

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

```
java
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

```
java
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

7

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:

Copy code

0

1. Find the Maximum Element in an Array

```
java
Copy code
int[] arr = {3, 5, 7, 2, 8};
```

55. 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]

56. 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}
```

java

Copy code

```
int[] arr = {3, 5, 7, 2, 8};
int max = arr[0];
for (int i = 1; i < arr.length; i++) {
    if (arr[i] > max) {
        max = arr[i];
    }
}
```

System.out.println("Max: " + max);

Output: Max: 8

2. Find the Minimum Element in an Array

java

Copy code

```
int[] arr = {3, 5, 7, 2, 8};
int min = arr[0];
for (int i = 1; i < arr.length; i++) {
    if (arr[i] < min) {
        min = arr[i];
    }
}
```

System.out.println("Min: " + min);

Output: Min: 2

3. Calculate the Sum of All Elements in an Array

java

Copy code

```
int[] arr = {1, 2, 3, 4, 5};
int sum = 0;
for (int num : arr) {
    sum += num;
}
```

System.out.println("Sum: " + sum);

Output: Sum: 15

4. Calculate the Average of Elements in an Array

java

Copy code

```
int[] arr = {1, 2, 3, 4, 5};
int sum = 0;
for (int num : arr) {
    sum += num;
}
```

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

System.out.println("Average: " + average);

Output: Average: 3.0

5. Count Occurrences of a Specific Element

java

Copy code

```
int[] arr = {1, 2, 3, 2, 2, 4};
int count = 0;
for (int num : arr) {
    if (num == 2) {
```

```
{1, 0, 1, 0},
```

```
{1, 1, 1, 1},
```

```
{1, 1, 1, 0}
```

```
};
```

```
int maxArea = 0;
```

```
int[] height = new int[matrix[0].length];
```

```
for (int i = 0; i < matrix.length; i++) {
```

```
    for (int j = 0; j < matrix[0].length; j++) {
```

```
        height[j] = matrix[i][j] == 0 ? 0 : height[j] + 1;
```

```
    }
```

```
    int area = calculateMaxArea(height);
```

```
    maxArea = Math.max(maxArea, area);
```

```
}
```

```
System.out.println("Largest Rectangle Area: " + maxArea);
```

```
private static int calculateMaxArea(int[] height) {
```

```
    int maxArea = 0;
```

```
    Stack<Integer> stack = new Stack<>();
```

```
    int index = 0;
```

```
    while (index < height.length) {
```

```
        if (stack.isEmpty() || height[index] >=
```

```
height[stack.peek()]) {
```

```
            stack.push(index++);
```

```
        } else {
```

```
            int topOfStack = stack.pop();
```

```
            int area = height[topOfStack] * (stack.isEmpty() ?
```

```
index : index - stack.peek() - 1);
```

```
            maxArea = Math.max(maxArea, area);
```

```
        }
```

```
    }
```

```
    while (!stack.isEmpty()) {
```

```
        int topOfStack = stack.pop();
```

```
        int area = height[topOfStack] * (stack.isEmpty() ?
```

```
index : index - stack.peek() - 1);
```

```
        maxArea = Math.max(maxArea, area);
```

```
    }
```

```
    return maxArea;
```

```
}
```

Output: Largest Rectangle Area: 8

58. Print the Diagonals of a Matrix

java

Copy code

```
int[][] matrix = {
```

```
    {1, 2, 3},
```

```
    {4, 5, 6},
```

```
    {7, 8, 9}
```

```
};
```

```
System.out.println("Primary Diagonal:");
```

```
for (int i = 0; i < matrix.length; i++) {
```

```
    System.out.print(matrix[i][i] + " ");
```

```
}
```

```
System.out.println("\nSecondary Diagonal:");
```

```
for (int i = 0; i < matrix.length; i++) {
```

```

for (int num : arr) {
    if (num == 2) {
        count++;
    }
}
System.out.println("Occurrences of 2: " +
count);

```

Output: Occurrences of 2: 3

6. Reverse an Array

```

java
Copy code
int[] arr = {1, 2, 3, 4, 5};
for (int i = 0; i < arr.length / 2; i++) {
    int temp = arr[i];
    arr[i] = arr[arr.length - 1 - i];
    arr[arr.length - 1 - i] = temp;
}
System.out.println("Reversed Array: " +
Arrays.toString(arr));

```

Output: Reversed Array: [5, 4, 3, 2, 1]

7. Check if an Array is Sorted in Ascending Order

```

java
Copy code
int[] arr = {1, 2, 3, 4, 5};
boolean isSorted = true;
for (int i = 1; i < arr.length; i++) {
    if (arr[i] < arr[i - 1]) {
        isSorted = false;
        break;
    }
}
System.out.println("Is Sorted: " + isSorted);

```

Output: Is Sorted: true

8. Find the Index of a Specific Element

```

java
Copy code
int[] arr = {1, 2, 3, 4, 5};
int target = 3;
int index = -1;
for (int i = 0; i < arr.length; i++) {
    if (arr[i] == target) {
        index = i;
        break;
    }
}
System.out.println("Index of 3: " + index);

```

Output: Index of 3: 2

9. Remove Duplicates from an Array

```

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

```

```

}
System.out.println("\nSecondary Diagonal:");
for (int i = 0; i < matrix.length; i++) {
    System.out.print(matrix[i][matrix.length - 1 - i] + " ");
}

```

Output:

mathematica

Copy code

Primary Diagonal:

1 5 9

Secondary Diagonal:

3 5 7

59. Check if a 2D Array is a Submatrix of Another 2D Array

```

java
Copy code
int[][] matrix = {
    {1, 2, 3, 4},
    {5, 6, 7, 8},
    {9, 10, 11, 12},
    {13, 14, 15, 16}
};
int[][] submatrix = {
    {6, 7},
    {10, 11}
};
boolean found = false;
for (int i = 0; i <= matrix.length - submatrix.length; i++) {
    for (int j = 0; j <= matrix[0].length - submatrix[0].length;
j++) {
        boolean match = true;
        for (int x = 0; x < submatrix.length; x++) {
            for (int y = 0; y < submatrix[0].length; y++) {
                if (matrix[i + x][j + y] != submatrix[x][y]) {
                    match = false;
                    break;
                }
            }
            if (!match) break;
        }
        if (match) {
            found = true;
            break;
        }
    }
    if (found) break;
}
System.out.println("Submatrix Found: " + found);

```

Output: Submatrix Found: true

60. Multiply Each Element of a Matrix by a Scalar

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},

```

Copy code

```
int[] arr = {1, 2, 2, 3, 4, 4, 5};
Set<Integer> set = new LinkedHashSet<>();
for (int num : arr) {
    set.add(num);
}
int[] result =
set.stream().mapToInt(Integer::intValue).to
Array();
System.out.println("Array without
duplicates: " + Arrays.toString(result));
```

Output: Array without duplicates: [1, 2, 3, 4, 5]

10. Rotate an Array to the Right by k Positions

java

Copy code

```
int[] arr = {1, 2, 3, 4, 5};
int k = 2;
int n = arr.length;
k = k % n;
int[] result = new int[n];
for (int i = 0; i < n; i++) {
    result[(i + k) % n] = arr[i];
}
System.out.println("Rotated Array: " +
Arrays.toString(result));
```

Output: Rotated Array: [4, 5, 1, 2, 3]

2. Intermediate Array Problems

11. Find the Second Largest Element

java

Copy code

```
int[] arr = {10, 20, 4, 45, 99};
int largest = Integer.MIN_VALUE;
int secondLargest = Integer.MIN_VALUE;
for (int num : arr) {
    if (num > largest) {
        secondLargest = largest;
        largest = num;
    } else if (num > secondLargest && num
< largest) {
        secondLargest = num;
    }
}
System.out.println("Second Largest: " +
secondLargest);
```

Output: Second Largest: 45

12. Find the Missing Number in a Consecutive Array

java

Copy code

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

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int scalar = 3;
int[][] result = new int[matrix.length][matrix[0].length];
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        result[i][j] = matrix[i][j] * scalar;
    }
}
System.out.println("Matrix after Scalar Multiplication:");
for (int[] row : result) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

Matrix after Scalar Multiplication:

```
[3, 6, 9]
[12, 15, 18]
[21, 24, 27]
```

61. Sort Each Row of a Matrix in Ascending Order

java

Copy code

```
int[][] matrix = {
    {3, 1, 2},
    {6, 4, 5},
    {9, 7, 8}
};
for (int[] row : matrix) {
    Arrays.sort(row);
}
System.out.println("Matrix with Sorted Rows:");
for (int[] row : matrix) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

Matrix with Sorted Rows:

```
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
```

62. Print a 2D Array in Spiral Order

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
```

Copy code

```
int[] arr = {1, 2, 3, 5};
int n = arr.length + 1;
int sum = n * (n + 1) / 2;
for (int num : arr) {
    sum -= num;
}
System.out.println("Missing Number: " +
sum);
```

Output: Missing Number: 4

13. Find the Majority Element (Element that Appears More than N/2 Times)

java

Copy code

```
int[] arr = {3, 3, 4, 2, 4, 4, 2, 4, 4};
int candidate = arr[0];
int count = 1;
for (int i = 1; i < arr.length; i++) {
    if (arr[i] == candidate) {
        count++;
    } else {
        count--;
        if (count == 0) {
            candidate = arr[i];
            count = 1;
        }
    }
}
count = 0;
for (int num : arr) {
    if (num == candidate) {
        count++;
    }
}
if (count > arr.length / 2) {
    System.out.println("Majority Element: " +
candidate);
} else {
    System.out.println("No Majority
Element");
}
```

Output: Majority Element: 4

14. Find the Intersection of Two Arrays

java

Copy code

```
int[] arr1 = {1, 2, 2, 1};
int[] arr2 = {2, 2};
Set<Integer> set1 = new HashSet<>();
for (int num : arr1) {
    set1.add(num);
}
Set<Integer> result = new HashSet<>();
for (int num : arr2) {
```

```
int[][] matrix = {
```

```
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int top = 0, bottom = matrix.length - 1, left = 0, right =
matrix[0].length - 1;
while (top <= bottom && left <= right) {
    for (int i = left; i <= right; i++) {
        System.out.print(matrix[top][i] + " ");
    }
    top++;
    for (int i = top; i <= bottom; i++) {
        System.out.print(matrix[i][right] + " ");
    }
    right--;
    if (top <= bottom) {
        for (int i = right; i >= left; i--) {
            System.out.print(matrix[bottom][i] + " ");
        }
        bottom--;
    }
    if (left <= right) {
        for (int i = bottom; i >= top; i--) {
            System.out.print(matrix[i][left] + " ");
        }
        left++;
    }
}
```

Output: 1 2 3 6 9 8 7 4 5

63. Generate a 2D Matrix with Random Numbers

java

Copy code

```
int rows = 3, cols = 3;
int[][] matrix = new int[rows][cols];
Random rand = new Random();
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        matrix[i][j] = rand.nextInt(100); // Random number
between 0 and 99
    }
}
System.out.println("Random Matrix:");
for (int[] row : matrix) {
    System.out.println(Arrays.toString(row));
}
```

Output: (Example)

csharp

Copy code

Random Matrix:
[23, 56, 12]

```

}
Set<Integer> result = new HashSet<>();
for (int num : arr2) {
    if (set1.contains(num)) {
        result.add(num);
    }
}
System.out.println("Intersection: " + result);
Output: Intersection: [2]
15. Find the Union of Two Arrays
java
Copy code
int[] arr1 = {1, 2, 2, 1};
int[] arr2 = {2, 3, 4};
Set<Integer> result = new HashSet<>();
for (int num : arr1) {
    result.add(num);
}
for (int num : arr2) {
    result.add(num);
}
System.out.println("Union: " + result);
Output: Union: [1, 2, 3, 4]
16. Find the Longest Consecutive Sequence
java
Copy code
int[] arr = {100, 4, 200, 1, 3, 2};
Set<Integer> set = new HashSet<>();
for (int num : arr) {
    set.add(num);
}
int longest = 0;
for (int num : arr) {
    if (!set.contains(num - 1)) {
        int currentNum = num;
        int currentStreak = 1;
        while (set.contains(currentNum + 1)) {
            currentNum += 1;
            currentStreak += 1;
        }
        longest = Math.max(longest, currentStreak);
    }
}
System.out.println("Longest Consecutive Sequence: " + longest);
Output: Longest Consecutive Sequence: 4
17. Find Duplicate Elements in an Array
java
Copy code
int[] arr = {1, 2, 3, 4, 5, 1, 2};
Set<Integer> set = new HashSet<>();
Set<Integer> duplicates = new HashSet<>();

```

64. Sum of Diagonals in a Square Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int primaryDiagonalSum = 0, secondaryDiagonalSum = 0;
for (int i = 0; i < matrix.length; i++) {
    primaryDiagonalSum += matrix[i][i];
    secondaryDiagonalSum += matrix[i][matrix.length - 1 - i];
}
System.out.println("Primary Diagonal Sum: " + primaryDiagonalSum);
System.out.println("Secondary Diagonal Sum: " + secondaryDiagonalSum);
Output:

```

mathematica

Copy code
Primary Diagonal Sum: 15
Secondary Diagonal Sum: 15

65. Generate a Spiral Matrix

```

java
Copy code
int n = 4;
int[][] matrix = new int[n][n];
int value = 1;
int top = 0, bottom = n - 1, left = 0, right = n - 1;

while (top <= bottom && left <= right) {
    for (int i = left; i <= right; i++) {
        matrix[top][i] = value++;
    }
    top++;
    for (int i = top; i <= bottom; i++) {
        matrix[i][right] = value++;
    }
    right--;
    if (top <= bottom) {
        for (int i = right; i >= left; i--) {
            matrix[bottom][i] = value++;
        }
        bottom--;
    }
    if (left <= right) {
        for (int i = bottom; i >= top; i--) {
            matrix[i][left] = value++;
        }
        left++;
    }
}

```

```
int[] arr = {1, 2, 3, 4, 5, 1, 2};
Set<Integer> set = new HashSet<>();
Set<Integer> duplicates = new HashSet<>();
for (int num : arr) {
    if (!set.add(num)) {
        duplicates.add(num);
    }
}
```

System.out.println("Duplicates: " + duplicates);

Output: Duplicates: [1, 2]

18. Rotate a 2D Array (Matrix) by 90 Degrees

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int n = matrix.length;
for (int i = 0; i < n; i++) {
    for (int j = i; j < n; j++) {
        int temp = matrix[i][j];
        matrix[i][j] = matrix[j][i];
        matrix[j][i] = temp;
    }
}
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n / 2; j++) {
        int temp = matrix[i][j];
        matrix[i][j] = matrix[i][n - 1 - j];
        matrix[i][n - 1 - j] = temp;
    }
}
```

System.out.println("Rotated Matrix: ");

```
for (int[] row : matrix) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

Rotated Matrix:

[7, 4, 1]

[8, 5, 2]

[9, 6, 3]

19. Find the Largest Sum of Contiguous Subarray (Kadane's Algorithm)

java

Copy code

```
int[] arr = {-2, 1, -3, 4, -1, 2, 1, -5, 4};
int maxCurrent = arr[0];
int maxGlobal = arr[0];
for (int i = 1; i < arr.length; i++) {
```

```
    if (left <= right) {
        for (int i = bottom; i >= top; i--) {
            matrix[i][left] = value++;
        }
        left++;
    }
}
System.out.println("Spiral Matrix:");
for (int[] row : matrix) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

Spiral Matrix:

[1, 2, 3, 4]

[12, 13, 14, 5]

[11, 16, 15, 6]

[10, 9, 8, 7]

66. Count Occurrences of Each Element in a Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 2},
    {3, 1, 1},
    {4, 5, 1}
};
Map<Integer, Integer> countMap = new HashMap<>();
for (int[] row : matrix) {
    for (int num : row) {
        countMap.put(num, countMap.getOrDefault(num, 0)
+ 1);
    }
}
System.out.println("Element Occurrences: " + countMap);
```

Output: Element Occurrences: {1=5, 2=2, 3=1, 4=1, 5=1}

Print Matrix Elements in Anti-Clockwise Spiral Order

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int top = 0, bottom = matrix.length - 1, left = 0, right =
matrix[0].length - 1;
while (top <= bottom && left <= right) {
    for (int i = top; i <= bottom; i++) {
        System.out.print(matrix[i][left] + " ");
    }
    left++;
    for (int i = left; i <= right; i++) {
```

```
int maxCurrent = arr[0];
int maxGlobal = arr[0];
for (int i = 1; i < arr.length; i++) {
    maxCurrent = Math.max(arr[i],
    maxCurrent + arr[i]);
    maxGlobal = Math.max(maxGlobal,
    maxCurrent);
}
System.out.println("Maximum Subarray
Sum: " + maxGlobal);
```

Output: Maximum Subarray Sum: 6

20. Find All Pairs in an Array that Sum to a Target Value

```
java
Copy code
int[] arr = {1, 4, 5, 7, 8, 9};
int target = 8;
Map<Integer, Integer> map = new
HashMap<>();
for (int num : arr) {
    int complement = target - num;
    if (map.containsKey(complement)) {
        System.out.println("Pair: (" +
        complement + ", " + num + ")");
    }
    map.put(num, map.getOrDefault(num, 0)
    + 1);
}
```

Output: Pair: (1, 7)

3. Advanced Array Problems

21. Find the Kth Largest Element in an Array

```
java
Copy code
int[] arr = {3, 2, 1, 5, 6, 4};
int k = 2;
PriorityQueue<Integer> minHeap = new
PriorityQueue<>(k);
for (int num : arr) {
    minHeap.add(num);
    if (minHeap.size() > k) {
        minHeap.poll();
    }
}
System.out.println("Kth Largest Element: " +
    minHeap.peek());
```

Output: Kth Largest Element: 5

22. Find All Subarrays with a Sum Equal to K

```
java
Copy code
int[] arr = {1, 1, 1, 1, 1};
int k = 2;
int count = 0;
for (int start = 0; start < arr.length; start++) {
```

```
    left++;
    for (int i = left; i <= right; i++) {
        System.out.print(matrix[bottom][i] + " ");
    }
    bottom--;
    if (top <= bottom) {
        for (int i = bottom; i >= top; i--) {
            System.out.print(matrix[i][right] + " ");
        }
        right--;
    }
    if (left <= right) {
        for (int i = right; i >= left; i--) {
            System.out.print(matrix[top][i] + " ");
        }
        top++;
    }
}
```

Output: 1 4 7 8 9 6 3 2 5

68. Create a Matrix with Multiples of a Number

```
java
Copy code
int n = 3;
int multiplier = 3;
int[][] matrix = new int[n][n];
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        matrix[i][j] = multiplier * (i * n + j + 1);
    }
}
System.out.println("Matrix with Multiples of " + multiplier
    + ":");
for (int[] row : matrix) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

Matrix with Multiples of 3:

[3, 6, 9]

[12, 15, 18]

[21, 24, 27]

69. Print Matrix Elements in Zigzag Order

```
java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println("Zigzag Order:");
for (int i = 0; i < matrix.length; i++) {
    if (i % 2 == 0) {
```



```

int k = 2;
int count = 0;
for (int start = 0; start < arr.length; start++) {
    int sum = 0;
    for (int end = start; end < arr.length;
end++) {
        sum += arr[end];
        if (sum == k) {
            count++;
        }
    }
}

```

System.out.println("Count of Subarrays with Sum = " + k + ": " + count);

Output: Count of Subarrays with Sum = 2: 4

23. Merge Two Sorted Arrays

java

Copy code

```

int[] arr1 = {1, 3, 5, 7};
int[] arr2 = {2, 4, 6, 8};
int[] merged = new int[arr1.length +
arr2.length];
int i = 0, j = 0, k = 0;
while (i < arr1.length && j < arr2.length) {
    if (arr1[i] < arr2[j]) {
        merged[k++] = arr1[i++];
    } else {
        merged[k++] = arr2[j++];
    }
}
while (i < arr1.length) {
    merged[k++] = arr1[i++];
}
while (j < arr2.length) {
    merged[k++] = arr2[j++];
}
System.out.println("Merged Array: " +
Arrays.toString(merged));

```

Output: Merged Array: [1, 2, 3, 4, 5, 6, 7, 8]

24. Find the Longest Subarray with Distinct Elements

Copy code

```

int[] arr = {1, 2, 3, 4, 3, 2, 1};
Set<Integer> set = new HashSet<>();
int longest = 0;
int left = 0;
for (int right = 0; right < arr.length; right++) {
    while (set.contains(arr[right])) {
        set.remove(arr[left++]);
    }
    set.add(arr[right]);
    longest = Math.max(longest, right - left +

```

```

System.out.println("Zigzag Order:");
for (int i = 0; i < matrix.length; i++) {
    if (i % 2 == 0) {
        for (int j = 0; j < matrix[0].length; j++) {
            System.out.print(matrix[i][j] + " ");
        }
    } else {
        for (int j = matrix[0].length - 1; j >= 0; j--) {
            System.out.print(matrix[i][j] + " ");
        }
    }
}

```

Output: 1 2 3 6 5 4 7 8 9

Transpose and Reverse Rows of a Matrix

java

Copy code

```

int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int[][] transposed = new int[matrix[0].length]
[matrix.length];
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        transposed[j][i] = matrix[i][j];
    }
}
System.out.println("Transposed and Reversed Rows
Matrix:");
for (int[] row : transposed) {
    int[] reversedRow = new int[row.length];
    for (int k = 0; k < row.length; k++) {
        reversedRow[k] = row[row.length - 1 - k];
    }
    System.out.println(Arrays.toString(reversedRow));
}

```

Output:

csharp

Copy code

Transposed and Reversed Rows Matrix:

[7, 4, 1]

[8, 5, 2]

[9, 6, 3]

Sum of Border Elements in a Matrix

java

Copy code

```

int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int sum = 0;

```

```

    }
    set.add(arr[right]);
    longest = Math.max(longest, right - left +
1);
}
System.out.println("Longest Subarray with
Distinct Elements: " + longest);
Output: Longest Subarray with Distinct
Elements: 4

```

25. Find the Median of Two Sorted Arrays

```

java
Copy code
int[] nums1 = {1, 3};
int[] nums2 = {2};
int[] merged = new int[nums1.length +
nums2.length];
int i = 0, j = 0, k = 0;
while (i < nums1.length && j
< nums2.length) {
    if (nums1[i] < nums2[j]) {
        merged[k++] = nums1[i++];
    } else {
        merged[k++] = nums2[j++];
    }
}
while (i < nums1.length) {
    merged[k++] = nums1[i++];
}
while (j < nums2.length) {
    merged[k++] = nums2[j++];
}
double median;
if (merged.length % 2 == 0) {
    median = (merged[merged.length / 2 - 1]
+ merged[merged.length / 2]) / 2.0;
} else {
    median = merged[merged.length / 2];
}
System.out.println("Median: " + median);
Output: Median: 2.0

```

26. Find the Intersection of Two Arrays with Count

```

java
Copy code
int[] nums1 = {1, 2, 2, 1};
int[] nums2 = {2, 2};
Map<Integer, Integer> countMap = new
HashMap<>();
for (int num : nums1) {
    countMap.put(num,
countMap.getOrDefault(num, 0) + 1);
}
List<Integer> result = new ArrayList<>();

```

```

    {7, 8, 9}
};
int sum = 0;
for (int i = 0; i < matrix.length; i++) {
    sum += matrix[i][0] + matrix[i][matrix[i].length - 1];
}
for (int j = 1; j < matrix[0].length - 1; j++) {
    sum += matrix[0][j] + matrix[matrix.length - 1][j];
}
System.out.println("Sum of Border Elements: " + sum);

```

Output: Sum of Border Elements: 40

72. Find the Maximal Square of 1's in a Binary Matrix

```

java
Copy code
int[][] matrix = {
    {1, 0, 1, 0},
    {1, 1, 1, 1},
    {1, 1, 1, 0}
};
int maxSide = 0;
int[][] dp = new int[matrix.length][matrix[0].length];

for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[0].length; j++) {
        if (matrix[i][j] == 1) {
            if (i == 0 || j == 0) {
                dp[i][j] = 1;
            } else {
                dp[i][j] = Math.min(dp[i - 1][j], Math.min(dp[i][j -
1], dp[i - 1][j - 1])) + 1;
            }
            maxSide = Math.max(maxSide, dp[i][j]);
        }
    }
}
int maxSquareArea = maxSide * maxSide;
System.out.println("Maximal Square Area of 1's: " +
maxSquareArea);
Output: Maximal Square Area of 1's: 4

```

73. Rotate a Matrix by 90 Degrees Clockwise

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int n = matrix.length;
int[][] rotated = new int[n][n];

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

```

```

countMap.getOrDefault(num, 0) + 1),
}
List<Integer> result = new ArrayList<>();
for (int num : nums2) {
    if (countMap.containsKey(num) &&
countMap.get(num) > 0) {
        result.add(num);
        countMap.put(num,
countMap.get(num) - 1);
    }
}
System.out.println("Intersection with
Count: " + result);

```

Output: Intersection with Count: [2, 2]

27. Find the Longest Increasing Subsequence

```

java
Copy code
int[] nums = {10, 9, 2, 5, 3, 7, 101, 18};
int[] dp = new int[nums.length];
Arrays.fill(dp, 1);
for (int i = 1; i < nums.length; i++) {
    for (int j = 0; j < i; j++) {
        if (nums[i] > nums[j]) {
            dp[i] = Math.max(dp[i], dp[j] + 1);
        }
    }
}
System.out.println("Length of Longest
Increasing Subsequence: " +
Arrays.stream(dp).max().getAsInt());
Output: Length of Longest Increasing
Subsequence: 4

```

28. Find the Subarray with Maximum Sum

```

java
Copy code
int[] arr = {-2, 1, -3, 4, -1, 2, 1, -5, 4};
int maxCurrent = arr[0];
int maxGlobal = arr[0];
for (int i = 1; i < arr.length; i++) {
    maxCurrent = Math.max(arr[i],
maxCurrent + arr[i]);
    maxGlobal = Math.max(maxGlobal,
maxCurrent);
}
System.out.println("Maximum Subarray
Sum: " + maxGlobal);

```

Output: Maximum Subarray Sum: 6

29. Find the Missing Number in a 1 to N Array

```

java
Copy code
int[] arr = {1, 2, 4, 5};
int n = 5;

```

```

for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        rotated[j][n - 1 - i] = matrix[i][j];
    }
}
System.out.println("Rotated Matrix:");
for (int[] row : rotated) {
    System.out.println(Arrays.toString(row));
}

```

Output:

csharp

Copy code

Rotated Matrix:

```

[7, 4, 1]
[8, 5, 2]
[9, 6, 3]

```

74. Find the Minimum Path Sum in a Matrix (Dynamic Programming)

```

java
Copy code
int[][] matrix = {
    {1, 3, 1},
    {1, 5, 1},
    {4, 2, 1}
};
int m = matrix.length;
int n = matrix[0].length;
int[][] dp = new int[m][n];

dp[0][0] = matrix[0][0];
for (int i = 1; i < m; i++) {
    dp[i][0] = dp[i - 1][0] + matrix[i][0];
}
for (int j = 1; j < n; j++) {
    dp[0][j] = dp[0][j - 1] + matrix[0][j];
}
for (int i = 1; i < m; i++) {
    for (int j = 1; j < n; j++) {
        dp[i][j] = Math.min(dp[i - 1][j], dp[i][j - 1]) + matrix[i][j];
    }
}
System.out.println("Minimum Path Sum: " + dp[m - 1][n - 1]);

```

Output: Minimum Path Sum: 7

75. Find Common Elements in All Rows of a Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3, 4},
    {3, 4, 5, 6},
    {4, 5, 6, 7}
}

```

```
Copy code
int[] arr = {1, 2, 4, 5};
int n = 5;
int sum = n * (n + 1) / 2;
for (int num : arr) {
    sum -= num;
}
```

```
System.out.println("Missing Number: " +
sum);
```

Output: Missing Number: 3

30. Find the First Non-Repeating Character in an Array

```
java
Copy code
char[] arr = {'a', 'b', 'c', 'a', 'b', 'c', 'd'};
Map<Character, Integer> countMap = new
LinkedHashMap<>();
for (char c : arr) {
    countMap.put(c,
countMap.getOrDefault(c, 0) + 1);
}
char result = 0;
for (char c : countMap.keySet()) {
    if (countMap.get(c) == 1) {
        result = c;
        break;
    }
}
System.out.println("First Non-Repeating
Character: " + result);
```

Output: First Non-Repeating Character: d

4. 2D Arrays and Matrix Problems

31. Print a 2D Matrix

```
java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
for (int[] row : matrix) {
    System.out.println(Arrays.toString(row));
}
```

Output:

```
csharp
Copy code
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
```

32. Transpose of a Matrix

```
java
Copy code
int[][] matrix = {
```

```
{1, 2, 3, 4},
{3, 4, 5, 6},
{4, 5, 6, 7}
};
Set<Integer> commonElements = new HashSet<>();
Set<Integer> rowElements = new HashSet<>();
for (int num : matrix[0]) {
    rowElements.add(num);
}
for (int i = 1; i < matrix.length; i++) {
    Set<Integer> currentRowElements = new HashSet<>();
    for (int num : matrix[i]) {
        if (rowElements.contains(num)) {
            currentRowElements.add(num);
        }
    }
    rowElements = currentRowElements;
}
commonElements = rowElements;
System.out.println("Common Elements in All Rows: " +
commonElements);
```

Output: Common Elements in All Rows: [4]

76. Check if a Matrix is Symmetric

```
java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {2, 4, 5},
    {3, 5, 6}
};
boolean isSymmetric = true;
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        if (matrix[i][j] != matrix[j][i]) {
            isSymmetric = false;
            break;
        }
    }
    if (!isSymmetric) break;
}
System.out.println("Matrix is Symmetric: " +
isSymmetric);
```

Output: Matrix is Symmetric: true

77. Find the Median of a Matrix

```
java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
List<Integer> elements = new ArrayList<>();
for (int[] row : matrix) {
```

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int[][] transposed = new
int[matrix[0].length][matrix.length];
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[0].length; j++) {
        transposed[j][i] = matrix[i][j];
    }
}
System.out.println("Transposed Matrix: ");
for (int[] row : transposed) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

Transposed Matrix:

[1, 4, 7]

[2, 5, 8]

[3, 6, 9]

33. Print the Diagonals of a Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int n = matrix.length;
System.out.println("Primary Diagonal:");
for (int i = 0; i < n; i++) {
    System.out.print(matrix[i][i] + " ");
}
System.out.println("\nSecondary
Diagonal:");
for (int i = 0; i < n; i++) {
    System.out.print(matrix[i][n - 1 - i] + " ");
}
```

Output:

mathematica

Copy code

Primary Diagonal:

1 5 9

Secondary Diagonal:

3 5 7

34. Find the Largest Sum of Submatrix

java

Copy code

};

```
List<Integer> elements = new ArrayList<>();
for (int[] row : matrix) {
    for (int num : row) {
        elements.add(num);
    }
}
Collections.sort(elements);
int median = elements.get(elements.size() / 2);
System.out.println("Median of Matrix: " + median);
Output: Median of Matrix: 5
```

78. Print a Matrix in a Specific Pattern (e.g., Zigzag)

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
for (int i = 0; i < matrix.length; i++) {
    if (i % 2 == 0) {
        for (int j = 0; j < matrix[i].length; j++) {
            System.out.print(matrix[i][j] + " ");
        }
    } else {
        for (int j = matrix[i].length - 1; j >= 0; j--) {
            System.out.print(matrix[i][j] + " ");
        }
    }
}
```

Output: 1 2 3 6 5 4 7 8 9

79. Fill a Matrix in Diagonal Order

java

Copy code

```
int n = 4;
int[][] matrix = new int[n][n];
int value = 1;
for (int sum = 0; sum < 2 * n - 1; sum++) {
    int row = sum < n ? 0 : sum - n + 1;
    int col = sum < n ? sum : n - 1;
    while (row < n && col >= 0) {
        matrix[row][col] = value++;
        row++;
        col--;
    }
}
System.out.println("Matrix Filled in Diagonal Order:");
for (int[] row : matrix) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

```

java
Copy code
int[][] matrix = {
    {1, 2, -1},
    {-1, 2, 3},
    {1, -1, 2}
};
int maxSum = Integer.MIN_VALUE;
int[][] dp = new int[matrix.length]
[matrix[0].length];
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[0].length; j++) {
        dp[i][j] = matrix[i][j];
        if (i > 0) dp[i][j] += dp[i - 1][j];
        if (j > 0) dp[i][j] += dp[i][j - 1];
        if (i > 0 && j > 0) dp[i][j] -= dp[i - 1][j - 1];
    }
}
for (int i1 = 0; i1 < matrix.length; i1++) {
    for (int j1 = 0; j1 < matrix[0].length; j1++) {
        for (int i2 = i1; i2 < matrix.length; i2++) {
            for (int j2 = j1; j2 < matrix[0].length; j2++) {
                int sum = dp[i2][j2];
                if (i1 > 0) sum -= dp[i1 - 1][j2];
                if (j1 > 0) sum -= dp[i2][j1 - 1];
                if (i1 > 0 && j1 > 0) sum += dp[i1 - 1][j1 - 1];
                maxSum = Math.max(maxSum, sum);
            }
        }
    }
}

```

System.out.println("Largest Sum of Submatrix: " + maxSum);

Output: Largest Sum of Submatrix: 10

35. Spiral Order of a Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int m = matrix.length;
int n = matrix[0].length;
int top = 0, bottom = m - 1, left = 0, right = n - 1;
while (top <= bottom && left <= right) {
    for (int i = left; i <= right; i++) {
        System.out.print(matrix[top][i] + " ");
    }
    top++;
    for (int i = top; i <= bottom; i++) {
        System.out.print(matrix[i][right] + " ");
    }
    right--;
    for (int i = right; i >= left; i--) {
        System.out.print(matrix[bottom][i] + " ");
    }
    bottom--;
    for (int i = bottom; i >= top; i--) {
        System.out.print(matrix[i][left] + " ");
    }
    left++;
}

```

Output.

csharp

Copy code

Matrix Filled in Diagonal Order:

[1, 2, 4, 7]

[3, 5, 8, 11]

[6, 9, 12, 14]

[10, 13, 15, 16]

80. Find All Unique Elements in a Matrix

java

Copy code

```

int[][] matrix = {
    {1, 2, 2},
    {3, 1, 1},
    {4, 5, 1}
};
Set<Integer> uniqueElements = new HashSet<>();
for (int[] row : matrix) {
    for (int num : row) {
        uniqueElements.add(num);
    }
}

```

System.out.println("Unique Elements in Matrix: " + uniqueElements);

Output: Unique Elements in Matrix: [1, 2, 3, 4, 5]

4o mini

continue

ChatGPT

81. Diagonal Sum of a Matrix

java

Copy code

```

int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int primaryDiagonalSum = 0;
int secondaryDiagonalSum = 0;
for (int i = 0; i < matrix.length; i++) {
    primaryDiagonalSum += matrix[i][i];
    secondaryDiagonalSum += matrix[i][matrix.length - 1 - i];
}

```

System.out.println("Primary Diagonal Sum: " + primaryDiagonalSum);

System.out.println("Secondary Diagonal Sum: " + secondaryDiagonalSum);

Output:

mathematica

Copy code

Primary Diagonal Sum: 15

Secondary Diagonal Sum: 15

82. Flatten a 2D Matrix into a 1D Array

```

        System.out.print(matrix[top][i] + " ");
    }
    top++;
    for (int i = top; i <= bottom; i++) {
        System.out.print(matrix[i][right] + " ");
    }
    right--;
    if (top <= bottom) {
        for (int i = right; i >= left; i--) {
            System.out.print(matrix[bottom][i] + " ");
        }
        bottom--;
    }
    if (left <= right) {
        for (int i = bottom; i >= top; i--) {
            System.out.print(matrix[i][left] + " ");
        }
        left++;
    }
}

```

Output: 1 2 3 6 9 8 7 4 5

36. Count Number of Islands in a Matrix

```

java
Copy code
int[][] grid = {
    {1, 1, 0, 0, 0},
    {1, 1, 0, 0, 1},
    {0, 0, 0, 1, 1},
    {0, 0, 0, 0, 0},
    {1, 0, 1, 0, 1}
};
int numIslands = 0;
int[][] directions = {{0, 1}, {1, 0}, {0, -1}, {-1, 0}};
for (int i = 0; i < grid.length; i++) {
    for (int j = 0; j < grid[0].length; j++) {
        if (grid[i][j] == 1) {
            numIslands++;
            Queue<int[]> queue = new
LinkedList<>();
            queue.offer(new int[]{i, j});
            grid[i][j] = 0;
            while (!queue.isEmpty()) {
                int[] cell = queue.poll();
                for (int[] direction : directions) {
                    int newRow = cell[0] +
direction[0];
                    int newCol = cell[1] +
direction[1];
                    if (newRow >= 0 && newRow
< grid.length &&
newCol >= 0 && newCol

```

Primary Diagonal Sum: 15

Secondary Diagonal Sum: 15

82. Flatten a 2D Matrix into a 1D Array

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int[] flattened = new int[matrix.length * matrix[0].length];
int index = 0;
for (int[] row : matrix) {
    for (int num : row) {
        flattened[index++] = num;
    }
}
System.out.println("Flattened Array: " +
Arrays.toString(flattened));
Output: Flattened Array: [1, 2, 3, 4, 5, 6, 7, 8, 9]

```

83. Transpose a Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int[][] transposed = new int[matrix[0].length]
[matrix.length];

for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        transposed[j][i] = matrix[i][j];
    }
}
System.out.println("Transposed Matrix:");
for (int[] row : transposed) {
    System.out.println(Arrays.toString(row));
}

```

Output:

```

csharp
Copy code
Transposed Matrix:
[1, 4, 7]
[2, 5, 8]
[3, 6, 9]

```

84. Print Matrix in Spiral Order

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},

```



```

        break;
    }
}
if (found) break;
}
System.out.println("Submatrix Found: " +
found);

```

Output: Submatrix Found: true

38. Find the Boundary Elements of a Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int m = matrix.length;
int n = matrix[0].length;
System.out.println("Boundary Elements:");
for (int i = 0; i < n; i++) {
    System.out.print(matrix[0][i] + " ");
}
for (int i = 1; i < m; i++) {
    System.out.print(matrix[i][n - 1] + " ");
}
for (int i = n - 2; i >= 0; i--) {
    System.out.print(matrix[m - 1][i] + " ");
}
for (int i = m - 2; i > 0; i--) {
    System.out.print(matrix[i][0] + " ");
}

```

Output: Boundary Elements: 1 2 3 6 9 8 7 4

39. Check if a Matrix is Symmetric

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {2, 4, 5},
    {3, 5, 6}
};
boolean isSymmetric = true;
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[0].length; j++) {
        if (matrix[i][j] != matrix[j][i]) {
            isSymmetric = false;
            break;
        }
    }
}
if (!isSymmetric) break;
}
System.out.println("Matrix is Symmetric: " +
isSymmetric);

```

Output: Matrix is Identity: true

86. Find the Maximum Element in Each Row

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
for (int i = 0; i < matrix.length; i++) {
    int max = matrix[i][0];
    for (int j = 1; j < matrix[i].length; j++) {
        if (matrix[i][j] > max) {
            max = matrix[i][j];
        }
    }
    System.out.println("Maximum in Row " + i + ": " + max);
}

```

Output:

```

mathematica
Copy code
Maximum in Row 0: 3
Maximum in Row 1: 6
Maximum in Row 2: 9

```

87. Find the Minimum Element in Each Column

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
for (int j = 0; j < matrix[0].length; j++) {
    int min = matrix[0][j];
    for (int i = 1; i < matrix.length; i++) {
        if (matrix[i][j] < min) {
            min = matrix[i][j];
        }
    }
    System.out.println("Minimum in Column " + j + ": " +
min);
}

```

Output:

```

mathematica
Copy code
Minimum in Column 0: 1
Minimum in Column 1: 2
Minimum in Column 2: 3

```

88. Count the Occurrences of Each Element in a Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},

```

```

}
System.out.println("Matrix is Symmetric: " +
isSymmetric);

```

Output: Matrix is Symmetric: true

40. Find the Maximum Value in a Matrix

java

Copy code

```

int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int maxValue = matrix[0][0];
for (int[] row : matrix) {
    for (int value : row) {
        maxValue = Math.max(maxValue,
value);
    }
}
System.out.println("Maximum Value in
Matrix: " + maxValue);

```

Output: Maximum Value in Matrix: 9

4o mini

continue

ChatGPT

4. 2D Arrays and Matrix Problems (continued)

41. Rotate a Matrix 90 Degrees Clockwise

java

Copy code

```

int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int n = matrix.length;
int[][] rotated = new int[n][n];
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        rotated[j][n - 1 - i] = matrix[i][j];
    }
}
System.out.println("Rotated Matrix:");
for (int[] row : rotated) {
    System.out.println(Arrays.toString(row));
}

```

Output:

csharp

Copy code

Rotated Matrix:

[7, 4, 1]

[8, 5, 2]

[9, 6, 3]

Copy code

```

int[][] matrix = {
    {1, 2, 2},
    {3, 1, 1},
    {4, 5, 1}
};
Map<Integer, Integer> frequencyMap = new HashMap<>
();
for (int[] row : matrix) {
    for (int num : row) {
        frequencyMap.put(num,
frequencyMap.getOrDefault(num, 0) + 1);
    }
}
System.out.println("Element Frequencies: " +
frequencyMap);

```

Output: Element Frequencies: {1=4, 2=2, 3=1, 4=1, 5=1}

89. 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},
    {0, 1, 1, 0}
};
int maxArea = 0;
int[] heights = new int[matrix[0].length];
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        heights[j] = matrix[i][j] == 0 ? 0 : heights[j] + 1;
    }
    maxArea = Math.max(maxArea,
maxRectangleArea(heights));
}
System.out.println("Largest Rectangle of 1's Area: " +
maxArea);
static int maxRectangleArea(int[] heights) {
    Stack<Integer> stack = new Stack<>();
    int maxArea = 0;
    int index = 0;
    while (index < heights.length) {
        if (stack.isEmpty() || heights[index] >=
heights[stack.peek()]) {
            stack.push(index++);
        } else {
            int top = stack.pop();
            maxArea = Math.max(maxArea, heights[top] *
(stack.isEmpty() ? index : index - stack.peek() - 1));
        }
    }
    while (!stack.isEmpty()) {
        int top = stack.pop();
        maxArea = Math.max(maxArea, heights[top] *
(index - stack.isEmpty() ? index : index - stack.peek() - 1));
    }
}

```

```
[1, 4, 1]
```

```
[8, 5, 2]
```

```
[9, 6, 3]
```

42. Check if a Matrix is Magic Square

java

Copy code

```
int[][] matrix = {
    {2, 7, 6},
    {9, 5, 1},
    {4, 3, 8}
};
int sum = Arrays.stream(matrix[0]).sum();
boolean isMagic = true;
// Check rows
for (int i = 0; i < matrix.length; i++) {
    if (Arrays.stream(matrix[i]).sum() != sum)
    {
        isMagic = false;
        break;
    }
}
// Check columns
for (int j = 0; j < matrix[0].length; j++) {
    int colSum = 0;
    for (int i = 0; i < matrix.length; i++) {
        colSum += matrix[i][j];
    }
    if (colSum != sum) {
        isMagic = false;
        break;
    }
}
// Check diagonals
int diag1Sum = 0, diag2Sum = 0;
for (int i = 0; i < matrix.length; i++) {
    diag1Sum += matrix[i][i];
    diag2Sum += matrix[i][matrix.length - 1 -
i];
}
if (diag1Sum != sum || diag2Sum != sum) {
    isMagic = false;
}
System.out.println("Matrix is Magic Square:
" + isMagic);
```

Output: Matrix is Magic Square: true

43. Find the Median of a 2D Matrix

java

Copy code

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

```
}
```

```
while (!stack.isEmpty()) {
    int top = stack.pop();
    maxArea = Math.max(maxArea, heights[top] *
(stack.isEmpty() ? index : index - stack.peek() - 1));
}
```

```
return maxArea;
```

```
}
```

Output: Largest Rectangle of 1's Area: 8

90. Merge Two Sorted Arrays

java

Copy code

```
int[] arr1 = {1, 3, 5, 7};
int[] arr2 = {2, 4, 6, 8};
int[] merged = new int[arr1.length + arr2.length];
int i = 0, j = 0, k = 0;
while (i < arr1.length && j < arr2.length) {
    if (arr1[i] <= arr2[j]) {
        merged[k++] = arr1[i++];
    } else {
        merged[k++] = arr2[j++];
    }
}
while (i < arr1.length) {
    merged[k++] = arr1[i++];
}
while (j < arr2.length) {
    merged[k++] = arr2[j++];
}
System.out.println("Merged Array: " +
Arrays.toString(merged));
```

Output: Merged Array: [1, 2, 3, 4, 5, 6, 7, 8]

91. Find the Common Elements in Two 2D Arrays

java

Copy code

```
int[][] matrix1 = {
    {1, 2, 3},
    {4, 5, 6}
};
int[][] matrix2 = {
    {4, 5, 6},
    {7, 8, 9}
};
Set<Integer> set1 = new HashSet<>();
Set<Integer> commonElements = new HashSet<>();
for (int[] row : matrix1) {
    for (int num : row) {
        set1.add(num);
    }
}
```

```

    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
List<Integer> list = new ArrayList<>();
for (int[] row : matrix) {
    for (int num : row) {
        list.add(num);
    }
}
Collections.sort(list);
int median = list.get(list.size() / 2);
System.out.println("Median of Matrix: " +
    median);

```

Output: Median of Matrix: 5

44. Print the Upper Triangular Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println("Upper Triangular
Matrix:");
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        if (i <= j) {
            System.out.print(matrix[i][j] + " ");
        } else {
            System.out.print("0 ");
        }
    }
    System.out.println();
}

```

Output:

```

yaml
Copy code
Upper Triangular Matrix:
1 2 3
0 5 6
0 0 9

```

45. Print the Lower Triangular Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println("Lower Triangular
Matrix:");
for (int i = 0; i < matrix.length; i++) {

```

```

    set1.add(num);
}
}
for (int[] row : matrix2) {
    for (int num : row) {
        if (set1.contains(num)) {
            commonElements.add(num);
        }
    }
}
System.out.println("Common Elements: " +
    commonElements);

```

Output: Common Elements: [4, 5, 6]

92. Rotate a Matrix 90 Degrees Clockwise

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int n = matrix.length;
int[][] rotated = new int[n][n];

for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        rotated[j][n - 1 - i] = matrix[i][j];
    }
}
System.out.println("Rotated Matrix:");
for (int[] row : rotated) {
    System.out.println(Arrays.toString(row));
}

```

Output:

```

csharp
Copy code
Rotated Matrix:
[7, 4, 1]
[8, 5, 2]
[9, 6, 3]

```

93. Find the Sum of Each Diagonal in a Square Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int primaryDiagonalSum = 0;
int secondaryDiagonalSum = 0;
for (int i = 0; i < matrix.length; i++) {
    primaryDiagonalSum += matrix[i][i];
    secondaryDiagonalSum += matrix[i][matrix.length - 1 - i];
}

```

```
System.out.println("Lower Triangular
Matrix:");
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        if (i >= j) {
            System.out.print(matrix[i][j] + " ");
        } else {
            System.out.print("0 ");
        }
    }
    System.out.println();
}
```

Output:

yaml

Copy code

Lower Triangular Matrix:

1 0 0

4 5 0

7 8 9

46. Find the Sum of Each Row in a 2D Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println("Sum of Each Row:");
for (int[] row : matrix) {
    int sum = 0;
    for (int num : row) {
        sum += num;
    }
    System.out.println(sum);
}
```

Output:

mathematica

Copy code

Sum of Each Row:

6

15

24

47. Find the Sum of Each Column in a 2D Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println("Sum of Each Column:");
```

```
for (int i = 0; i < matrix.length; i++) {
    primaryDiagonalSum += matrix[i][i];
    secondaryDiagonalSum += matrix[i][matrix.length - 1 -
i];
}
System.out.println("Primary Diagonal Sum: " +
primaryDiagonalSum);
System.out.println("Secondary Diagonal Sum: " +
secondaryDiagonalSum);
```

Output:

mathematica

Copy code

Primary Diagonal Sum: 15

Secondary Diagonal Sum: 15

94. Check if a Matrix is Symmetric

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {2, 4, 5},
    {3, 5, 6}
};
boolean isSymmetric = true;

for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        if (matrix[i][j] != matrix[j][i]) {
            isSymmetric = false;
            break;
        }
    }
    if (!isSymmetric) break;
}
```

```
System.out.println("Matrix is Symmetric: " +
isSymmetric);
```

Output: Matrix is Symmetric: true

95. Find the Row with the Maximum Sum in a Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int maxRowSum = Integer.MIN_VALUE;
int rowIndex = -1;

for (int i = 0; i < matrix.length; i++) {
    int sum = 0;
```

```

    }, 0, 0);
};
System.out.println("Sum of Each Column:");
for (int j = 0; j < matrix[0].length; j++) {
    int sum = 0;
    for (int i = 0; i < matrix.length; i++) {
        sum += matrix[i][j];
    }
    System.out.println(sum);
}

```

Output:

mathematica
 Copy code
 Sum of Each Column:
 12
 15
 18

48. Find the Saddle Point of a Matrix

```

java
Copy code
int[][] matrix = {
    {3, 8, 7},
    {4, 5, 6},
    {1, 2, 9}
};
boolean foundSaddlePoint = false;
for (int i = 0; i < matrix.length; i++) {
    int minRow = matrix[i][0];
    int minColIndex = 0;
    for (int j = 1; j < matrix[i].length; j++) {
        if (matrix[i][j] < minRow) {
            minRow = matrix[i][j];
            minColIndex = j;
        }
    }
    int maxCol = minRow;
    for (int k = 0; k < matrix.length; k++) {
        if (matrix[k][minColIndex] > maxCol) {
            maxCol = matrix[k][minColIndex];
        }
    }
    if (maxCol == minRow) {
        System.out.println("Saddle Point: " +
maxCol);
        foundSaddlePoint = true;
        break;
    }
}
if (!foundSaddlePoint) {
    System.out.println("No Saddle Point
Found.");
}

```

```

for (int i = 0; i < matrix.length; i++) {
    int sum = 0;
    for (int num : matrix[i]) {
        sum += num;
    }
    if (sum > maxRowSum) {
        maxRowSum = sum;
        rowIndex = i;
    }
}

```

```

System.out.println("Row with Maximum Sum: " +
rowIndex);
System.out.println("Maximum Sum: " + maxRowSum);

```

Output:

mathematica
 Copy code
 Row with Maximum Sum: 2
 Maximum Sum: 24

96. Find the Column with the Maximum Sum in a Matrix

```

java
Copy code
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int maxColumnSum = Integer.MIN_VALUE;
int columnIndex = -1;
for (int j = 0; j < matrix[0].length; j++) {
    int sum = 0;
    for (int i = 0; i < matrix.length; i++) {
        sum += matrix[i][j];
    }
    if (sum > maxColumnSum) {
        maxColumnSum = sum;
        columnIndex = j;
    }
}
System.out.println("Column with Maximum Sum: " +
columnIndex);
System.out.println("Maximum Sum: " + maxColumnSum);

```

Output:

mathematica
 Copy code
 Column with Maximum Sum: 2
 Maximum Sum: 18

97. Find the Saddle Point in a Matrix

java
 Copy code

Output: Saddle Point: 5

49. Find the Maximum Value in Each Row of a Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println("Maximum Value in Each Row:");
for (int[] row : matrix) {
    int max = row[0];
    for (int num : row) {
        max = Math.max(max, num);
    }
    System.out.println(max);
}
```

Output:

sql

Copy code

Maximum Value in Each Row:

3
6
9

50. Find the Minimum Value in Each Column of a Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println("Minimum Value in Each Column:");
for (int j = 0; j < matrix[0].length; j++) {
    int min = matrix[0][j];
    for (int i = 0; i < matrix.length; i++) {
        min = Math.min(min, matrix[i][j]);
    }
    System.out.println(min);
}
```

Output:

sql

Copy code

Minimum Value in Each Column:

1
2
~

97. Find the Saddle Point in a Matrix

java

Copy code

```
int[][] matrix = {
    {3, 8, 7},
    {5, 6, 4},
    {1, 2, 9}
};
for (int i = 0; i < matrix.length; i++) {
    int rowMin = matrix[i][0];
    int colIndex = 0;
    for (int j = 1; j < matrix[i].length; j++) {
        if (matrix[i][j] < rowMin) {
            rowMin = matrix[i][j];
            colIndex = j;
        }
    }
    boolean isSaddlePoint = true;
    for (int k = 0; k < matrix.length; k++) {
        if (matrix[k][colIndex] > rowMin) {
            isSaddlePoint = false;
            break;
        }
    }

    if (isSaddlePoint) {
        System.out.println("Saddle Point: " + rowMin);
    }
}
```

Output: Saddle Point: 4

98. Count the Number of Islands in a Binary Matrix

java

Copy code

```
int[][] matrix = {
    {1, 1, 0, 0, 0},
    {1, 1, 0, 0, 1},
    {0, 0, 0, 1, 1},
    {0, 0, 0, 0, 0},
    {1, 0, 1, 0, 1}
};
boolean[][] visited = new boolean[matrix.length][matrix[0].length];
int[] rowNum = {-1, 0, 0, 1};
int[] colNum = {0, -1, 1, 0};
int islandCount = 0;

for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[0].length; j++) {
        if (matrix[i][j] == 1 && !visited[i][j]) {
            markIsland(matrix, visited, i, j, rowNum, colNum);
            islandCount++;
        }
    }
}
```

1
2
3

51. Find the Number of Positive and Negative Numbers in a Matrix

java

Copy code

```
int[][] matrix = {
    {1, -2, 3},
    {-4, 5, -6},
    {7, -8, 9}
};
int positiveCount = 0;
int negativeCount = 0;
for (int[] row : matrix) {
    for (int num : row) {
        if (num > 0) positiveCount++;
        else if (num < 0) negativeCount++;
    }
}
System.out.println("Number of Positive Numbers: " + positiveCount);
System.out.println("Number of Negative Numbers: " + negativeCount);
```

Output:

mathematica

Copy code

Number of Positive Numbers: 6
Number of Negative Numbers: 4

52. Check if a Matrix is an Identity Matrix

java

Copy code

```
int[][] matrix = {
    {1, 0, 0},
    {0, 1, 0},
    {0, 0, 1}
};
boolean isIdentity = true;
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        if (i == j && matrix[i][j] != 1 || i != j && matrix[i][j] != 0) {
            isIdentity = false;
            break;
        }
    }
    if (!isIdentity) break;
}
System.out.println("Matrix is Identity: " + isIdentity);
```

Output: Matrix is Identity: true

53. Transpose a Matrix

```
islandCount++;
    }
}
}
System.out.println("Number of Islands: " + islandCount);
static void markIsland(int[][] matrix, boolean[][] visited,
int row, int col, int[] rowNum, int[] colNum) {
    int n = matrix.length;
    int m = matrix[0].length;
    visited[row][col] = true;
    for (int k = 0; k < 4; k++) {
        int newRow = row + rowNum[k];
        int newCol = col + colNum[k];
        if (isSafe(matrix, visited, newRow, newCol, n, m)) {
            markIsland(matrix, visited, newRow, newCol,
rowNum, colNum);
        }
    }
}
static boolean isSafe(int[][] matrix, boolean[][] visited, int
row, int col, int n, int m) {
    return (row >= 0) && (row < n) &&
        (col >= 0) && (col < m) &&
        (matrix[row][col] == 1 && !visited[row][col]);
}
```

Output: Number of Islands: 4

99. Find the Longest Common Prefix in 2D Matrix Strings

java

Copy code

```
String[][] matrix = {
    {"flower", "flow", "flight"},
    {"dog", "racecar", "car"}
};
for (String[] row : matrix) {
    String prefix = row[0];
    for (int i = 1; i < row.length; i++) {
        while (row[i].indexOf(prefix) != 0) {
            prefix = prefix.substring(0, prefix.length() - 1);
            if (prefix.isEmpty()) {
                System.out.println("Longest Common Prefix: " + prefix);
                break;
            }
        }
    }
    System.out.println("Longest Common Prefix: " + prefix);
}
```

Output:

mathematica

Copy code

Longest Common Prefix: fl
Longest Common Prefix:

100. Find the Maximum Sum Subarray in a 2D Matrix ``java
int[][] matrix = { {1, -2, -1, 4}, {2, -1, 2, 1}, {1, -3, 4, -2}, {1,

Output: Matrix is Identity: true

53. Transpose a Matrix

java

Copy code

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
int[][] transpose = new int[matrix[0].length]
[matrix.length];
for (int i = 0; i < matrix.length; i++) {
    for (int j = 0; j < matrix[i].length; j++) {
        transpose[j][i] = matrix[i][j];
    }
}
System.out.println("Transpose of Matrix:");
for (int[] row : transpose) {
    System.out.println(Arrays.toString(row));
}
```

Output:

csharp

Copy code

Transpose of Matrix:

```
[1, 4, 7]
[2, 5, 8]
[3, 6, 9]
```

54. Flatten a 2D Matrix to a 1D Array

java

Copy code

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

☐ Problems

☐ Solve

100.

Longest Common Prefix.

Find the Maximum Sum Subarray in a 2D Matrix ```java
int[][] matrix = { {1, -2, -1, 4}, {2, -1, 2, 1}, {1, -3, 4, -2}, {1, 2, 2, 1} }; int maxSum = Integer.MIN_VALUE;
css
Copy code
for (int left = 0; left < matrix[0].length; left++) {
sql
Copy code
int[] temp = new int[matrix.length];
 for (int right = left; right < matrix[0].length; right++) {
 for (int i = 0; i < matrix.length; i++) {
 temp[i] += matrix[i][right];
 }

 int currentSum = kadane(temp);
 maxSum = Math.max(maxSum, currentSum);
 }
}
System.out.println("Maximum Sum Subarray: " + maxSum);
static int kadane(int[] arr) {
 int maxSoFar = Integer.MIN_VALUE;
 int maxEndingHere = 0;

 for (int num : arr) {
 maxEndingHere += num;
 maxSoFar = Math.max(maxSoFar, maxEndingHere);
 if (maxEndingHere < 0) {
 maxEndingHere = 0;
 }
 }

 return maxSoFar;
}
...
Output: `Maximum Sum Subarray: 12`

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

☐ Level 2

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

<input type="checkbox"/> <u>Find whether an array is a subset of another array</u>	<input type="checkbox"/> <u>Solve</u>
<input type="checkbox"/> <u>Implement two Stacks in an array</u>	<input type="checkbox"/> <u>Solve</u>
<input type="checkbox"/> <u>Majority Element</u>	<input type="checkbox"/> <u>Solve</u>
<input type="checkbox"/> <u>Wave Array</u>	<input type="checkbox"/> <u>Solve</u>
<input type="checkbox"/> <u>Trapping Rainwater</u>	<input type="checkbox"/> <u>Solve</u>

☐ Level 3

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

