};
        int[] B = {2, 4, 9};

        mergeAndSort(A, B);
    }
}
```

17. Write a Java program to find the maximum product of two integers in a given array of integers.

Example:

Input :

nums = { 2, 3, 5, 7, -7, 5, 8, -5 }

Output: Pair is (7, 8), Maximum Product: 56

```
---- import java.util.Arrays;

public class MaxProductPair {
    static void findMaxProductPair(int[] nums) {
        if (nums.length < 2) {
            System.out.println("Array must have at least two elements.");
            return;
        }

        // Sort the array
        Arrays.sort(nums);

        int n = nums.length;

        // Maximum product can be from:
        // 1. The two largest positive numbers (nums[n-1] * nums[n-2])
        // 2. The two smallest negative numbers (nums[0] * nums[1])
        int product1 = nums[n - 1] * nums[n - 2];
        int product2 = nums[0] * nums[1];

        if (product1 > product2) {
            System.out.println("Pair is (" + nums[n - 2] + ", " + nums[n - 1] + "),
Maximum Product: " + product1);
        } else {
            System.out.println("Pair is (" + nums[0] + ", " + nums[1] + "), Maximum
Product: " + product2);
        }
    }
    public static void main(String[] args) {
        int[] nums = {2, 3, 5, 7, -7, 5, 8, -5};
        findMaxProductPair(nums);
    }
}
```

18. Print a Matrix

○ Given an m x n matrix, print all its elements row-wise.

--- import java.util.Scanner;

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

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

        int[][] matrix = new int[m][n];

        // Input matrix elements
        System.out.println("Enter the matrix elements row-wise:");
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Print matrix row-wise
        System.out.println("Matrix elements (row-wise):");
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println(); // Move to the next row
        }

        scanner.close();
    }
}
```

19. Transpose of a Matrix

○ Given a matrix, return its transpose (swap rows and columns).

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

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

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

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

int[][] matrix = new int[m][n];
int[][] transpose = new int[n][m]; // Transposed matrix

// Input matrix elements
System.out.println("Enter the matrix elements row-wise:");
for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++) {
        matrix[i][j] = scanner.nextInt();
    }
}

// Compute the transpose
for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++) {
        transpose[j][i] = matrix[i][j]; // Swap rows and columns
    }
```

```
        }

        // Print the transposed matrix
        System.out.println("Transposed Matrix:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                System.out.print(transpose[i][j] + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}
```

20. Sum of Two Matrices

○ Given two matrices of the same size, compute their sum.

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

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

        // Input matrix dimensions
        System.out.print("Enter the number of rows (m): ");
```

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

int[][] matrix1 = new int[m][n];
int[][] matrix2 = new int[m][n];
int[][] sumMatrix = new int[m][n];

// Input first matrix
System.out.println("Enter elements of first matrix:");
for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++) {
        matrix1[i][j] = scanner.nextInt();
    }
}

// Input second matrix
System.out.println("Enter elements of second matrix:");
for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++) {
        matrix2[i][j] = scanner.nextInt();
    }
}
```

```
        // Compute the sum of matrices
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                sumMatrix[i][j] = matrix1[i][j] + matrix2[i][j];
            }
        }

        // Print the sum matrix
        System.out.println("Sum of the two matrices:");
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(sumMatrix[i][j] + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}
```

21. Row-wise and Column-wise Sum

○ Find the sum of each row and each column of a given matrix.

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

public class MatrixRowColumnSum {
```

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

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

    int[][] matrix = new int[m][n];

    // Input matrix elements
    System.out.println("Enter the matrix elements row-wise:");
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            matrix[i][j] = scanner.nextInt();
        }
    }

    // Compute and print row-wise sum
    System.out.println("Row-wise Sum:");
    for (int i = 0; i < m; i++) {
        int rowSum = 0;
        for (int j = 0; j < n; j++) {
                rowSum += matrix[i][j];
            }
            System.out.println("Sum of row " + (i + 1) + ": " + rowSum);
        }

        // Compute and print column-wise sum
        System.out.println("Column-wise Sum:");
        for (int j = 0; j < n; j++) {
            int colSum = 0;
            for (int i = 0; i < m; i++) {
                colSum += matrix[i][j];
            }
            System.out.println("Sum of column " + (j + 1) + ": " + colSum);
        }

        scanner.close();
    }
}
```

22. Find the Maximum Element in a Matrix

○ Find the largest element in a given matrix.

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

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

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

        int[][] matrix = new int[m][n];

        // Input matrix elements
        System.out.println("Enter the matrix elements row-wise:");
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Find the maximum element
        int maxElement = matrix[0][0]; // Assume first element is the largest initially
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                if (matrix[i][j] > maxElement) {
                    maxElement = matrix[i][j]; // Update maxElement if a larger value is
found
```

```
                }
            }
        }

        // Print the maximum element
        System.out.println("Maximum element in the matrix: " + maxElement);

        scanner.close();
    }
}
```

23. Matrix Multiplication

○ Multiply two matrices and return the resultant matrix.

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

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

        // Input dimensions of Matrix A
        System.out.print("Enter the number of rows of Matrix A: ");
        int m = scanner.nextInt();
        System.out.print("Enter the number of columns of Matrix A (or rows of
Matrix B): ");
        int n = scanner.nextInt();
```

```
// Input dimensions of Matrix B
System.out.print("Enter the number of columns of Matrix B: ");
int p = scanner.nextInt();

int[][] A = new int[m][n];
int[][] B = new int[n][p];
int[][] C = new int[m][p]; // Resultant matrix

// Input elements for Matrix A
System.out.println("Enter elements of Matrix A:");
for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++) {
        A[i][j] = scanner.nextInt();
    }
}

// Input elements for Matrix B
System.out.println("Enter elements of Matrix B:");
for (int i = 0; i < n; i++) {
    for (int j = 0; j < p; j++) {
        B[i][j] = scanner.nextInt();
    }
}
```

```
        // Matrix Multiplication Logic
        for (int i = 0; i < m; i++) { // Row of A
            for (int j = 0; j < p; j++) { // Column of B
                for (int k = 0; k < n; k++) { // Row of B
                    C[i][j] += A[i][k] * B[k][j];
                }
            }
        }

        // Print the resultant matrix
        System.out.println("Resultant Matrix after Multiplication:");
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < p; j++) {
                System.out.print(C[i][j] + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}
```

24. Rotate a Matrix by 90 Degrees

○ Rotate a given N x N matrix by 90 degrees clockwise.

---- import java.util.Scanner;

```
public class RotateMatrix90 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input matrix size (N x N)
        System.out.print("Enter the size of the square matrix (N): ");
        int n = scanner.nextInt();

        int[][] matrix = new int[n][n];

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

        // Rotate the matrix by 90 degrees
        rotate90Clockwise(matrix, n);

        // Print rotated matrix
        System.out.println("Matrix after 90-degree rotation:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }

        scanner.close();
    }

    // Function to rotate the matrix 90 degrees clockwise
    static void rotate90Clockwise(int[][] matrix, int n) {
        // Step 1: Transpose the matrix (swap rows and columns)
        for (int i = 0; i < n; i++) {
            for (int j = i; j < n; j++) {
                int temp = matrix[i][j];
                matrix[i][j] = matrix[j][i];
                matrix[j][i] = temp;
            }
        }

        // Step 2: Reverse each row to get the final rotated matrix
        for (int i = 0; i < n; i++) {
            int left = 0, right = n - 1;
            while (left < right) {
                int temp = matrix[i][left];
                matrix[i][left] = matrix[i][right];
                matrix[i][right] = temp;
                left++;
                right--;
            }
        }
    }
}
```

25. Find the Diagonal Sum ○ Compute the sum of both diagonals in a square matrix.

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

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

        // Input matrix size (N x N)
        System.out.print("Enter the size of the square matrix (N): ");
        int n = scanner.nextInt();

        int[][] matrix = new int[n][n];

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

        // Compute diagonal sum
        int sum = 0;
        for (int i = 0; i < n; i++) {
            sum += matrix[i][i]; // Main diagonal
            sum += matrix[i][n - i - 1]; // Secondary diagonal
        }

        // If N is odd, subtract the middle element (double-counted)
        if (n % 2 == 1) {
            sum -= matrix[n / 2][n / 2];
        }

        // Print the diagonal sum
        System.out.println("Sum of both diagonals: " + sum);

        scanner.close();
    }
}
```