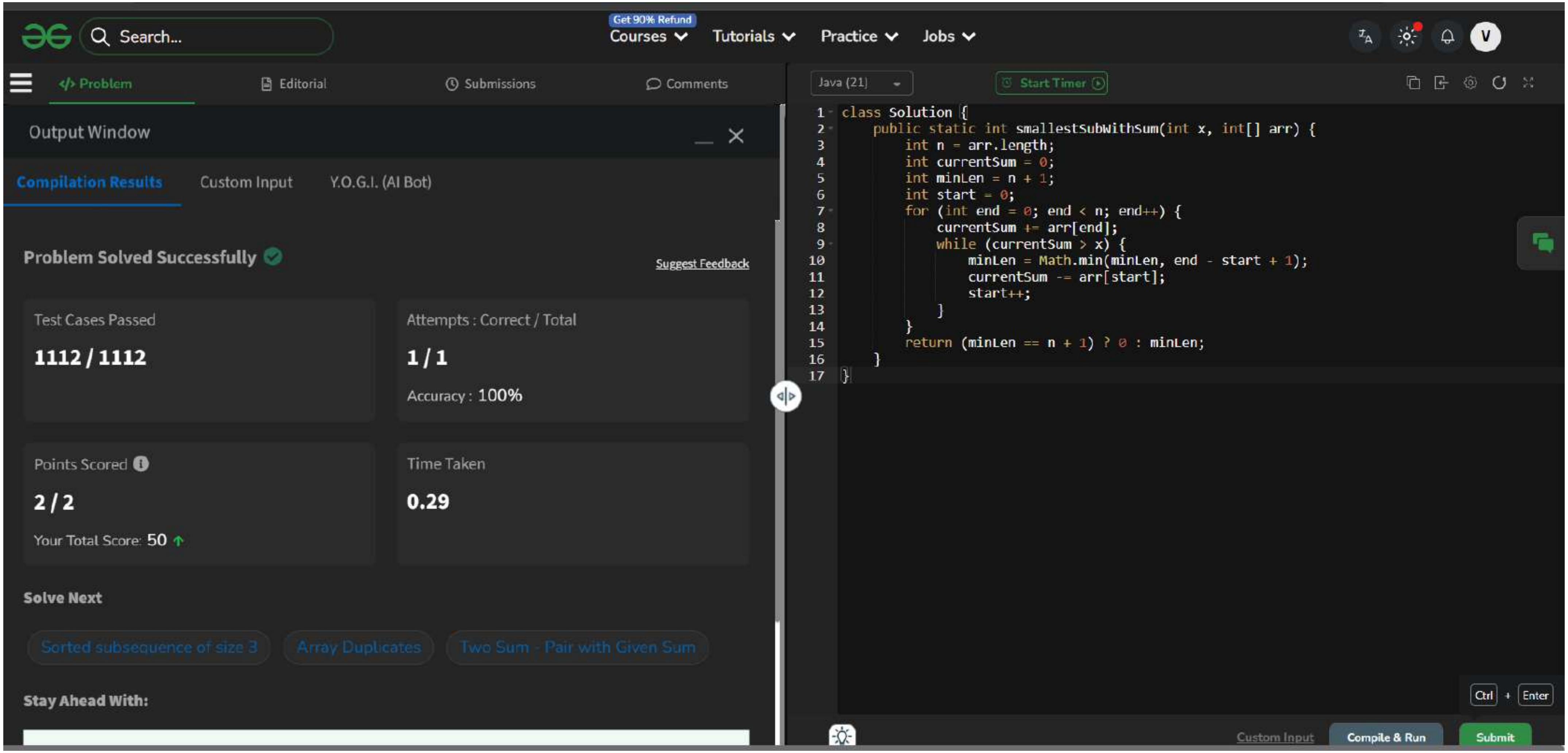


Problem Solved Successfully ✓

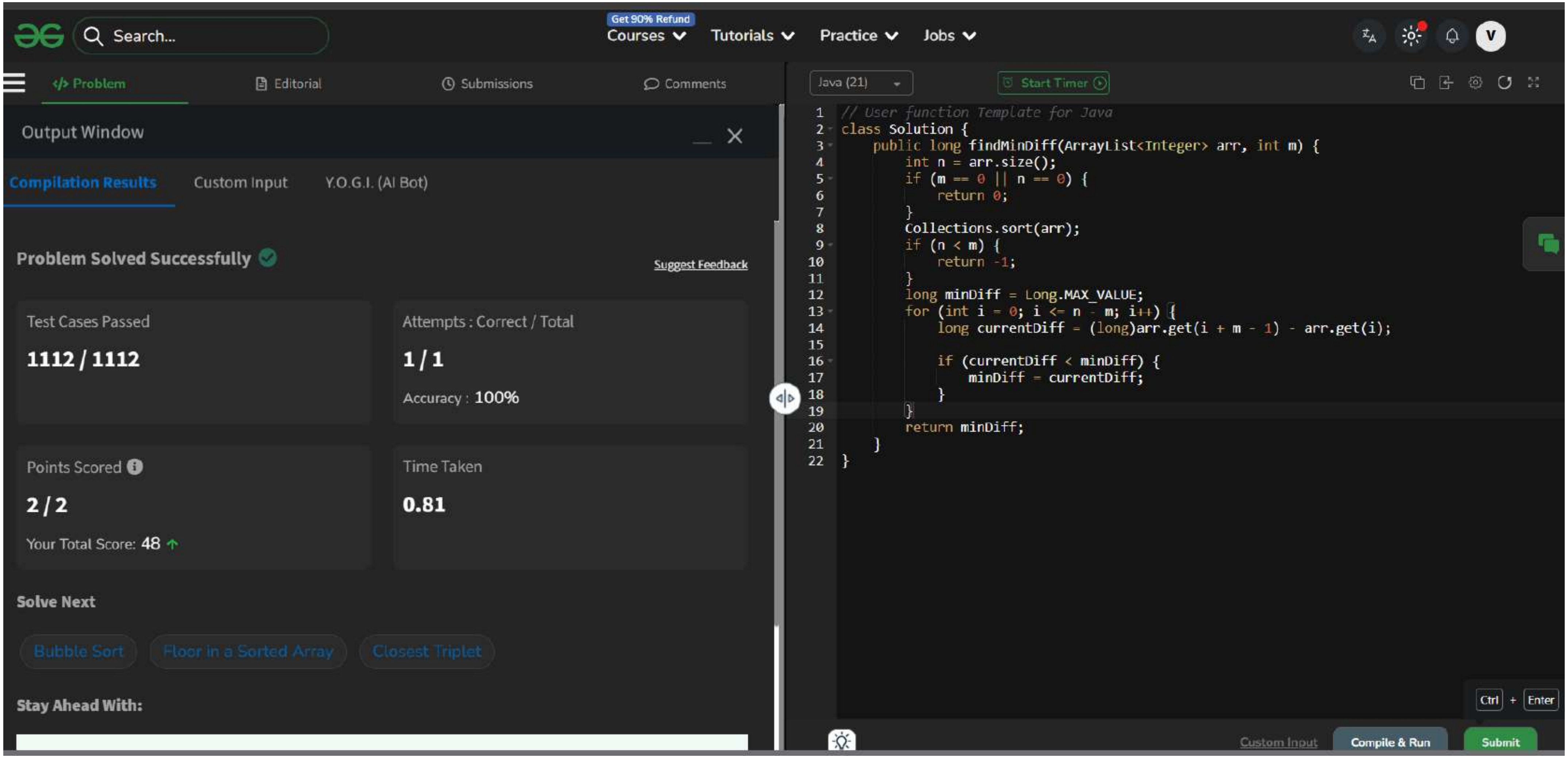
Test Cases Passed
1112 / 1112

Attempts : Correct / Total
1 / 1
Accuracy : 100%

Points Scored
2 / 2
Your Total Score: 50 ↑

Time Taken
0.29

```java
class Solution {
    public static int smallestSubWithSum(int x, int[] arr) {
        int n = arr.length;
        int currentSum = 0;
        int minLen = n + 1;
        int start = 0;
        for (int end = 0; end < n; end++) {
            currentSum += arr[end];
            while (currentSum > x) {
                minLen = Math.min(minLen, end - start + 1);
                currentSum -= arr[start];
                start++;
            }
        }
        return (minLen == n + 1) ? 0 : minLen;
    }
}
```



Output Window

Compilation Results   Custom Input   Y.O.G.I. (AI Bot)

Problem Solved Successfully

Suggest Feedback

Test Cases Passed

1112 / 1112

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

2 / 2

Your Total Score: 48

Time Taken

0.81

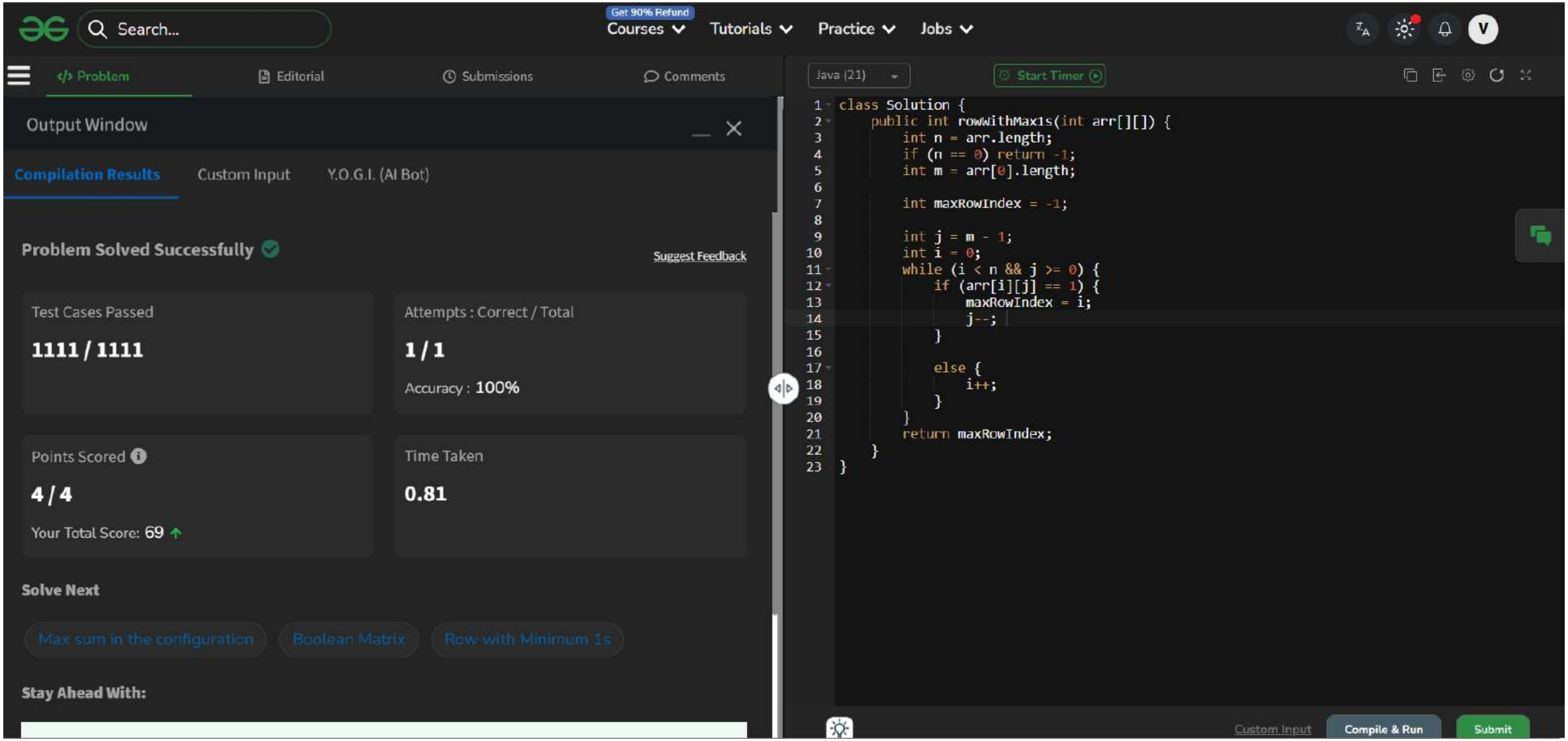
Solve Next

Bubble Sort   Floor in a Sorted Array   Closest Triplet

Stay Ahead With:

```java
// User function Template for Java
class Solution {
    public long findMinDiff(ArrayList<Integer> arr, int m) {
        int n = arr.size();
        if (m == 0 || n == 0) {
            return 0;
        }
        Collections.sort(arr);
        if (n < m) {
            return -1;
        }
        long minDiff = Long.MAX_VALUE;
        for (int i = 0; i <= n - m; i++) {
            long currentDiff = (long)arr.get(i + m - 1) - arr.get(i);

            if (currentDiff < minDiff) {
                minDiff = currentDiff;
            }
        }
        return minDiff;
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed	Attempts : Correct / Total
1111 / 1111	1 / 1 Accuracy : 100%

Points Scored	Time Taken
4 / 4 Your Total Score: 69 ↑	0.81

Solve Next

Max sum in the configuration

Boolean Matrix

Row with Minimum 1s

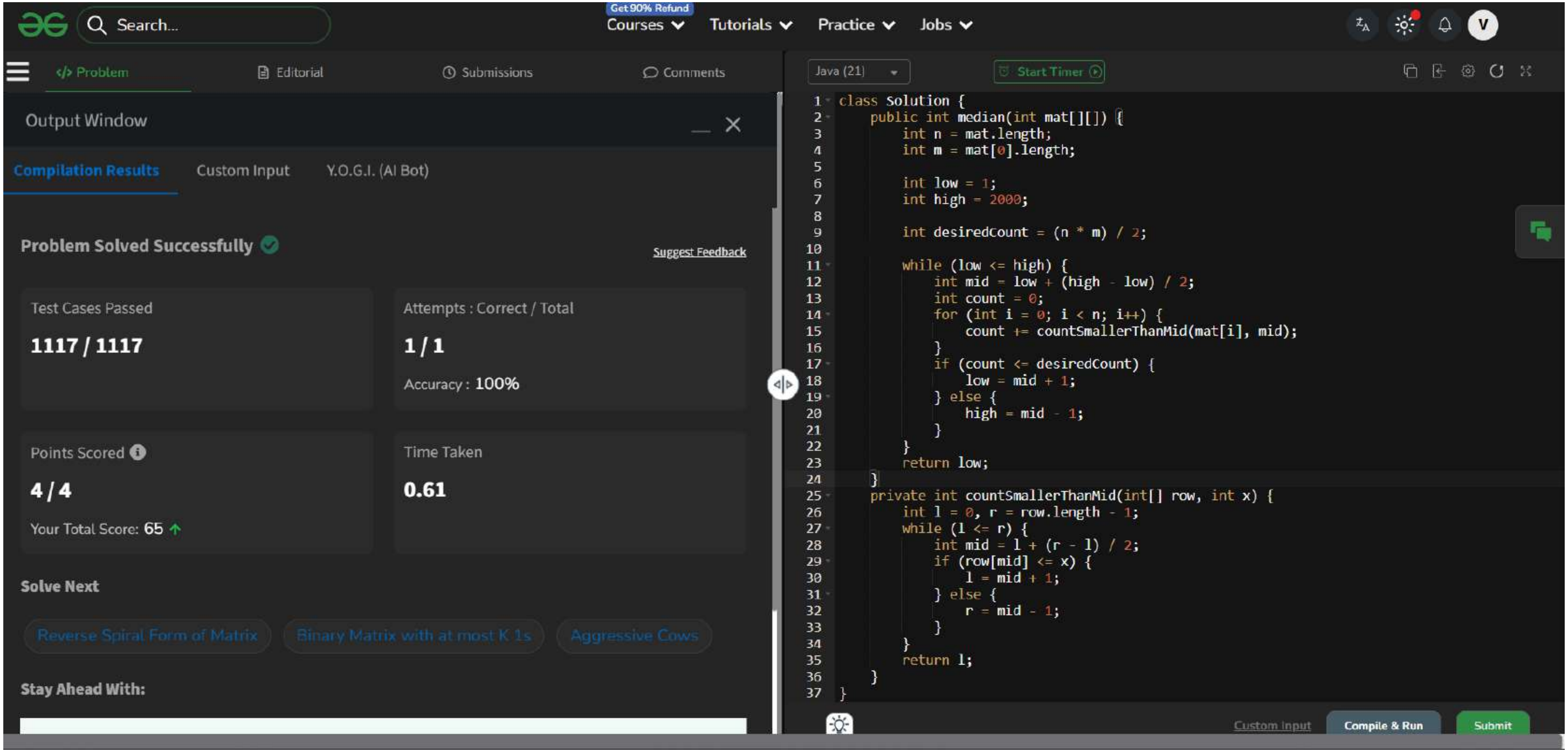
Stay Ahead With:

```java
class Solution {
    public int rowWithMax1s(int arr[][]) {
        int n = arr.length;
        if (n == 0) return -1;
        int m = arr[0].length;

        int maxRowIndex = -1;

        int j = m - 1;
        int i = 0;
        while (i < n && j >= 0) {
            if (arr[i][j] == 1) {
                maxRowIndex = i;
                j--;
            }

            else {
                i++;
            }
        }
        return maxRowIndex;
    }
}
```



Problem Solved Successfully

Test Cases Passed

1117 / 1117

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

4 / 4

Your Total Score: 65

Time Taken

0.61

Solve Next

Reverse Spiral Form of Matrix

Binary Matrix with at most K 1s

Aggressive Cows

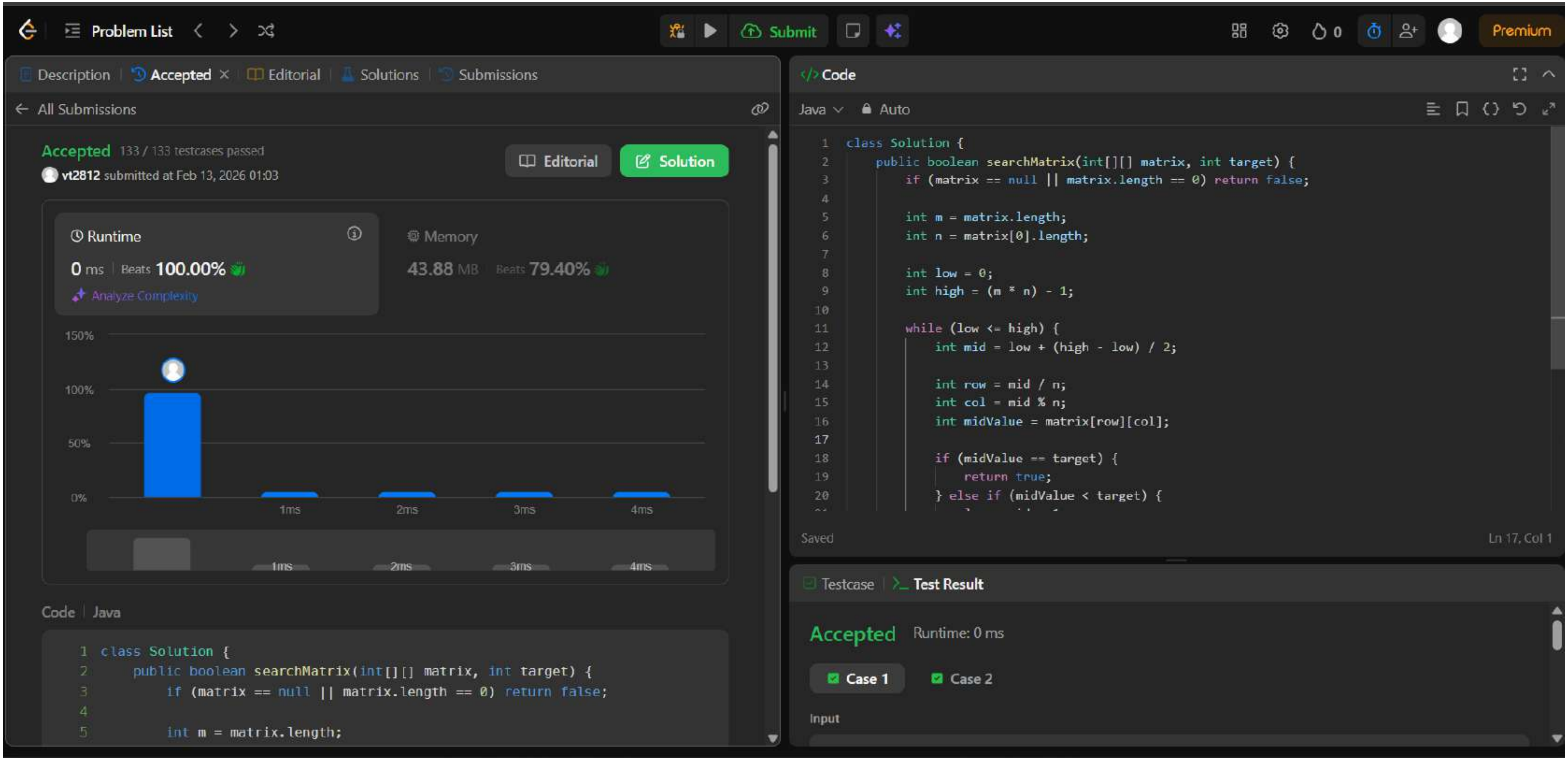
Stay Ahead With:

```java
class Solution {
    public int median(int mat[][]) {
        int n = mat.length;
        int m = mat[0].length;

        int low = 1;
        int high = 2000;

        int desiredCount = (n * m) / 2;

        while (low <= high) {
            int mid = low + (high - low) / 2;
            int count = 0;
            for (int i = 0; i < n; i++) {
                count += countSmallerThanMid(mat[i], mid);
            }
            if (count <= desiredCount) {
                low = mid + 1;
            } else {
                high = mid - 1;
            }
        }
        return low;
    }
    private int countSmallerThanMid(int[] row, int x) {
        int l = 0, r = row.length - 1;
        while (l <= r) {
            int mid = l + (r - l) / 2;
            if (row[mid] <= x) {
                l = mid + 1;
            } else {
                r = mid - 1;
            }
        }
        return l;
    }
}
```



Accepted 133 / 133 testcases passed

vt2812 submitted at Feb 13, 2026 01:03

Runtime

0 ms | Beats 100.00%

Memory

43.88 MB | Beats 79.40%

```java
class Solution {
    public boolean searchMatrix(int[][] matrix, int target) {
        if (matrix == null || matrix.length == 0) return false;

        int m = matrix.length;
        int n = matrix[0].length;

        int low = 0;
        int high = (m * n) - 1;

        while (low <= high) {
            int mid = low + (high - low) / 2;

            int row = mid / n;
            int col = mid % n;
            int midValue = matrix[row][col];

            if (midValue == target) {
                return true;
            } else if (midValue < target) {
```

Saved

Ln 17, Col 1

Code | Java

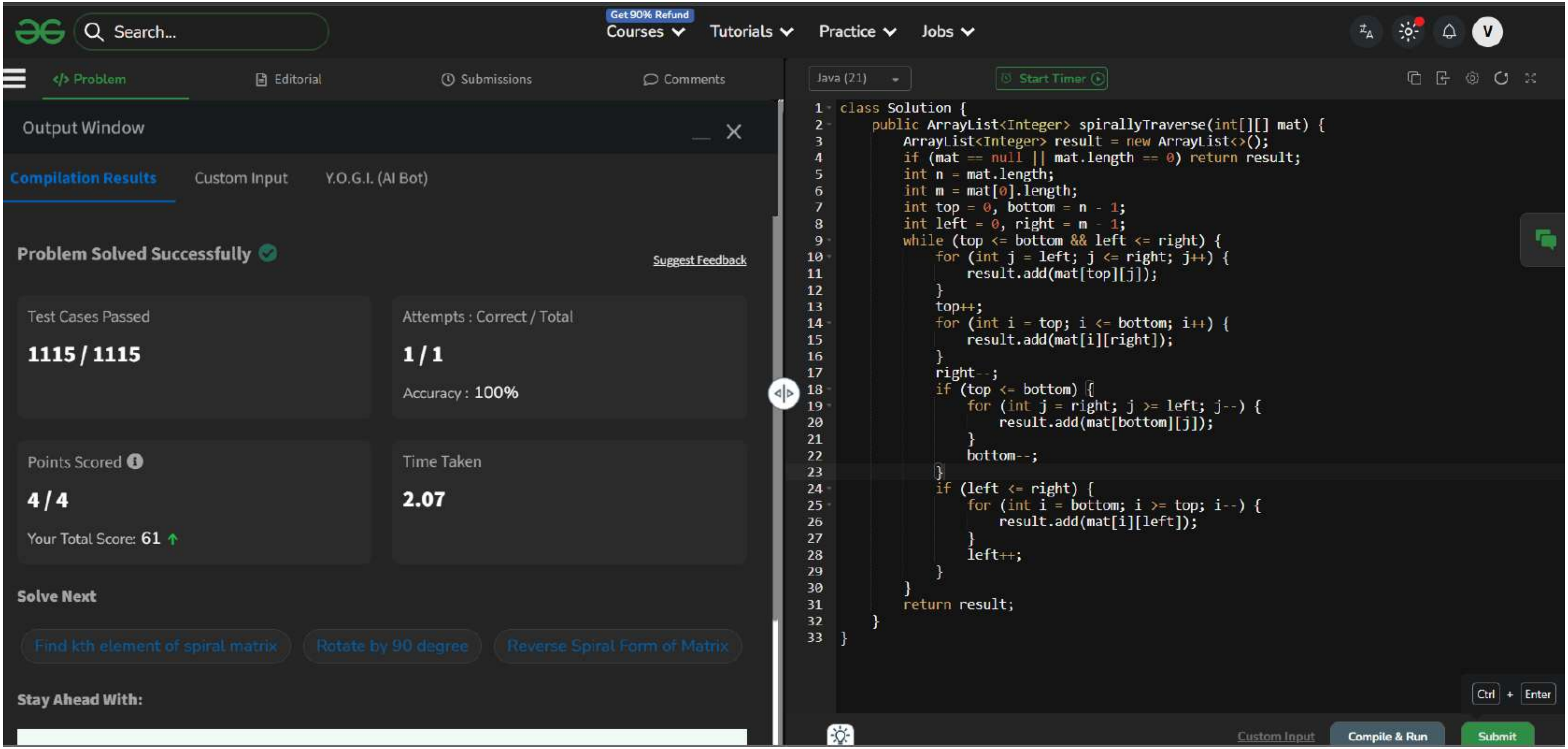
```java
class Solution {
    public boolean searchMatrix(int[][] matrix, int target) {
        if (matrix == null || matrix.length == 0) return false;

        int m = matrix.length;
```

Accepted Runtime: 0 ms

☑ Case 1 ☑ Case 2

Input



Problem Solved Successfully ✓

Test Cases Passed
1115 / 1115

Attempts : Correct / Total
1 / 1
Accuracy : 100%

Points Scored
4 / 4
Your Total Score: 61 ↑

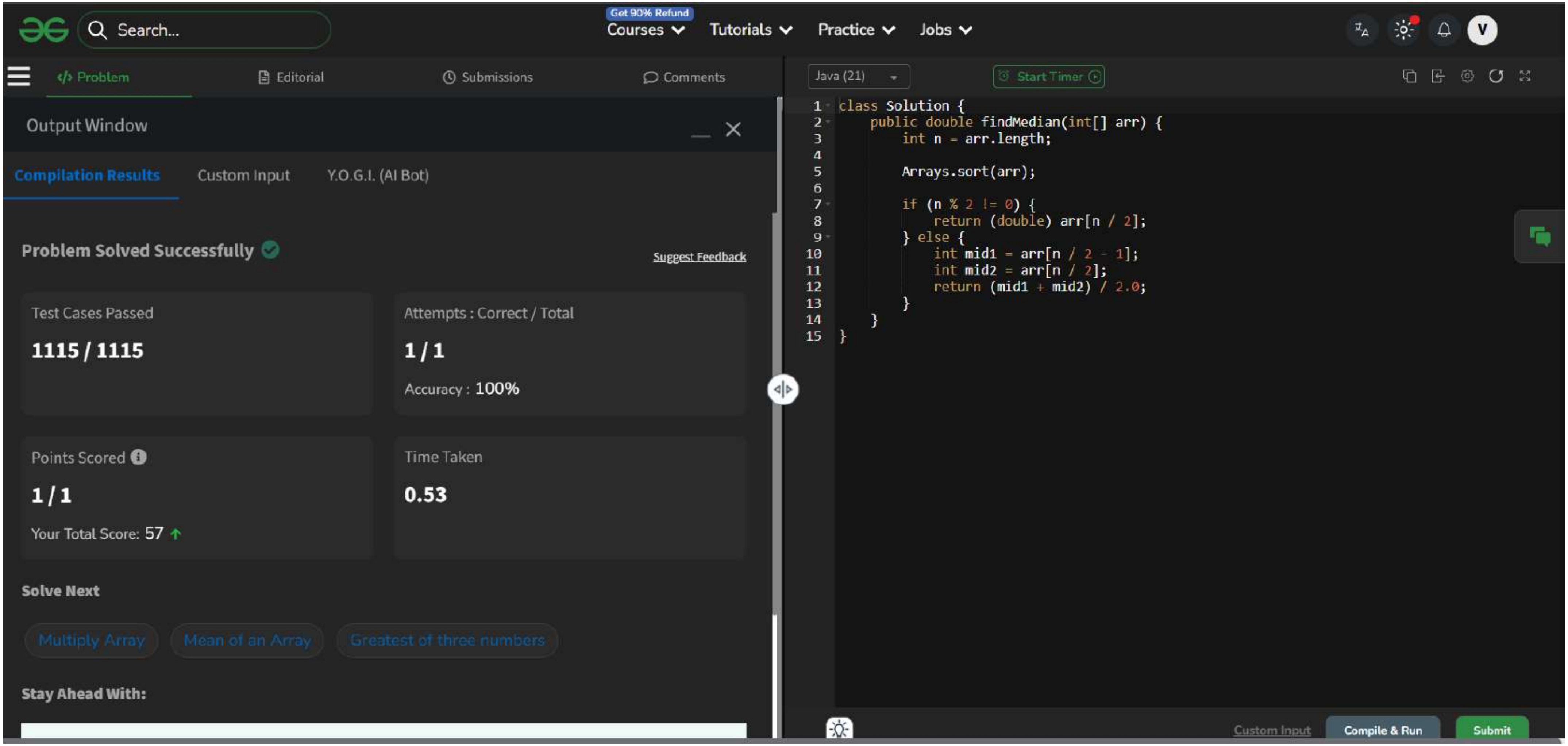
Time Taken
2.07

Solve Next

Find kth element of spiral matrix
Rotate by 90 degree
Reverse Spiral Form of Matrix

Stay Ahead With:

```java
class Solution {
    public ArrayList<Integer> spirallyTraverse(int[][] mat) {
        ArrayList<Integer> result = new ArrayList<>();
        if (mat == null || mat.length == 0) return result;
        int n = mat.length;
        int m = mat[0].length;
        int top = 0, bottom = n - 1;
        int left = 0, right = m - 1;
        while (top <= bottom && left <= right) {
            for (int j = left; j <= right; j++) {
                result.add(mat[top][j]);
            }
            top++;
            for (int i = top; i <= bottom; i++) {
                result.add(mat[i][right]);
            }
            right--;
            if (top <= bottom) {
                for (int j = right; j >= left; j--) {
                    result.add(mat[bottom][j]);
                }
                bottom--;
            }
            if (left <= right) {
                for (int i = bottom; i >= top; i--) {
                    result.add(mat[i][left]);
                }
                left++;
            }
        }
        return result;
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed
1115 / 1115

Attempts : Correct / Total
1 / 1
Accuracy : 100%

Points Scored
1 / 1
Your Total Score: 57 ↑

Time Taken
0.53

Solve Next

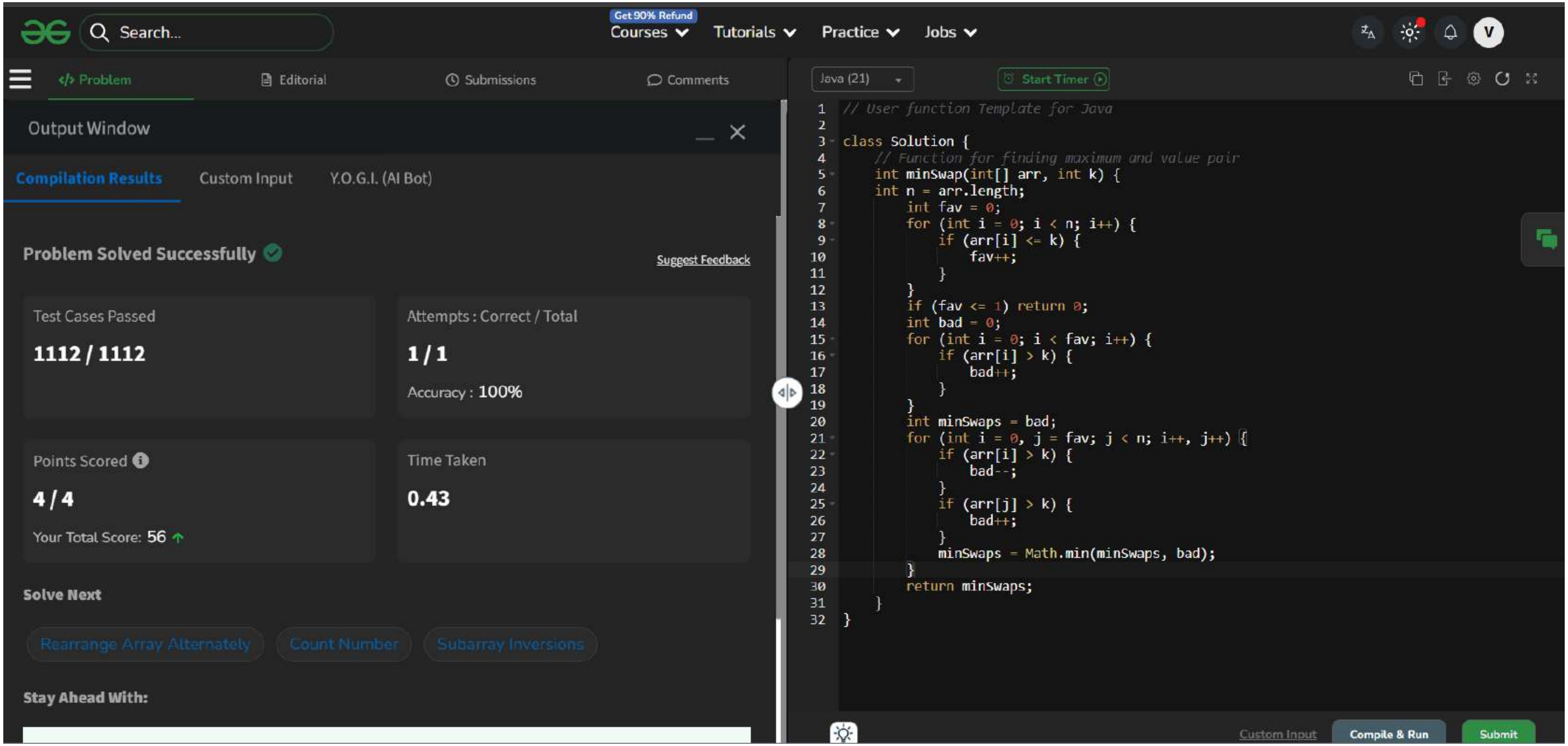
Multiply Array | Mean of an Array | Greatest of three numbers

Stay Ahead With:

```java
class Solution {
    public double findMedian(int[] arr) {
        int n = arr.length;

        Arrays.sort(arr);

        if (n % 2 != 0) {
            return (double) arr[n / 2];
        } else {
            int mid1 = arr[n / 2 - 1];
            int mid2 = arr[n / 2];
            return (mid1 + mid2) / 2.0;
        }
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed

1112 / 1112

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

4 / 4

Your Total Score: 56 ↑

Time Taken

0.43

Solve Next

Rearrange Array Alternately

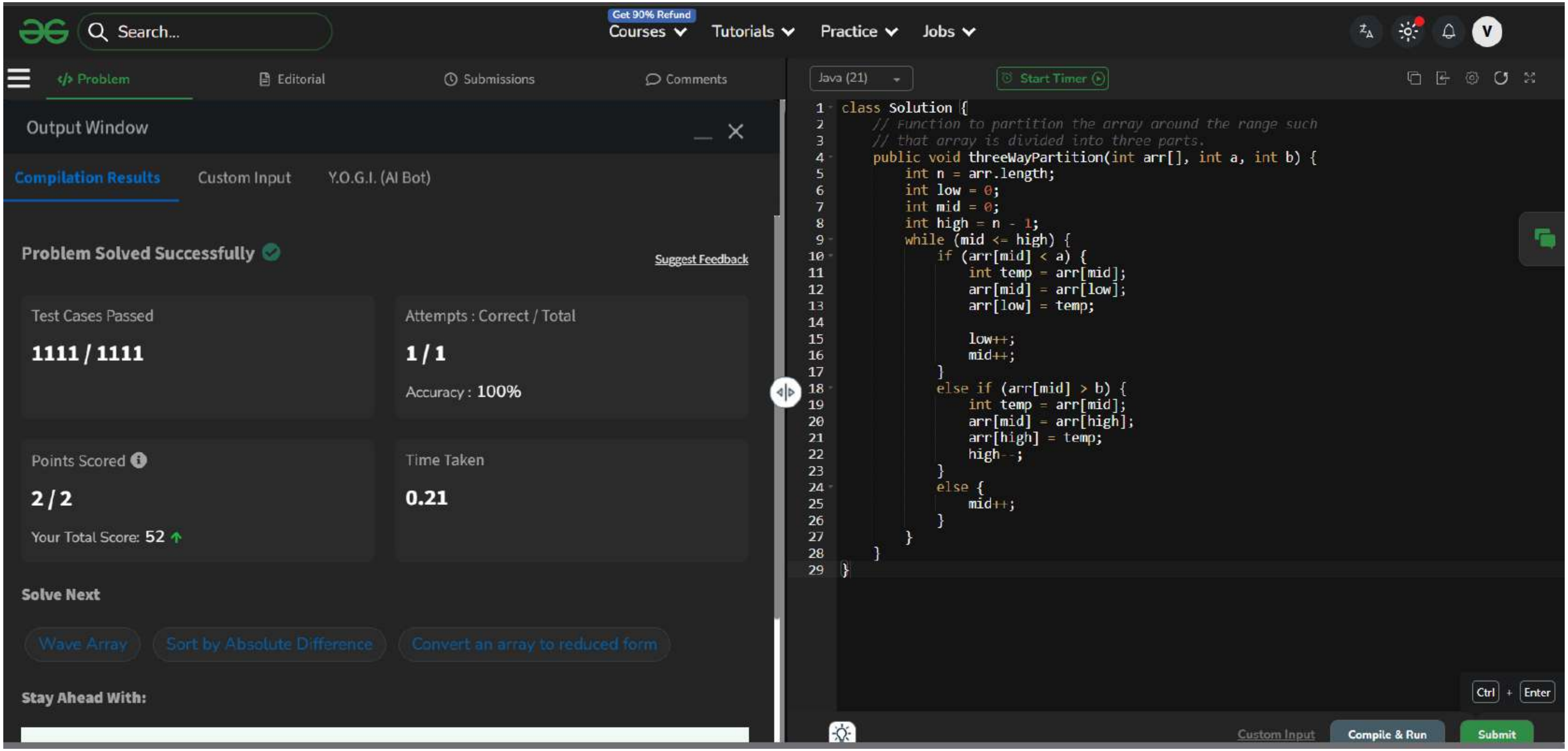
Count Number

Subarray Inversions

Stay Ahead With:

```java
// User function Template for Java

class Solution {
    // Function for finding maximum and value pair
    int minSwap(int[] arr, int k) {
    int n = arr.length;
        int fav = 0;
        for (int i = 0; i < n; i++) {
            if (arr[i] <= k) {
                fav++;
            }
        }
        if (fav <= 1) return 0;
        int bad = 0;
        for (int i = 0; i < fav; i++) {
            if (arr[i] > k) {
                bad++;
            }
        }
        int minSwaps = bad;
        for (int i = 0, j = fav; j < n; i++, j++) {
            if (arr[i] > k) {
                bad--;
            }
            if (arr[j] > k) {
                bad++;
            }
            minSwaps = Math.min(minSwaps, bad);
        }
        return minSwaps;
    }
}
```



Output Window

Compilation Results | Custom Input | Y.O.G.I. (AI Bot)

Problem Solved Successfully ✓

Suggest Feedback

Test Cases Passed

1111 / 1111

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

2 / 2

Your Total Score: 52 ↑

Time Taken

0.21

Solve Next

Wave Array | Sort by Absolute Difference | Convert an array to reduced form

Stay Ahead With:

```java
class Solution {
    // Function to partition the array around the range such
    // that array is divided into three parts.
    public void threeWayPartition(int arr[], int a, int b) {
        int n = arr.length;
        int low = 0;
        int mid = 0;
        int high = n - 1;
        while (mid <= high) {
            if (arr[mid] < a) {
                int temp = arr[mid];
                arr[mid] = arr[low];
                arr[low] = temp;

                low++;
                mid++;
            }
            else if (arr[mid] > b) {
                int temp = arr[mid];
                arr[mid] = arr[high];
                arr[high] = temp;
                high--;
            }
            else {
                mid++;
            }
        }
    }
}
```