## Minimize the Heights II

Difficulty: **Medium** Accuracy: **15.06%** Submissions: **770K+** Points: **4** Average Time: **25m**

Given an array **arr[]** denoting heights of **n** towers and a positive integer **k**.

For **each** tower, you must perform **exactly one** of the following operations **exactly once**.

- **Increase** the height of the tower by **k**
- **Decrease** the height of the tower by **k**

Find out the **minimum** possible difference between the height of the shortest and tallest towers after you have modified each tower.

You can find a slight modification of the problem here.
**Note:** It is **compulsory** to increase or decrease the height by k for each tower. After the operation,

Java (21) Start Timer

```java
class Solution {
    public int getMinDiff(int[] arr, int k) {
     int n = arr.length;
        if (n == 1) return 0;
        Arrays.sort(arr);
        int res = arr[n - 1] - arr[0];
        int smallest = arr[0] + k;
        int largest = arr[n - 1] - k;

        for (int i = 0; i < n - 1; i++) {
            int minHeight = Math.min(smallest, arr[i + 1] - k);
            int maxHeight = Math.max(largest, arr[i] + k);
            if (minHeight < 0) continue;
            res = Math.min(res, maxHeight - minHeight);
        }

        return res;

    }
}
```

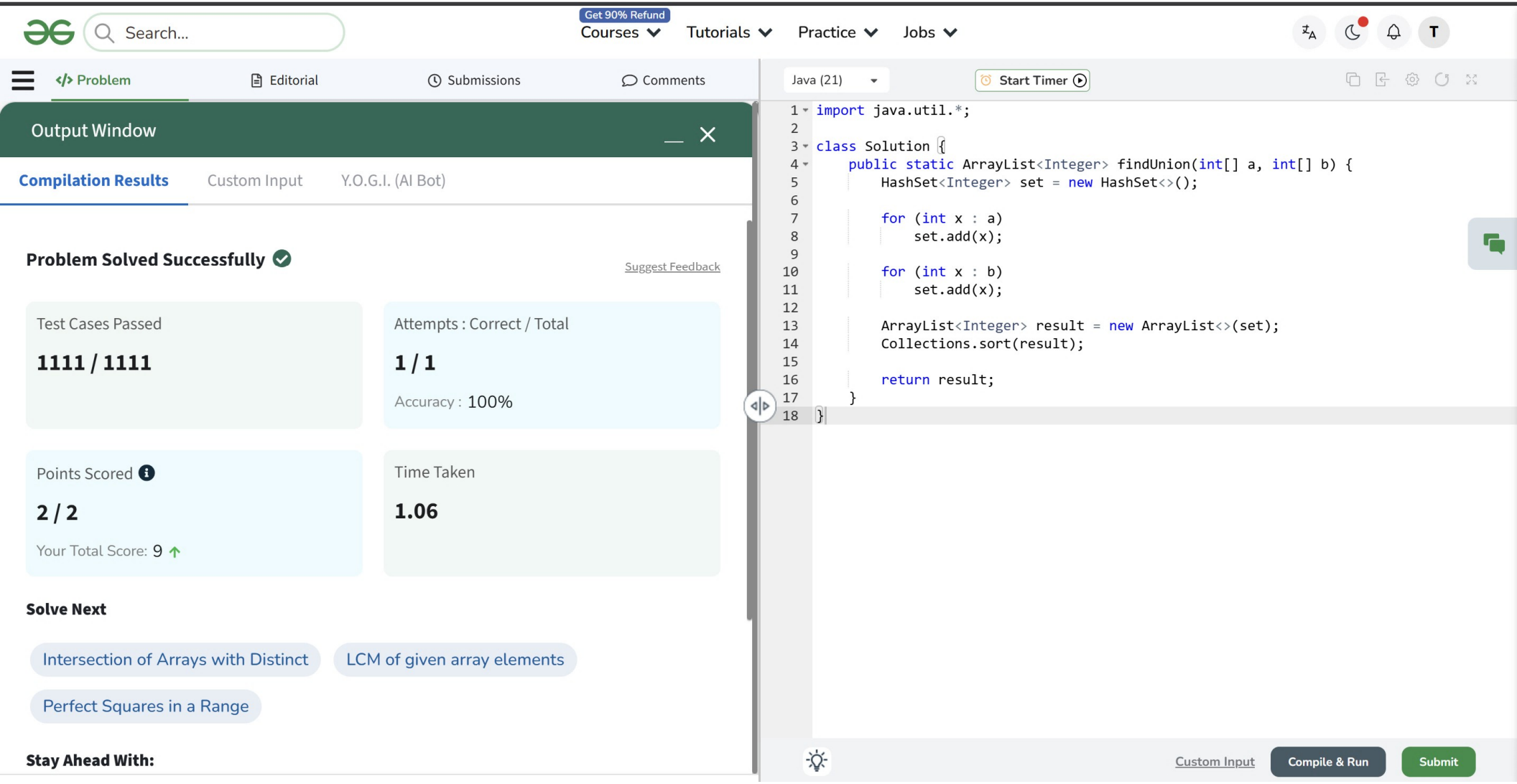
Output Window

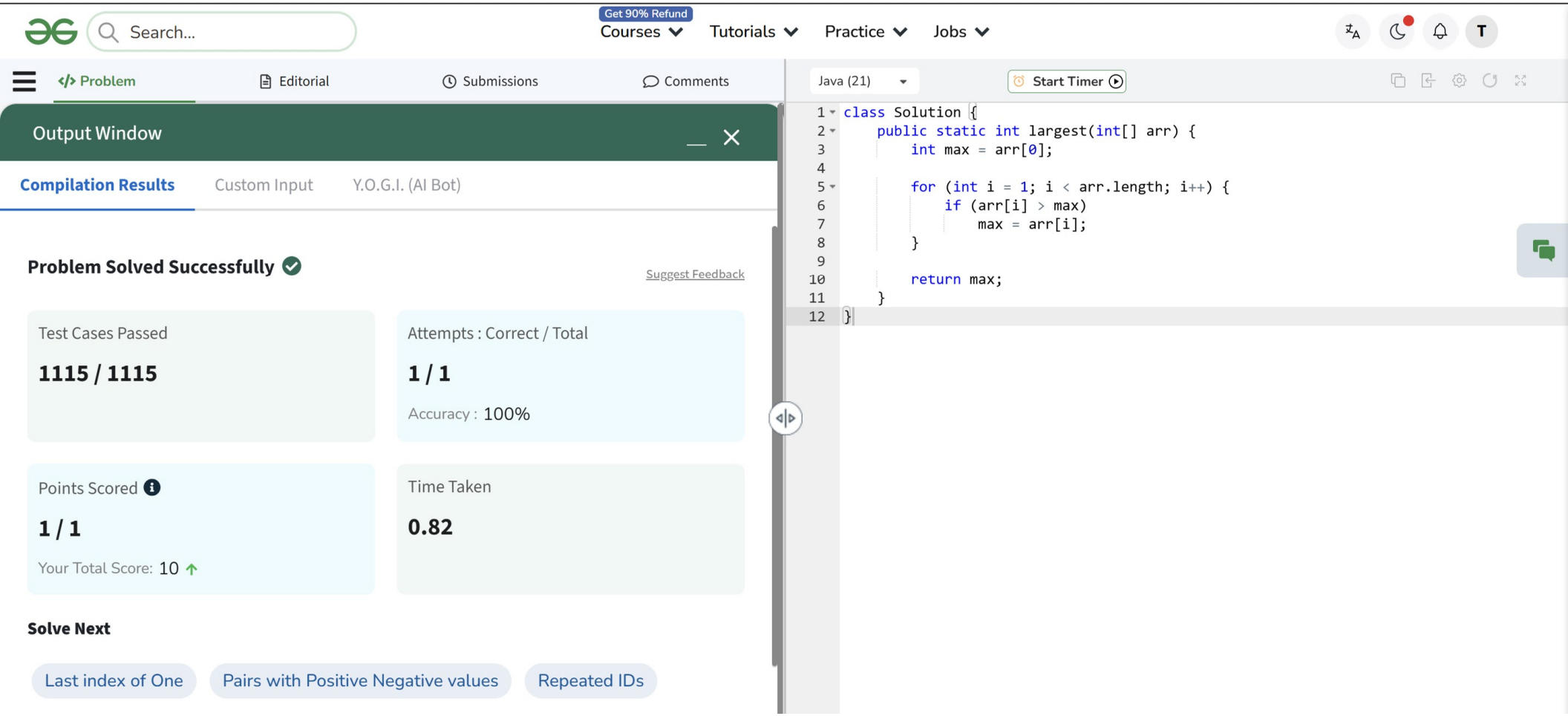
Compilation Results | Custom Input | Y.O.G.I. (AI Bot)

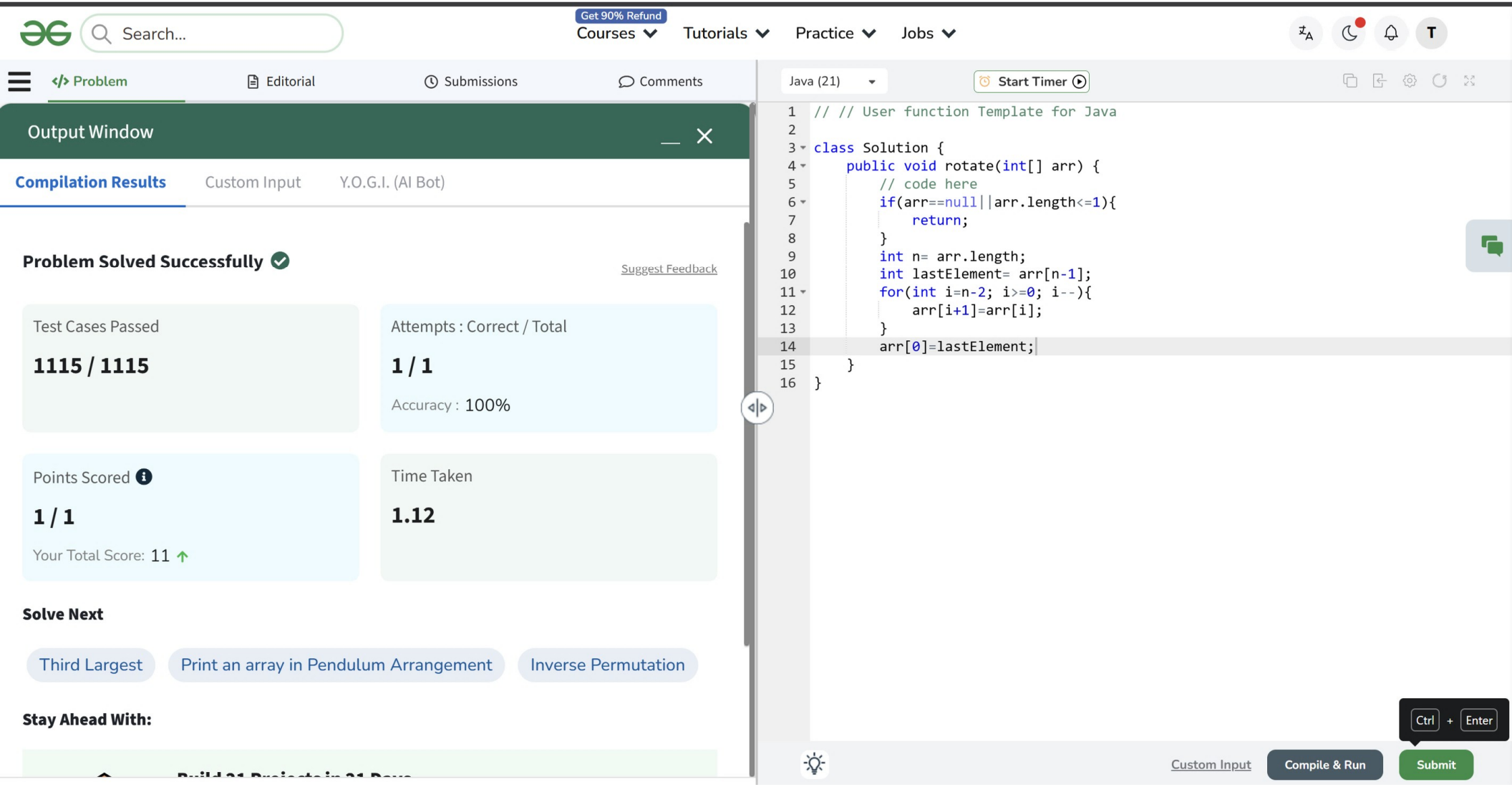
**Problem Solved Successfully** Suggest Feedback

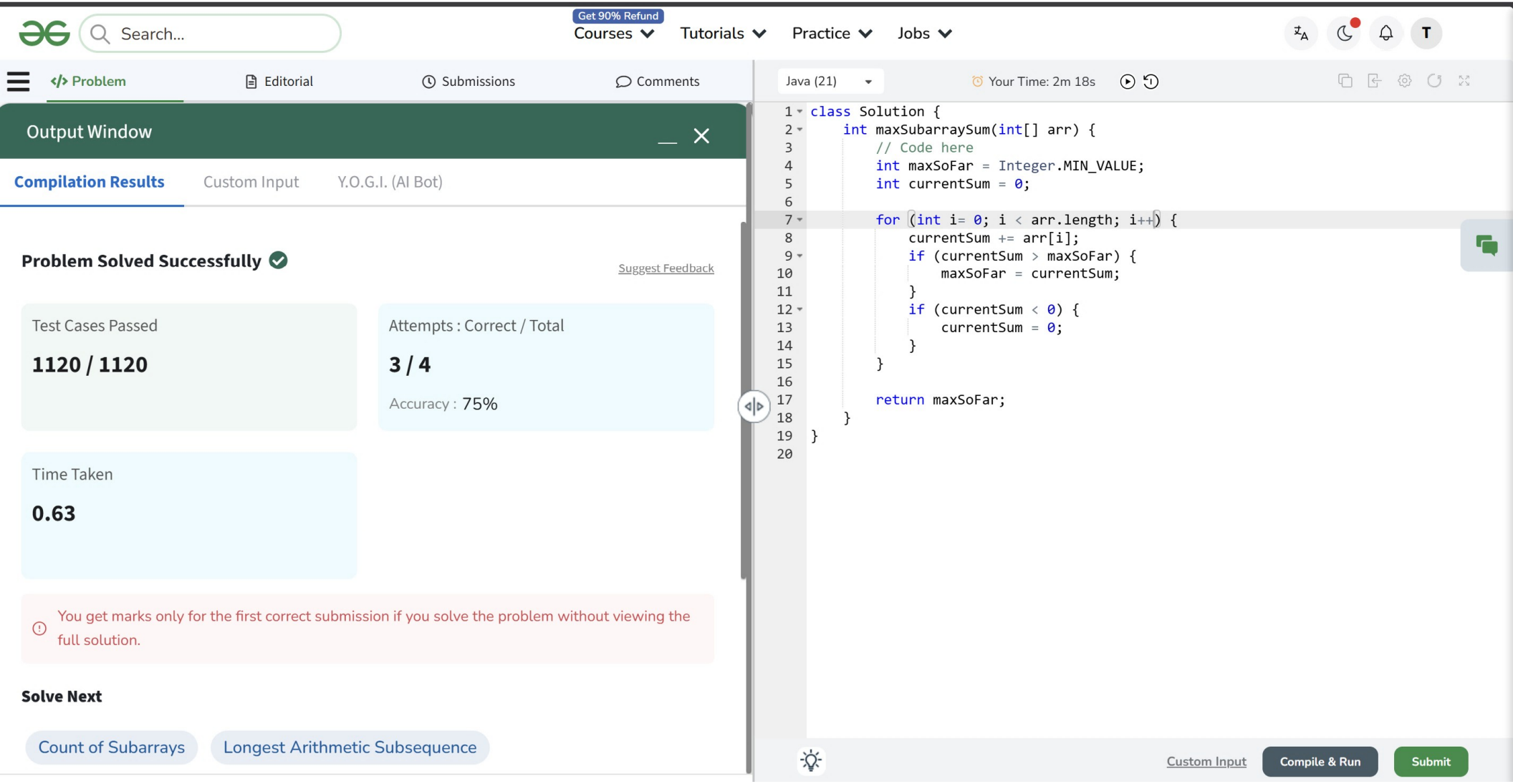
| Test Cases Passed | Attempts : Correct / Total |
|---|---|
| **1115 / 1115** | **2 / 2** |
| | Accuracy : 100% |

Custom Input Compile & Run Submit


Search...
Get 90% Refund
Courses
Tutorials
Practice
Jobs
T
Problem
Editorial
Submissions
Comments
Java (21)
Start Timer
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: 9
Time Taken
1.06
Solve Next
Intersection of Arrays with Distinct
LCM of given array elements
Perfect Squares in a Range
Stay Ahead With:
1 import java.util.*;
2
3 class Solution {
4 public static ArrayList<Integer> findUnion(int[] a, int[] b) {
5 HashSet<Integer> set = new HashSet<>();
6
7 for (int x : a)
8 set.add(x);
9
10 for (int x : b)
11 set.add(x);
12
13 ArrayList<Integer> result = new ArrayList<>(set);
14 Collections.sort(result);
15
16 return result;
17 }
18 }
Custom Input
Compile & Run
Submit


Search...
Get 90% Refund
Courses
Tutorials
Practice
Jobs
T
Problem
Editorial
Submissions
Comments
Java (21)
Start Timer
Output Window
Compilation Results
Custom Input
Y.O.G.I. (AI Bot)
Problem Solved Successfully
Suggest Feedback
Test Cases Passed
1115 / 1115
Attempts : Correct / Total
1 / 1
Accuracy : 100%
Points Scored
1 / 1
Your Total Score: 10
Time Taken
0.82
Solve Next
Last index of One
Pairs with Positive Negative values
Repeated IDs
class Solution {
public static 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;
}
}


Search...
Get 90% Refund
Courses
Tutorials
Practice
Jobs
T
Problem
Editorial
Submissions
Comments
Output Window
Compilation Results
Custom Input
Y.O.G.I. (AI Bot)
Problem Solved Successfully
Suggest Feedback
Test Cases Passed
1115 / 1115
Attempts : Correct / Total
1 / 1
Accuracy : 100%
Points Scored
1 / 1
Your Total Score: 11
Time Taken
1.12
Solve Next
Third Largest
Print an array in Pendulum Arrangement
Inverse Permutation
Stay Ahead With:
Java (21)
Start Timer
// // User function Template for Java
class Solution {
public void rotate(int[] arr) {
// code here
if(arr==null||arr.length<=1){
return;
}
int n= arr.length;
int lastElement= arr[n-1];
for(int i=n-2; i>=0; i--){
arr[i+1]=arr[i];
}
arr[0]=lastElement;
}
}
Ctrl + Enter
Custom Input
Compile & Run
Submit


Search...
Get 90% Refund
Courses
Tutorials
Practice
Jobs
T
Problem
Editorial
Submissions
Comments
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
3 / 4
Accuracy : 75%
Time Taken
0.63
You get marks only for the first correct submission if you solve the problem without viewing the full solution.
Solve Next
Count of Subarrays
Longest Arithmetic Subsequence
Java (21)
Your Time: 2m 18s
1 class Solution {
2 int maxSubarraySum(int[] arr) {
3 // Code here
4 int maxSoFar = Integer.MIN_VALUE;
5 int currentSum = 0;
6
7 for (int i= 0; i < arr.length; i++) {
8 currentSum += arr[i];
9 if (currentSum > maxSoFar) {
10 maxSoFar = currentSum;
11 }
12 if (currentSum < 0) {
13 currentSum = 0;
14 }
15 }
16
17 return maxSoFar;
18 }
19 }
20
Custom Input
Compile & Run
Submit