100+ Array

Monday, July 29, 2024

12:36 AM

Array , 2D Array most Important Interview programming Question write 100+ Array Question with output using Java chapter wise ChatGPT

1. Basics of Arrays

38. Sum of Array Elements

```
int[] array = {1, 2, 3, 4, 5};
int sum = 0;
for (int i = 0; i < array.length; i++) {
  sum += array[i];
System.out.println(sum);
Output: 15
39. Finding Maximum Element in an Array
int[] array = {1, 3, 7, 0, 5};
int max = array[0];
for (int i = 1; i < array.length; i++) {
  if (array[i] > max) {
     max = array[i];
  }
}
System.out.println(max);
Output:
Copy code
40. Finding Minimum Element in an Array
Copy code
int[] array = {1, 3, 7, 0, 5};
int min = array[0];
for (int i = 1; i < array.length; i++) {
  if (array[i] < min) {
    min = array[i];
  }
System.out.println(min);
Output:
```

```
1. Find the Maximum Element in an Array java Copy code int[] arr = {3, 5, 7, 2, 8};
```

Copy code

```
Multiply Two Matrices
    java
     Copy code
     int[][] matrix1 = {
       {1, 2},
       \{3, 4\}
     };
    int[][] matrix2 = {
       {5, 6},
       \{7, 8\}
    };
     int[][] result = new int[matrix1.length][matrix2[0].length];
    for (int i = 0; i < matrix1.length; i++) {
       for (int j = 0; j < matrix2[0].length; j++) {
         for (int k = 0; k < matrix2.length; k++) {
            result[i][j] += matrix1[i][k] * matrix2[k][j];
         }
       }
    }
    System.out.println("Matrix Multiplication Result:");
    for (int[] row : result) {
       System.out.println(Arrays.toString(row));
    }
    Output:
     csharp
     Copy code
     Matrix Multiplication Result:
    [19, 22]
    [43, 50]
    Sum of All Elements in a 2D Matrix
    java
    Copy code
    int[][] matrix = {
       {1, 2, 3},
       {4, 5, 6},
       \{7, 8, 9\}
    };
    int sum = 0;
    for (int[] row : matrix) {
       for (int num : row) {
         sum += num;
       }
    System.out.println("Sum of All Elements: " + sum);
     Output: Sum of All Elements: 45
57
    Find the Largest Rectangle of 1's in a Binary Matrix
    java
     Copy code
    int[][] matrix = {
       \{1, 0, 1, 0\},\
       {1, 1, 1, 1},
```

{1, 1, 1, 0}

```
\{1, 0, 1, 0\},\
   Copy code
                                                      \{1, 1, 1, 1\},\
   int[] arr = {3, 5, 7, 2, 8};
                                                      \{1, 1, 1, 0\}
   int max = arr[0];
                                                    };
   for (int i = 1; i < arr.length; i++) {
                                                    int maxArea = 0;
      if (arr[i] > max) {
                                                    int[] height = new int[matrix[0].length];
        max = arr[i];
     }
                                                    for (int i = 0; i < matrix.length; i++) {
                                                      for (int j = 0; j < matrix[0].length; j++) {
   System.out.println("Max: " + max);
                                                         height[j] = matrix[i][j] == 0 ? 0 : height[j] + 1;
   Output: Max: 8
2. Find the Minimum Element in an Array
                                                      int area = calculateMaxArea(height);
   iava
                                                      maxArea = Math.max(maxArea, area);
   Copy code
                                                    }
   int[] arr = {3, 5, 7, 2, 8};
                                                    System.out.println("Largest Rectangle Area: " + maxArea);
   int min = arr[0];
                                                    private static int calculateMaxArea(int[] height) {
   for (int i = 1; i < arr.length; i++) {
                                                      int maxArea = 0;
      if (arr[i] < min) {
                                                      Stack<Integer> stack = new Stack<>();
        min = arr[i];
                                                      int index = 0;
     }
                                                      while (index < height.length) {
                                                         if (stack.isEmpty() || height[index] >=
   System.out.println("Min: " + min);
                                                    height[stack.peek()]) {
   Output: Min: 2
                                                           stack.push(index++);
3. Calculate the Sum of All Elements in an
                                                        } else {
   Array
                                                           int topOfStack = stack.pop();
   java
                                                           int area = height[topOfStack] * (stack.isEmpty() ?
   Copy code
                                                    index : index - stack.peek() - 1);
   int[] arr = {1, 2, 3, 4, 5};
                                                           maxArea = Math.max(maxArea, area);
   int sum = 0;
                                                        }
   for (int num: arr) {
      sum += num;
                                                      while (!stack.isEmpty()) {
                                                         int topOfStack = stack.pop();
   System.out.println("Sum: " + sum);
                                                         int area = height[topOfStack] * (stack.isEmpty()?
   Output: Sum: 15
                                                    index : index - stack.peek() - 1);
4. Calculate the Average of Elements in an
                                                         maxArea = Math.max(maxArea, area);
   Array
                                                      }
   java
                                                      return maxArea;
   Copy code
   int[] arr = {1, 2, 3, 4, 5};
                                                    Output: Largest Rectangle Area: 8
   int sum = 0;
                                                    Print the Diagonals of a Matrix
   for (int num: arr) {
                                                    java
      sum += num;
                                                    Copy code
   }
                                                    int[][] matrix = {
   double average = (double) sum / arr.length;
                                                      {1, 2, 3},
   System.out.println("Average: " + average);
                                                      \{4, 5, 6\},\
   Output: Average: 3.0
                                                      \{7, 8, 9\}
5. Count Occurrences of a Specific Element
   java
                                                    System.out.println("Primary Diagonal:");
   Copy code
                                                    for (int i = 0; i < matrix.length; i++) {
   int[] arr = {1, 2, 3, 2, 2, 4};
                                                      System.out.print(matrix[i][i] + " ");
   int count = 0;
                                                    }
   for (int num: arr) {
                                                    System.out.println("\nSecondary Diagonal:");
      if (num == 2) {
                                                    for (int i = 0; i < matrix.length; i++) {
```

```
for (int num : arr) {
                                                     System.out.println("\nSecondary Diagonal:");
      if (num == 2) {
                                                    for (int i = 0; i < matrix.length; i++) {
        count++;
                                                       System.out.print(matrix[i][matrix.length - 1 - i] + " ");
     }
                                                    }
                                                     Output:
   System.out.println("Occurrences of 2: " +
                                                     mathematica
   count);
                                                     Copy code
   Output: Occurrences of 2: 3
                                                     Primary Diagonal:
6. Reverse an Array
                                                     159
   java
                                                     Secondary Diagonal:
   Copy code
                                                     357
   int[] arr = {1, 2, 3, 4, 5};
                                                59.
                                                    Check if a 2D Array is a Submatrix of Another 2D Array
   for (int i = 0; i < arr.length / 2; i++) {
                                                    java
      int temp = arr[i];
                                                     Copy code
      arr[i] = arr[arr.length - 1 - i];
                                                    int[][] matrix = {
      arr[arr.length - 1 - i] = temp;
                                                       {1, 2, 3, 4},
                                                       {5, 6, 7, 8},
   System.out.println("Reversed Array: " +
                                                       {9, 10, 11, 12},
   Arrays.toString(arr));
                                                       {13, 14, 15, 16}
   Output: Reversed Array: [5, 4, 3, 2, 1]
                                                    };
7. Check if an Array is Sorted in Ascending
                                                    int[][] submatrix = {
   Order
                                                       {6, 7},
   iava
                                                       {10, 11}
   Copy code
                                                    };
   int[] arr = {1, 2, 3, 4, 5};
                                                     boolean found = false;
   boolean isSorted = true;
                                                    for (int i = 0; i <= matrix.length - submatrix.length; i++) {
   for (int i = 1; i < arr.length; i++) {
                                                       for (int j = 0; j <= matrix[0].length - submatrix[0].length;
      if (arr[i] < arr[i - 1]) {
                                                    j++) {
        isSorted = false;
                                                         boolean match = true;
        break;
                                                         for (int x = 0; x < \text{submatrix.length}; x++) {
      }
                                                            for (int y = 0; y < submatrix[0].length; y++) {
   }
                                                              if (matrix[i + x][j + y] != submatrix[x][y]) {
   System.out.println("Is Sorted: " + isSorted);
                                                                 match = false;
   Output: Is Sorted: true
                                                                 break;
8. Find the Index of a Specific Element
                                                              }
   java
                                                            }
   Copy code
                                                            if (!match) break;
   int[] arr = {1, 2, 3, 4, 5};
   int target = 3;
                                                         if (match) {
   int index = -1;
                                                            found = true;
   for (int i = 0; i < arr.length; i++) {
                                                            break;
      if (arr[i] == target) {
                                                         }
        index = i;
        break;
                                                       if (found) break;
      }
                                                    System.out.println("Submatrix Found: " + found);
   System.out.println("Index of 3: " + index);
                                                     Output: Submatrix Found: true
   Output: Index of 3: 2
                                                60.
                                                    Multiply Each Element of a Matrix by a Scalar
9. Remove Duplicates from an Array
                                                    java
   java
                                                    Copy code
   Copy code
                                                    int[][] matrix = {
   int[] arr = {1, 2, 2, 3, 4, 4, 5};
                                                       {1, 2, 3},
```

```
copy code
      Copy code
                                                      int[][] matrix = {
      int[] arr = {1, 2, 2, 3, 4, 4, 5};
                                                         {1, 2, 3},
      Set<Integer> set = new LinkedHashSet<>();
                                                         {4, 5, 6},
      for (int num : arr) {
                                                         \{7, 8, 9\}
        set.add(num);
                                                      };
      }
                                                      int scalar = 3;
      int[] result =
                                                      int[][] result = new int[matrix.length][matrix[0].length];
      set.stream().mapToInt(Integer::intValue).to
                                                      for (int i = 0; i < matrix.length; i++) {
      Array();
                                                         for (int j = 0; j < matrix[i].length; j++) {
      System.out.println("Array without
                                                           result[i][j] = matrix[i][j] * scalar;
      duplicates: " + Arrays.toString(result));
                                                         }
                                                      }
      Output: Array without duplicates: [1, 2, 3, 4,
                                                      System.out.println("Matrix after Scalar Multiplication:");
                                                      for (int[] row : result) {
 10. Rotate an Array to the Right by k Positions
                                                         System.out.println(Arrays.toString(row));
                                                      }
      Copy code
                                                      Output:
      int[] arr = {1, 2, 3, 4, 5};
                                                      csharp
      int k = 2;
                                                      Copy code
      int n = arr.length;
                                                      Matrix after Scalar Multiplication:
      k = k \% n;
                                                      [3, 6, 9]
      int[] result = new int[n];
                                                      [12, 15, 18]
      for (int i = 0; i < n; i++) {
                                                      [21, 24, 27]
        result[(i + k) \% n] = arr[i];
                                                      Sort Each Row of a Matrix in Ascending Order
                                                      java
      System.out.println("Rotated Array: " +
                                                      Copy code
      Arrays.toString(result));
                                                      int[][] matrix = {
      Output: Rotated Array: [4, 5, 1, 2, 3]
                                                         {3, 1, 2},
2. Intermediate Array Problems
                                                         \{6, 4, 5\},\
11. Find the Second Largest Element
                                                         \{9, 7, 8\}
     java
                                                      };
      Copy code
                                                      for (int[] row : matrix) {
      int[] arr = {10, 20, 4, 45, 99};
                                                         Arrays.sort(row);
      int largest = Integer.MIN_VALUE;
      int secondLargest = Integer.MIN VALUE;
                                                      System.out.println("Matrix with Sorted Rows:");
      for (int num: arr) {
                                                      for (int[] row : matrix) {
        if (num > largest) {
                                                         System.out.println(Arrays.toString(row));
          secondLargest = largest;
                                                      }
          largest = num;
        } else if (num > secondLargest && num
                                                      Output:
      < largest) {
                                                      csharp
          secondLargest = num;
                                                       Copy code
        }
                                                       Matrix with Sorted Rows:
                                                      [1, 2, 3]
      System.out.println("Second Largest: " +
                                                      [4, 5, 6]
      secondLargest);
                                                      [7, 8, 9]
      Output: Second Largest: 45
                                                      Print a 2D Array in Spiral Order
 12. Find the Missing Number in a Consecutive
                                                      java
      Array
                                                      Copy code
      java
                                                      int[][] matrix = {
      Copy code
                                                         \{1, 2, 3\},\
      int[] arr = {1, 2, 3, 5};
                                                         {4, 5, 6},
```

```
Copy code
                                                        {1, 2, 3},
     int[] arr = {1, 2, 3, 5};
                                                        {4, 5, 6},
     int n = arr.length + 1;
                                                       \{7, 8, 9\}
     int sum = n * (n + 1) / 2;
                                                     };
     for (int num: arr) {
                                                     int top = 0, bottom = matrix.length - 1, left = 0, right =
       sum -= num;
                                                     matrix[0].length - 1;
                                                     while (top <= bottom && left <= right) {
     System.out.println("Missing Number: " +
                                                       for (int i = left; i <= right; i++) {
     sum);
                                                          System.out.print(matrix[top][i] + " ");
     Output: Missing Number: 4
                                                       }
13. Find the Majority Element (Element that
                                                       top++;
     Appears More than N/2 Times)
                                                        for (int i = top; i <= bottom; i++) {
    java
                                                          System.out.print(matrix[i][right] + " ");
     Copy code
                                                       }
     int[] arr = {3, 3, 4, 2, 4, 4, 2, 4, 4};
                                                        right--;
     int candidate = arr[0];
                                                        if (top <= bottom) {
     int count = 1;
                                                          for (int i = right; i >= left; i--) {
     for (int i = 1; i < arr.length; i++) {
                                                            System.out.print(matrix[bottom][i] + " ");
       if (arr[i] == candidate) {
         count++;
                                                          bottom--;
       } else {
         count--;
                                                       if (left <= right) {</pre>
         if (count == 0) {
                                                          for (int i = bottom; i >= top; i--) {
           candidate = arr[i];
                                                            System.out.print(matrix[i][left] + " ");
           count = 1;
         }
                                                          left++;
       }
                                                       }
     }
     count = 0;
                                                     Output: 1 2 3 6 9 8 7 4 5
     for (int num : arr) {
                                                     Generate a 2D Matrix with Random Numbers
       if (num == candidate) {
                                                     java
         count++;
                                                     Copy code
       }
                                                     int rows = 3, cols = 3;
                                                     int[][] matrix = new int[rows][cols];
     if (count > arr.length / 2) {
                                                     Random rand = new Random();
       System.out.println("Majority Element: " +
                                                     for (int i = 0; i < rows; i++) {
     candidate);
                                                       for (int j = 0; j < cols; j++) {
     } else {
                                                          matrix[i][j] = rand.nextInt(100); // Random number
       System.out.println("No Majority
                                                     between 0 and 99
     Element");
                                                       }
                                                     }
     Output: Majority Element: 4
                                                     System.out.println("Random Matrix:");
14. Find the Intersection of Two Arrays
                                                     for (int[] row : matrix) {
    java
                                                        System.out.println(Arrays.toString(row));
     Copy code
                                                     }
     int[] arr1 = {1, 2, 2, 1};
     int[] arr2 = {2, 2};
                                                     Output: (Example)
     Set<Integer> set1 = new HashSet<>();
     for (int num: arr1) {
                                                     csharp
       set1.add(num);
                                                     Copy code
                                                     Random Matrix:
     Set<Integer> result = new HashSet<>();
                                                     [23, 56, 12]
     for lint num . arr? \ \
```

```
Random Matrix:
    Set<Integer> result = new HashSet<>();
                                                    [23, 56, 12]
    for (int num: arr2) {
                                                    [77, 45, 89]
       if (set1.contains(num)) {
                                                    [34, 67, 98]
         result.add(num);
                                                    Sum of Diagonals in a Square Matrix
       }
                                                    java
                                                    Copy code
    System.out.println("Intersection: " + result);
                                                    int[][] matrix = {
    Output: Intersection: [2]
                                                      {1, 2, 3},
15. Find the Union of Two Arrays
                                                      {4, 5, 6},
    java
                                                      {7, 8, 9}
    Copy code
                                                    };
    int[] arr1 = {1, 2, 2, 1};
                                                    int primaryDiagonalSum = 0, secondaryDiagonalSum = 0;
    int[] arr2 = {2, 3, 4};
                                                    for (int i = 0; i < matrix.length; i++) {
    Set<Integer> result = new HashSet<>();
                                                      primaryDiagonalSum += matrix[i][i];
    for (int num: arr1) {
                                                      secondaryDiagonalSum += matrix[i][matrix.length - 1 -
       result.add(num);
                                                    i];
                                                    }
    for (int num: arr2) {
                                                    System.out.println("Primary Diagonal Sum: " +
       result.add(num);
                                                    primaryDiagonalSum);
                                                    System.out.println("Secondary Diagonal Sum: " +
    System.out.println("Union: " + result);
                                                    secondaryDiagonalSum);
    Output: Union: [1, 2, 3, 4]
                                                    Output:
16. Find the Longest Consecutive Sequence
                                                    mathematica
    java
                                                    Copy code
    Copy code
                                                    Primary Diagonal Sum: 15
    int[] arr = {100, 4, 200, 1, 3, 2};
                                                    Secondary Diagonal Sum: 15
    Set<Integer> set = new HashSet<>();
                                                    Generate a Spiral Matrix
    for (int num: arr) {
                                                    java
       set.add(num);
                                                    Copy code
    }
                                                    int n = 4;
    int longest = 0;
                                                    int[][] matrix = new int[n][n];
    for (int num: arr) {
                                                    int value = 1;
       if (!set.contains(num - 1)) {
                                                    int top = 0, bottom = n - 1, left = 0, right = n - 1;
         int currentNum = num;
         int currentStreak = 1;
                                                    while (top <= bottom && left <= right) {
         while (set.contains(currentNum + 1)) {
                                                      for (int i = left; i \le right; i++) {
           currentNum += 1;
                                                         matrix[top][i] = value++;
           currentStreak += 1;
                                                      }
         }
                                                      top++;
         longest = Math.max(longest,
                                                      for (int i = top; i \le bottom; i++) {
    currentStreak);
                                                         matrix[i][right] = value++;
       }
                                                      }
                                                      right--;
    System.out.println("Longest Consecutive
                                                      if (top <= bottom) {
    Sequence: " + longest);
                                                         for (int i = right; i >= left; i--) {
    Output: Longest Consecutive Sequence: 4
                                                           matrix[bottom][i] = value++;
17. Find Duplicate Elements in an Array
    java
                                                         bottom--;
    Copy code
                                                      }
    int[] arr = {1, 2, 3, 4, 5, 1, 2};
                                                      if (left <= right) {
    Set<Integer> set = new HashSet<>();
                                                         for (int i = bottom; i >= top; i--) {
    SetaIntegers dunlicates = new HashSetas/1.
```

```
int[] arr = {1, 2, 3, 4, 5, 1, 2};
                                                        if (left <= right) {</pre>
     Set<Integer> set = new HashSet<>();
                                                           for (int i = bottom; i >= top; i--) {
     Set<Integer> duplicates = new HashSet<>();
                                                             matrix[i][left] = value++;
     for (int num: arr) {
                                                           }
       if (!set.add(num)) {
                                                           left++;
         duplicates.add(num);
                                                         }
       }
                                                      }
     }
                                                      System.out.println("Spiral Matrix:");
     System.out.println("Duplicates: " +
                                                      for (int[] row : matrix) {
     duplicates);
                                                         System.out.println(Arrays.toString(row));
     Output: Duplicates: [1, 2]
                                                      }
18. Rotate a 2D Array (Matrix) by 90 Degrees
                                                      Output:
     java
                                                      csharp
     Copy code
                                                      Copy code
     int[][] matrix = {
                                                      Spiral Matrix:
       \{1, 2, 3\},\
                                                      [1, 2, 3, 4]
       {4, 5, 6},
                                                      [12, 13, 14, 5]
       {7, 8, 9}
                                                      [11, 16, 15, 6]
     };
                                                      [10, 9, 8, 7]
     int n = matrix.length;
                                                 66.
                                                      Count Occurrences of Each Element in a Matrix
     for (int i = 0; i < n; i++) {
                                                      java
       for (int j = i; j < n; j++) {
                                                      Copy code
         int temp = matrix[i][j];
                                                      int[][] matrix = {
         matrix[i][j] = matrix[j][i];
                                                         {1, 2, 2},
         matrix[j][i] = temp;
                                                         {3, 1, 1},
       }
                                                         \{4, 5, 1\}
                                                      };
     for (int i = 0; i < n; i++) {
                                                      Map<Integer, Integer> countMap = new HashMap<>();
       for (int j = 0; j < n / 2; j++) {
                                                      for (int[] row : matrix) {
         int temp = matrix[i][j];
                                                         for (int num : row) {
         matrix[i][j] = matrix[i][n - 1 - j];
                                                           countMap.put(num, countMap.getOrDefault(num, 0)
         matrix[i][n - 1 - j] = temp;
                                                      + 1);
       }
                                                         }
     }
                                                      }
     System.out.println("Rotated Matrix: ");
                                                      System.out.println("Element Occurrences: " + countMap);
     for (int[] row : matrix) {
                                                      Output: Element Occurrences: {1=5, 2=2, 3=1, 4=1, 5=1}
       System.out.println(Arrays.toString(row)
                                                      Print Matrix Elements in Anti-Clockwise Spiral Order
     }
                                                      iava
     Output:
                                                      Copy code
     csharp
                                                      int[][] matrix = {
     Copy code
                                                         {1, 2, 3},
     Rotated Matrix:
                                                         {4, 5, 6},
     [7, 4, 1]
                                                         \{7, 8, 9\}
     [8, 5, 2]
                                                      };
     [9, 6, 3]
                                                      int top = 0, bottom = matrix.length - 1, left = 0, right =
19. Find the Largest Sum of Contiguous
                                                      matrix[0].length - 1;
     Subarray (Kadane's Algorithm)
                                                      while (top <= bottom && left <= right) {
     java
                                                         for (int i = top; i \le bottom; i++) {
     Copy code
                                                           System.out.print(matrix[i][left] + " ");
     int[] arr = {-2, 1, -3, 4, -1, 2, 1, -5, 4};
                                                         }
     int maxCurrent = arr[0];
                                                         left++;
     int maxGlobal = arr[0];
                                                         for (int i = left; i \le right; i++) {
     for (int i = 1: i < arr.length: i++) {
```

```
int maxCurrent = arr[0];
                                                        left++;
     int maxGlobal = arr[0];
                                                        for (int i = left; i <= right; i++) {
     for (int i = 1; i < arr.length; i++) {
                                                          System.out.print(matrix[bottom][i] + " ");
        maxCurrent = Math.max(arr[i],
                                                        }
     maxCurrent + arr[i]);
                                                        bottom--;
        maxGlobal = Math.max(maxGlobal,
                                                        if (top <= bottom) {
     maxCurrent);
                                                          for (int i = bottom; i >= top; i--) {
     }
                                                             System.out.print(matrix[i][right] + " ");
     System.out.println("Maximum Subarray
     Sum: " + maxGlobal);
                                                          right--;
     Output: Maximum Subarray Sum: 6
                                                        }
20. Find All Pairs in an Array that Sum to a
                                                        if (left <= right) {
     Target Value
                                                          for (int i = right; i >= left; i--) {
     java
                                                             System.out.print(matrix[top][i] + " ");
     Copy code
                                                          }
     int[] arr = {1, 4, 5, 7, 8, 9};
                                                          top++;
     int target = 8;
                                                        }
     Map<Integer, Integer> map = new
                                                     }
     HashMap<>();
                                                     Output: 147896325
     for (int num: arr) {
                                                     Create a Matrix with Multiples of a Number
        int complement = target - num;
                                                     java
        if (map.containsKey(complement)) {
                                                     Copy code
          System.out.println("Pair: (" +
                                                     int n = 3;
     complement + ", " + num + ")");
                                                     int multiplier = 3;
                                                     int[][] matrix = new int[n][n];
        map.put(num, map.getOrDefault(num, 0)
                                                     for (int i = 0; i < n; i++) {
     + 1);
                                                        for (int j = 0; j < n; j++) {
     }
                                                          matrix[i][j] = multiplier * (i * n + j + 1);
     Output: Pair: (1, 7)
                                                        }
3. Advanced Array Problems
21. Find the Kth Largest Element in an Array
                                                     System.out.println("Matrix with Multiples of " + multiplier
     java
                                                     + ":");
     Copy code
                                                     for (int[] row : matrix) {
     int[] arr = {3, 2, 1, 5, 6, 4};
                                                        System.out.println(Arrays.toString(row));
     int k = 2;
                                                     }
     PriorityQueue<Integer> minHeap = new
                                                     Output:
     PriorityQueue<>(k);
                                                      csharp
     for (int num: arr) {
                                                      Copy code
        minHeap.add(num);
                                                      Matrix with Multiples of 3:
        if (minHeap.size() > k) {
                                                     [3, 6, 9]
          minHeap.poll();
                                                     [12, 15, 18]
        }
                                                      [21, 24, 27]
                                                     Print Matrix Elements in Zigzag Order
     System.out.println("Kth Largest Element:
                                                     java
     minHeap.peek());
                                                     Copy code
     Output: Kth Largest Element: 5
                                                     int[][] matrix = {
22. Find All Subarrays with a Sum Equal to K
                                                        \{1, 2, 3\},\
     java
                                                        {4, 5, 6},
     Copy code
                                                        \{7, 8, 9\}
     int[] arr = {1, 1, 1, 1, 1};
                                                     };
     int k = 2;
                                                     System.out.println("Zigzag Order:");
     int count = 0;
                                                     for (int i = 0; i < matrix.length; i++) {
     for (int start = 0; start < arr.length; start++) {
```

```
int k = 2;
                                                      System.out.println("Zigzag Order:");
     int count = 0;
                                                      for (int i = 0; i < matrix.length; i++) {
     for (int start = 0; start < arr.length; start++) {
                                                        if (i \% 2 == 0) {
       int sum = 0;
                                                          for (int j = 0; j < matrix[0].length; j++) {
       for (int end = start; end < arr.length;
                                                             System.out.print(matrix[i][j] + " ");
     end++) {
                                                           }
         sum += arr[end];
                                                        } else {
         if (sum == k) {
                                                          for (int j = matrix[0].length - 1; j \ge 0; j--) {
            count++;
                                                             System.out.print(matrix[i][j] + " ");
         }
       }
                                                        }
     System.out.println("Count of Subarrays with
                                                      Output: 1 2 3 6 5 4 7 8 9
     Sum = " + k + ": " + count);
                                                      Transpose and Reverse Rows of a Matrix
     Output: Count of Subarrays with Sum = 2: 4
                                                      iava
23. Merge Two Sorted Arrays
                                                      Copy code
     java
                                                      int[][] matrix = {
     Copy code
                                                        {1, 2, 3},
     int[] arr1 = {1, 3, 5, 7};
                                                        {4, 5, 6},
     int[] arr2 = {2, 4, 6, 8};
                                                        \{7, 8, 9\}
     int[] merged = new int[arr1.length +
     arr2.length];
                                                      int[][] transposed = new int[matrix[0].length]
     int i = 0, j = 0, k = 0;
                                                      [matrix.length];
     while (i < arr1.length && j < arr2.length) {
                                                      for (int i = 0; i < matrix.length; i++) {
       if (arr1[i] < arr2[j]) {
                                                        for (int j = 0; j < matrix[i].length; j++) {
         merged[k++] = arr1[i++];
                                                           transposed[j][i] = matrix[i][j];
       } else {
                                                        }
         merged[k++] = arr2[j++];
                                                      }
       }
                                                      System.out.println("Transposed and Reversed Rows
                                                      Matrix:");
     while (i < arr1.length) {
                                                      for (int[] row : transposed) {
       merged[k++] = arr1[i++];
                                                        int[] reversedRow = new int[row.length];
                                                        for (int k = 0; k < row.length; k++) {
     while (j < arr2.length) {
                                                           reversedRow[k] = row[row.length - 1 - k];
       merged[k++] = arr2[j++];
                                                        }
                                                        System.out.println(Arrays.toString(reversedRow));
     System.out.println("Merged Array: " +
     Arrays.toString(merged));
                                                      Output:
                                                      csharp
     Output: Merged Array: [1, 2, 3, 4, 5, 6, 7, 8]
                                                      Copy code
24. Find the Longest Subarray with Distinct
                                                      Transposed and Reversed Rows Matrix:
     Elements
                                                      [7, 4, 1]
     Copy code
                                                      [8, 5, 2]
     int[] arr = {1, 2, 3, 4, 3, 2, 1};
                                                      [9, 6, 3]
     Set<Integer> set = new HashSet<>();
                                                 71. Sum of Border Elements in a Matrix
     int longest = 0;
                                                      java
     int left = 0;
                                                      Copy code
     for (int right = 0; right < arr.length; right++) {
                                                      int[][] matrix = {
       while (set.contains(arr[right])) {
                                                        {1, 2, 3},
         set.remove(arr[left++]);
                                                        {4, 5, 6},
       }
                                                        \{7, 8, 9\}
       set.add(arr[right]);
                                                      };
       longest = Math.max(longest, right - left +
```

```
}
                                                       \{7, 8, 9\}
       set.add(arr[right]);
                                                    };
       longest = Math.max(longest, right - left +
                                                    int sum = 0;
                                                    for (int i = 0; i < matrix.length; i++) {
    }
                                                       sum += matrix[i][0] + matrix[i][matrix[i].length - 1];
    System.out.println("Longest Subarray with
    Distinct Elements: " + longest);
                                                    for (int j = 1; j < matrix[0].length - 1; j++) {
    Output: Longest Subarray with Distinct
                                                       sum += matrix[0][j] + matrix[matrix.length - 1][j];
    Elements: 4
25. Find the Median of Two Sorted Arrays
                                                    System.out.println("Sum of Border Elements: " + sum);
    java
    Copy code
                                                     Output: Sum of Border Elements: 40
    int[] nums1 = {1, 3};
                                                72.
                                                    Find the Maximal Square of 1's in a Binary Matrix
    int[] nums2 = {2};
                                                    java
    int[] merged = new int[nums1.length +
                                                     Copy code
    nums2.length];
                                                    int[][] matrix = {
    int i = 0, j = 0, k = 0;
                                                       \{1, 0, 1, 0\},\
    while (i < nums1.length && j
                                                       {1, 1, 1, 1},
    < nums2.length) {
                                                       {1, 1, 1, 0}
       if (nums1[i] < nums2[j]) {
                                                    };
         merged[k++] = nums1[i++];
                                                     int maxSide = 0;
       } else {
                                                    int[][] dp = new int[matrix.length][matrix[0].length];
         merged[k++] = nums2[j++];
       }
                                                    for (int i = 0; i < matrix.length; i++) {
                                                       for (int j = 0; j < matrix[0].length; j++) {
    while (i < nums1.length) {
                                                         if (matrix[i][j] == 1) {
       merged[k++] = nums1[i++];
                                                           if (i == 0 | | i == 0) {
                                                              dp[i][j] = 1;
    while (j < nums2.length) {
                                                           } else {
       merged[k++] = nums2[j++];
                                                              dp[i][j] = Math.min(dp[i-1][j], Math.min(dp[i][j-1][j])
                                                     1], dp[i - 1][j - 1])) + 1;
    double median;
    if (merged.length % 2 == 0) {
                                                           maxSide = Math.max(maxSide, dp[i][j]);
       median = (merged[merged.length / 2 - 1]
                                                         }
    + merged[merged.length / 2]) / 2.0;
                                                       }
    } else {
       median = merged[merged.length / 2];
                                                    int maxSquareArea = maxSide * maxSide;
                                                     System.out.println("Maximal Square Area of 1's: " +
    System.out.println("Median: " + median);
                                                     maxSquareArea);
    Output: Median: 2.0
                                                     Output: Maximal Square Area of 1's: 4
26. Find the Intersection of Two Arrays with 73.
                                                     Rotate a Matrix by 90 Degrees Clockwise
    Count
                                                    java
    java
                                                     Copy code
    Copy code
                                                    int[][] matrix = {
    int[] nums1 = {1, 2, 2, 1};
                                                       {1, 2, 3},
    int[] nums2 = {2, 2};
                                                       {4, 5, 6},
    Map<Integer, Integer> countMap = new
                                                       \{7, 8, 9\}
    HashMap<>();
                                                    };
    for (int num: nums1) {
                                                    int n = matrix.length;
       countMap.put(num,
                                                     int[][] rotated = new int[n][n];
    countMap.getOrDefault(num, 0) + 1);
    }
                                                    for (int i = 0; i < n; i++) {
    List<Integer> result = new ArrayList<>();
                                                       for (int i = 0: i < n: i++) {
```

```
countiviap.getoi Detauttitutti, oj + 1),
                                                     for (int i = 0; i < n; i++) {
    List<Integer> result = new ArrayList<>();
                                                       for (int j = 0; j < n; j++) {
    for (int num: nums2) {
                                                          rotated[j][n - 1 - i] = matrix[i][j];
       if (countMap.containsKey(num) &&
                                                       }
    countMap.get(num) > 0) {
                                                     }
         result.add(num);
                                                     System.out.println("Rotated Matrix:");
         countMap.put(num,
                                                     for (int[] row : rotated) {
    countMap.get(num) - 1);
                                                       System.out.println(Arrays.toString(row));
       }
    }
                                                     Output:
    System.out.println("Intersection with
                                                     csharp
    Count: " + result);
                                                     Copy code
                                                     Rotated Matrix:
    Output: Intersection with Count: [2, 2]
                                                     [7, 4, 1]
27. Find the Longest Increasing Subsequence
                                                     [8, 5, 2]
    java
                                                     [9, 6, 3]
    Copy code
                                                74.
                                                    Find the Minimum Path Sum in a Matrix (Dynamic
    int[] nums = {10, 9, 2, 5, 3, 7, 101, 18};
                                                     Programming)
    int[] dp = new int[nums.length];
                                                     java
    Arrays.fill(dp, 1);
                                                     Copy code
    for (int i = 1; i < nums.length; i++) {
                                                     int[][] matrix = {
       for (int j = 0; j < i; j++) {
                                                       {1, 3, 1},
         if (nums[i] > nums[j]) {
                                                       {1, 5, 1},
           dp[i] = Math.max(dp[i], dp[j] + 1);
                                                       \{4, 2, 1\}
         }
                                                     };
       }
                                                     int m = matrix.length;
                                                     int n = matrix[0].length;
    System.out.println("Length of Longest
                                                     int[][] dp = new int[m][n];
    Increasing Subsequence: " +
    Arrays.stream(dp).max().getAsInt());
                                                     dp[0][0] = matrix[0][0];
    Output: Length of Longest Increasing
                                                     for (int i = 1; i < m; i++) {
    Subsequence: 4
                                                       dp[i][0] = dp[i - 1][0] + matrix[i][0];
28. Find the Subarray with Maximum Sum
                                                     }
    java
                                                     for (int j = 1; j < n; j++) {
    Copy code
                                                       dp[0][j] = dp[0][j - 1] + matrix[0][j];
    int[] arr = {-2, 1, -3, 4, -1, 2, 1, -5, 4};
    int maxCurrent = arr[0];
                                                     for (int i = 1; i < m; i++) {
    int maxGlobal = arr[0];
                                                       for (int j = 1; j < n; j++) {
    for (int i = 1; i < arr.length; i++) {
                                                         dp[i][j] = Math.min(dp[i-1][j], dp[i][j-1]) + matrix[i]
       maxCurrent = Math.max(arr[i],
                                                     [j];
    maxCurrent + arr[i]);
                                                       }
       maxGlobal = Math.max(maxGlobal,
    maxCurrent);
                                                     System.out.println("Minimum Path Sum: " + dp[m - 1][n -
    }
    System.out.println("Maximum Subarray
                                                     Output: Minimum Path Sum: 7
    Sum: " + maxGlobal);
                                                75.
                                                     Find Common Elements in All Rows of a Matrix
    Output: Maximum Subarray Sum: 6
                                                     java
29. Find the Missing Number in a 1 to N Array
                                                     Copy code
    java
                                                     int[][] matrix = {
    Copy code
                                                       \{1, 2, 3, 4\},\
    int[] arr = {1, 2, 4, 5};
                                                       {3, 4, 5, 6},
    int n = 5;
                                                       \{4, 5, 6, 7\}
```

```
copy couc
                                                        \{1, 2, 3, 4\},\
     int[] arr = {1, 2, 4, 5};
                                                        {3, 4, 5, 6},
     int n = 5;
                                                        {4, 5, 6, 7}
     int sum = n * (n + 1) / 2;
                                                     };
     for (int num: arr) {
                                                     Set<Integer> commonElements = new HashSet<>();
        sum -= num;
                                                     Set<Integer> rowElements = new HashSet<>();
     }
                                                     for (int num : matrix[0]) {
     System.out.println("Missing Number: " +
                                                        rowElements.add(num);
     Output: Missing Number: 3
                                                     for (int i = 1; i < matrix.length; i++) {
30. Find the First Non-Repeating Character in
                                                        Set<Integer> currentRowElements = new HashSet<>();
     an Array
                                                        for (int num : matrix[i]) {
     java
                                                          if (rowElements.contains(num)) {
     Copy code
                                                            currentRowElements.add(num);
     char[] arr = {'a', 'b', 'c', 'a', 'b', 'c', 'd'};
                                                          }
     Map<Character, Integer> countMap = new
                                                        }
     LinkedHashMap<>();
                                                        rowElements = currentRowElements;
     for (char c : arr) {
        countMap.put(c,
                                                     commonElements = rowElements;
     countMap.getOrDefault(c, 0) + 1);
                                                     System.out.println("Common Elements in All Rows: " +
                                                     commonElements);
     char result = 0;
                                                     Output: Common Elements in All Rows: [4]
     for (char c : countMap.keySet()) {
                                                76. Check if a Matrix is Symmetric
        if (countMap.get(c) == 1) {
                                                     java
          result = c;
                                                     Copy code
          break;
                                                     int[][] matrix = {
        }
                                                        {1, 2, 3},
                                                        \{2, 4, 5\},\
     System.out.println("First Non-Repeating
                                                        {3, 5, 6}
     Character: " + result);
                                                     };
     Output: First Non-Repeating Character: d
                                                     boolean isSymmetric = true;
4. 2D Arrays and Matrix Problems
                                                     for (int i = 0; i < matrix.length; i++) {
31. Print a 2D Matrix
                                                        for (int j = 0; j < matrix[i].length; <math>j++) {
     java
                                                          if (matrix[i][j] != matrix[j][i]) {
     Copy code
                                                            isSymmetric = false;
     int[][] matrix = {
                                                            break;
        {1, 2, 3},
                                                          }
        \{4, 5, 6\},\
                                                        }
        \{7, 8, 9\}
                                                        if (!isSymmetric) break;
     };
     for (int[] row : matrix) {
                                                     System.out.println("Matrix is Symmetric: " +
        System.out.println(Arrays.toString(row));
                                                     isSymmetric);
     }
                                                     Output: Matrix is Symmetric: true
     Output:
                                                77.
                                                     Find the Median of a Matrix
     csharp
                                                     java
     Copy code
                                                     Copy code
     [1, 2, 3]
                                                     int[][] matrix = {
     [4, 5, 6]
                                                       {1, 2, 3},
     [7, 8, 9]
                                                        {4, 5, 6},
32. Transpose of a Matrix
                                                        \{7, 8, 9\}
     java
                                                     };
     Copy code
                                                     List<Integer> elements = new ArrayList<>();
     int[][] matrix = {
                                                    for (int[] row : matrix) {
```

	J	[<i>]</i> ;
	Copy code	List <integer> elements = new ArrayList<>();</integer>
	int[][] matrix = {	for (int[] row : matrix) {
	{1, 2, 3},	for (int num : row) {
	{4, 5, 6},	elements.add(num);
	{7, 8, 9}	}
	} ;	\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
	int[][] transposed = new	Collections.sort(elements);
	int[matrix[0].length][matrix.length];	· · · · · · · · · · · · · · · · · · ·
	for (int i = 0; i < matrix.length; i++) {	int median = elements.get(elements.size() / 2);
	for (int j = 0; j < matrix[0].length; j++) {	System.out.println("Median of Matrix: " + median);
	transposad[i][i] = matriv[i][i]	Output: Median of Matrix: 5
	78.	Print a Matrix in a Specific Pattern (e.g., Zigzag)
	}	java
	}	Copy code
	System.out.println("Transposed Matrix: ");	int[][] matrix = {
	for (int[] row : transposed) {	{1, 2, 3},
	System.out.println(Arrays.toString(row));	{4, 5, 6},
	}	{7, 8, 9}
	Output:	};
	csharp	for (int i = 0; i < matrix.length; i++) {
	Copy code	if (i % 2 == 0) {
	Transposed Matrix:	, ,
	[1, 4, 7]	for (int j = 0; j < matrix[i].length; j++) {
	[2, 5, 8]	System.out.print(matrix[i][j] + " ");
	[3, 6, 9]	}
22	Print the Diagonals of a Matrix	} else {
55.	_	for (int j = matrix[i].length - 1; j >= 0; j) {
	java	System.out.print(matrix[i][j] + " ");
	Copy code	}
	int[][] matrix = {	}
	{1, 2, 3},	}
	{4, 5, 6},	Output: 1 2 3 6 5 4 7 8 9
	{7, 8, 9} 79.	Fill a Matrix in Diagonal Order
	} ;	java
	int n = matrix.length;	Copy code
	System.out.println("Primary Diagonal:");	int n = 4;
	for (int i = 0; i < n; i++) {	int[][] matrix = new int[n][n];
	System.out.print(matrix[i][i] + " ");	int value = 1;
	}	· · · · · · · · · · · · · · · · · · ·
	System.out.println("\nSecondary	for (int sum = 0; sum < 2 * n - 1; sum++) {
	Diagonal:");	int row = sum < n ? 0 : sum - n + 1;
	for (int i = 0; i < n; i++) {	int col = sum < n ? sum : n - 1;
	System.out.print(matrix[i][n - 1 - i] + " ");	while (row < n && col >= 0) {
	System.out.print(matrix[i][$i - 1 - i$] + j ,	matrix[row][col] = value++;
		row++;
	Output:	col;
	mathematica	}
	Copy code	}
	Primary Diagonal:	System.out.println("Matrix Filled in Diagonal Order:");
	159	for (int[] row : matrix) {
	Secondary Diagonal:	System.out.println(Arrays.toString(row));
	357	}
34.	Find the Largest Sum of Submatrix	Outnut
	java	Output:
	Copy code	csharp
		Copy code

	java	csharp
	Copy code	Copy code
	int[][] matrix = {	
	{1, 2, -1},	Matrix Filled in Diagonal Order:
	{-1, 2, 3},	[1, 2, 4, 7]
	{1, -1, 2}	[3, 5, 8, 11]
	};	[6, 9, 12, 14]
	int mayCum - Intoger MINI \/ALLE	[10, 13, 15, 16]
	int[][] dp = new int[matrix.length]	Find All Unique Elements in a Matrix
	[matrix[0].length];	java Convende
	for (int i = 0; i < matrix.length; i++) {	Copy code
	for (int j = 0; j < matrix[0].length; j++) {	int[][] matrix = {
	dp[i][j] = matrix[i][j];	{1, 2, 2},
	if (i > 0) dp[i][j] += dp[i - 1][j];	{3, 1, 1}, {4, 5, 1}
	if (j > 0) dp[i][j] += dp[i][j - 1];	
	if $(i > 0 & j > 0)$ dp[i][j] -= dp[i - 1][j - 1];	};
	}	Set <integer> uniqueElements = new HashSet<>();</integer>
	}	for (int[] row : matrix) {
	for (int i1 = 0; i1 < matrix.length; i1++) {	for (int num : row) {
	for (int j1 = 0; j1 < matrix[0].length; j1++) {	uniqueElements.add(num);
	for (int i2 = i1; i2 < matrix.length; i2++) {	}
	for (int $j2 = j1$; $j2 < matrix[0].length; j2++)$	}
	{	System.out.println("Unique Elements in Matrix: " +
	int sum = dp[i2][j2];	uniqueElements);
	if (i1 > 0) sum -= dp[i1 - 1][j2];	Output: Unique Elements in Matrix: [1, 2, 3, 4, 5]
	if (j1 > 0) sum -= dp[i2][j1 - 1];	4o mini
	if (i1 > 0 && j1 > 0) sum += dp[i1 -	continue
	1][i1 - 1]·	ChatGPT
	maxSum = Math.max(maxSum, sum);	Diagonal Sum of a Matrix
	}	java
	}	Copy code
	}	int[][] matrix = {
	1	{1, 2, 3},
	System.out.println("Largest Sum of	{4, 5, 6},
	Submatrix: " + maxSum);	{7, 8, 9}
	Output: Largest Sum of Submatrix: 10	};
25	Spiral Order of a Matrix	int primaryDiagonalSum = 0;
33.	java	int secondaryDiagonalSum = 0;
	Copy code	for (int i = 0; i < matrix.length; i++) {
	int[][] matrix = {	primaryDiagonalSum += matrix[i][i];
	{1, 2, 3},	secondaryDiagonalSum += matrix[i][matrix.length - 1 -
	{4, 5, 6},	[i];
	{7, 8, 9}	}
	{ <i>i</i> , <i>i</i> , <i>i</i> , <i>i</i> };	System.out.println("Primary Diagonal Sum: " +
	int m = matrix.length;	primaryDiagonalSum);
	int n = matrix[0].length;	System.out.println("Secondary Diagonal Sum: " +
	int top = 0, bottom = $m - 1$, left = 0, right =	secondaryDiagonalSum);
		Output:
	n - 1; while (top <= bottom && left <= right) {	mathematica
	for (int i = left; i <= right; i++) {	Copy code
	System.out.print(matrix[top][i] + " ");	Primary Diagonal Sum: 15
	1	Secondary Diagonal Sum: 15
	82.	Flatten a 2D Matrix into a 1D Array

```
i ililialy Diagonal Jaili. 13
          System.out.print(matrix[top][i] + " ");
                                                       Secondary Diagonal Sum: 15
       }
                                                       Flatten a 2D Matrix into a 1D Array
       top++;
                                                       java
       for (int i = top; i \le bottom; i++) {
                                                       Copy code
          System.out.print(matrix[i][right] + " ");
                                                       int[][] matrix = {
                                                          {1, 2, 3},
       right--;
                                                          {4, 5, 6},
       if (top <= bottom) {
                                                          \{7, 8, 9\}
          for (int i = right; i >= left; i--) {
                                                       };
       System.out.print(matrix[bottom][i] + " ");
                                                       int[] flattened = new int[matrix.length * matrix[0].length];
          }
                                                       int index = 0;
          bottom--;
                                                       for (int[] row : matrix) {
       }
                                                          for (int num: row) {
       if (left <= right) {
                                                            flattened[index++] = num;
          for (int i = bottom; i >= top; i--) {
                                                          }
            System.out.print(matrix[i][left] + " ");
          }
                                                       System.out.println("Flattened Array: " +
          left++;
                                                       Arrays.toString(flattened));
       }
                                                       Output: Flattened Array: [1, 2, 3, 4, 5, 6, 7, 8, 9]
     }
                                                  83.
                                                       Transpose a Matrix
                                                       java
     Output: 123698745
                                                       Copy code
36. Count Number of Islands in a Matrix
                                                       int[][] matrix = {
     java
                                                          \{1, 2, 3\},\
     Copy code
                                                          \{4, 5, 6\},\
     int[][] grid = {
                                                          {7, 8, 9}
       \{1, 1, 0, 0, 0\},\
                                                       };
       \{1, 1, 0, 0, 1\},\
                                                       int[][] transposed = new int[matrix[0].length]
       \{0, 0, 0, 1, 1\},\
                                                       [matrix.length];
       \{0, 0, 0, 0, 0, 0\},\
       {1, 0, 1, 0, 1}
                                                       for (int i = 0; i < matrix.length; i++) {
     };
                                                          for (int j = 0; j < matrix[i].length; j++) {
     int numIslands = 0;
                                                            transposed[j][i] = matrix[i][j];
     int[][] directions = {{0, 1}, {1, 0}, {0, -1}, {-1,
                                                          }
     0}};
                                                       }
     for (int i = 0; i < grid.length; i++) {
                                                       System.out.println("Transposed Matrix:");
       for (int j = 0; j < grid[0].length; j++) {
                                                       for (int[] row : transposed) {
          if (grid[i][j] == 1) {
                                                          System.out.println(Arrays.toString(row));
            numIslands++;
            Queue<int[]> queue = new
                                                       Output:
     LinkedList<>();
                                                       csharp
            queue.offer(new int[]{i, j});
                                                       Copy code
            grid[i][j] = 0;
                                                       Transposed Matrix:
            while (!queue.isEmpty()) {
                                                       [1, 4, 7]
               int[] cell = queue.poll();
                                                       [2, 5, 8]
               for (int[] direction : directions) {
                                                       [3, 6, 9]
                 int newRow = cell[0] +
                                                       Print Matrix in Spiral Order
     direction[0];
                                                       java
                 int newCol = cell[1] +
                                                       Copy code
     direction[1];
                                                       int[][] matrix = {
                 if (newRow >= 0 && newRow
                                                          \{1, 2, 3\},\
     < grid.length &&
                                                          {4, 5, 6},
                    now(Cal >= 0 8.8. now(Cal
```

```
if (newRow >= 0 && newRow
                                                          {1, 2, 3},
     < grid.length &&
                                                          \{4, 5, 6\},\
                   newCol >= 0 && newCol
                                                          \{7, 8, 9\}
     < grid[0].length &&
                                                       };
                   grid[newRow][newCol] == 1) {
                                                       int top = 0, bottom = matrix.length - 1, left = 0, right =
                   queue.offer(new int[]
                                                       matrix[0].length - 1;
     {newRow, newCol});
                                                       while (top <= bottom && left <= right) {
                   grid[newRow][newCol] = 0;
                                                          for (int i = left; i <= right; i++) {
                 }
                                                            System.out.print(matrix[top][i] + " ");
                                                          }
            }
                                                          top++;
         }
                                                          for (int i = top; i \le bottom; i++) {
       }
                                                            System.out.print(matrix[i][right] + " ");
     }
                                                          }
     System.out.println("Number of Islands: " +
                                                          right--;
     numIslands);
                                                          if (top <= bottom) {</pre>
     Output: Number of Islands: 6
                                                            for (int i = right; i >= left; i--) {
37. Check if a Matrix is a Submatrix
                                                               System.out.print(matrix[bottom][i] + " ");
     java
                                                            }
     Copy code
                                                            bottom--;
     int[][] matrix = {
       \{1, 2, 3, 4, 5\},\
                                                          if (left <= right) {</pre>
       \{6, 7, 8, 9, 10\},\
                                                            for (int i = bottom; i >= top; i--) {
       {11, 12, 13, 14, 15},
                                                               System.out.print(matrix[i][left] + " ");
       {16, 17, 18, 19, 20}
     };
                                                            left++;
     int[][] submatrix = {
                                                          }
       {7, 8},
       {12, 13}
                                                       Output: 1 2 3 6 9 8 7 4 5
     };
                                                       Check if a Matrix is an Identity Matrix
     boolean found = false;
                                                       java
     for (int i = 0; i <= matrix.length -
                                                       Copy code
     submatrix.length; i++) {
                                                       int[][] matrix = {
       for (int j = 0; j \le matrix[0].length -
                                                          \{1, 0, 0\},\
     submatrix[0].length; j++) {
                                                          \{0, 1, 0\},\
          boolean match = true;
                                                          \{0, 0, 1\}
          for (int x = 0; x < submatrix.length; x++)
                                                       };
     {
                                                       boolean isIdentity = true;
      for (int y = 0; y < submatrix[0].length; y++)
                                                       for (int i = 0; i < matrix.length; i++) {
               if (matrix[i + x][j + y] !=
                                                          for (int j = 0; j < matrix[i].length; j++) {
     submatrix[x][y]) {
                                                            if (i == j && matrix[i][j] != 1 || i != j && matrix[i][j] !=
                 match = false;
                                                       0) {
                 break;
                                                               isIdentity = false;
              }
                                                               break;
                                                            }
            if (!match) break;
                                                          if (!isIdentity) break;
         if (match) {
            found = true;
                                                       System.out.println("Matrix is Identity: " + isIdentity);
            break;
                                                       Output: Matrix is Identity: true
         }
                                                  86. Find the Maximum Element in Each Row
       }
```

```
break;
                                                      Output: Matrix is Identity: true
         }
                                                 86. Find the Maximum Element in Each Row
       }
                                                      iava
       if (found) break;
                                                      Copy code
                                                      int[][] matrix = {
     System.out.println("Submatrix Found: " +
                                                        {1, 2, 3},
     found);
                                                        \{4, 5, 6\},\
     Output: Submatrix Found: true
                                                        \{7, 8, 9\}
38. Find the Boundary Elements of a Matrix
                                                      };
                                                      for (int i = 0; i < matrix.length; i++) {
     Copy code
                                                        int max = matrix[i][0];
     int[][] matrix = {
                                                        for (int j = 1; j < matrix[i].length; j++) {
       \{1, 2, 3\},\
                                                           if (matrix[i][j] > max) {
       {4, 5, 6},
                                                             max = matrix[i][j];
       \{7, 8, 9\}
                                                          }
     };
                                                        }
     int m = matrix.length;
                                                        System.out.println("Maximum in Row " + i + ": " + max);
     int n = matrix[0].length;
     System.out.println("Boundary Elements:");
                                                      Output:
     for (int i = 0; i < n; i++) {
                                                      mathematica
       System.out.print(matrix[0][i] + " ");
                                                      Copy code
                                                      Maximum in Row 0: 3
     for (int i = 1; i < m; i++) {
                                                      Maximum in Row 1: 6
       System.out.print(matrix[i][n - 1] + " ");
                                                      Maximum in Row 2:9
                                                      Find the Minimum Element in Each Column
     for (int i = n - 2; i >= 0; i--) {
                                                      iava
       System.out.print(matrix[m - 1][i] + " ");
                                                      Copy code
                                                      int[][] matrix = {
     for (int i = m - 2; i > 0; i--) {
                                                        {1, 2, 3},
       System.out.print(matrix[i][0] + " ");
                                                        \{4, 5, 6\},\
     }
                                                        \{7, 8, 9\}
                                                      };
     Output: Boundary Elements: 1 2 3 6 9 8 7 4
                                                      for (int j = 0; j < matrix[0].length; j++) {
39. Check if a Matrix is Symmetric
                                                        int min = matrix[0][j];
    java
                                                        for (int i = 1; i < matrix.length; i++) {
     Copy code
                                                           if (matrix[i][j] < min) {</pre>
     int[][] matrix = {
                                                             min = matrix[i][j];
       {1, 2, 3},
                                                           }
       \{2, 4, 5\},\
       {3, 5, 6}
                                                        System.out.println("Minimum in Column " + j + ": " +
     };
                                                      min);
     boolean isSymmetric = true;
     for (int i = 0; i < matrix.length; i++) {
                                                      Output:
       for (int j = 0; j < matrix[0].length; j++) {
                                                      mathematica
         if (matrix[i][j] != matrix[j][i]) {
                                                      Copy code
            isSymmetric = false;
                                                      Minimum in Column 0: 1
            break;
                                                      Minimum in Column 1: 2
         }
                                                      Minimum in Column 2: 3
       }
                                                      Count the Occurrences of Each Element in a Matrix
       if (!isSymmetric) break;
                                                      java
                                                      Copy code
     System.out.println("Matrix is Symmetric:
                                                      int[][] matrix = {
     isSymmetric):
```

```
Copy code
     System.out.println("Matrix is Symmetric:
                                                      int[][] matrix = {
     isSymmetric);
                                                        \{1, 2, 2\},\
                                                        {3, 1, 1},
     Output: Matrix is Symmetric: true
                                                        \{4, 5, 1\}
40. Find the Maximum Value in a Matrix
                                                      };
     iava
                                                      Map<Integer, Integer> frequencyMap = new HashMap<>
     Copy code
     int[][] matrix = {
                                                      for (int[] row : matrix) {
        {1, 2, 3},
                                                        for (int num: row) {
        \{4, 5, 6\},\
                                                           frequencyMap.put(num,
        \{7, 8, 9\}
                                                      frequencyMap.getOrDefault(num, 0) + 1);
     };
                                                        }
     int maxValue = matrix[0][0];
     for (int[] row : matrix) {
                                                      System.out.println("Element Frequencies: " +
        for (int value : row) {
                                                      frequencyMap);
          maxValue = Math.max(maxValue,
                                                      Output: Element Frequencies: {1=4, 2=2, 3=1, 4=1, 5=1}
     value);
                                                 89.
                                                      Find the Largest Rectangle of 1's in a Binary Matrix
        }
                                                      java
                                                      Copy code
     System.out.println("Maximum Value in
                                                      int[][] matrix = {
     Matrix: " + maxValue);
                                                        \{1, 0, 1, 0\},\
     Output: Maximum Value in Matrix: 9
                                                        {1, 1, 1, 1},
40 mini
                                                        \{1, 1, 1, 0\},\
continue
                                                        \{0, 1, 1, 0\}
ChatGPT
                                                      };
4. 2D Arrays and Matrix Problems (continued)
                                                      int maxArea = 0;
41. Rotate a Matrix 90 Degrees Clockwise
                                                      int[] heights = new int[matrix[0].length];
     java
                                                      for (int i = 0; i < matrix.length; i++) {
     Copy code
                                                        for (int j = 0; j < matrix[i].length; j++) {
     int[][] matrix = {
                                                           heights[j] = matrix[i][j] == 0 ? 0 : heights[j] + 1;
        {1, 2, 3},
        \{4, 5, 6\},\
                                                        maxArea = Math.max(maxArea,
        \{7, 8, 9\}
                                                      maxRectangleArea(heights));
     };
                                                      }
     int n = matrix.length;
                                                      System.out.println("Largest Rectangle of 1's Area: " +
     int[][] rotated = new int[n][n];
                                                      maxArea);
     for (int i = 0; i < n; i++) {
                                                      static int maxRectangleArea(int[] heights) {
        for (int j = 0; j < n; j++) {
                                                        Stack<Integer> stack = new Stack<>();
          rotated[j][n - 1 - i] = matrix[i][j];
                                                        int maxArea = 0;
        }
                                                        int index = 0;
                                                        while (index < heights.length) {
     System.out.println("Rotated Matrix:");
                                                           if (stack.isEmpty() | | heights[index] >=
     for (int[] row : rotated) {
                                                      heights[stack.peek()]) {
        System.out.println(Arrays.toString(row));
                                                             stack.push(index++);
     }
                                                          } else {
     Output:
                                                             int top = stack.pop();
     csharp
                                                             maxArea = Math.max(maxArea, heights[top] *
     Copy code
                                                      (stack.isEmpty() ? index : index - stack.peek() - 1));
     Rotated Matrix:
                                                           }
     [7, 4, 1]
                                                        }
     [8, 5, 2]
                                                        while (!stack.isEmpty()) {
     [9, 6, 3]
                                                           int tan - ctack nan//
```

```
[1, 4, 1]
     [8, 5, 2]
                                                        while (!stack.isEmpty()) {
     [9, 6, 3]
                                                          int top = stack.pop();
42. Check if a Matrix is Magic Square
                                                          maxArea = Math.max(maxArea, heights[top] *
     iava
                                                      (stack.isEmpty() ? index : index - stack.peek() - 1));
     Copy code
                                                        }
     int[][] matrix = {
       {2, 7, 6},
                                                        return maxArea;
       {9, 5, 1},
       {4, 3, 8}
                                                     Output: Largest Rectangle of 1's Area: 8
     };
                                                     Merge Two Sorted Arrays
     int sum = Arrays.stream(matrix[0]).sum();
     boolean isMagic = true;
                                                     java
     // Check rows
                                                     Copy code
     for (int i = 0; i < matrix.length; i++) {
                                                     int[] arr1 = {1, 3, 5, 7};
       if (Arrays.stream(matrix[i]).sum() != sum)
                                                     int[] arr2 = {2, 4, 6, 8};
     {
                                                     int[] merged = new int[arr1.length + arr2.length];
         isMagic = false;
                                                     int i = 0, j = 0, k = 0;
         break;
                                                     while (i < arr1.length && j < arr2.length) {
       }
                                                        if (arr1[i] <= arr2[j]) {
                                                          merged[k++] = arr1[i++];
     // Check columns
     for (int j = 0; j < matrix[0].length; j++) {
                                                          merged[k++] = arr2[j++];
       int colSum = 0;
                                                        }
       for (int i = 0; i < matrix.length; i++) {
                                                     }
         colSum += matrix[i][j];
                                                     while (i < arr1.length) {
       }
                                                        merged[k++] = arr1[i++];
       if (colSum != sum) {
         isMagic = false;
                                                     while (j < arr2.length) {
         break;
                                                        merged[k++] = arr2[j++];
       }
     }
                                                     System.out.println("Merged Array: " +
     // Check diagonals
                                                     Arrays.toString(merged));
     int diag1Sum = 0, diag2Sum = 0;
     for (int i = 0; i < matrix.length; i++) {
                                                      Output: Merged Array: [1, 2, 3, 4, 5, 6, 7, 8]
       diag1Sum += matrix[i][i];
                                                     Find the Common Elements in Two 2D Arrays
       diag2Sum += matrix[i][matrix.length - 1 -
                                                     java
    i];
                                                     Copy code
     }
                                                     int[][] matrix1 = {
     if (diag1Sum != sum || diag2Sum != sum) {
                                                        {1, 2, 3},
       isMagic = false;
                                                        \{4, 5, 6\}
                                                     };
     System.out.println("Matrix is Magic Square:
                                                     int[][] matrix2 = {
     " + isMagic);
                                                        \{4, 5, 6\},\
                                                        \{7, 8, 9\}
     Output: Matrix is Magic Square: true
                                                     };
43. Find the Median of a 2D Matrix
                                                     Set<Integer> set1 = new HashSet<>();
    java
                                                     Set<Integer> commonElements = new HashSet<>();
     Copy code
                                                     for (int[] row : matrix1) {
     int[][] matrix = {
                                                       for (int num: row) {
       {1, 2, 3},
                                                          set1.add(num);
       \{4, 5, 6\},\
       \{7, 8, 9\}
```

```
{1, 4, 5},
                                                           set1.add(num);
       {4, 5, 6},
                                                        }
       \{7, 8, 9\}
                                                      }
     };
                                                      for (int[] row : matrix2) {
     List<Integer> list = new ArrayList<>();
                                                        for (int num : row) {
     for (int[] row : matrix) {
                                                           if (set1.contains(num)) {
       for (int num : row) {
                                                             commonElements.add(num);
         list.add(num);
                                                          }
       }
                                                        }
     }
     Collections.sort(list);
                                                      System.out.println("Common Elements: " +
     int median = list.get(list.size() / 2);
                                                      commonElements);
     System.out.println("Median of Matrix: " +
                                                      Output: Common Elements: [4, 5, 6]
     median);
                                                      Rotate a Matrix 90 Degrees Clockwise
     Output: Median of Matrix: 5
                                                      java
44. Print the Upper Triangular Matrix
                                                      Copy code
     java
                                                      int[][] matrix = {
     Copy code
                                                        {1, 2, 3},
     int[][] matrix = {
                                                        {4, 5, 6},
       {1, 2, 3},
                                                        \{7, 8, 9\}
       {4, 5, 6},
       \{7, 8, 9\}
                                                      int n = matrix.length;
     };
                                                      int[][] rotated = new int[n][n];
     System.out.println("Upper Triangular
     Matrix:");
                                                      for (int i = 0; i < n; i++) {
     for (int i = 0; i < matrix.length; i++) {
                                                        for (int j = 0; j < n; j++) {
       for (int j = 0; j < matrix[i].length; j++) {
                                                           rotated[j][n - 1 - i] = matrix[i][j];
         if (i \le j) {
            System.out.print(matrix[i][j] + " ");
                                                      }
                                                      System.out.println("Rotated Matrix:");
            System.out.print("0");
                                                      for (int[] row : rotated) {
         }
                                                        System.out.println(Arrays.toString(row));
       }
       System.out.println();
                                                      Output:
                                                      csharp
     Output:
                                                      Copy code
     yaml
                                                      Rotated Matrix:
     Copy code
                                                      [7, 4, 1]
     Upper Triangular Matrix:
                                                      [8, 5, 2]
     123
                                                      [9, 6, 3]
     056
                                                 93. Find the Sum of Each Diagonal in a Square Matrix
     009
                                                      java
45. Print the Lower Triangular Matrix
                                                      Copy code
     java
                                                      int[][] matrix = {
     Copy code
                                                        {1, 2, 3},
     int[][] matrix = {
                                                        \{4, 5, 6\},\
       {1, 2, 3},
                                                        \{7, 8, 9\}
       {4, 5, 6},
                                                      };
       \{7, 8, 9\}
                                                      int primaryDiagonalSum = 0;
                                                      int secondaryDiagonalSum = 0;
     System.out.println("Lower Triangular
                                                      for (int i = 0; i < matrix.length; i++) {
     Matrix:");
                                                        primaryDiagonalSum += matrix[i][i];
     for (int i = 0; i < matrix.length; i++) {
                                                        secondaryDiagonalSum += matrix[i][matrix.length - 1 -
```

```
System.out.printing Lower mangular
                                                    for (int i = 0; i < matrix.length; i++) {
    Matrix:");
                                                       primaryDiagonalSum += matrix[i][i];
    for (int i = 0; i < matrix.length; i++) {
                                                       secondaryDiagonalSum += matrix[i][matrix.length - 1 -
       for (int j = 0; j < matrix[i].length; j++) {
                                                    i];
         if (i >= i) {
                                                    }
           System.out.print(matrix[i][j] + " ");
                                                    System.out.println("Primary Diagonal Sum: " +
                                                     primaryDiagonalSum);
           System.out.print("0");
                                                    System.out.println("Secondary Diagonal Sum: " +
         }
                                                    secondaryDiagonalSum);
       }
                                                    Output:
       System.out.println();
                                                    mathematica
                                                    Copy code
    Output:
                                                    Primary Diagonal Sum: 15
    yaml
                                                    Secondary Diagonal Sum: 15
    Copy code
                                               94.
                                                    Check if a Matrix is Symmetric
    Lower Triangular Matrix:
    100
                                                    java
    450
                                                    Copy code
    789
                                                    int[][] matrix = {
46. Find the Sum of Each Row in a 2D Matrix
                                                       \{1, 2, 3\},\
                                                       \{2, 4, 5\},\
    java
                                                       {3, 5, 6}
    Copy code
                                                    };
    int[][] matrix = {
                                                    boolean isSymmetric = true;
       {1, 2, 3},
       {4, 5, 6},
                                                    for (int i = 0; i < matrix.length; i++) {
       \{7, 8, 9\}
                                                       for (int j = 0; j < matrix[i].length; j++) {
    };
                                                         if (matrix[i][j] != matrix[j][i]) {
    System.out.println("Sum of Each Row:");
                                                           isSymmetric = false;
    for (int[] row : matrix) {
                                                           break;
       int sum = 0;
                                                         }
       for (int num : row) {
                                                       }
         sum += num;
                                                       if (!isSymmetric) break;
       }
       System.out.println(sum);
    }
                                                    System.out.println("Matrix is Symmetric: " +
    Output:
                                                    isSymmetric);
    mathematica
    Copy code
                                                    Output: Matrix is Symmetric: true
    Sum of Each Row:
                                                    Find the Row with the Maximum Sum in a Matrix
    6
    15
                                                    java
    24
                                                    Copy code
47. Find the Sum of Each Column in a 2D
                                                    int[][] matrix = {
    Matrix
                                                       {1, 2, 3},
    java
                                                       {4, 5, 6},
    Copy code
                                                       \{7, 8, 9\}
    int[][] matrix = {
       \{1, 2, 3\},\
                                                    int maxRowSum = Integer.MIN VALUE;
       {4, 5, 6},
                                                    int rowIndex = -1;
       \{7, 8, 9\}
    };
                                                    for (int i = 0; i < matrix.length; i++) {
    System.out.println("Sum of Each Column:");
                                                       int sum = 0;
```

```
(,, ,, ,,
    };
                                                   for (int i = 0; i < matrix.length; i++) {
    System.out.println("Sum of Each Column:");
                                                     int sum = 0;
    for (int j = 0; j < matrix[0].length; j++) {
                                                     for (int num: matrix[i]) {
       int sum = 0;
                                                        sum += num;
       for (int i = 0; i < matrix.length; i++) {
         sum += matrix[i][j];
                                                     if (sum > maxRowSum) {
                                                        maxRowSum = sum;
       System.out.println(sum);
                                                        rowIndex = i;
    }
                                                     }
    Output:
                                                   }
    mathematica
    Copy code
                                                   System.out.println("Row with Maximum Sum: " +
    Sum of Each Column:
                                                   rowIndex);
    12
                                                   System.out.println("Maximum Sum: " + maxRowSum);
    15
    18
                                                   Output:
48. Find the Saddle Point of a Matrix
    java
                                                   mathematica
    Copy code
                                                   Copy code
    int[][] matrix = {
                                                   Row with Maximum Sum: 2
       \{3, 8, 7\},\
                                                   Maximum Sum: 24
       \{4, 5, 6\},\
                                              96.
                                                   Find the Column with the Maximum Sum in a Matrix
       \{1, 2, 9\}
                                                   java
    };
                                                   Copy code
    boolean foundSaddlePoint = false;
                                                   int[][] matrix = {
    for (int i = 0; i < matrix.length; i++) {
                                                     \{1, 2, 3\},\
       int minRow = matrix[i][0];
                                                     \{4, 5, 6\},\
       int minColIndex = 0;
                                                     {7, 8, 9}
       for (int j = 1; j < matrix[i].length; j++) {
                                                   };
         if (matrix[i][j] < minRow) {</pre>
                                                   int maxColumnSum = Integer.MIN VALUE;
           minRow = matrix[i][j];
                                                   int columnIndex = -1;
           minColIndex = j;
                                                   for (int j = 0; j < matrix[0].length; j++) {
         }
                                                     int sum = 0;
       }
                                                     for (int i = 0; i < matrix.length; i++) {
       int maxCol = minRow;
                                                        sum += matrix[i][j];
       for (int k = 0; k < matrix.length; k++) {
                                                     }
         if (matrix[k][minColIndex] > maxCol) {
                                                     if (sum > maxColumnSum) {
           maxCol = matrix[k][minColIndex];
                                                        maxColumnSum = sum;
         }
                                                        columnIndex = j;
       }
                                                     }
       if (maxCol == minRow) {
         System.out.println("Saddle Point: " +
                                                   System.out.println("Column with Maximum Sum: " +
    maxCol);
                                                   columnIndex);
         foundSaddlePoint = true;
                                                   System.out.println("Maximum Sum: " + maxColumnSum);
         break;
                                                   Output:
      }
                                                   mathematica
    }
                                                   Copy code
    if (!foundSaddlePoint) {
                                                   Column with Maximum Sum: 2
       System.out.println("No Saddle Point
                                                   Maximum Sum: 18
    Found.");
                                              97. Find the Saddle Point in a Matrix
    }
                                                   iava
                                                  Copy code
```

```
97. | Find the Saddle Point in a Watrix
                                                      java
                                                      Copy code
     Output: Saddle Point: 5
                                                      int[][] matrix = {
49. Find the Maximum Value in Each Row of a
                                                        {3, 8, 7},
     Matrix
                                                        {5, 6, 4},
    java
                                                        \{1, 2, 9\}
     Copy code
                                                      };
     int[][] matrix = {
                                                      for (int i = 0; i < matrix.length; i++) {
       {1, 2, 3},
                                                        int rowMin = matrix[i][0];
       {4, 5, 6},
                                                        int collndex = 0;
       {7, 8, 9}
                                                        for (int j = 1; j < matrix[i].length; j++) {
     };
                                                           if (matrix[i][j] < rowMin) {</pre>
     System.out.println("Maximum Value in Each
                                                             rowMin = matrix[i][j];
     Row:");
                                                             colIndex = j;
     for (int[] row : matrix) {
                                                           }
       int max = row[0];
                                                        }
       for (int num : row) {
                                                        boolean isSaddlePoint = true;
         max = Math.max(max, num);
                                                        for (int k = 0; k < matrix.length; k++) {
                                                           if (matrix[k][colIndex] > rowMin) {
       System.out.println(max);
                                                             isSaddlePoint = false;
                                                             break;
     Output:
                                                           }
     sql
                                                        }
     Copy code
     Maximum Value in Each Row:
                                                        if (isSaddlePoint) {
                                                           System.out.println("Saddle Point: " + rowMin);
     6
                                                        }
50. Find the Minimum Value in Each Column of
                                                      Output: Saddle Point: 4
     a Matrix
                                                 98.
                                                      Count the Number of Islands in a Binary Matrix
    java
                                                      java
     Copy code
                                                      Copy code
     int[][] matrix = {
                                                      int[][] matrix = {
       {1, 2, 3},
                                                        \{1, 1, 0, 0, 0\}
       {4, 5, 6},
                                                        \{1, 1, 0, 0, 1\},\
       \{7, 8, 9\}
                                                        \{0, 0, 0, 1, 1\},\
     };
                                                        \{0, 0, 0, 0, 0, 0\},\
     System.out.println("Minimum Value in Each
                                                        {1, 0, 1, 0, 1}
     Column:");
                                                      };
     for (int j = 0; j < matrix[0].length; j++) {
                                                      boolean[][] visited = new boolean[matrix.length]
       int min = matrix[0][j];
                                                      [matrix[0].length];
       for (int i = 0; i < matrix.length; i++) {
                                                      int[] rowNum = {-1, 0, 0, 1};
         min = Math.min(min, matrix[i][j]);
                                                      int[] colNum = {0, -1, 1, 0};
       }
                                                      int islandCount = 0;
       System.out.println(min);
                                                      for (int i = 0; i < matrix.length; i++) {
     Output:
                                                        for (int j = 0; j < matrix[0].length; j++) {
     sql
                                                           if (matrix[i][j] == 1 && !visited[i][j]) {
     Copy code
                                                             markIsland(matrix, visited, i, j, rowNum, colNum);
     Minimum Value in Each Column:
                                                             islandCount++;
     1
                                                           }
     2
                                                        }
```

```
isianucountii,
     1
     2
                                                       }
     3
                                                     }
51. Find the Number of Positive and Negative
                                                     System.out.println("Number of Islands: " + islandCount);
     Numbers in a Matrix
                                                     static void markIsland(int[][] matrix, boolean[][] visited,
     java
                                                     int row, int col, int[] rowNum, int[] colNum) {
     Copy code
                                                       int n = matrix.length;
     int[][] matrix = {
                                                       int m = matrix[0].length;
       \{1, -2, 3\},\
                                                       visited[row][col] = true;
       \{-4, 5, -6\},\
                                                       for (int k = 0; k < 4; k++) {
       \{7, -8, 9\}
                                                          int newRow = row + rowNum[k];
     };
                                                          int newCol = col + colNum[k];
     int positiveCount = 0;
                                                          if (isSafe(matrix, visited, newRow, newCol, n, m)) {
     int negativeCount = 0;
                                                            markIsland(matrix, visited, newRow, newCol,
     for (int[] row : matrix) {
                                                     rowNum, colNum);
       for (int num : row) {
                                                         }
         if (num > 0) positiveCount++;
                                                       }
         else if (num < 0) negativeCount++;
       }
                                                     static boolean isSafe(int[][] matrix, boolean[][] visited, int
     }
                                                     row, int col, int n, int m) {
     System.out.println("Number of Positive
                                                       return (row >= 0) && (row < n) &&
     Numbers: " + positiveCount);
                                                           (col >= 0) \&\& (col < m) \&\&
     System.out.println("Number of Negative
                                                           (matrix[row][col] == 1 && !visited[row][col]);
     Numbers: " + negativeCount);
     Output:
                                                     Output: Number of Islands: 4
     mathematica
                                                     Find the Longest Common Prefix in 2D Matrix Strings
     Copy code
                                                     java
     Number of Positive Numbers: 6
                                                     Copy code
     Number of Negative Numbers: 4
                                                     String[][] matrix = {
52. Check if a Matrix is an Identity Matrix
                                                       {"flower", "flow", "flight"},
    java
                                                       {"dog", "racecar", "car"}
     Copy code
                                                     };
     int[][] matrix = {
                                                     for (String[] row : matrix) {
       \{1, 0, 0\},\
                                                       String prefix = row[0];
       \{0, 1, 0\},\
                                                       for (int i = 1; i < row.length; i++) {
       \{0, 0, 1\}
                                                          while (row[i].indexOf(prefix) != 0) {
     };
                                                            prefix = prefix.substring(0, prefix.length() - 1);
     boolean isIdentity = true;
                                                            if (prefix.isEmpty()) {
     for (int i = 0; i < matrix.length; i++) {
                                                       System.out.println("Longest Common Prefix: " + prefix);
       for (int j = 0; j < matrix[i].length; j++) {
                                                              break;
         if (i == j && matrix[i][j] != 1 || i != j &&
                                                            }
     matrix[i][j] != 0) {
                                                          }
           isIdentity = false;
           break;
                                                       System.out.println("Longest Common Prefix: " + prefix);
         }
       }
                                                     Output:
       if (!isIdentity) break;
                                                     mathematica
                                                     Copy code
     System.out.println("Matrix is Identity: " +
                                                     Longest Common Prefix: fl
     isIdentity);
                                                     Longest Common Prefix:
                                               100. Find the Maximum Sum Subarray in a 2D Matrix "java
     Output: Matrix is Identity: true
                                                     int[][] matrix = { {1, -2, -1, 4}, {2, -1, 2, 1}, {1, -3, 4, -2}, {1,
```

```
100.
                                                    Find the Maximum Sum Subarray in a 2D Matrix ```java
    Output: Matrix is Identity: true
                                                    int[][] matrix = { {1, -2, -1, 4}, {2, -1, 2, 1}, {1, -3, 4, -2}, {1,
53. Transpose a Matrix
                                                     2, 2, 1} }; int maxSum = Integer.MIN VALUE;
    java
                                                    CSS
    Copy code
                                                     Copy code
    int[][] matrix = {
                                                    for (int left = 0; left < matrix[0].length; left++) {
       {1, 2, 3},
                                                    sql
       {4, 5, 6},
                                                     Copy code
       \{7, 8, 9\}
                                                    int[] temp = new int[matrix.length];
    };
                                                       for (int right = left; right < matrix[0].length; right++) {
    int[][] transpose = new int[matrix[0].length]
                                                         for (int i = 0; i < matrix.length; i++) {
    [matrix.length];
                                                            temp[i] += matrix[i][right];
    for (int i = 0; i < matrix.length; i++) {
                                                         }
       for (int j = 0; j < matrix[i].length; j++) {
         transpose[j][i] = matrix[i][j];
                                                         int currentSum = kadane(temp);
       }
                                                         maxSum = Math.max(maxSum, currentSum);
    }
                                                       }
    System.out.println("Transpose of Matrix:");
    for (int[] row : transpose) {
                                                     System.out.println("Maximum Sum Subarray: " +
       System.out.println(Arrays.toString(row));
                                                     maxSum);
    }
                                                    static int kadane(int[] arr) {
    Output:
                                                       int maxSoFar = Integer.MIN VALUE;
    csharp
                                                       int maxEndingHere = 0;
    Copy code
    Transpose of Matrix:
                                                       for (int num: arr) {
    [1, 4, 7]
                                                         maxEndingHere += num;
    [2, 5, 8]
                                                         maxSoFar = Math.max(maxSoFar, maxEndingHere);
    [3, 6, 9]
                                                         if (maxEndingHere < 0) {
54. Flatten a 2D Matrix to a 1D Array
                                                            maxEndingHere = 0;
    java
                                                         }
    Copy code
                                                       }
    int[][] matrix = {
       {1, 2, 3},
                                                       return maxSoFar;
       {4, 5, 6},
                                                    }
       \{7, 8, 9\}
                                                     **Output:** `Maximum Sum Subarray: 12`
    int[] flatArray = new int[matrix.length *
    matrix[0].length];
    int index = 0;
    for (int[] row : matrix) {
       for (int num : row) {
         flatArray[index++] = num;
       }
    System.out.println("Flattened Array: " +
    Arrays.toString(flatArray));
    Output: Flattened Array: [1, 2, 3, 4, 5, 6, 7,
    8, 9]
              Level 1
```

Solve

Problems

Find a peak element which is not smaller than its neighbors	<u>Solve</u>
Find the minimum and maximum element in an array	Solve
Write a program to reverse the array	Solve
Write a program to sort the given array	<u>Solve</u>
Find the Kth largest and Kth smallest number in an array	<u>Solve</u>
Find the occurrence of an integer in the array	<u>Solve</u>
Sort the array of 0s, 1s, and 2s	<u>Solve</u>
Subarray with given Sum	<u>Solve</u>
Move all the negative elements to one side of the array	<u>Solve</u>
Find the Union and Intersection of the two sorted arrays	Solve

Level 2

☐ Problems	Solve
Write a program to cyclically rotate an array by one	<u>Solve</u>
Find the missing integer	<u>Solve</u>
Count Pairs with the given sum	Solve
Find duplicates in an array	<u>Solve</u>
Sort an Array using the Quicksort algorithm	<u>Solve</u>
Find common elements in three sorted arrays	<u>Solve</u>
Find the first repeating element in an array of integers	<u>Solve</u>
Find the first non-repeating element in a given array of integers	<u>Solve</u>
Subarrays with equal 1s and 0s	<u>Solve</u>
Rearrange the array in alternating positive and negative items	<u>Solve</u>
Find if there is any subarray with a sum equal to zero	Solve
Find the Largest sum contiguous Subarray	Solve
Find the factorial of a large number	Solve
Find Maximum Product Subarray	<u>Solve</u>
Find the longest consecutive subsequence	Solve
Find the minimum element in a rotated and sorted array	Solve
Max sum in the configuration	Solve
Minimum Platforms	Solve
Minimize the maximum difference between the heights	Solve
Minimum number of jumps to reach the end	Solve
Stock Span problem	Solve
Find a triplet that sums to a given value	<u>Solve</u>
Smallest positive missing number	Solve
Find the row with a maximum number of 1's	Solve
Print the matrix in a Spiral manner	<u>Solve</u>

Find whether an array is a subset of another array	Solve Solve
Implement two Stacks in an array	Solve
Majority Element	Solve
Wave Array	Solve
Trapping Rainwater	Solve

Level 3

☐ Problems	Solve
Maximum Index	Solve
Max sum path in two arrays	Solve
Find Missing And Repeating	Solve
Stock buy and sell Problem	<u>Solve</u>
Pair with the given sum in a sorted array	<u>Solve</u>
Chocolate Distribution Problem	<u>Solve</u>
Partition Equal Subset Sum	<u>Solve</u>
Smallest Positive integer that can't be represented as a sum	<u>Solve</u>
Coin Change Problem	Solve_
Longest Alternating subsequence	<u>Solve</u>



