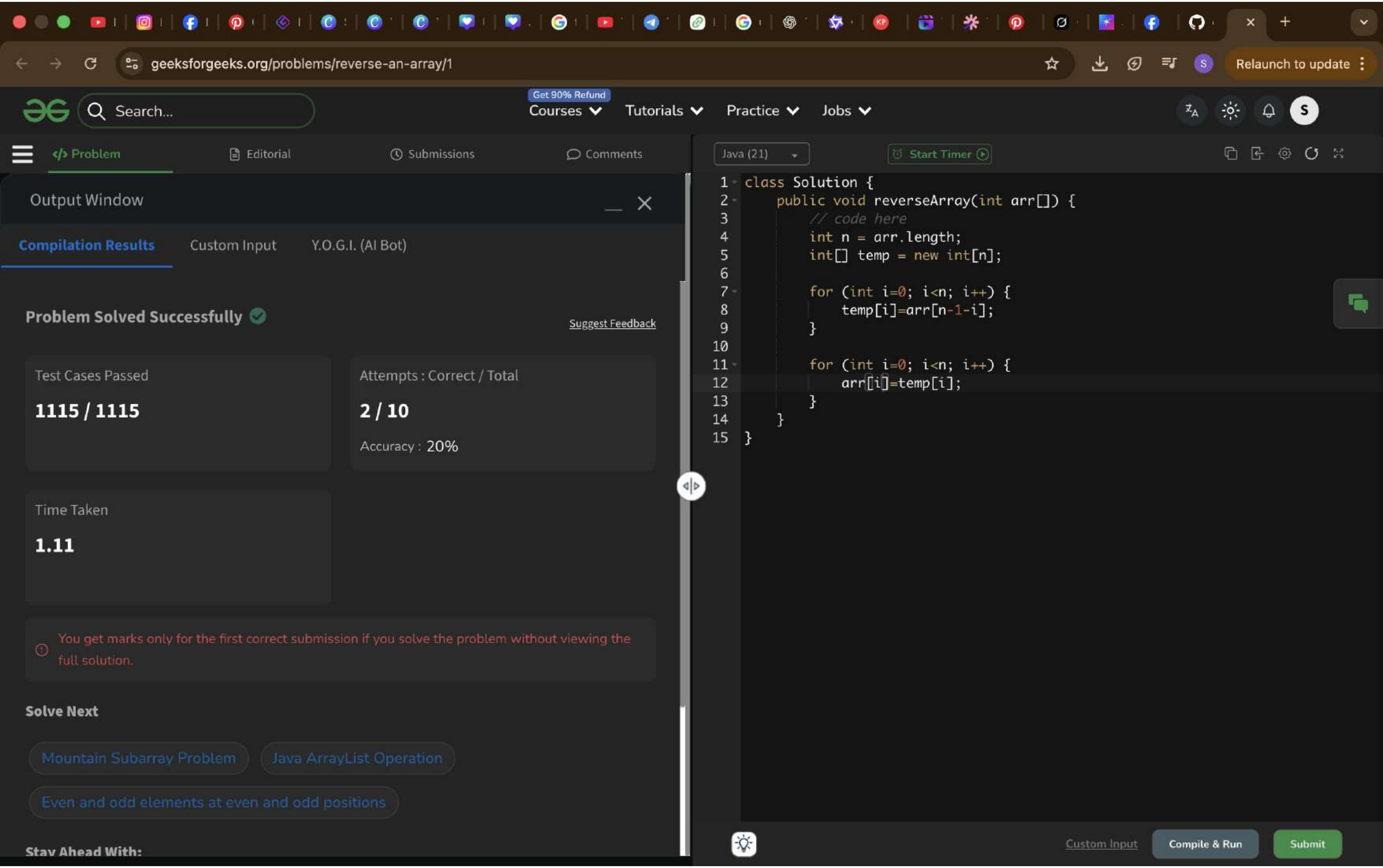


Output Window

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

Problem Solved Successfully ✓

Suggest Feedback

Test Cases Passed	Attempts : Correct / Total
1115 / 1115	**2 / 10** Accuracy : 20%

Time Taken

1.11

You get marks only for the first correct submission if you solve the problem without viewing the full solution.

Solve Next

Mountain Subarray Problem

Java ArrayList Operation

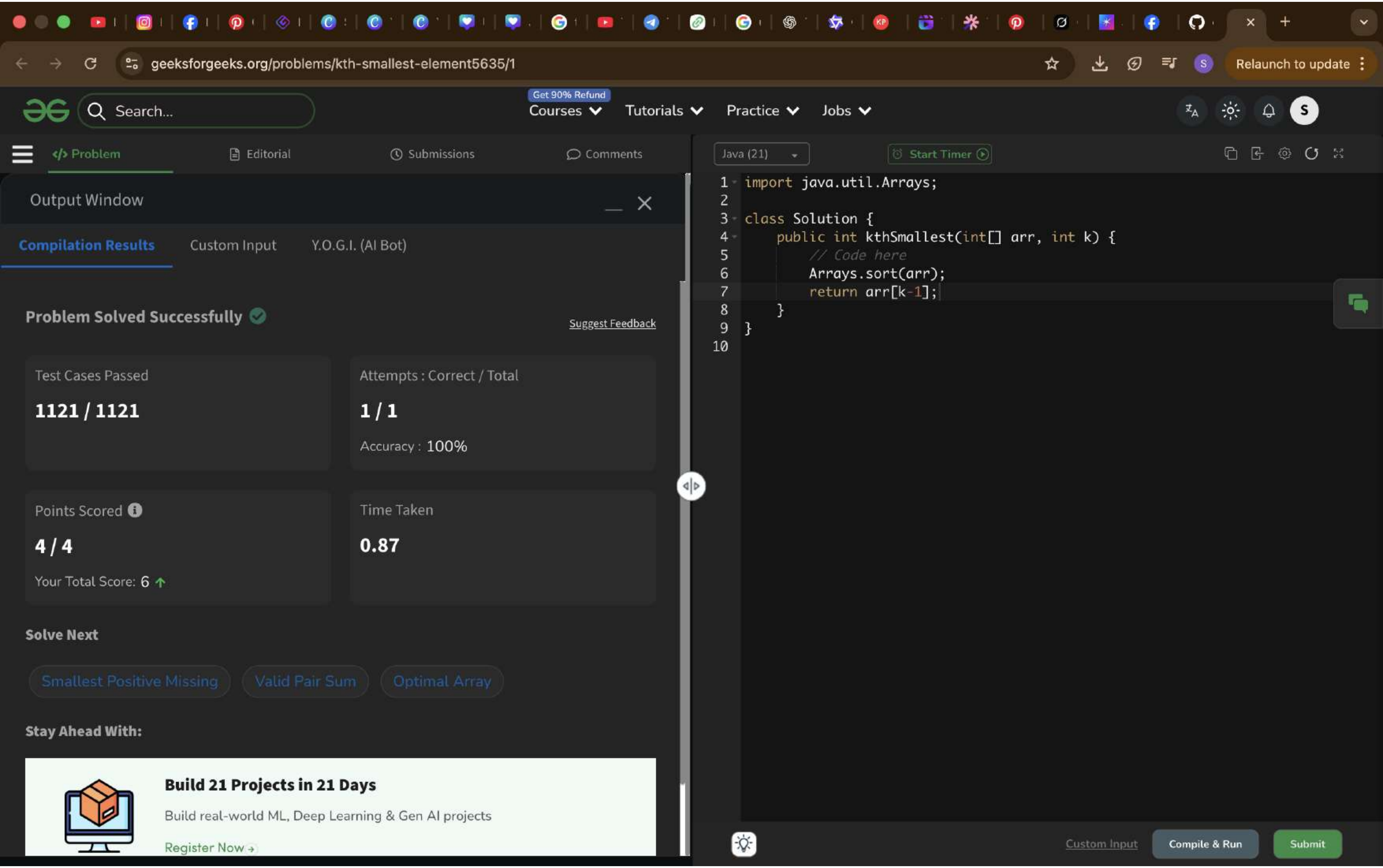
Even and odd elements at even and odd positions

Stay Ahead With:

```java
class Solution {
    public void reverseArray(int arr[]) {
        // code here
        int n = arr.length;
        int[] temp = new int[n];

        for (int i=0; i<n; i++) {
            temp[i]=arr[n-1-i];
        }

        for (int i=0; i<n; i++) {
            arr[i]=temp[i];
        }
    }
}
```



```java
import java.util.Arrays;

class Solution {
    public int kthSmallest(int[] arr, int k) {
        // Code here
        Arrays.sort(arr);
        return arr[k-1];
    }
}
```

Problem Solved Successfully ✓

Suggest Feedback

Test Cases Passed

1121 / 1121

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

4 / 4

Your Total Score: 6 ↑

Time Taken

0.87

Solve Next

Smallest Positive Missing

Valid Pair Sum

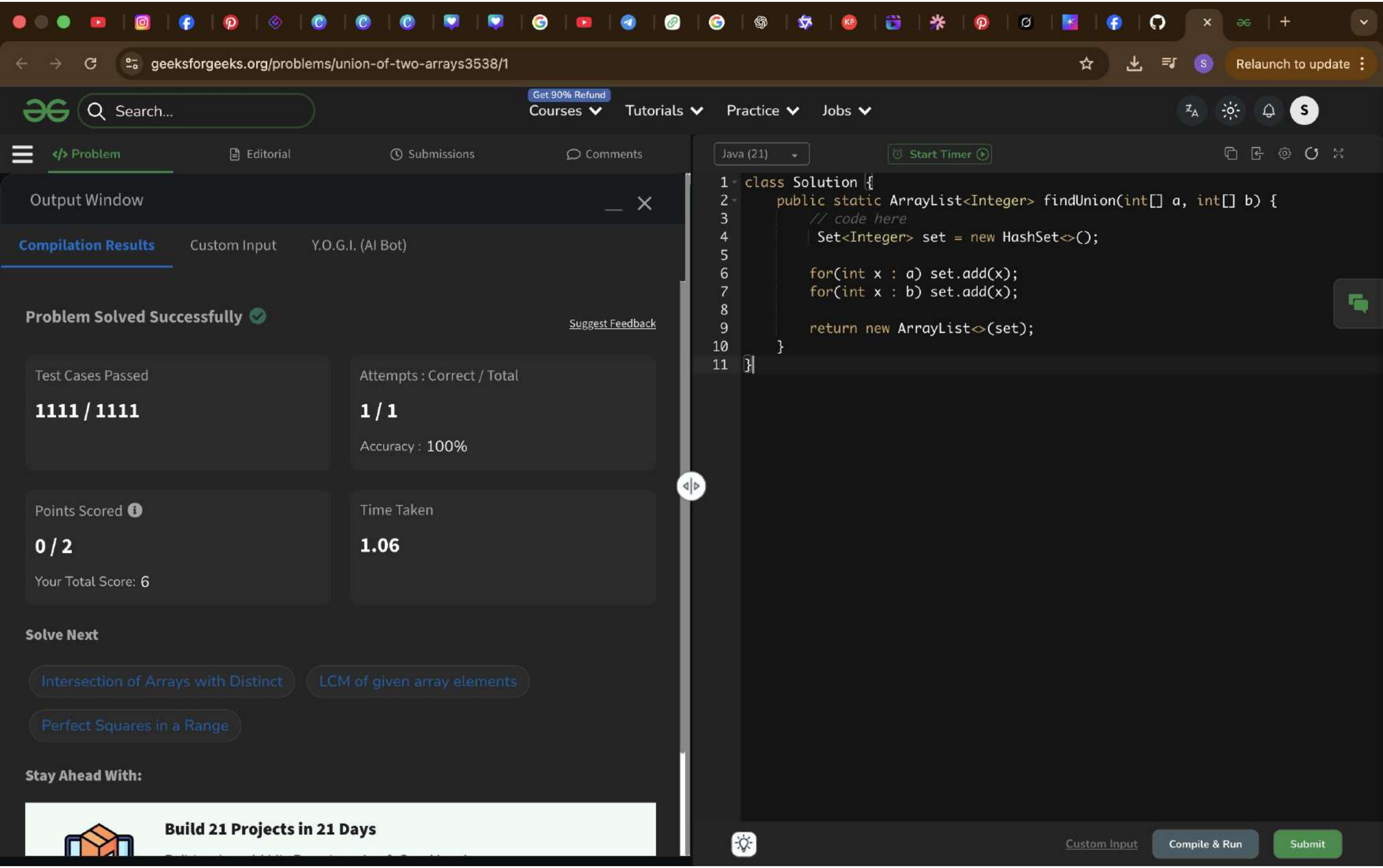
Optimal Array

Stay Ahead With:

Build 21 Projects in 21 Days

Build real-world ML, Deep Learning & Gen AI projects

Register Now →



Problem Solved Successfully ✓

Test Cases Passed

1111 / 1111

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

0 / 2

Your Total Score: 6

Time Taken

1.06

Solve Next

Intersection of Arrays with Distinct

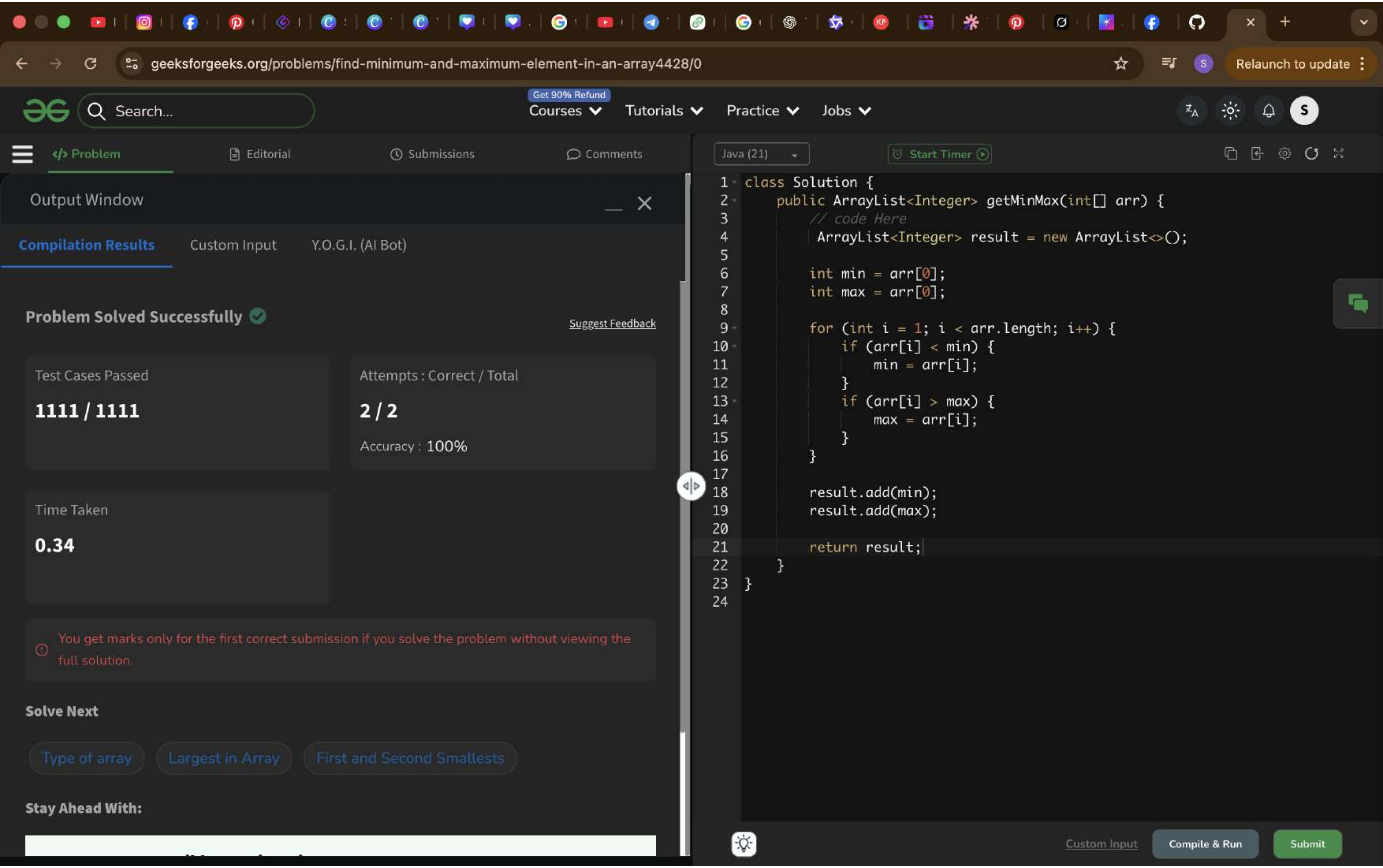
LCM of given array elements

Perfect Squares in a Range

Stay Ahead With:

Build 21 Projects in 21 Days

```java
class Solution {
    public static ArrayList<Integer> findUnion(int[] a, int[] b) {
        // code here
         Set<Integer> set = new HashSet<>();
        
        for(int x : a) set.add(x);
        for(int x : b) set.add(x);
        
        return new ArrayList<>(set);
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed

1111 / 1111

Attempts : Correct / Total

2 / 2

Accuracy : 100%

Time Taken

0.34

You get marks only for the first correct submission if you solve the problem without viewing the full solution.

Solve Next

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

Stay Ahead With:

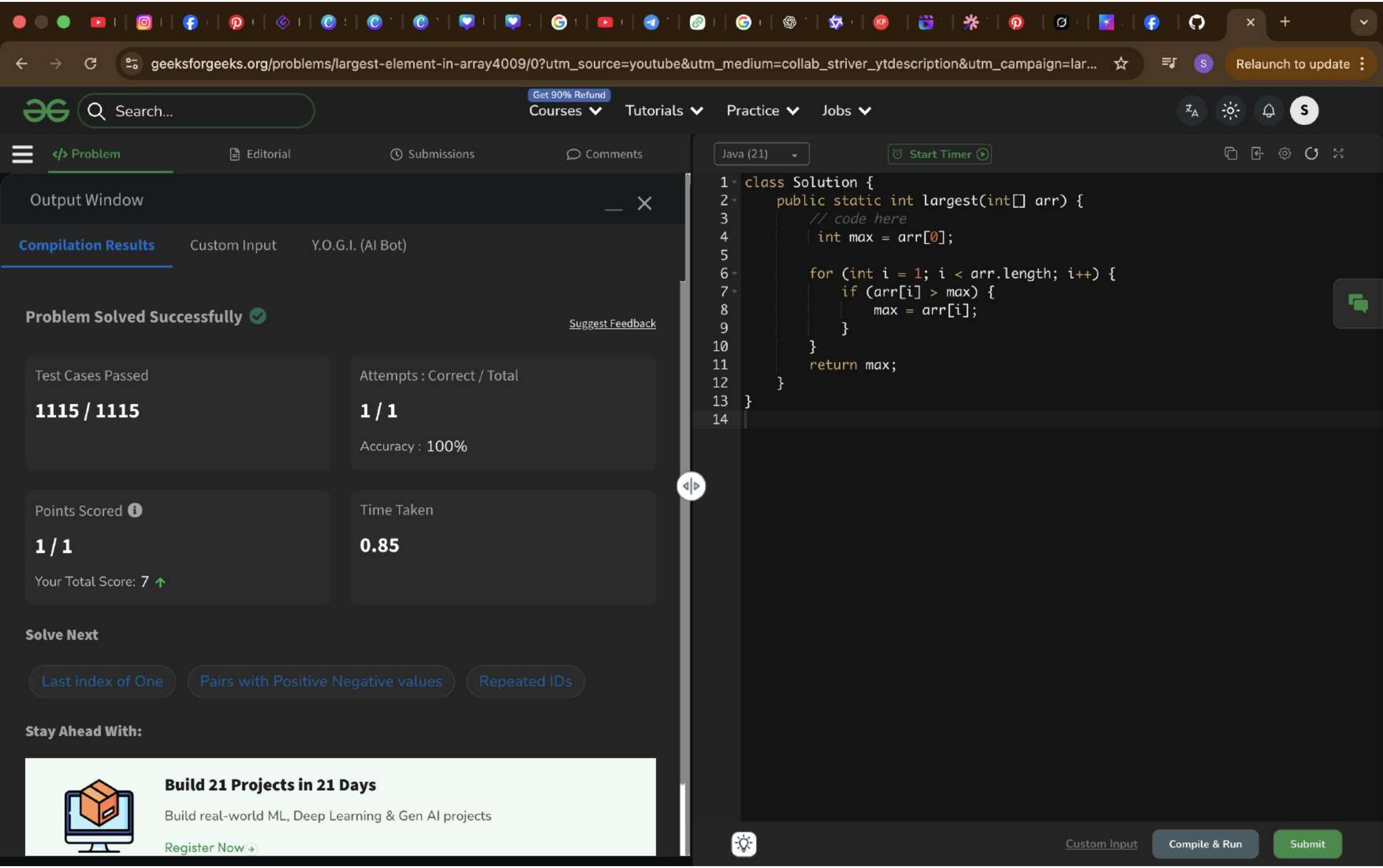
```java
class Solution {
    public ArrayList<Integer> getMinMax(int[] arr) {
        // code Here
         ArrayList<Integer> result = new ArrayList<>();

        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];
            }
        }

        result.add(min);
        result.add(max);

        return result;
    }
}
```



Problem Solved Successfully ✓

Test Cases Passed

1115 / 1115

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

1 / 1

Your Total Score: 7 ↑

Time Taken

0.85

Solve Next

Last index of One | Pairs with Positive Negative values | Repeated IDs

Stay Ahead With:

Build 21 Projects in 21 Days

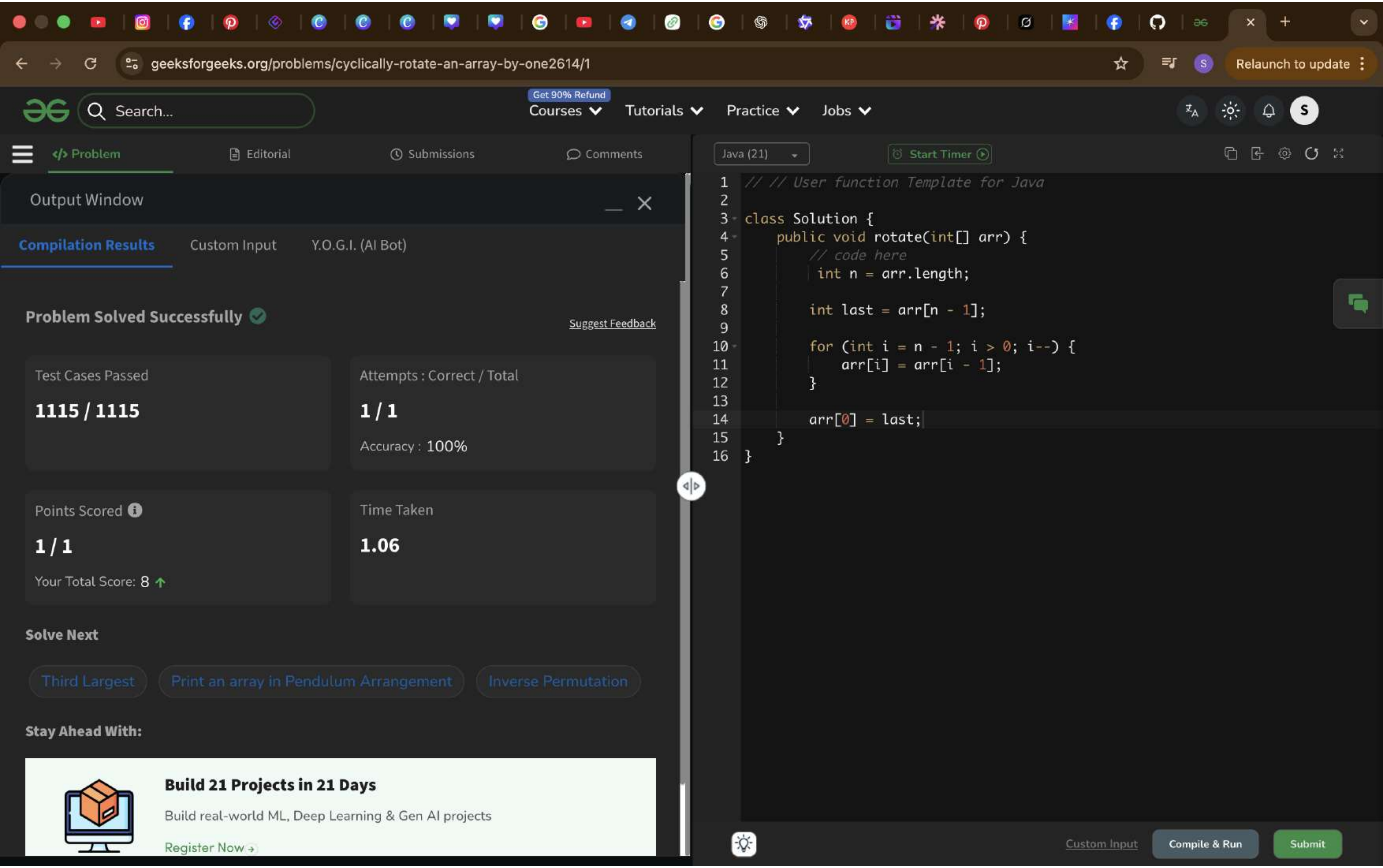
Build real-world ML, Deep Learning & Gen AI projects

Register Now →

```java
class Solution {
    public static int largest(int[] arr) {
        // code here
         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: 8 ↑

Time Taken

1.06

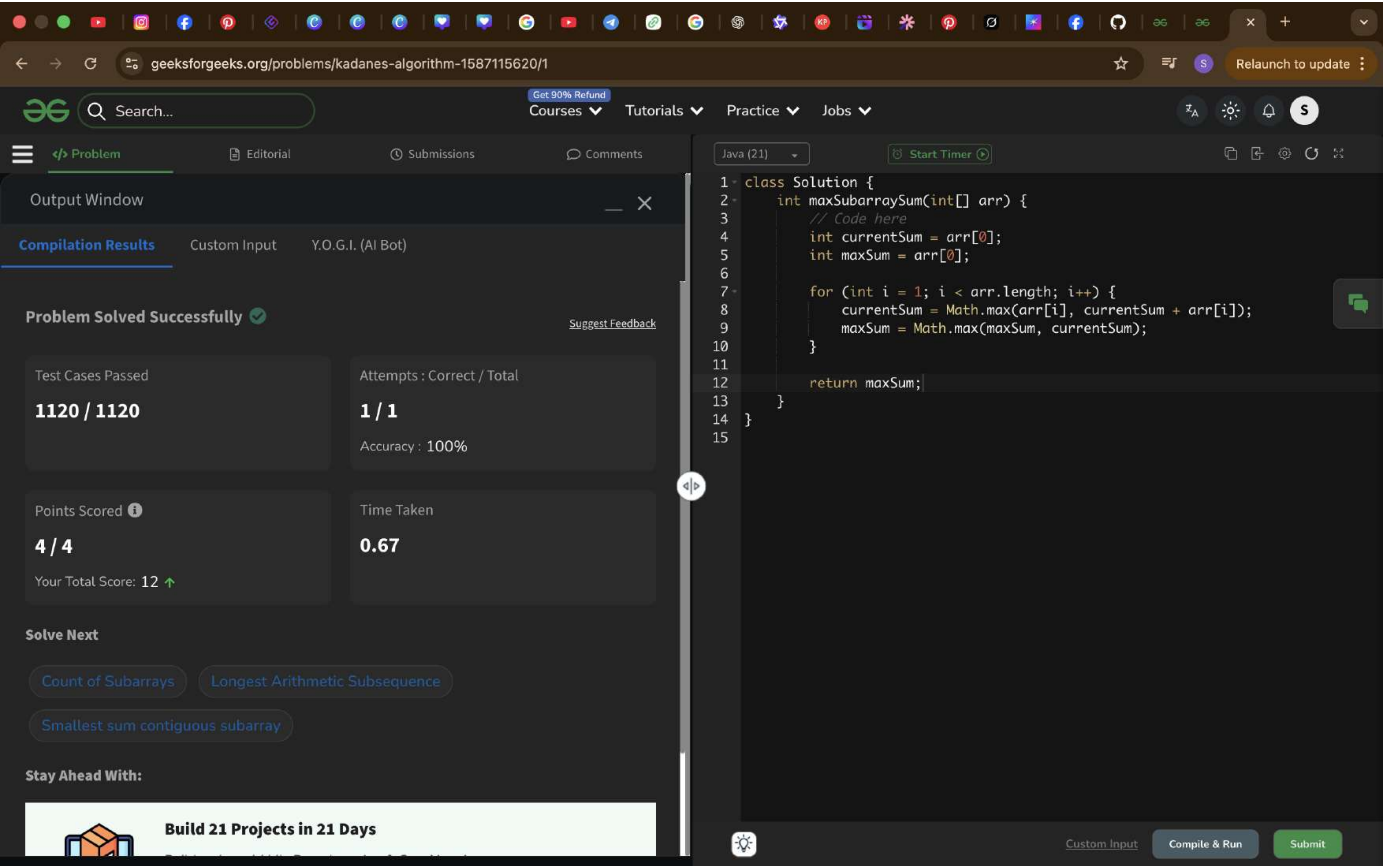
Solve Next

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

Stay Ahead With:

```java
// // User function Template for Java

class Solution {
    public void rotate(int[] arr) {
        // code here
         int n = arr.length;
        
        int last = arr[n - 1];
        
        for (int i = n - 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: 12 ↑

Time Taken

0.67

Solve Next

Count of Subarrays

Longest Arithmetic Subsequence

Smallest sum contiguous subarray

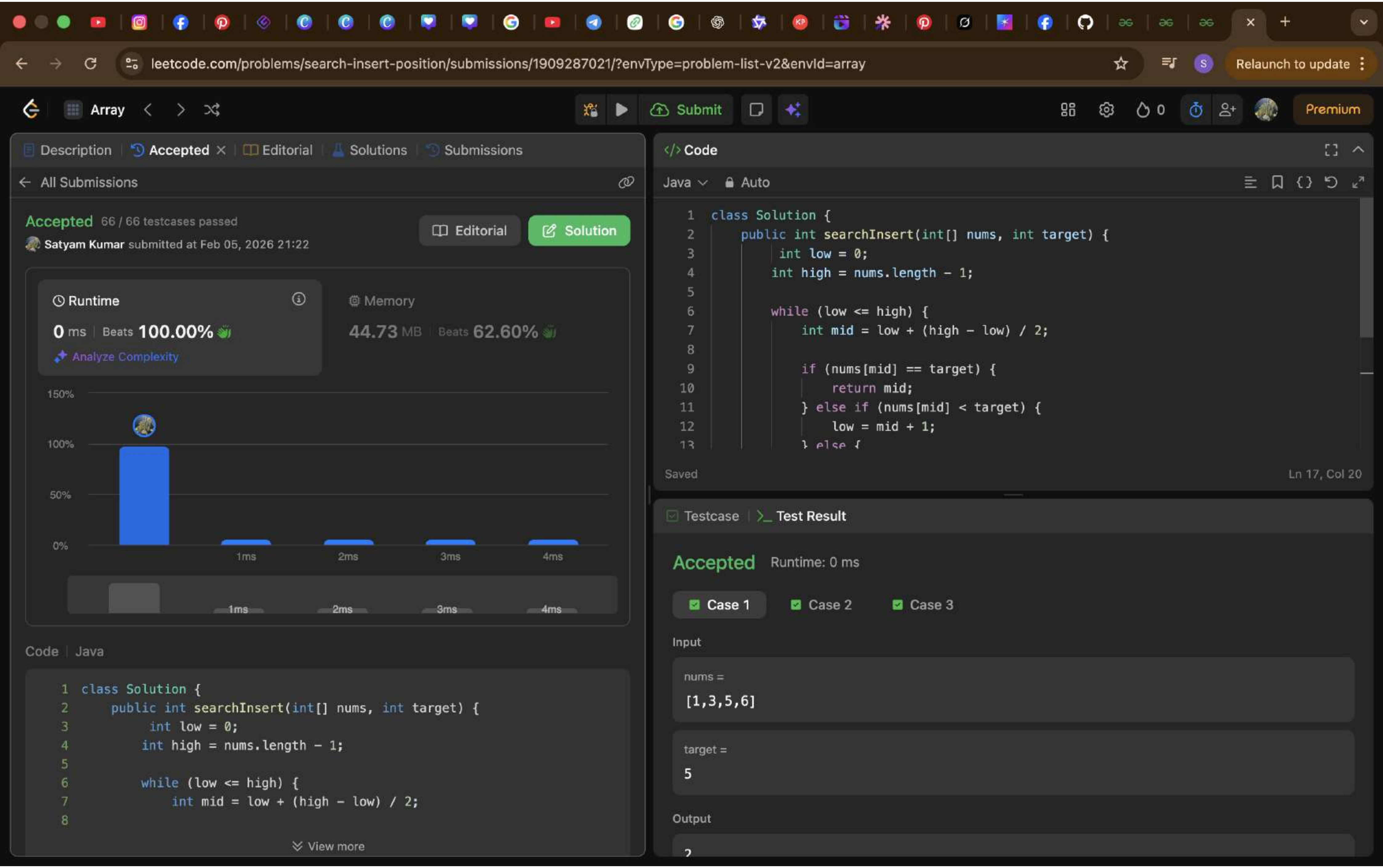
Stay Ahead With:

Build 21 Projects in 21 Days

```java
class Solution {
    int maxSubarraySum(int[] arr) {
        // Code here
        int currentSum = arr[0];
        int maxSum = arr[0];

        for (int i = 1; i < arr.length; i++) {
            currentSum = Math.max(arr[i], currentSum + arr[i]);
            maxSum = Math.max(maxSum, currentSum);
        }

        return maxSum;
    }
}
```



Accepted 66 / 66 testcases passed

Satyam Kumar submitted at Feb 05, 2026 21:22

Runtime: 0 ms | Beats 100.00%

Memory: 44.73 MB | Beats 62.60%

Code | 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;

```

```
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 {
```

Accepted Runtime: 0 ms

☑ Case 1 ☑ Case 2 ☑ Case 3

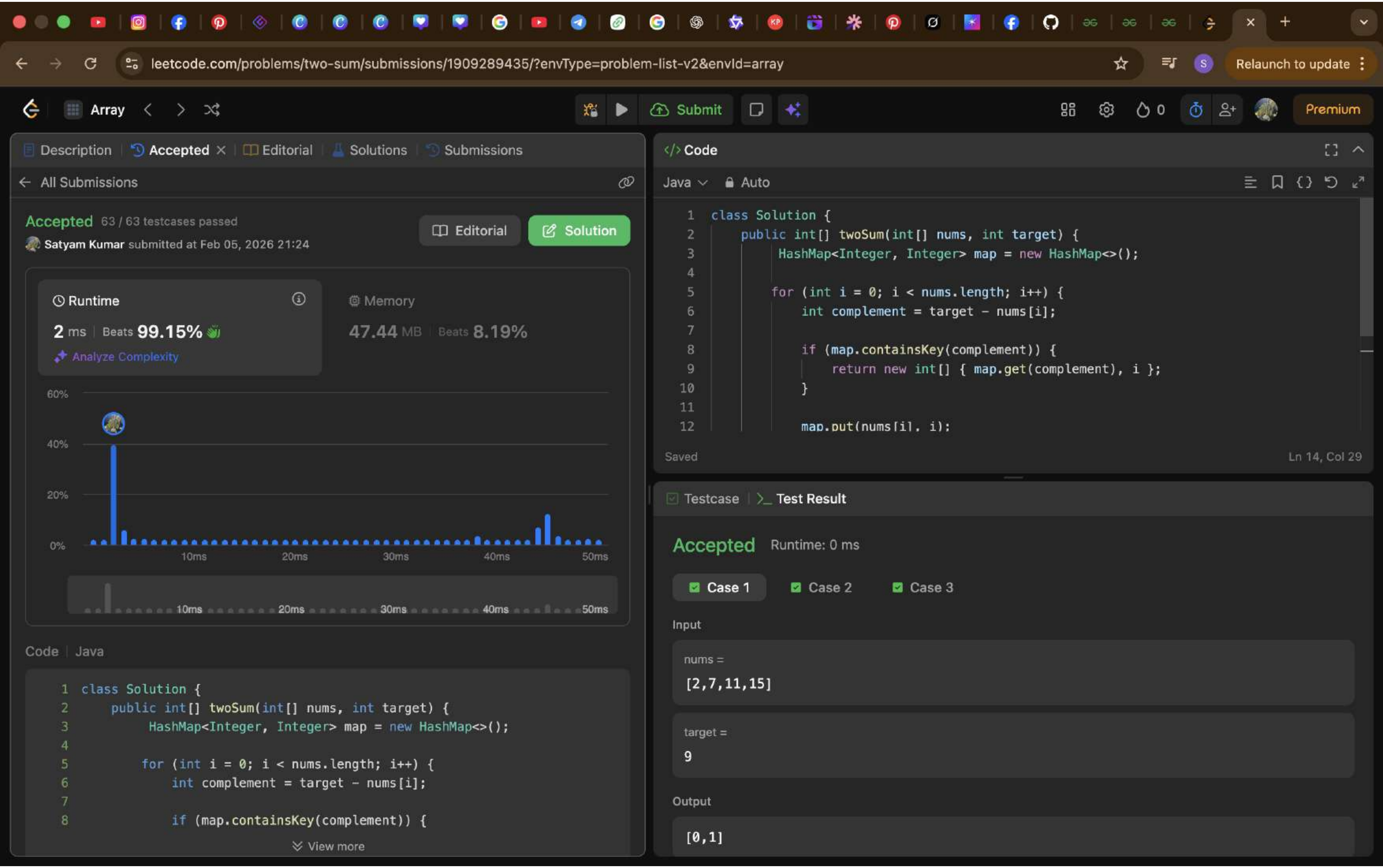
Input

nums =
[1,3,5,6]

target =
5

Output

2



Accepted 63 / 63 testcases passed

Satyam Kumar submitted at Feb 05, 2026 21:24

Runtime

2 ms | Beats 99.15%

Memory

47.44 MB | Beats 8.19%

Code | Java

```java
class Solution {
    public int[] twoSum(int[] nums, int target) {
        HashMap<Integer, Integer> map = new HashMap<>();

        for (int i = 0; i < nums.length; i++) {
            int complement = target - nums[i];

            if (map.containsKey(complement)) {
```

```java
class Solution {
    public int[] twoSum(int[] nums, int target) {
        HashMap<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);
```

Saved   Ln 14, Col 29

Accepted Runtime: 0 ms

☑ Case 1 ☑ Case 2 ☑ Case 3

Input

nums =

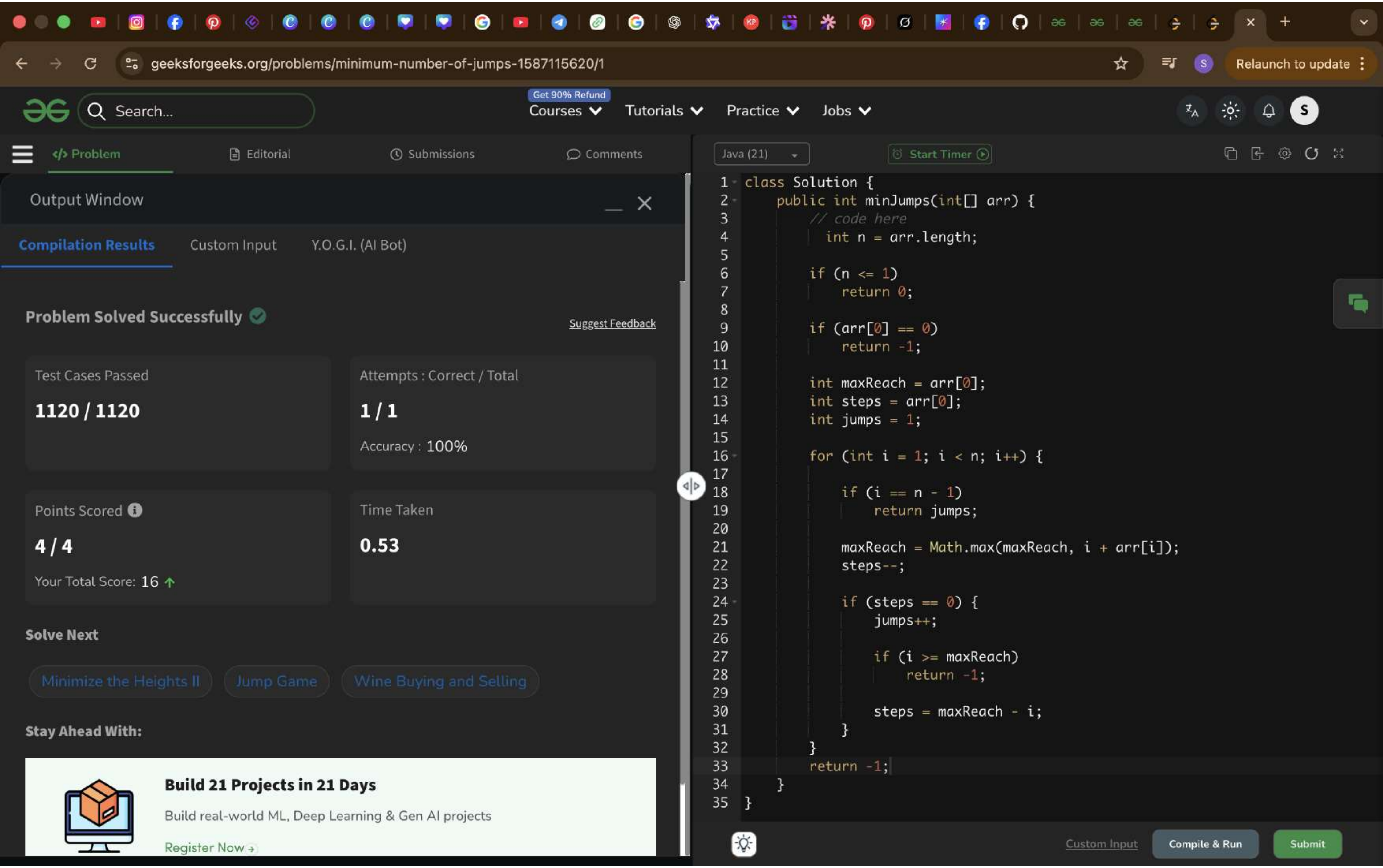
```
[2,7,11,15]
```

target =

```
9
```

Output

```
[0,1]
```



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: 16 ↑

Time Taken
0.53

Solve Next

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

Stay Ahead With:

Build 21 Projects in 21 Days
Build real-world ML, Deep Learning & Gen AI projects
Register Now →

```java
class Solution {
    public int minJumps(int[] arr) {
        // code here
         int n = arr.length;

        if (n <= 1)
            return 0;

        if (arr[0] == 0)
            return -1;

        int maxReach = arr[0];
        int steps = arr[0];
        int jumps = 1;

        for (int i = 1; i < n; i++) {

            if (i == n - 1)
                return jumps;

            maxReach = Math.max(maxReach, i + arr[i]);
            steps--;

            if (steps == 0) {
                jumps++;

                if (i >= maxReach)
                    return -1;

                steps = maxReach - i;
            }
        }
        return -1;
    }
}
```