# Array coding question

**080_Shital Hiray_JH**

1. **Find the Largest and Smallest Element**
   - **Given an array, find the smallest and largest elements in it.**

Ans:

```
public class SmallLarge {
   public static void main(String[] args) {
      int[] arr = {12, 45, 2, 89, 34, 7, 99, 23};

      int min = arr[0], max = arr[0];

      for (int num : arr) {
         if (num < min) min = num;
         if (num > max) max = num;
      }

      System.out.println("Smallest element: " + min);
      System.out.println("Largest element: " + max);
   }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac SmallLarge.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java SmallLarge
Smallest element: 2
Largest element: 99
```

2. **Reverse an Array**
   - **Reverse the given array in place.**

**Ans:**

```
public class ReverseArray {
   public static void main(String[] args) {
      int[] arr = {1, 2, 3, 4, 5};

      System.out.print("Original array: ");
      for (int num : arr) {
         System.out.print(num + " ");
      }

      System.out.print("\nReversed array: ");
      for (int i = arr.length - 1; i >= 0; i--) {
         System.out.print(arr[i] + " ");
      }
   }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac ReverseArray.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java ReverseArray
Original array: 1 2 3 4 5
Reversed array: 5 4 3 2 1
```

3. **Find the Second Largest Element**
   - **Find the second-largest element in the given array.**

Ans:

```
public class SecondLargest {
```

```
    public static void main(String[] args) {
        int[] arr = {12, 45, 2, 89, 34, 7, 99, 23};

        int largest = Integer.MIN_VALUE;
        int secondLargest = Integer.MIN_VALUE;

        for (int num : arr) {
            if (num > largest) {
                secondLargest = largest;
                largest = num;
            } else if (num > secondLargest && num != largest) {
                secondLargest = num;
            }
        }

        if (secondLargest == Integer.MIN_VALUE) {
            System.out.println("No second-largest element found.");
        } else {
            System.out.println("Second largest element: " + secondLargest);
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac SecondLargest.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java SecondLargest
Second largest element: 89
```

4. **Count Even and Odd Numbers**
    - **Count the number of even and odd numbers in an array.**

**Ans:**

```
public class CountEvenOdd {
    public static void main(String[] args) {
        int[] arr = {12, 45, 2, 89, 34, 7, 99, 23};

        int evenCount = 0, oddCount = 0;

        for (int num : arr) {
            if (num % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }

        System.out.println("Even numbers count: " + evenCount);
        System.out.println("Odd numbers count: " + oddCount);
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac CountEvenOdd.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java CountEvenOdd
Even numbers count: 3
Odd numbers count: 5
```

5. **Find Sum and Average**
    - **Compute the sum and average of all elements in the array.**

**Ans:**

```
public class SumAndAverage {
    public static void main(String[] args) {
        int[] arr = {12, 45, 2, 89, 34, 7, 99, 23};
```

```
        int sum = 0;

        for (int num : arr) {
            sum += num;
        }

        double average = (double) sum / arr.length;

        System.out.println("Sum of elements: " + sum);
        System.out.println("Average of elements: " + average);
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac SumAndAverage.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java SumAndAverage
Sum of elements: 311
Average of elements: 38.875
```

6. **Remove Duplicates from a Sorted Array**
    - **Remove duplicate elements from a sorted array without using extra space.**

**Ans:**

```
import java.util.Arrays;

public class RemoveDuplicates {
    public static int removeDuplicates(int[] arr) {
        if (arr.length == 0) return 0;

        int index = 1;

        for (int i = 1; i < arr.length; i++) {
            if (arr[i] != arr[i - 1]) {
                arr[index] = arr[i];
                index++;
            }
        }
        return index;
    }

    public static void main(String[] args) {
        int[] arr = {1, 1, 2, 2, 3, 4, 4, 5};
        int newLength = removeDuplicates(arr);

        System.out.println("Array after removing duplicates:");
        for (int i = 0; i < newLength; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac RemoveDuplicates.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java RemoveDuplicates
Array after removing duplicates:
1 2 3 4 5
```

7. **Rotate an Array**
    - **Rotate the array to the right by k positions.**

**Ans:**

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

```
public class RotateArray {
    public static void rotate(int[] arr, int k) {
        int n = arr.length;
        k = k % n;
        reverse(arr, 0, n - 1);
        reverse(arr, 0, k - 1);
        reverse(arr, k, n - 1);
    }

    private static void reverse(int[] arr, int start, int end) {
        while (start < end) {
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;
            start++;
            end--;
        }
    }

    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7};
        int k = 3;

        rotate(arr, k);

        System.out.println("Rotated Array: " + Arrays.toString(arr));
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac RotateArray.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java RotateArray
Rotated Array: [5, 6, 7, 1, 2, 3, 4]
```

8. **Merge Two Sorted Arrays**
    - **Merge two sorted arrays into a single sorted array without using extra space.**

**Ans:**

```
import java.util.Arrays;

public class MergeSortedArrays {
    public static void merge(int[] arr1, int m, int[] arr2, int n) {
        int i = m - 1;
        int j = n - 1;
        int k = m + n - 1;

        while (i >= 0 && j >= 0) {
            if (arr1[i] > arr2[j]) {
                arr1[k] = arr1[i];
                i--;
            } else {
                arr1[k] = arr2[j];
                j--;
            }
            k--;
        }

        while (j >= 0) {
            arr1[k] = arr2[j];
```

```
            j--;
            k--;
        }
    }

    public static void main(String[] args) {
        int[] arr1 = {1, 3, 5, 0, 0, 0};
        int[] arr2 = {2, 4, 6};
        int m = 3, n = 3;

        merge(arr1, m, arr2, n);

        System.out.println("Merged Array: " + Arrays.toString(arr1));
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac MergeSortedArrays.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java MergeSortedArrays
Merged Array: [1, 2, 3, 4, 5, 6]
```

9. **Find Missing Number in an Array**
    - **Given an array of size n-1 containing numbers from 1 to n, find the missing number.**

**Ans:**

```
public class FindMissingNumber {
    public static int findMissing(int[] arr, int n) {
        int totalSum = n * (n + 1) / 2;
        int arraySum = 0;

        for (int num : arr) {
            arraySum += num;
        }

        return totalSum - arraySum;
    }

    public static void main(String[] args) {
        int[] arr = {1, 2, 4, 5, 6};
        int n = 6;

        System.out.println("Missing Number: " + findMissing(arr, n));
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac FindMissingNumber.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java FindMissingNumber
Missing Number: 3
```

10. **Find Intersection and Union of Two Arrays**
    - **Find the intersection and union of two unsorted arrays.**

**Ans:**

```
import java.util.HashSet;

public class ArrayUnionIntersection {
    public static void main(String[] args) {
        int[] arr1 = {1, 2, 3, 4, 5};
```

```
        int[] arr2 = {3, 4, 5, 6, 7};

        System.out.println("Union of arrays:");
        printUnion(arr1, arr2);

        System.out.println("\nIntersection of arrays:");
        printIntersection(arr1, arr2);
    }

    static void printUnion(int[] arr1, int[] arr2) {
        HashSet<Integer> unionSet = new HashSet<>();
        for (int num : arr1) {
            unionSet.add(num);
        }
        for (int num : arr2) {
            unionSet.add(num);
        }
        System.out.println(unionSet);
    }

    static void printIntersection(int[] arr1, int[] arr2) {
        HashSet<Integer> set1 = new HashSet<>();
        HashSet<Integer> intersectionSet = new HashSet<>();

        for (int num : arr1) {
            set1.add(num);
        }

        for (int num : arr2) {
            if (set1.contains(num)) {
                intersectionSet.add(num);
            }
        }
        System.out.println(intersectionSet);
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac ArrayUnionIntersection.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java ArrayUnionIntersection
Union of arrays:
[1, 2, 3, 4, 5, 6, 7]

Intersection of arrays:
[3, 4, 5]
```

11. **Find a Subarray with Given Sum**
    - **Given an array of integers, find the subarray that sums to a given value S.**

**Ans:**

```
public class SubarrayWithGivenSum {

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

        findSubarray(arr, S);
    }

    static void findSubarray(int[] arr, int S) {
        int left = 0, sum = 0;

        for (int right = 0; right < arr.length; right++) {
```

```
            sum += arr[right];

            while (sum > S && left <= right) {
                sum -= arr[left];
                left++;
            }

            if (sum == S) {
                System.out.println("Subarray found from index " + left + " to " + right);
                return;
            }
        }
        System.out.println("No subarray found with sum " + S);
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac SubarrayWithGivenSum.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java SubarrayWithGivenSum
Subarray found from index 2 to 4
```

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**

**Ans:**

```
import java.util.Scanner;

public class ArrayNumberAnalysis {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] numbers = new int[20];

        int evenCount = 0, oddCount = 0, multipleOf3Count = 0;

        System.out.println("Enter 20 integers:");
        for (int i = 0; i < 20; i++) {
            numbers[i] = sc.nextInt();

            if (numbers[i] % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }

            if (numbers[i] % 3 == 0) {
                multipleOf3Count++;
            }
        }

        System.out.println("\nResults:");
        System.out.println("Number of even numbers: " + evenCount);
        System.out.println("Number of odd numbers: " + oddCount);
        System.out.println("Number of multiples of 3: " + multipleOf3Count);

        sc.close();
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java ArrayNumberAnalysis
Enter 20 integers:
20
51
34
45
48
10
56
32
77
45
64
75
51
212
42
21
12
21
32
56

Results:
Number of even numbers: 12
Number of odd numbers: 8
Number of multiples of 3: 10
```

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.**

**Ans:**

```
import java.util.Scanner;
public class StudentMarksAnalysis {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int totalStudents = 20;
        int[] physics = new int[totalStudents];
        int[] chemistry = new int[totalStudents];
        int[] maths = new int[totalStudents];

        int highScorers = 0, lowScorers = 0;

        System.out.println("Enter marks for 20 students (out of 100) in Physics, Chemistry, and
Maths:");
        for (int i = 0; i < totalStudents; i++) {
            System.out.println("Student " + (i + 1) + ":");
            System.out.print("Physics: ");
            physics[i] = sc.nextInt();
            System.out.print("Chemistry: ");
            chemistry[i] = sc.nextInt();
            System.out.print("Maths: ");
            maths[i] = sc.nextInt();

            double aggregate = (physics[i] + chemistry[i] + maths[i]) / 3.0;

            if (aggregate >= 75) {
                highScorers++;
            } else if (aggregate <= 40) {
                lowScorers++;
            }
        }
```

```
        System.out.println("\nResults:");
        System.out.println("Number of students securing 75% and above: " + highScorers)
        System.out.println("Number of students securing 40% and below: " + lowScorers);

        sc.close();
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac StudentMarksAnalysis.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java StudentMarksAnalysis
Enter marks for 20 students (out of 100) in Physics, Chemistry, and Maths:
Student 1:
Physics: 75
Chemistry: 66
Maths: 55
Student 2:
Physics: 88
Chemistry: 75
Maths: 64
Student 3:
Physics: 96
Chemistry: 54
Maths: 85
Student 4:
Physics: 66
Chemistry: 88
Maths: 90
Student 5:
Physics: 55
Chemistry: 77
Maths: 75
Student 6:
Physics: 69
Chemistry: 58
Maths: 47
Student 7:
Physics: 66
Chemistry: 62
Maths: 84
Student 8:
Physics: 81
Chemistry: 76
Maths: 72
Student 9:
Physics: 95
Chemistry: 85
Maths: 81
C:\Windows\System32\cmd.e
Student 10:
Physics: 86
Chemistry: 81
Maths: 75
Student 11:
Physics: 64
Chemistry: 67
Maths: 69
Student 12:
Physics: 62
Chemistry: 61
Maths: 54
Student 13:
Physics: 56
Chemistry: 95
Maths: 64
Student 14:
Physics: 66
Chemistry: 67
Maths: 69
Student 15:
Physics: 63
Chemistry: 95
Maths: 75
Student 16:
Physics: 84
Chemistry: 85
Maths: 81
Student 17:
Physics: 80
Chemistry: 59
Maths: 96
Student 18:
Physics: 84
Chemistry: 75
Maths: 76
Student 19:
Physics: 65
Chemistry: 64
Maths: 68
Student 20:
Maths: 68
Student 20:
Physics: 84
Chemistry: 80
Maths: 83

Results:
Number of students securing 75% and above: 11
Number of students securing 40% and below: 0
```

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.**

**Ans:**

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

public class EvenOddArray {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] arr = new int[20];

        ArrayList<Integer> even = new ArrayList<>();
        ArrayList<Integer> odd = new ArrayList<>();

        System.out.println("Enter 20 integers:");
        for (int i = 0; i < 20; i++) {
            arr[i] = scanner.nextInt();

            if (arr[i] % 2 == 0) {
                even.add(arr[i]);
            } else {
                odd.add(arr[i]);
            }
        }


        int[] evenArray = even.stream().mapToInt(i -> i).toArray();
        int[] oddArray = odd.stream().mapToInt(i -> i).toArray();


        System.out.println("\nEven numbers:");
        for (int num : evenArray) {
            System.out.print(num + " ");
        }

        System.out.println("\nOdd numbers:");
        for (int num : oddArray) {
            System.out.print(num + " ");
        }

        scanner.close();
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac EvenOddArray.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java EvenOddArray
Enter 20 integers:
20
40
32
54
54
63
54
22
33
44
88
77
55
22
33
22
56
65
58
74

Even numbers:
20 40 32 54 54 54 22 44 88 22 22 56 58 74
Odd numbers:
63 33 77 55 33 65
```

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]**

**Ans:**

```
public class ZeroSumSubarraysSimple {
   public static void main(String[] args) {
     int[][] testCases = {
         {1, 3, -7, 3, 2, 3, 1, -3, -2, -2},
         {1, 2, -3, 4, 5, 6},
         {1, 2, -2, 3, 4, 5, 6}
      };

      for (int i = 0; i < testCases.length; i++) {
         System.out.println("Sub-arrays with 0 sum:");
         findZeroSumSubarrays(testCases[i]);
         System.out.println();
      }
   }

   static void findZeroSumSubarrays(int[] arr) {
      int n = arr.length;
      for (int start = 0; start < n; start++) {
```

```
            int sum = 0;
            for (int end = start; end < n; end++) {
                sum += arr[end];

                if (sum == 0) {
                    printSubarray(arr, start, end);
                }
            }
        }
    }

    static void printSubarray(int[] arr, int start, int end) {
        System.out.print("[");
        for (int i = start; i <= end; i++) {
            System.out.print(arr[i] + (i < end ? ", " : ""));
        }
        System.out.println("]");
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac ZeroSumSubarraysSimple.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java ZeroSumSubarraysSimple
Sub-arrays with 0 sum:
[1, 3, -7, 3]
[3, -7, 3, 2, 3, 1, -3, -2]

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

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]**

**Ans:**

```
import java.util.Arrays;

public class MergeSortedArrays1 {
    public static void main(String[] args) {
        int[] A = {1, 5, 6, 7, 8, 10};
        int[] B = {2, 4, 9};

        mergeArrays1(A, B);
    }

    static void mergeArrays1(int[] A, int[] B) {
        int p = A.length, q = B.length;
        int[] merged = new int[p + q];

        System.arraycopy(A, 0, merged, 0, p);
        System.arraycopy(B, 0, merged, p, q);
        Arrays.sort(merged);

        System.arraycopy(merged, 0, A, 0, p);
        System.arraycopy(merged, p, B, 0, q);

        System.out.println("Sorted Arrays:");
        System.out.println("A: " + Arrays.toString(A));
        System.out.println("B: " + Arrays.toString(B));
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac MergeSortedArrays1.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java MergeSortedArrays1
Sorted Arrays:
A: [1, 2, 4, 5, 6, 7]
B: [8, 9, 10]
```

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**

**Ans:**

```
public class MaxProductPair {
    public static void main(String[] args) {
        int[] nums = {2, 3, 5, 7, -7, 5, 8, -5};
        findMaxProductPair(nums);
    }

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

        int max1 = Integer.MIN_VALUE, max2 = Integer.MIN_VALUE;
        int min1 = Integer.MAX_VALUE, min2 = Integer.MAX_VALUE;

        for (int num : nums) {
            if (num > max1) {
                max2 = max1;
                max1 = num;
            } else if (num > max2) {
                max2 = num;
            }

            if (num < min1) {
```

```
                min2 = min1;
                min1 = num;
            } else if (num < min2) {
                min2 = num;
            }
        }

        int product1 = max1 * max2;
        int product2 = min1 * min2;

        if (product1 > product2) {
            System.out.println("Pair is (" + max1 + ", " + max2 + "), Maximum Product: " +
        product1);
        } else {
            System.out.println("Pair is (" + min1 + ", " + min2 + "), Maximum Product: " +
        product2);
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac MaxProductPair.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java MaxProductPair
Pair is (8, 7), Maximum Product: 56
```

18. **Print a Matrix**
    - **Given an m x n matrix, print all its elements row-wise.**

**Ans**:

```
public class PrintMatrix {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        printMatrix(matrix);
    }

    public static void printMatrix(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();
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac PrintMatrix.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java PrintMatrix
1 2 3
4 5 6
7 8 9
```

19. **Transpose of a Matrix**
    - **Given a matrix, return its transpose (swap rows and columns).**

**Ans:**

```
public class TransposeMatrix {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        int[][] transposedMatrix = transpose(matrix);
        printMatrix(transposedMatrix);
    }

    public static int[][] transpose(int[][] matrix) {
        int rows = matrix.length;
        int cols = matrix[0].length;
        int[][] transposed = new int[cols][rows];

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                transposed[j][i] = matrix[i][j];
            }
        }
        return transposed;
    }

    public static void printMatrix(int[][] matrix) {
        for (int[] row : matrix) {
            for (int elem : row) {
                System.out.print(elem + " ");
            }
            System.out.println();
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac TransposeMatrix.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java TransposeMatrix
1 4 7
2 5 8
3 6 9
```

20. **Sum of Two Matrices**
    - **Given two matrices of the same size, compute their sum.**

**Ans:**

```
public class SumOfMatrices {
    public static void main(String[] args) {
        int[][] matrix1 = {
            {1, 2, 3},
```

```
        {4, 5, 6},
        {7, 8, 9}
    };

    int[][] matrix2 = {
        {9, 8, 7},
        {6, 5, 4},
        {3, 2, 1}
    };

    int[][] sumMatrix = addMatrices(matrix1, matrix2);
    printMatrix(sumMatrix);
  }

  public static int[][] addMatrices(int[][] matrix1, int[][] matrix2) {
    int rows = matrix1.length;
    int cols = matrix1[0].length;
    int[][] sum = new int[rows][cols];

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            sum[i][j] = matrix1[i][j] + matrix2[i][j];
        }
    }
    return sum;
  }

  public static void printMatrix(int[][] matrix) {
    for (int[] row : matrix) {
        for (int elem : row) {
            System.out.print(elem + " ");
        }
        System.out.println();
    }
  }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac SumOfMatrices.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java SumOfMatrices
10 10 10
10 10 10
10 10 10
```

21. **Row-wise and Column-wise Sum**
    - **Find the sum of each row and each column of a given matrix.**

**Ans:**

```
public class RowColumnSum {
  public static void main(String[] args) {
    int[][] matrix = {
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9}
    };

    computeRowColumnSum(matrix);
  }

  public static void computeRowColumnSum(int[][] matrix) {
    int rows = matrix.length;
    int cols = matrix[0].length;
```

```
        System.out.println("Row-wise Sum:");
        for (int i = 0; i < rows; i++) {
            int rowSum = 0;
            for (int j = 0; j < cols; j++) {
                rowSum += matrix[i][j];
            }
            System.out.println("Row " + (i + 1) + ": " + rowSum);
        }

        System.out.println("Column-wise Sum:");
        for (int j = 0; j < cols; j++) {
            int colSum = 0;
            for (int i = 0; i < rows; i++) {
                colSum += matrix[i][j];
            }
            System.out.println("Column " + (j + 1) + ": " + colSum);
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac RowColumnSum.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java RowColumnSum
Row-wise Sum:
Row 1: 6
Row 2: 15
Row 3: 24
Column-wise Sum:
Column 1: 12
Column 2: 15
Column 3: 18
```

22. **Find the Maximum Element in a Matrix**
    - **Find the largest element in a given matrix.**

**Ans:**

```
public class MaxElementMatrix {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        int maxElement = findMaxElement(matrix);
        System.out.println("Maximum Element in the Matrix: " + maxElement);
    }

    public static int findMaxElement(int[][] matrix) {
        int max = Integer.MIN_VALUE;

        for (int[] row : matrix) {
            for (int elem : row) {
                if (elem > max) {
                    max = elem;
                }
            }
        }
        return max;
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac MaxElementMatrix.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java MaxElementMatrix
Maximum Element in the Matrix: 9
```

23. **Matrix Multiplication**
    - **Multiply two matrices and return the resultant matrix.**

**Ans:**

```
public class MatrixMultiplication {
    public static void main(String[] args) {
        int[][] matrix1 = {
            {1, 2, 3},
            {4, 5, 6}
        };

        int[][] matrix2 = {
            {7, 8},
            {9, 10},
            {11, 12}
        };

        int[][] result = multiplyMatrices(matrix1, matrix2);
        printMatrix(result);
    }

    public static int[][] multiplyMatrices(int[][] matrix1, int[][] matrix2) {
        int rows1 = matrix1.length;
        int cols1 = matrix1[0].length;
        int cols2 = matrix2[0].length;

        int[][] product = new int[rows1][cols2];

        for (int i = 0; i < rows1; i++) {
            for (int j = 0; j < cols2; j++) {
                for (int k = 0; k < cols1; k++) {
                    product[i][j] += matrix1[i][k] * matrix2[k][j];
                }
            }
        }
        return product;
    }

    public static void printMatrix(int[][] matrix) {
        for (int[] row : matrix) {
            for (int elem : row) {
                System.out.print(elem + " ");
            }
            System.out.println();
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac MatrixMultiplication.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java MatrixMultiplication
58 64
139 154
```

24. **Rotate a Matrix by 90 Degrees**
    - **Rotate a given N x N matrix by 90 degrees clockwise.**

**Ans:**

```
public class RotateMatrix90 {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        rotateMatrix(matrix);
        printMatrix(matrix);
    }
    public static void rotateMatrix(int[][] matrix) {
        int n = matrix.length;

        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;
            }
        }

        for (int i = 0; i < n; i++) {
            for (int j = 0, k = n - 1; j < k; j++, k--) {
                int temp = matrix[i][j];
                matrix[i][j] = matrix[i][k];
                matrix[i][k] = temp;
            }
        }
    }

    public static void printMatrix(int[][] matrix) {
        for (int[] row : matrix) {
            for (int elem : row) {
                System.out.print(elem + " ");
            }
            System.out.println();
        }
    }
}
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac RotateMatrix90.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java RotateMatrix90
7 4 1
8 5 2
9 6 3
```

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

**Ans:**

```
public class DiagonalSumMatrix {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
```

```
    };

    int sum = findDiagonalSum(matrix);
    System.out.println("Sum of both diagonals: " + sum);
}

public static int findDiagonalSum(int[][] matrix) {
    int n = matrix.length;
    int sum = 0;

    for (int i = 0; i < n; i++) {
        sum += matrix[i][i]; // Primary diagonal
        if (i != n - 1 - i) {
            sum += matrix[i][n - 1 - i]; // Secondary diagonal
        }
    }
    return sum;
}
  }
```

```
D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>javac DiagonalSumMatrix.java

D:\CDAC DATA FEB 25\OOPJ\Assignment\OOPJ Assignment-3>java DiagonalSumMatrix
Sum of both diagonals: 25
```