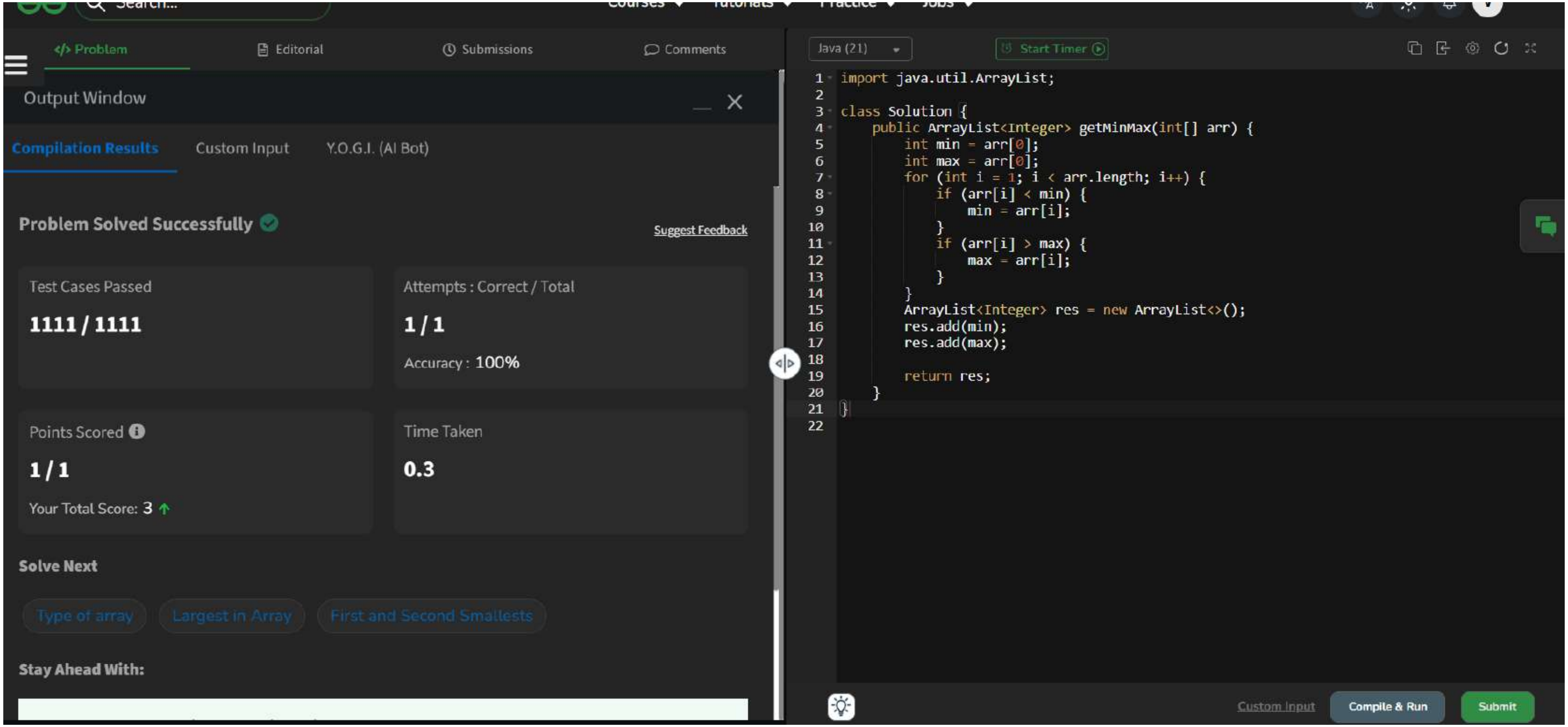


Output Window

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

1 / 1

Your Total Score: 3 ↑

Time Taken

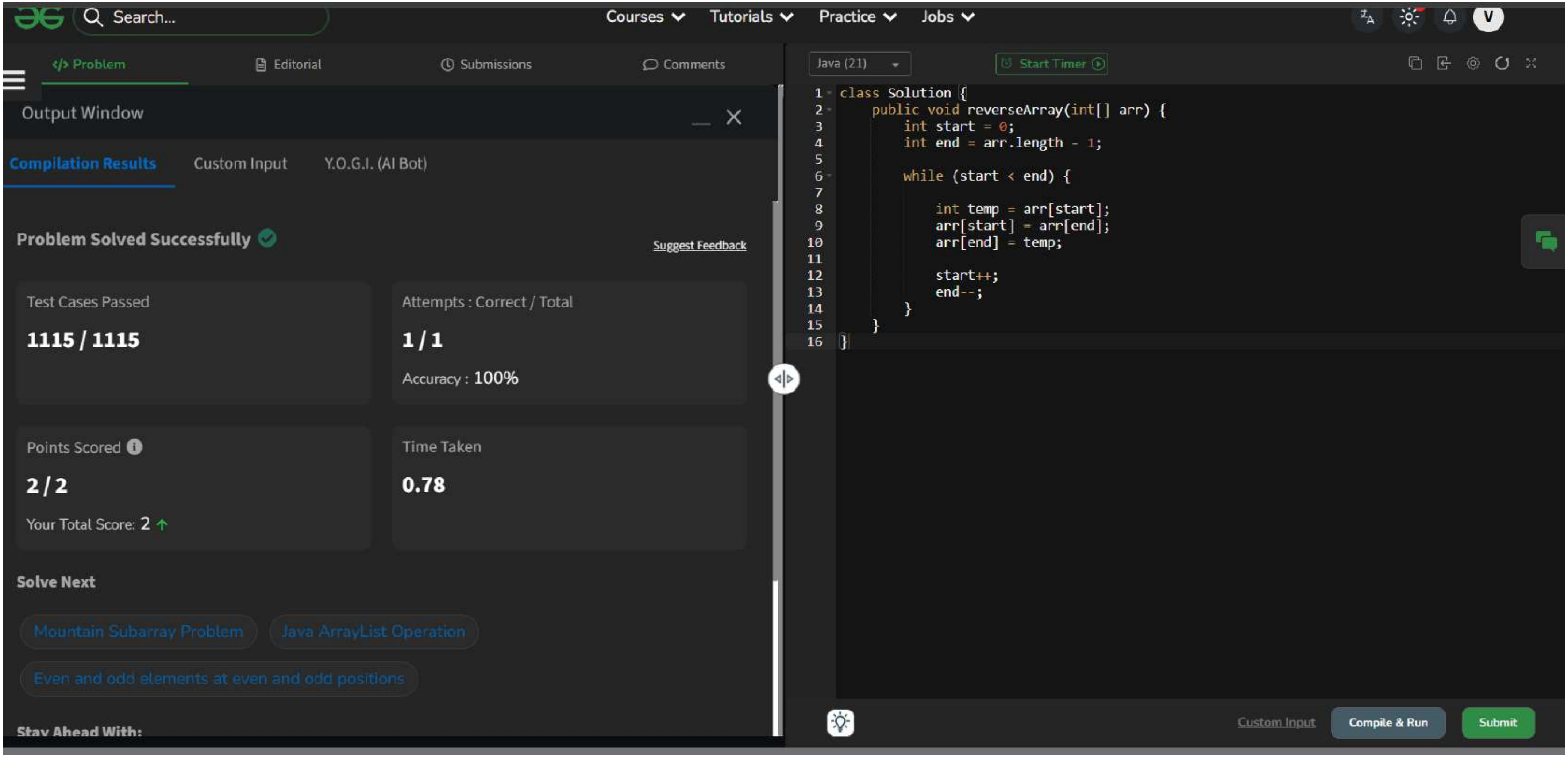
0.3

Solve Next

Type of array | Largest in Array | First and Second Smallests

Stay Ahead With:

```java
import java.util.ArrayList;

class Solution {
    public ArrayList<Integer> getMinMax(int[] arr) {
        int min = arr[0];
        int max = arr[0];
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] < min) {
                min = arr[i];
            }
            if (arr[i] > max) {
                max = arr[i];
            }
        }
        ArrayList<Integer> res = new ArrayList<>();
        res.add(min);
        res.add(max);

        return res;
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed
1115 / 1115

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

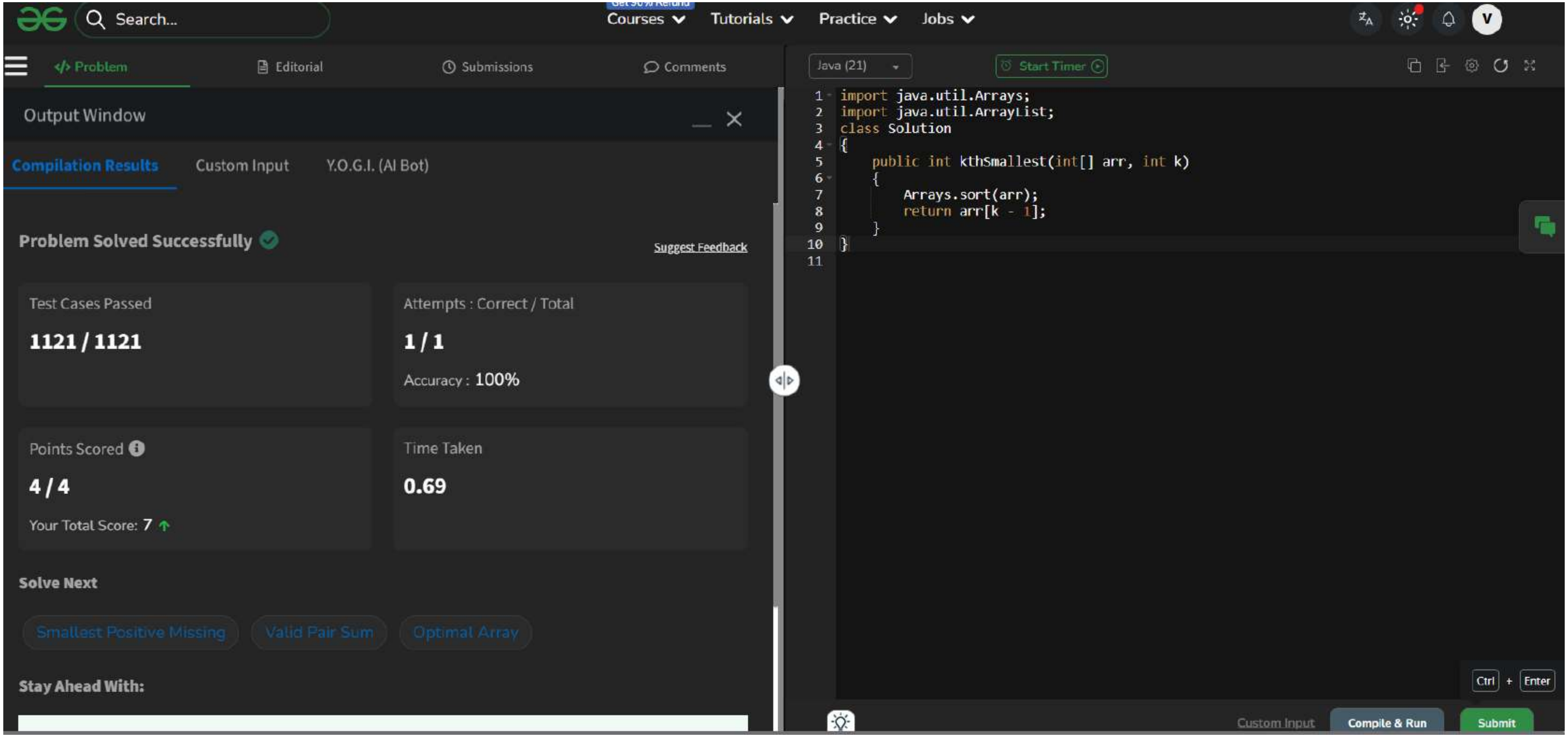
Points Scored
2 / 2
Your Total Score: 2 ↑

Time Taken
0.78

Solve Next

Mountain Subarray Problem
Java ArrayList Operation
Even and odd elements at even and odd positions
Stay Ahead With:
Custom Input
Compile & Run
Submit

```java
class Solution {
    public void reverseArray(int[] arr) {
        int start = 0;
        int end = arr.length - 1;

        while (start < end) {

            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;

            start++;
            end--;
        }
    }
}
```



Problem Solved Successfully ✓

Suggest Feedback

Test Cases Passed

1121 / 1121

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

4 / 4

Your Total Score: 7 ↑

Time Taken

0.69

Solve Next

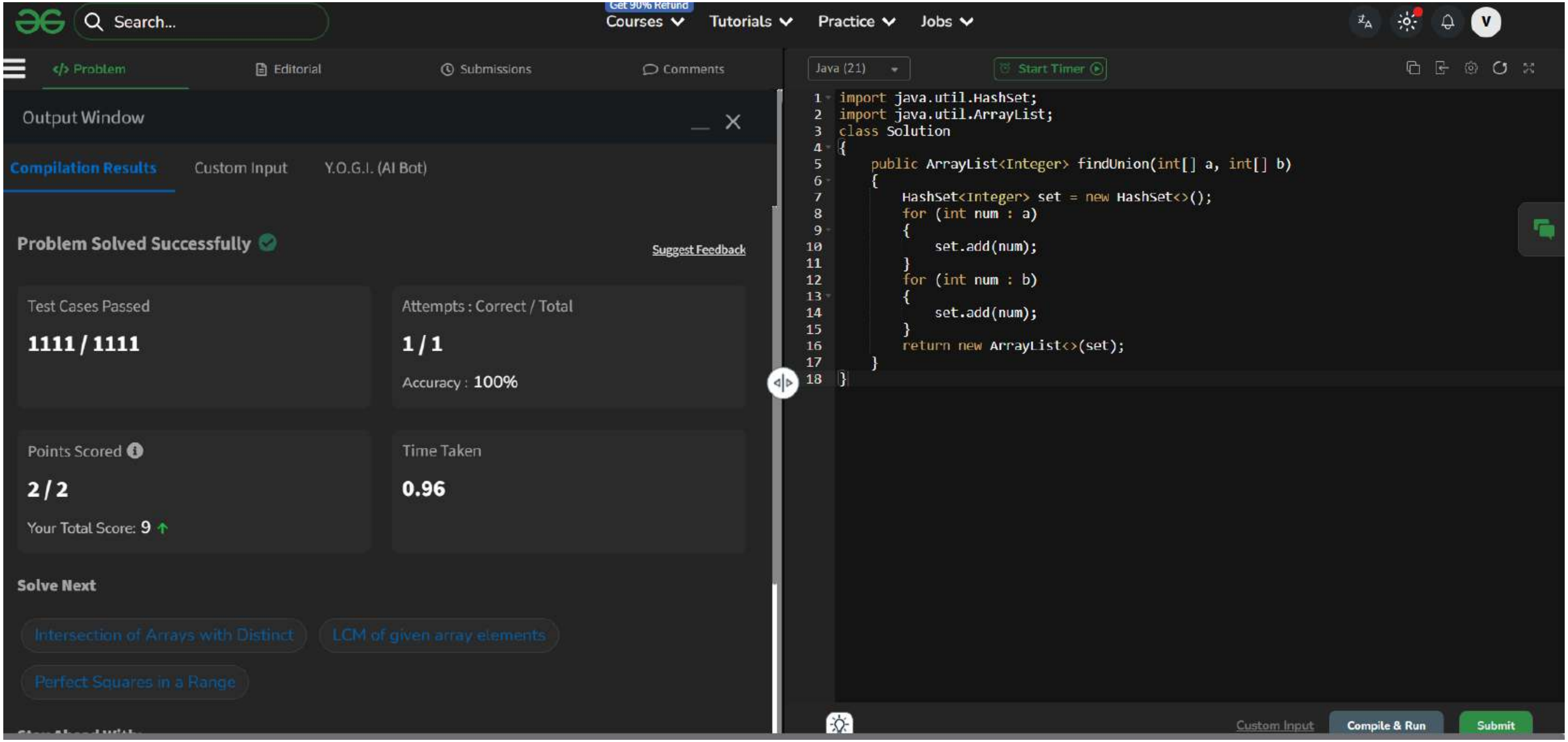
Smallest Positive Missing

Valid Pair Sum

Optimal Array

Stay Ahead With:

```java
import java.util.Arrays;
import java.util.ArrayList;
class Solution
{
    public int kthSmallest(int[] arr, int k)
    {
        Arrays.sort(arr);
        return arr[k - 1];
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed

1111 / 1111

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

2 / 2

Your Total Score: 9 ↑

Time Taken

0.96

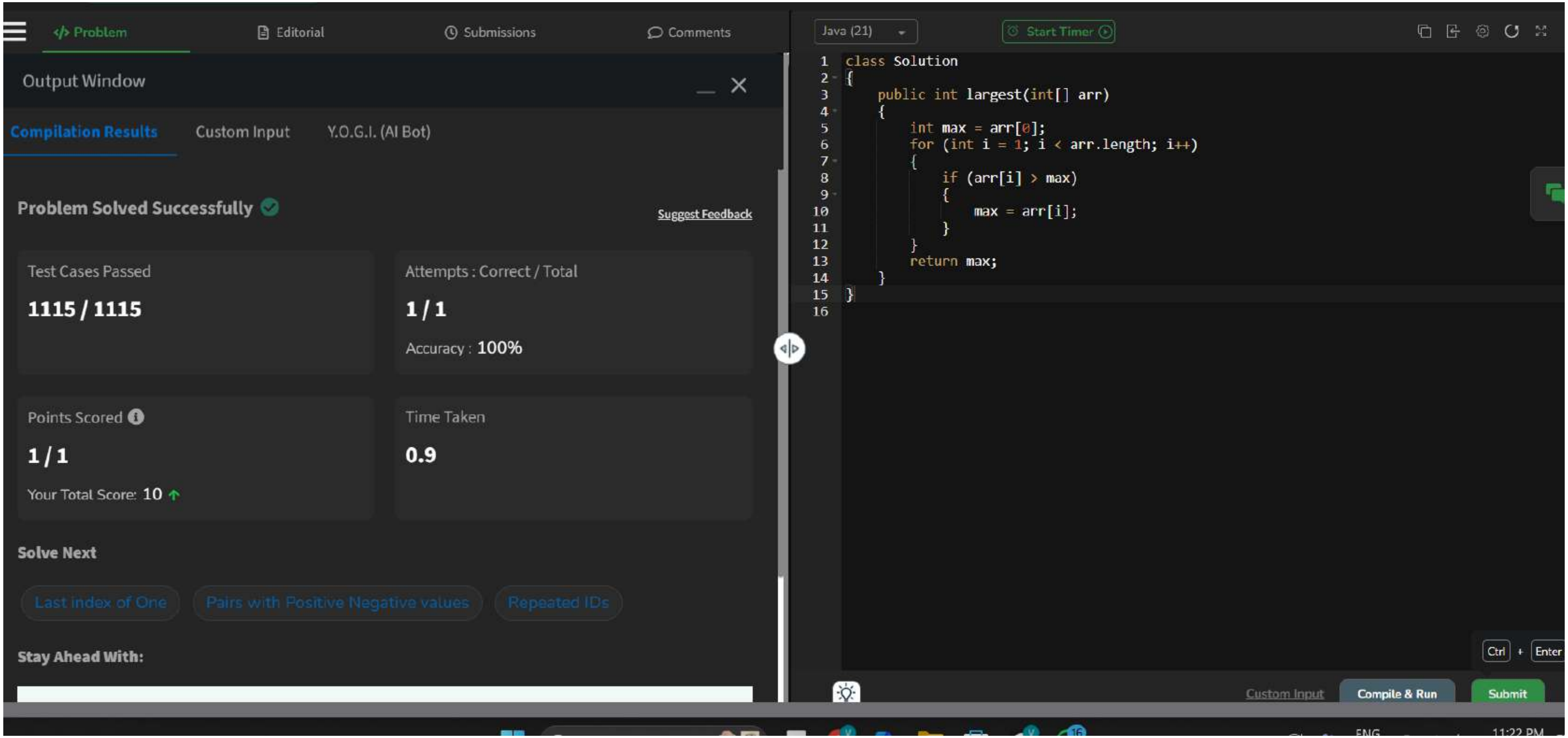
Solve Next

Intersection of Arrays with Distinct

LCM of given array elements

Perfect Squares in a Range

```java
import java.util.HashSet;
import java.util.ArrayList;
class Solution
{
    public ArrayList<Integer> findUnion(int[] a, int[] b)
    {
        HashSet<Integer> set = new HashSet<>();
        for (int num : a)
        {
            set.add(num);
        }
        for (int num : b)
        {
            set.add(num);
        }
        return new ArrayList<>(set);
    }
}
```



```
class Solution
{
    public int largest(int[] arr)
    {
        int max = arr[0];
        for (int i = 1; i < arr.length; i++)
        {
            if (arr[i] > max)
            {
                max = arr[i];
            }
        }
        return max;
    }
}
```

Problem Solved Successfully ✓

Test Cases Passed

1115 / 1115

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

1 / 1

Your Total Score: 10 ↑

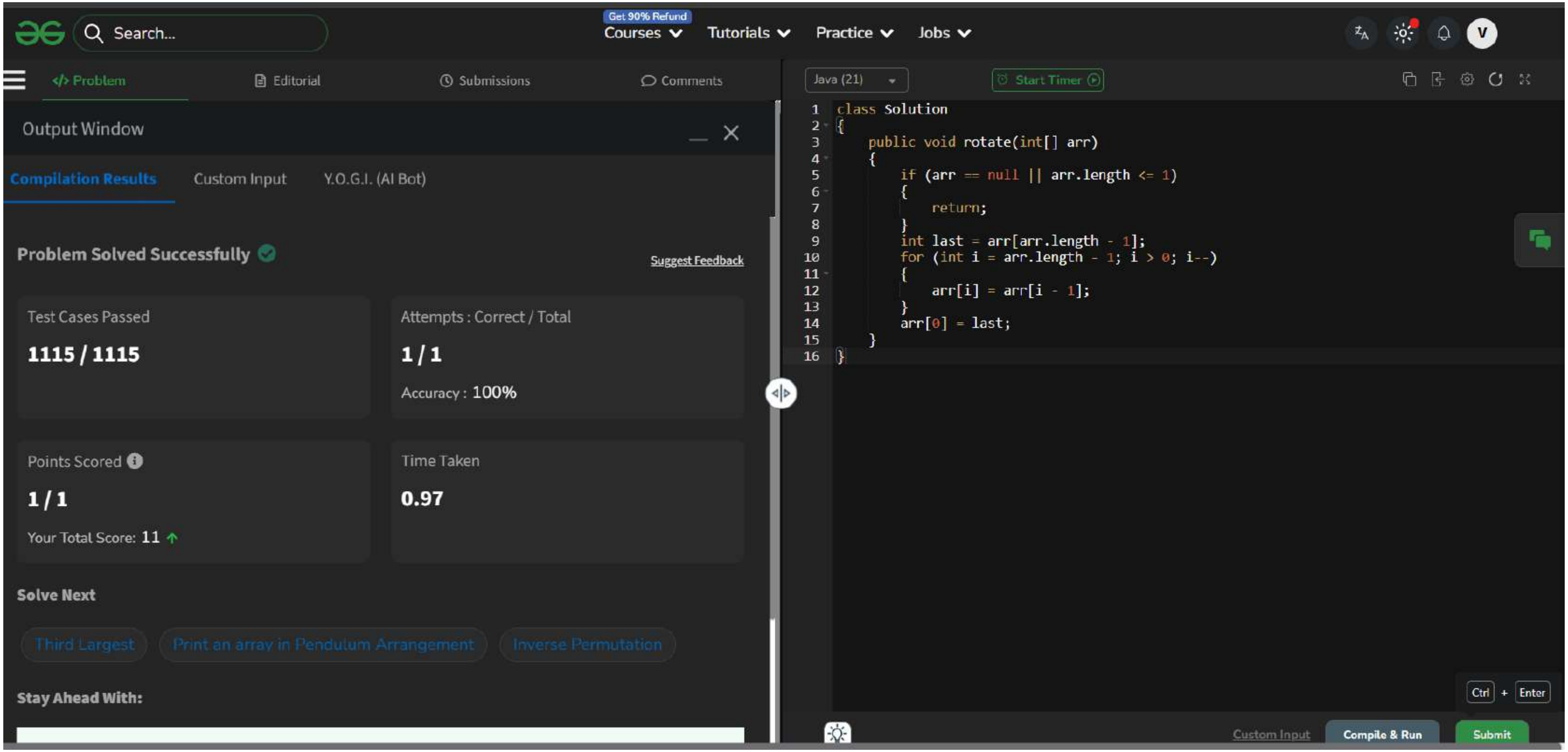
Time Taken

0.9

Solve Next

Last index of One   Pairs with Positive Negative values   Repeated IDs

Stay Ahead With:



Problem Solved Successfully ✓

Test Cases Passed

1115 / 1115

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

1 / 1

Your Total Score: 11 ↑

Time Taken

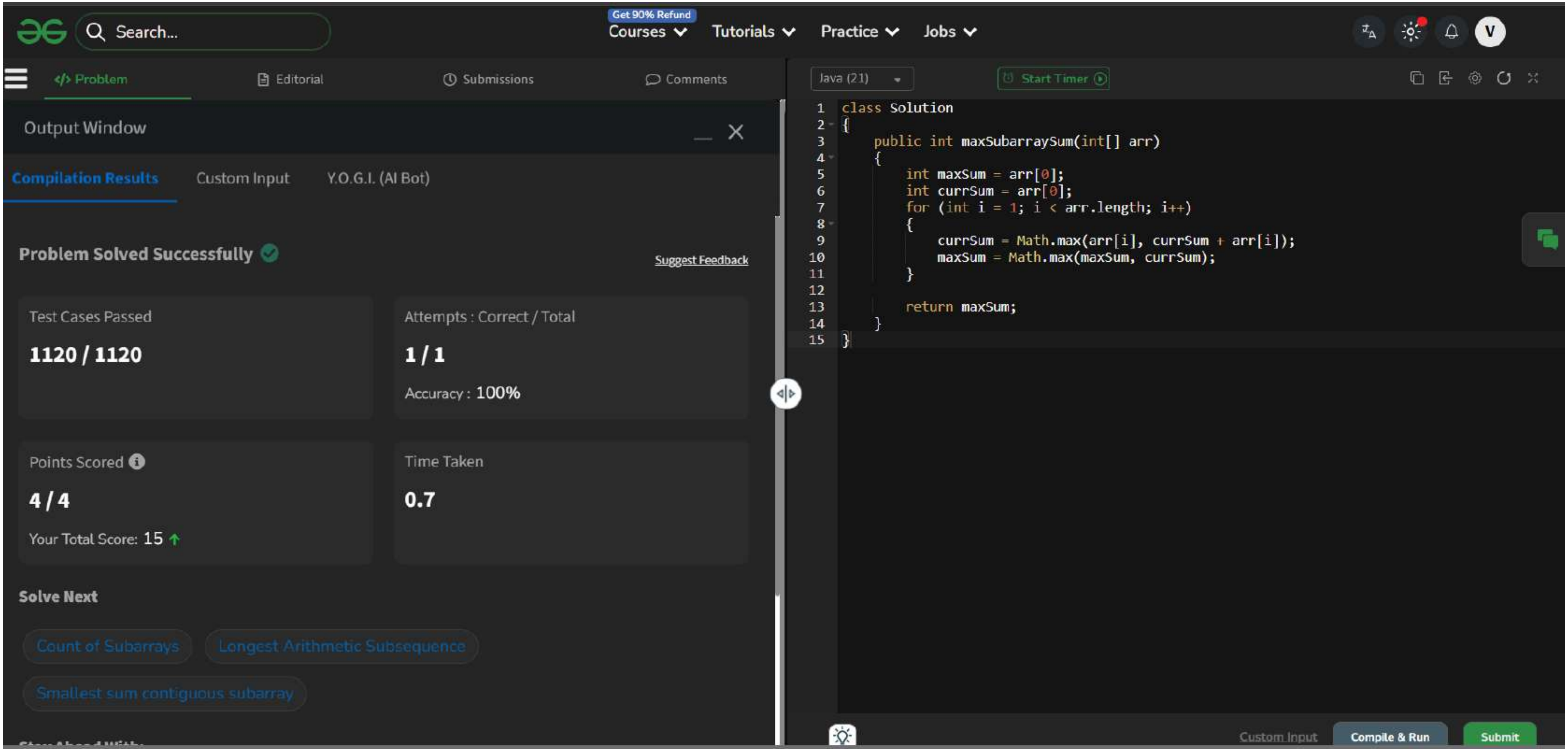
0.97

Solve Next

Third Largest | Print an array in Pendulum Arrangement | Inverse Permutation

Stay Ahead With:

```
class Solution
{
    public void rotate(int[] arr)
    {
        if (arr == null || arr.length <= 1)
        {
            return;
        }
        int last = arr[arr.length - 1];
        for (int i = arr.length - 1; i > 0; i--)
        {
            arr[i] = arr[i - 1];
        }
        arr[0] = last;
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed
1120 / 1120

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

Points Scored
4 / 4
Your Total Score: 15 ↑

Time Taken
0.7

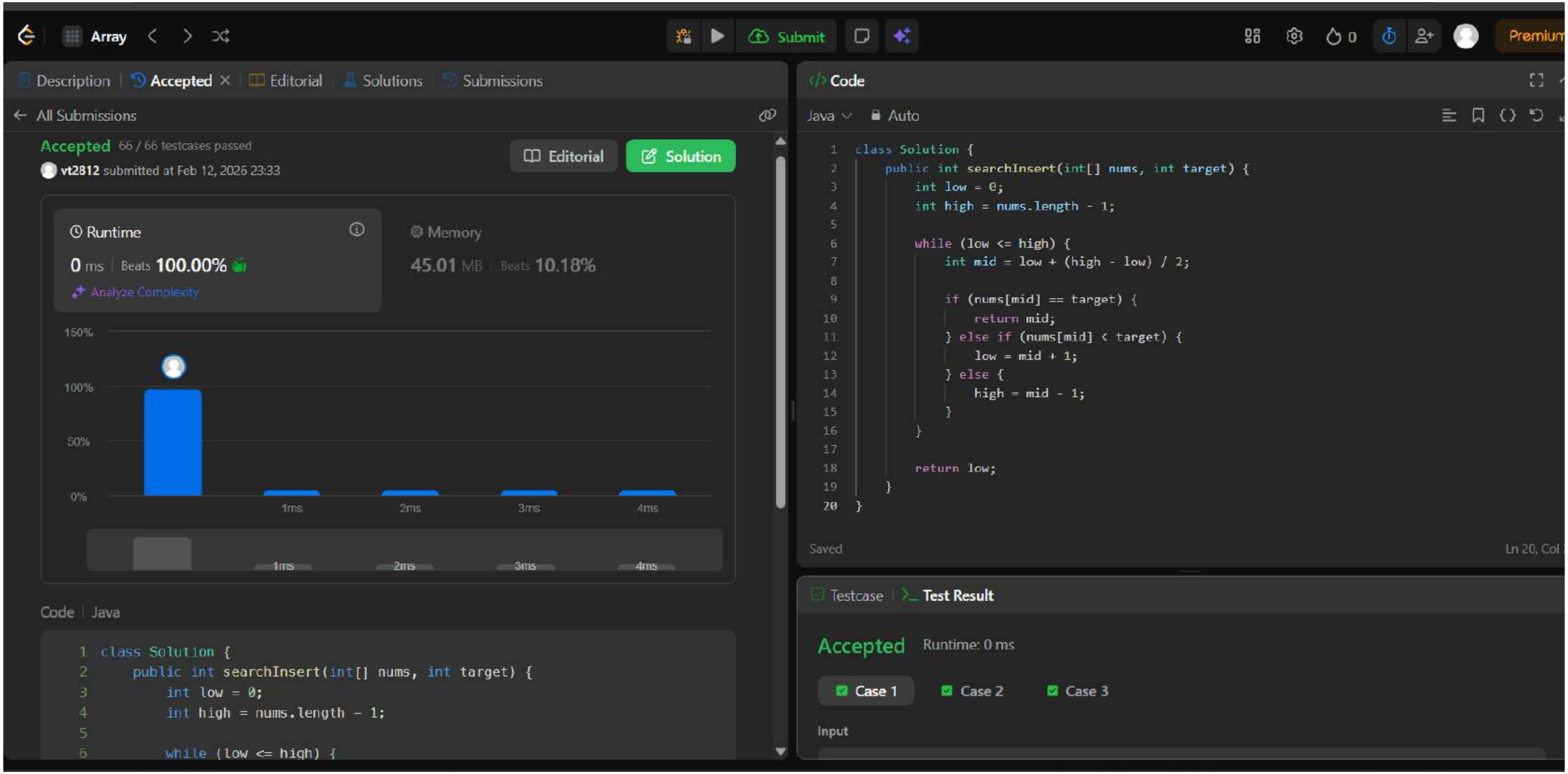
Solve Next

Count of Subarrays

Longest Arithmetic Subsequence

Smallest sum contiguous subarray

```java
class Solution
{
    public int maxSubarraySum(int[] arr)
    {
        int maxSum = arr[0];
        int currSum = arr[0];
        for (int i = 1; i < arr.length; i++)
        {
            currSum = Math.max(arr[i], currSum + arr[i]);
            maxSum = Math.max(maxSum, currSum);
        }

        return maxSum;
    }
}
```



Accepted 66 / 66 testcases passed

vt2812 submitted at Feb 12, 2026 23:33

Runtime

0 ms | Beats 100.00%

Memory

45.01 MB | Beats 10.18%

Code | Java

```java
class Solution {
    public int searchInsert(int[] nums, int target) {
        int low = 0;
        int high = nums.length - 1;

        while (low <= high) {
```

```java
class Solution {
    public int searchInsert(int[] nums, int target) {
        int low = 0;
        int high = nums.length - 1;

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

            if (nums[mid] == target) {
                return mid;
            } else if (nums[mid] < target) {
                low = mid + 1;
            } else {
                high = mid - 1;
            }
        }

        return low;
    }
}
```

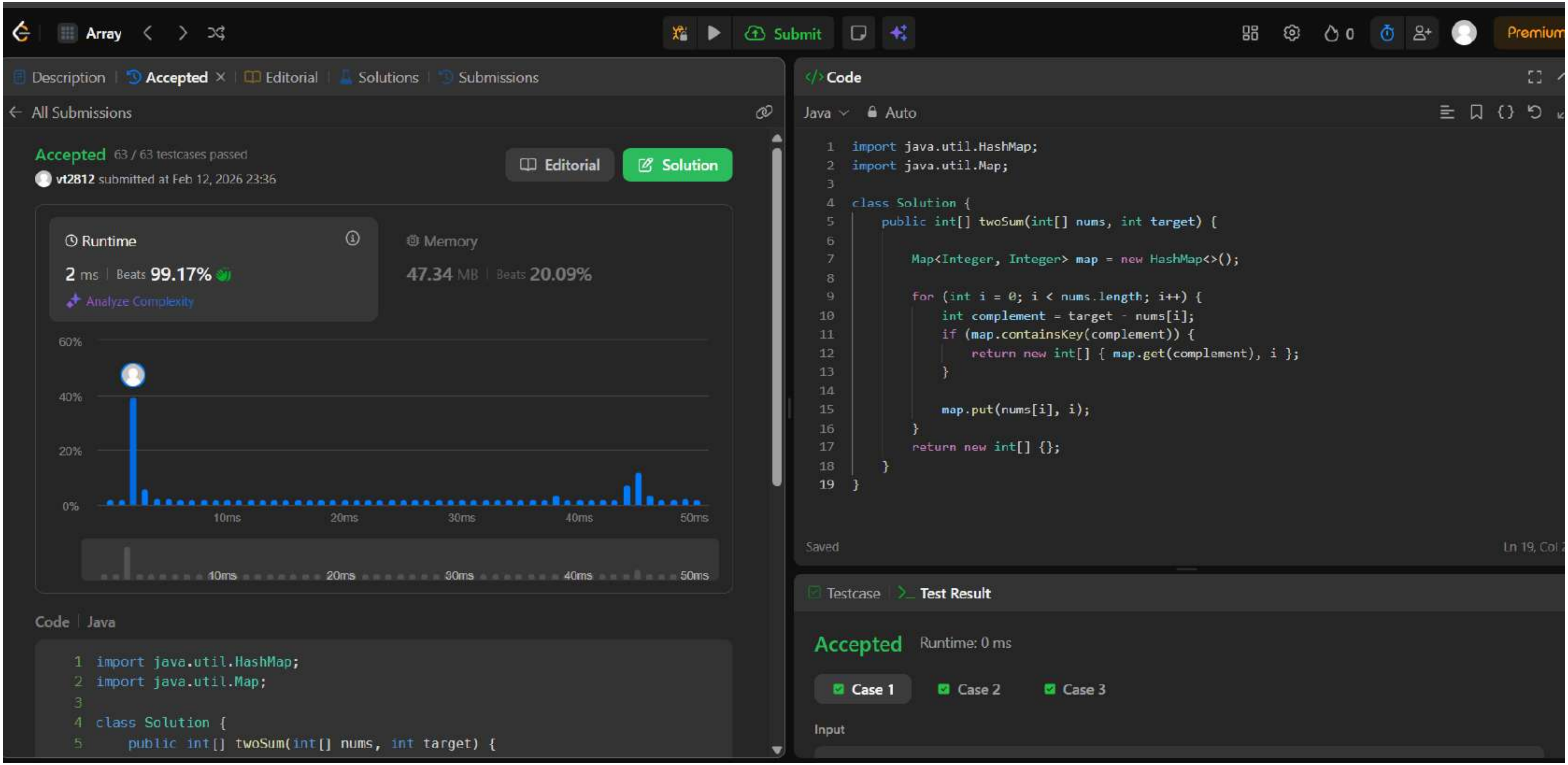
Saved

Ln 20, Col

Accepted Runtime: 0 ms

☑ Case 1 ☑ Case 2 ☑ Case 3

Input



Accepted 63 / 63 testcases passed

vt2812 submitted at Feb 12, 2026 23:36

Runtime: 2 ms | Beats 99.17%

Memory: 47.34 MB | Beats 20.09%

Code | Java

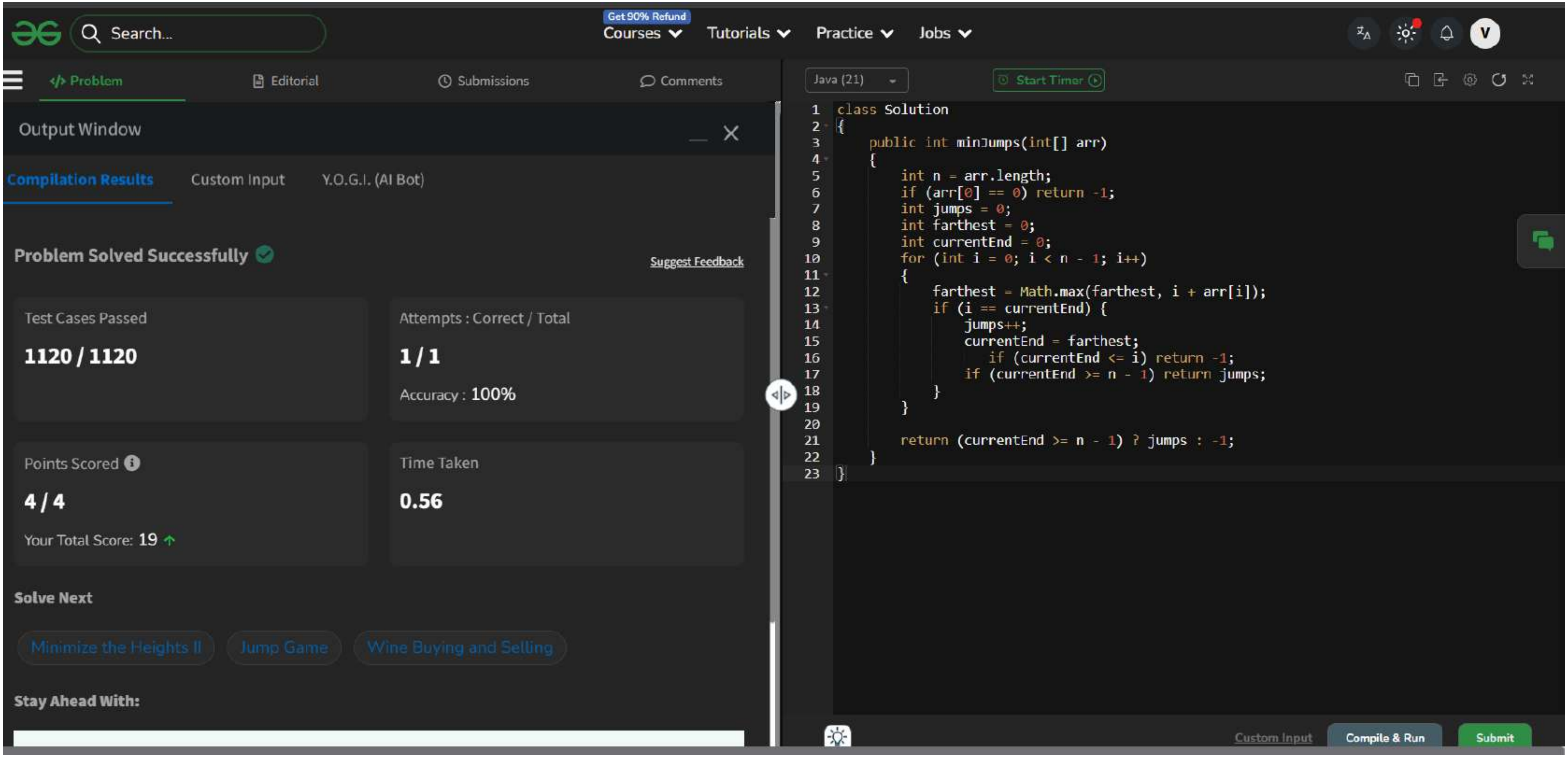
```java
import java.util.HashMap;
import java.util.Map;

class Solution {
    public int[] twoSum(int[] nums, int target) {
```

```java
import java.util.HashMap;
import java.util.Map;

class Solution {
    public int[] twoSum(int[] nums, int target) {

        Map<Integer, Integer> map = new HashMap<>();

        for (int i = 0; i < nums.length; i++) {
            int complement = target - nums[i];
            if (map.containsKey(complement)) {
                return new int[] { map.get(complement), i };
            }

            map.put(nums[i], i);
        }
        return new int[] {};
    }
}
```

Accepted Runtime: 0 ms

☑ Case 1 ☑ Case 2 ☑ Case 3

Input



Output Window

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

Problem Solved Successfully

Suggest Feedback

Test Cases Passed

1120 / 1120

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

4 / 4

Your Total Score: 19

Time Taken

0.56

Solve Next

Minimize the Heights II | Jump Game | Wine Buying and Selling

Stay Ahead With:

```java
class Solution
{
    public int minJumps(int[] arr)
    {
        int n = arr.length;
        if (arr[0] == 0) return -1;
        int jumps = 0;
        int farthest = 0;
        int currentEnd = 0;
        for (int i = 0; i < n - 1; i++)
        {
            farthest = Math.max(farthest, i + arr[i]);
            if (i == currentEnd) {
                jumps++;
                currentEnd = farthest;
                   if (currentEnd <= i) return -1;
                if (currentEnd >= n - 1) return jumps;
            }
        }

        return (currentEnd >= n - 1) ? jumps : -1;
    }
}
```