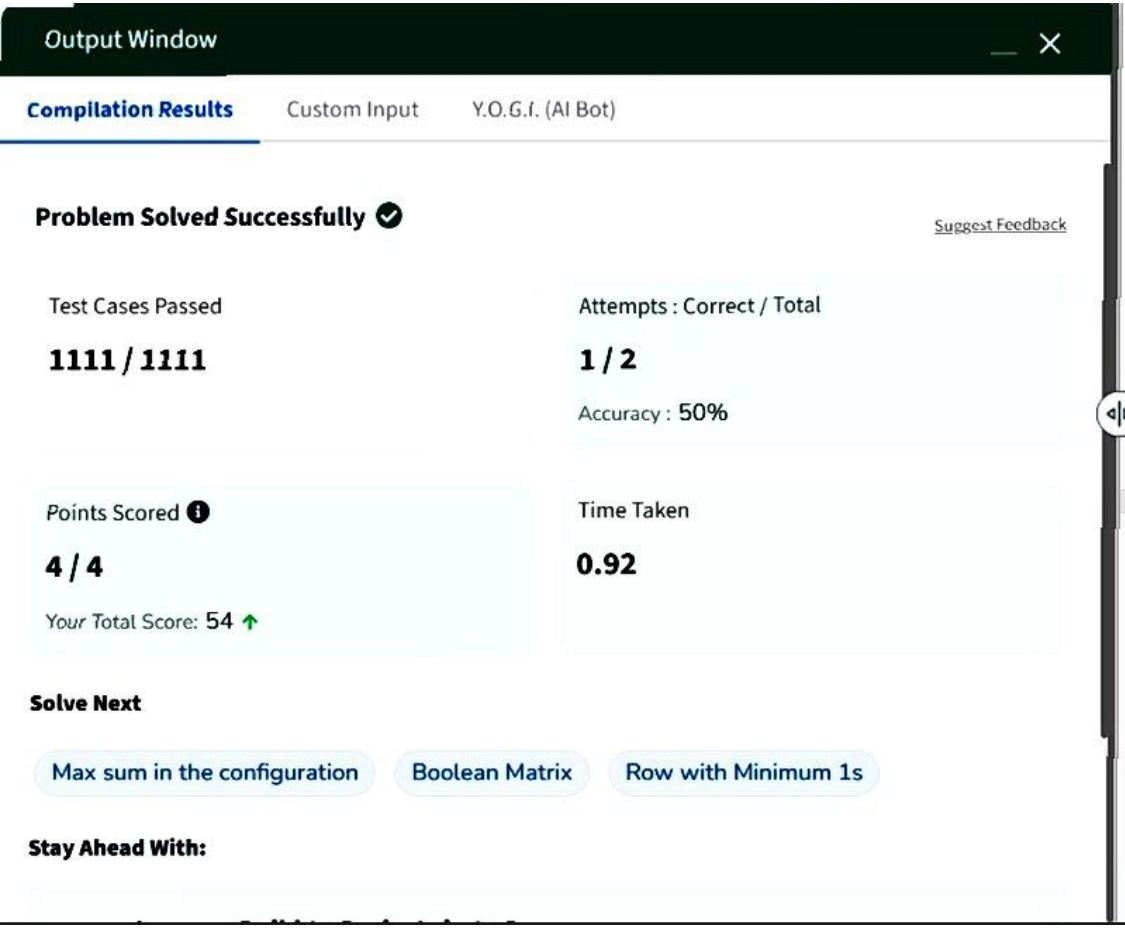
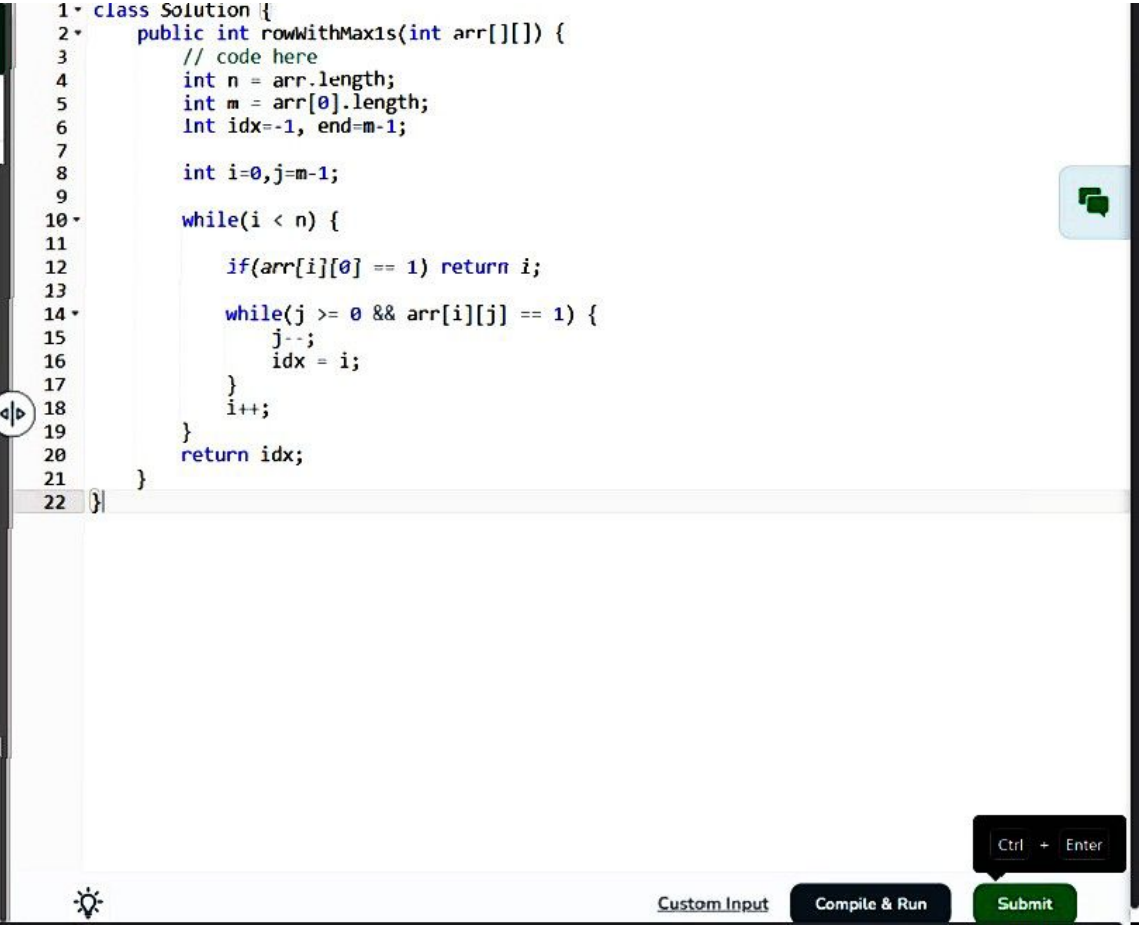
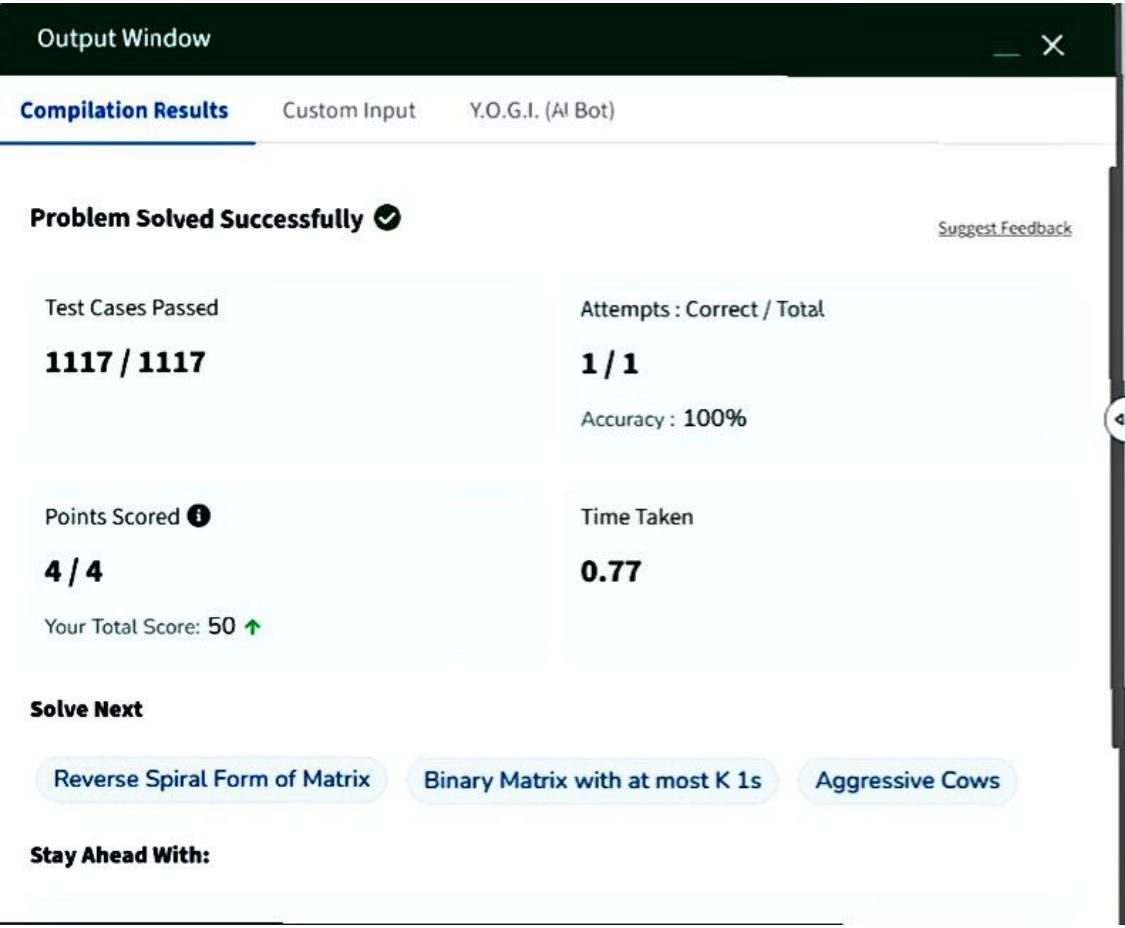

Output Window
Compilation Results
Custom Input
Y.O.G.I. (AI Bot)
Problem Solved Successfully
Suggest Feedback
Test Cases Passed
1111 / 1111
Attempts : Correct / Total
1 / 2
Accuracy : 50%
Points Scored
4 / 4
Your Total Score: 54
Time Taken
0.92
Solve Next
Max sum in the configuration
Boolean Matrix
Row with Minimum 1s
Stay Ahead With:


class Solution {
public int rowWithMax1s(int arr[][]) {
// code here
int n = arr.length;
int m = arr[0].length;
int idx=-1, end=m-1;
int i=0,j=m-1;
while(i < n) {
if(arr[i][0] == 1) return i;
while(j >= 0 && arr[i][j] == 1) {
j--;
idx = i;
}
i++;
}
return idx;
}
}
Ctrl + Enter
Custom Input
Compile & Run
Submit




```
14          }
15
16          int desired = (n * m + 1) / 2;
17
18          while (low < high) {
19              int mid = low + (high - low) / 2;
20              int count = 0;
21
22              for (int i = 0; i < n; i++) {
23                  count += upperBound(mat[i], mid);
24              }
25
26              if (count < desired) {
27                  low = mid + 1;
28              } else {
29                  high = mid;
30              }
31          }
32          return low;
33      }
34
35      private int upperBound(int[] row, int target) {
36          int l = 0, r = row.length;
37          while (l < r) {
38              int mid = l + (r - l) / 2;
39              if (row[mid] <= target) {
40                  l = mid + 1;
41              } else {
42                  r = mid;
43              }
44          }
45          return l;
46      }
47  }
48
49
50
```

Ctrl + Enter

Custom Input | Compile & Run | Submit

Description | Editorial | Solutions | **Accepted** × | Submissions

← All Submissions

**Accepted** 133 / 133 testcases passed

**SHREYA RAJ** submitted at Feb 18, 2026 14:39

Editorial | Solution

Runtime

**0** ms | Beats **100.00%**

Analyze Complexity

Memory

43.68 MB | Beats 95.73%


Code | Java

```java
class Solution {
    public boolean searchMatrix(int[][] matrix, int target)
    {
        int iRow = matrix.length;
        int iColumn = matrix[0].length;
```

**</> Code**

Java | Auto

```java
                {
                    low = mid + 1;
                }
            }
        }
        return false;
    }
}
```

Saved | Ln 35, Col 2

Testcase | **Test Result**

**Accepted** Runtime: 0 ms

**Case 1** | **Case 2**

Input

matrix =

```
[[1,3,5,7],[10,11,16,20],[23,30,34,60]]
```

target =

```
3
```

Output

```
true
```

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 / 2**

Accuracy : 50%

Points Scored

**4 / 4**

Your Total Score: 46 ↑

Time Taken

**2.59**

**Solve Next**

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

**Stay Ahead With:**

```java
class Solution {
    // Function to return a list of integers denoting spiral traversal of matrix.
    public ArrayList<Integer> spirallyTraverse(int mat[][]) {
        ArrayList<Integer> spiral = new ArrayList<>();
        int row_start = 0, row_end = mat.length - 1;
        int col_start = 0, col_end = mat[0].length - 1;
        while(row_start <= row_end && col_start <= col_end){
            for(int i = col_start; i <= col_end; i++){
                spiral.add(mat[row_start][i]);
            }
            row_start++;
            for(int i = row_start; i <= row_end; i++){
                spiral.add(mat[i][col_end]);
            }
            col_end--;
            if(row_start <= row_end){
                for(int i = col_end; i >= col_start; i--){
                    spiral.add(mat[row_end][i]);
                }
                row_end--;
            }
            if(col_start <= col_end){
                for(int i = row_end; i >= row_start; i--){
                    spiral.add(mat[i][col_start]);
                }
                col_start++;
            }
        }
        return spiral;
    }
}
```

Ctrl + Enter

Custom Input | Compile & Run | Submit

Description | Editorial | Solutions | Submissions

**Follow up:**

- How can we prove that at least one duplicate number must exist in `nums` ?
- Can you solve the problem in linear runtime complexity?

over
ore

Seen this question in a real interview before? 1/5

Yes No

Accepted **2,475,173** /3.9M Acceptance Rate **64.0**%

Topics

Companies

Similar Questions

Discussion (498)



25.3K 498 165 Online

Code

Java Auto

```java
class Solution {
    public int findDuplicate(int[] nums) {
        int slow = nums[0];
        int fast = nums[0];

        while (true) {
            slow = nums[slow];
            fast = nums[nums[fast]];

            if (slow == fast) {
```

Saved Ln 24, Col 2

Testcase Test Result

**Accepted** Runtime: 0 ms

Case 1 Case 2 Case 3

Input

nums =

[1,3,4,2,2]

Output

2

Expected

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 / 7**

Accuracy : 14%

Points Scored

**2 / 2**

Your Total Score: 41

Time Taken

**0.3**

**Solve Next**

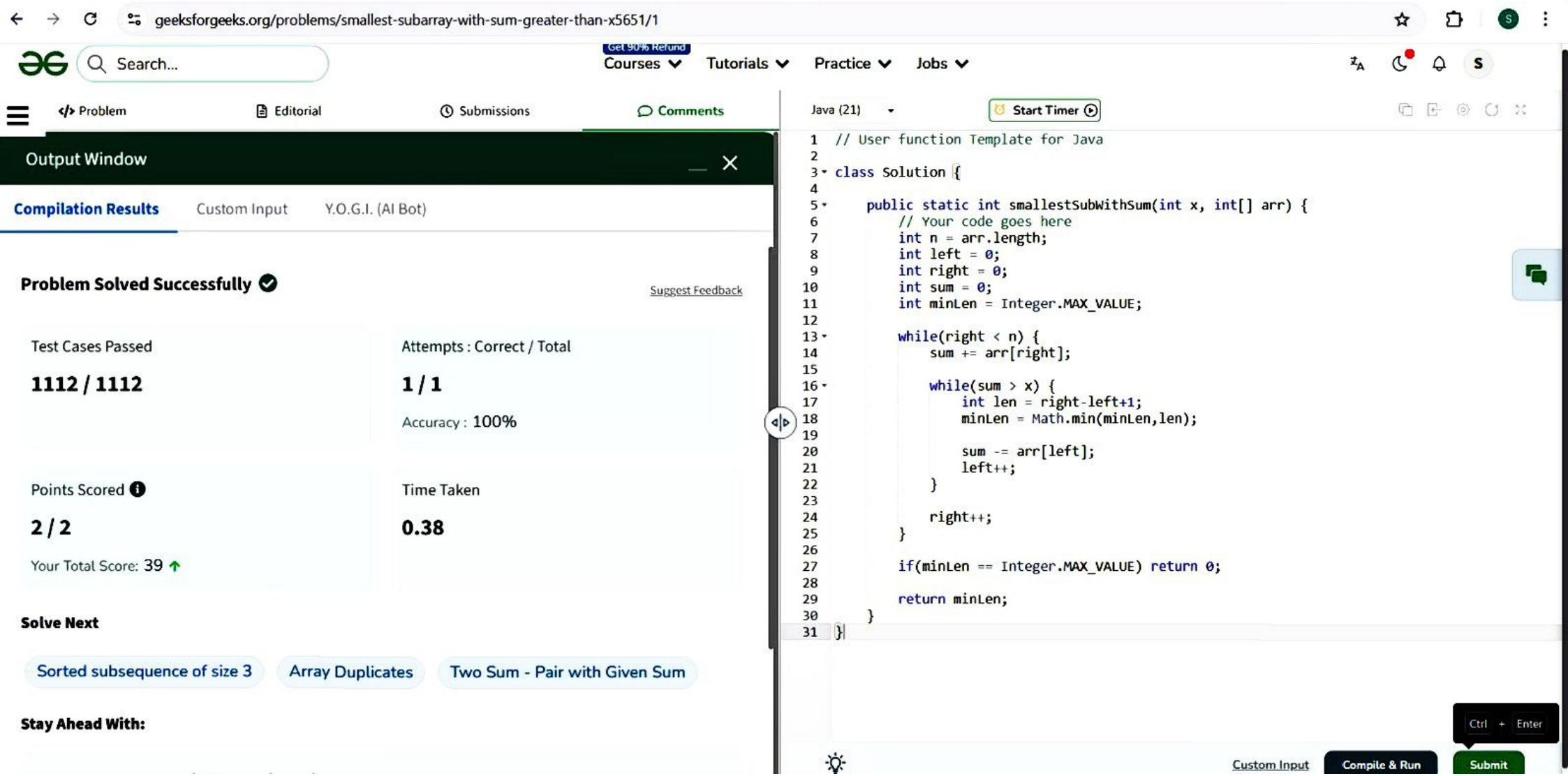
Wave Array | Sort by Absolute Difference | Convert an array to reduced form

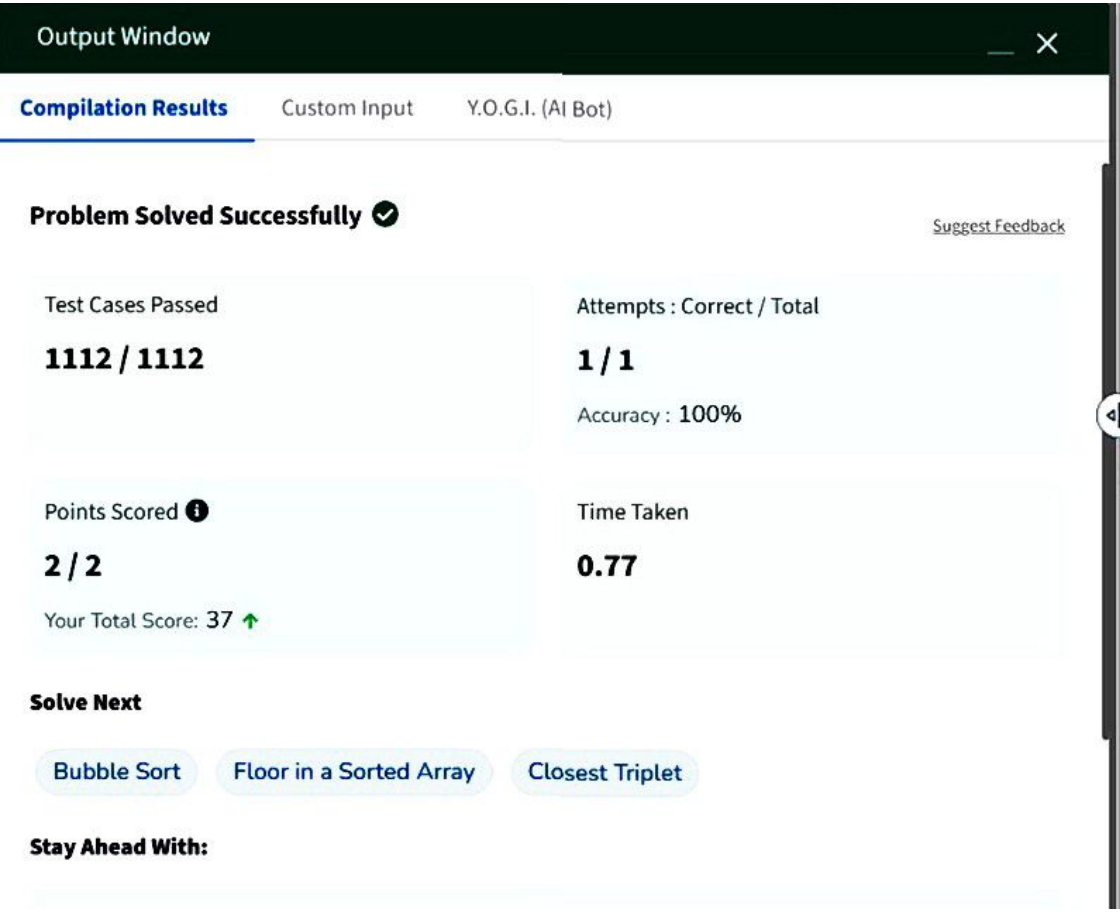
**Stay Ahead With:**

```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)
    {
        // code here
        int n = arr.length;
        int k[] = new int[n];
        int i=0;
        Arrays.sort(arr);
       while(i<n){
                if(arr[i]>b) k[i]=arr[i];
            else if(a<=arr[i] && arr[i]<=b)  k[i]=arr[i];
            else if(arr[i]<a)  k[i]=arr[i];

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


       }
    }
}

```

Ctrl + Enter

Custom Input | Compile & Run | Submit


geeksforgeeks.org/problems/smallest-subarray-with-sum-greater-than-x5651/1
Search...
Courses
Tutorials
Practice
Jobs
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
1112 / 1112
Attempts : Correct / Total
1 / 1
Accuracy : 100%
Points Scored
2 / 2
Time Taken
0.38
Your Total Score: 39
Solve Next
Sorted subsequence of size 3
Array Duplicates
Two Sum - Pair with Given Sum
Stay Ahead With:
// User function Template for Java
class Solution {
public static int smallestSubWithSum(int x, int[] arr) {
// Your code goes here
int n = arr.length;
int left = 0;
int right = 0;
int sum = 0;
int minLen = Integer.MAX_VALUE;
while(right < n) {
sum += arr[right];
while(sum > x) {
int len = right-left+1;
minLen = Math.min(minLen,len);
sum -= arr[left];
left++;
}
right++;
}
if(minLen == Integer.MAX_VALUE) return 0;
return minLen;
}
}
Ctrl + Enter
Custom Input
Compile & Run
Submit




```java
import java.util.ArrayList;
import java.util.Collections;

class Solution {
    public int findMinDiff(ArrayList<Integer> arr, int m) {
        // Sort the array
        Collections.sort(arr);

        int start = 0;
        int end = m - 1;
        int ans = Integer.MAX_VALUE;

        while (end < arr.size()) {
            ans = Math.min(ans, arr.get(end) - arr.get(start));
            start++;
            end++;
        }

        return ans;
    }
}
```