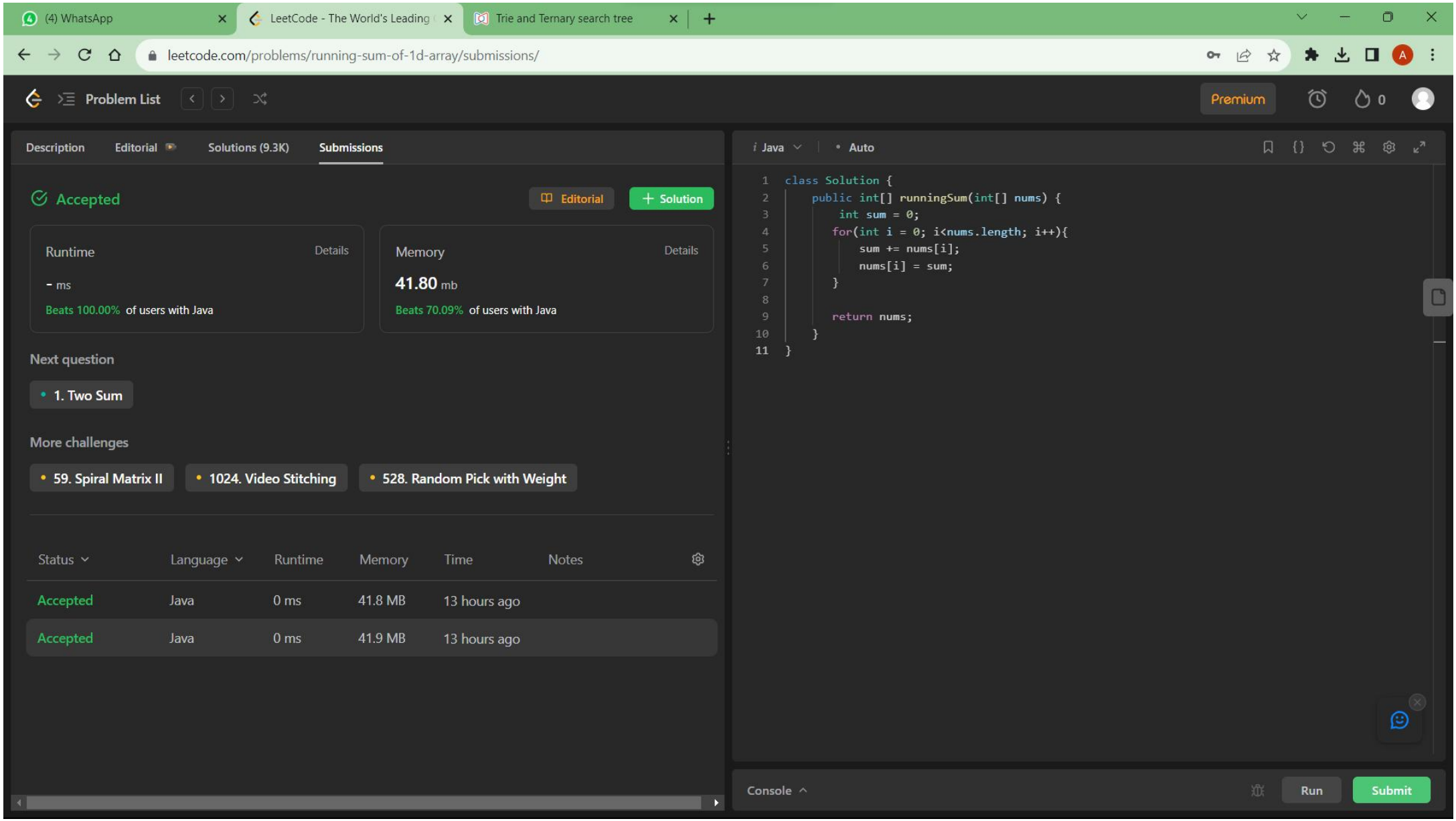
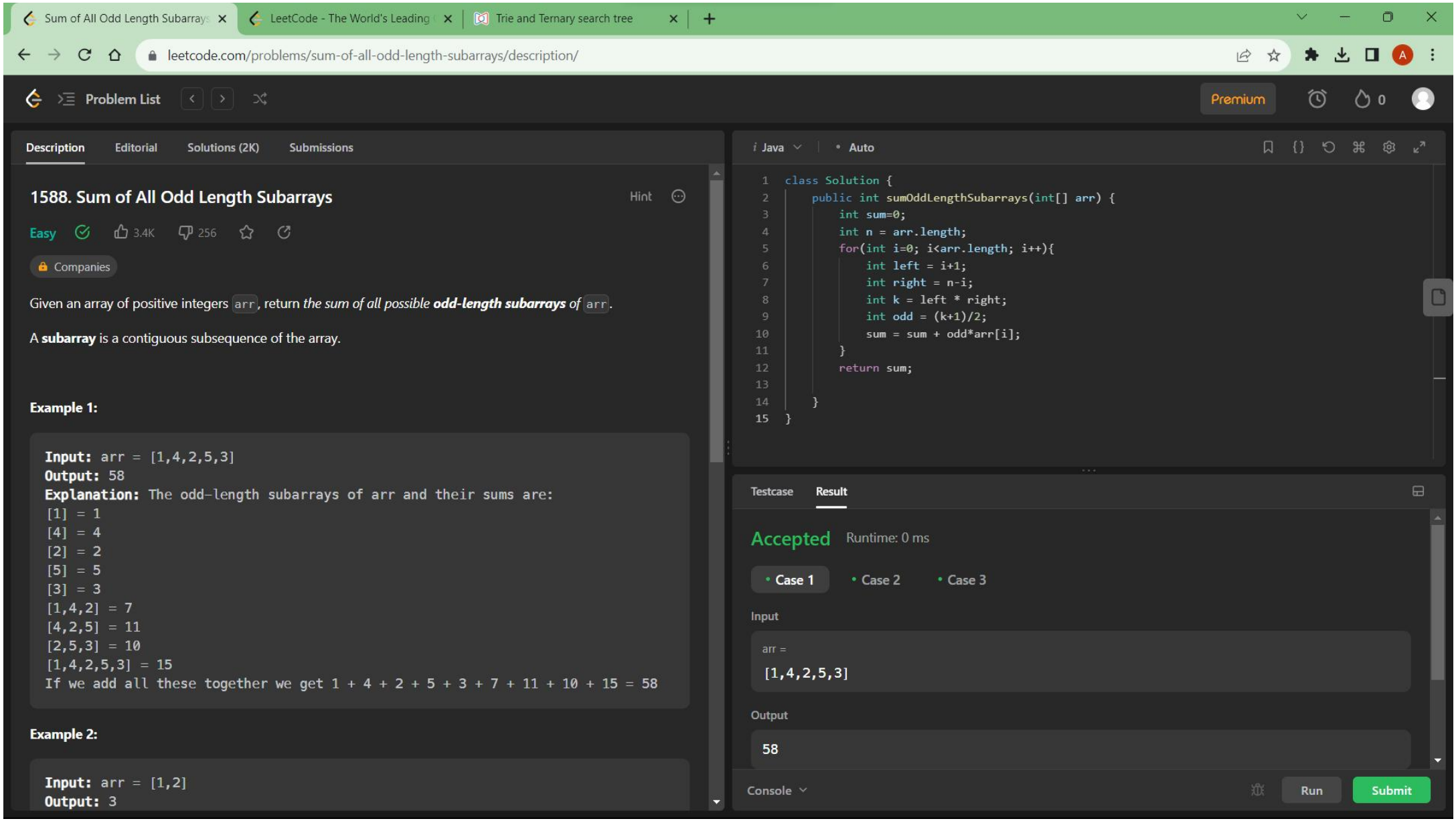
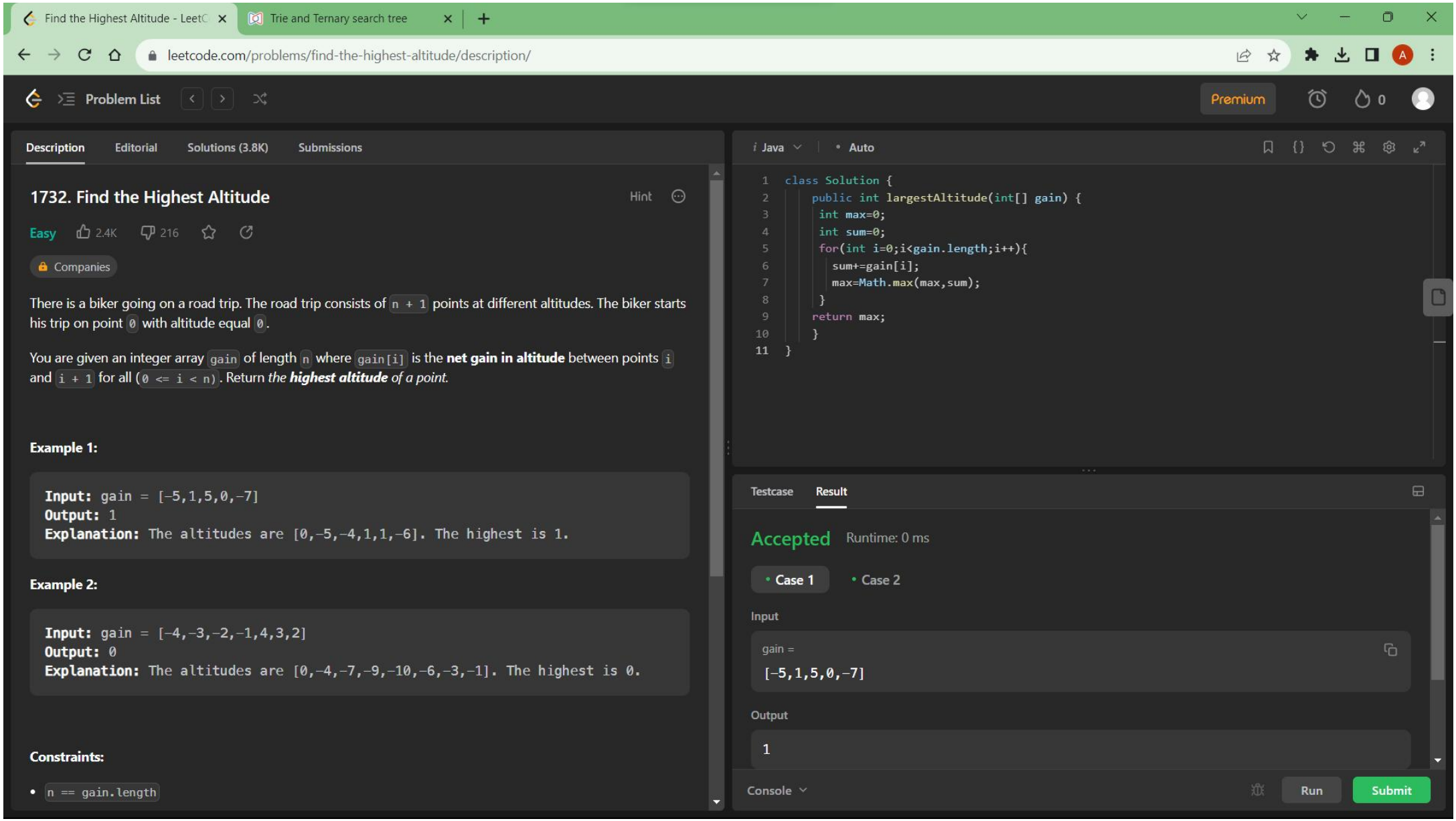
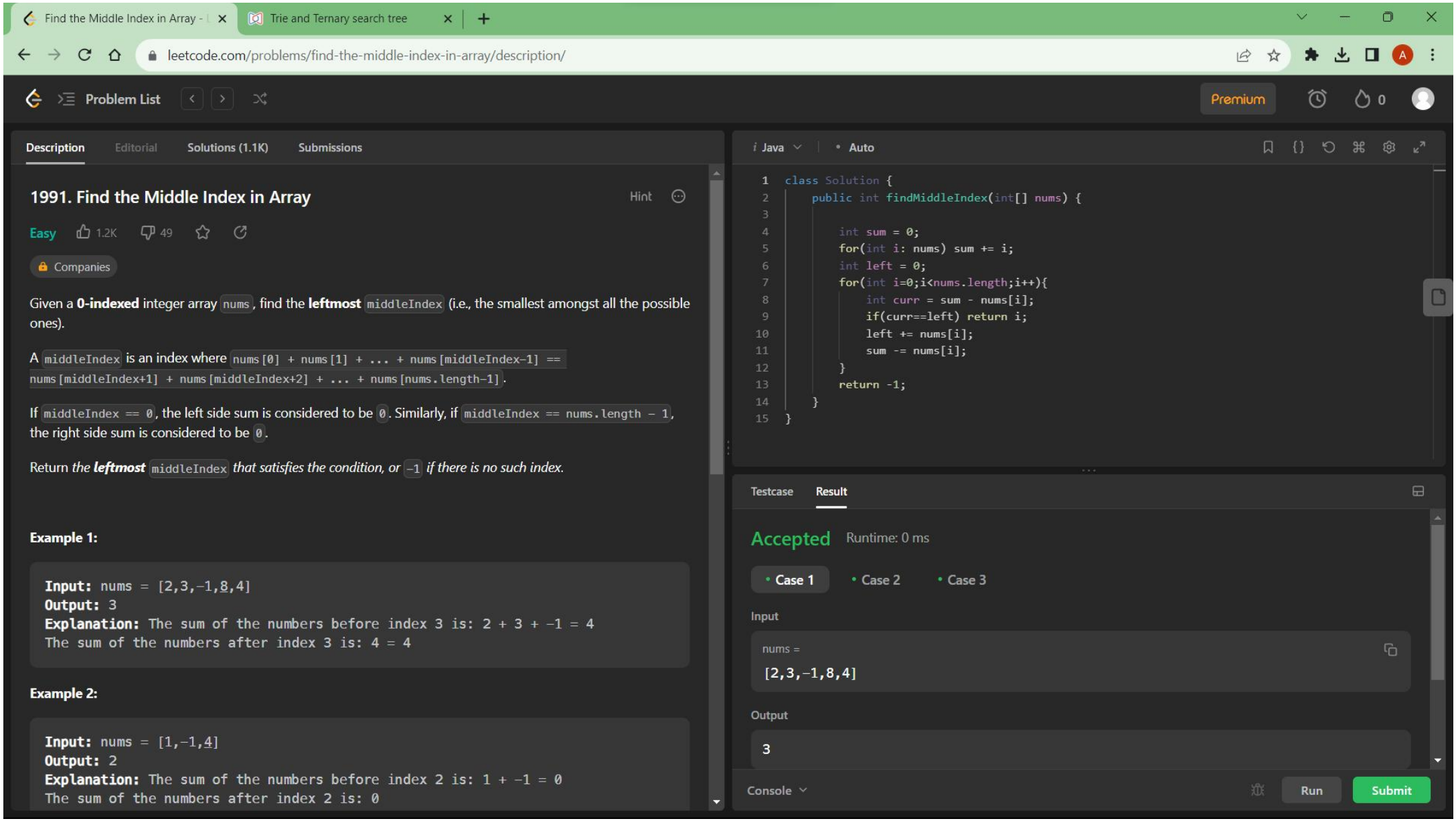
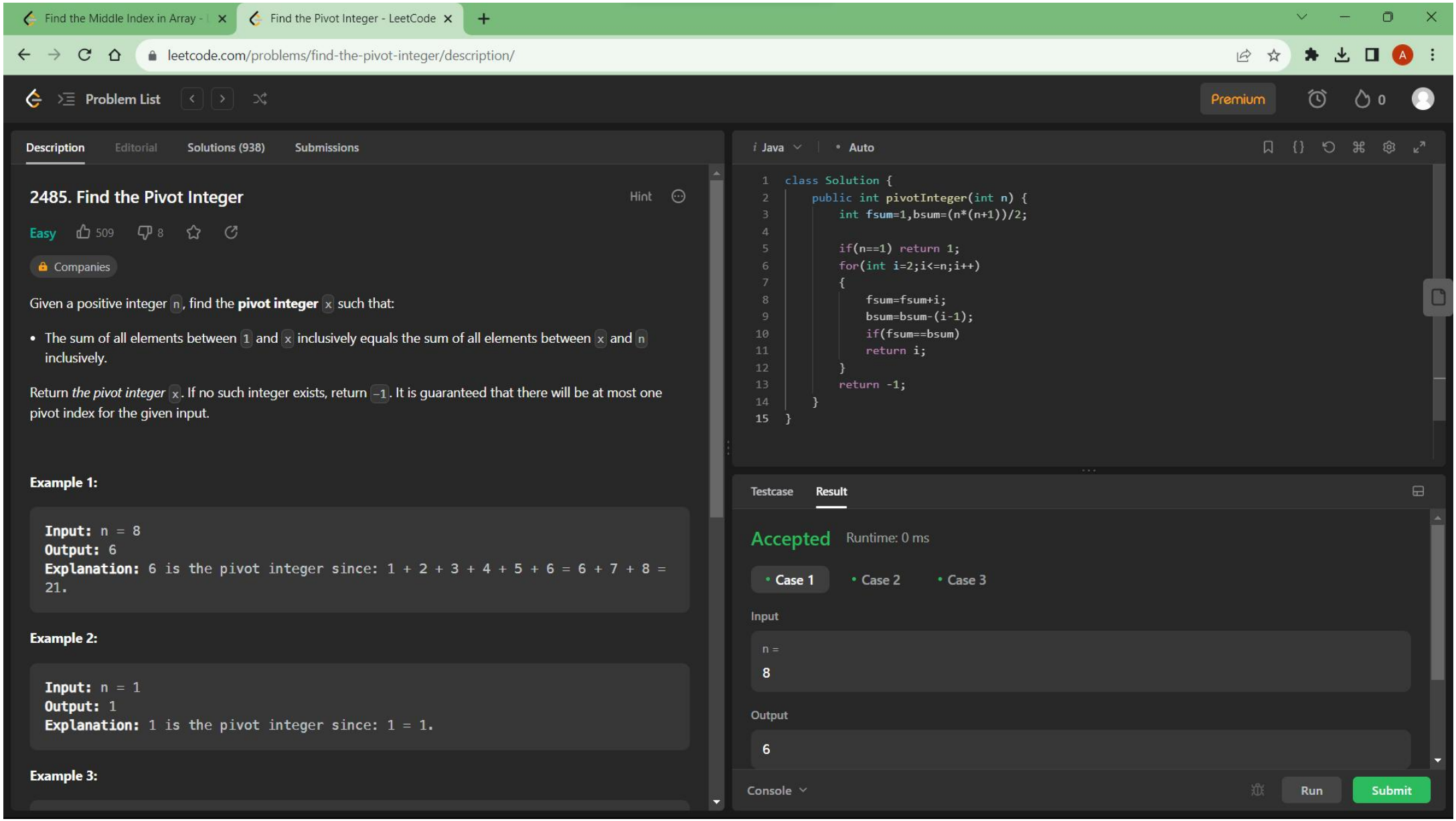
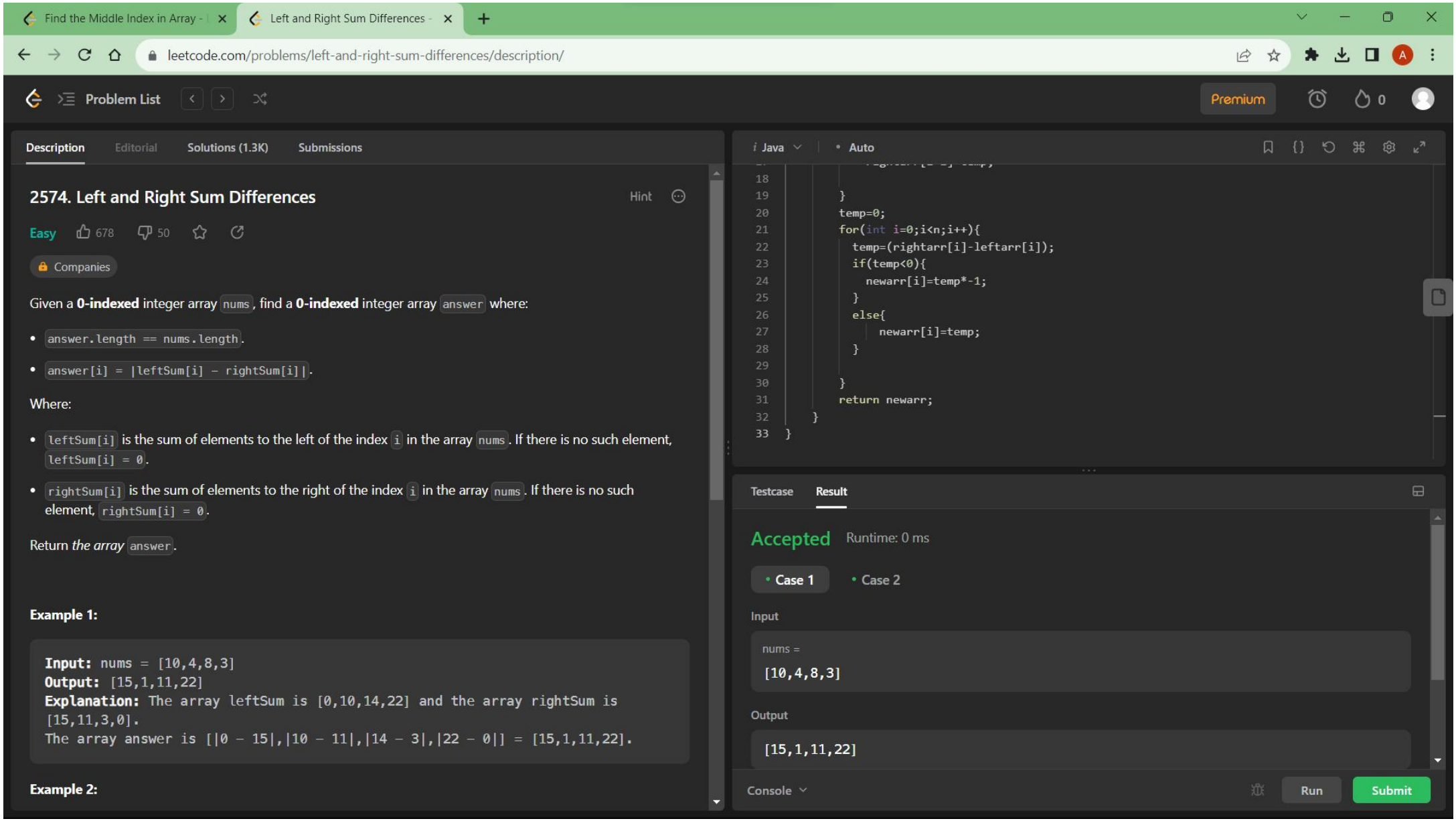

(4) WhatsApp
LeetCode - The World's Leading
Trie and Ternary search tree
leetcode.com/problems/running-sum-of-1d-array/submissions/
Problem List
Premium
0
Description
Editorial
Solutions (9.3K)
Submissions
Accepted
Editorial
+ Solution
Runtime
Details
- ms
Beats 100.00% of users with Java
Memory
Details
41.80 mb
Beats 70.09% of users with Java
Next question
1. Two Sum
More challenges
59. Spiral Matrix II
1024. Video Stitching
528. Random Pick with Weight
Status
Language
Runtime
Memory
Time
Notes
Accepted
Java
0 ms
41.8 MB
13 hours ago
Accepted
Java
0 ms
41.9 MB
13 hours ago
Java
Auto
1 class Solution {
2 public int[] runningSum(int[] nums) {
3 int sum = 0;
4 for(int i = 0; i<nums.length; i++){
5 sum += nums[i];
6 nums[i] = sum;
7 }
8
9 return nums;
10 }
11 }
Console
Run
Submit

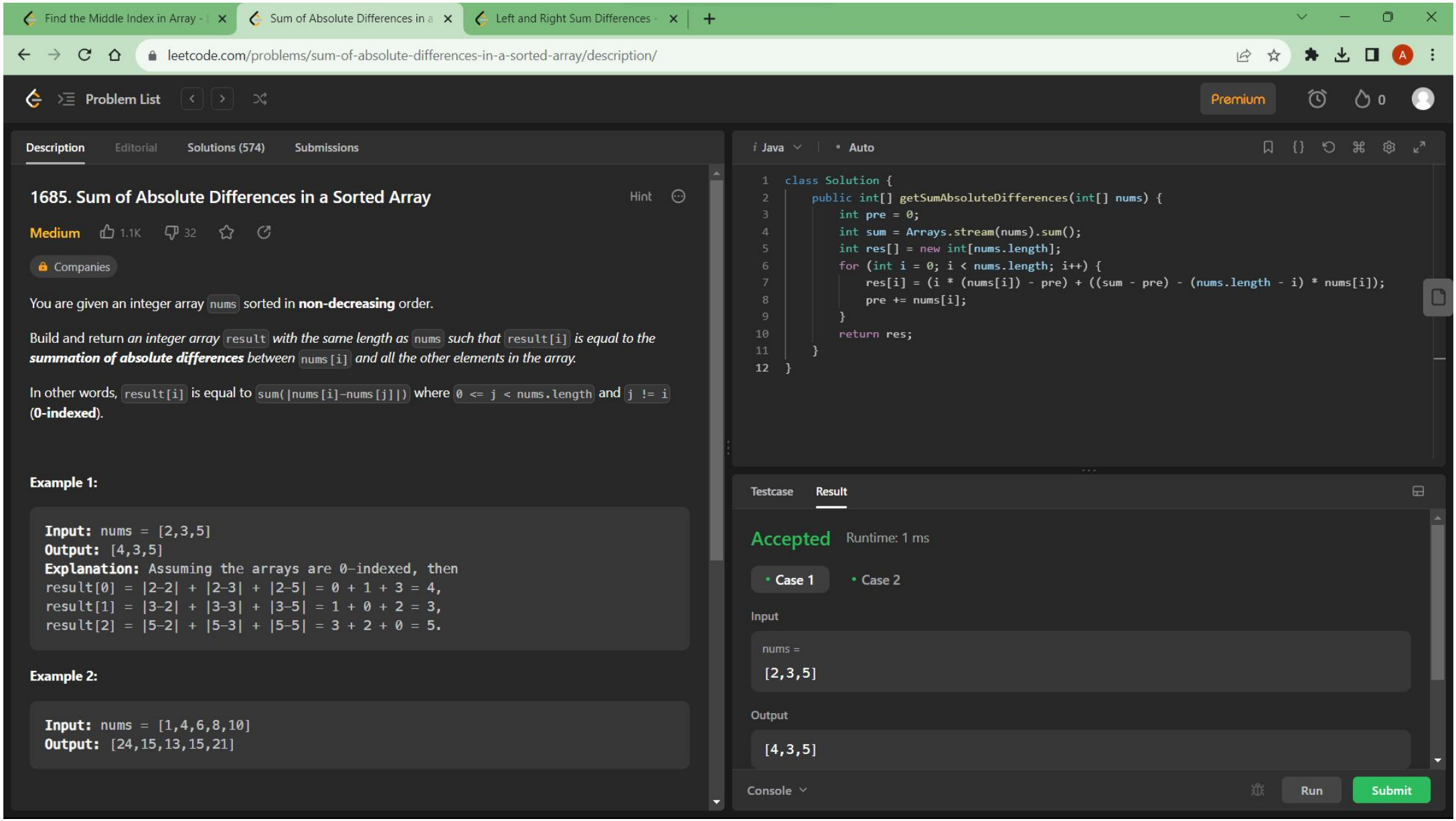

Sum of All Odd Length Subarrays
LeetCode - The World's Leading
Trie and Ternary search tree
leetcode.com/problems/sum-of-all-odd-length-subarrays/description/
Problem List
Premium
Description
Editorial
Solutions (2K)
Submissions
1588. Sum of All Odd Length Subarrays
Hint
Easy
3.4K
256
Companies
Given an array of positive integers arr, return the sum of all possible odd-length subarrays of arr.
A subarray is a contiguous subsequence of the array.
Example 1:
Input: arr = [1,4,2,5,3]
Output: 58
Explanation: The odd-length subarrays of arr and their sums are:
[1] = 1
[4] = 4
[2] = 2
[5] = 5
[3] = 3
[1,4,2] = 7
[4,2,5] = 11
[2,5,3] = 10
[1,4,2,5,3] = 15
If we add all these together we get 1 + 4 + 2 + 5 + 3 + 7 + 11 + 10 + 15 = 58
Example 2:
Input: arr = [1,2]
Output: 3
Java
Auto
class Solution {
public int sumOddLengthSubarrays(int[] arr) {
int sum=0;
int n = arr.length;
for(int i=0; i<arr.length; i++){
int left = i+1;
int right = n-i;
int k = left * right;
int odd = (k+1)/2;
sum = sum + odd*arr[i];
}
return sum;
}
}
Testcase
Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
arr =
[1,4,2,5,3]
Output
58
Console
Run
Submit

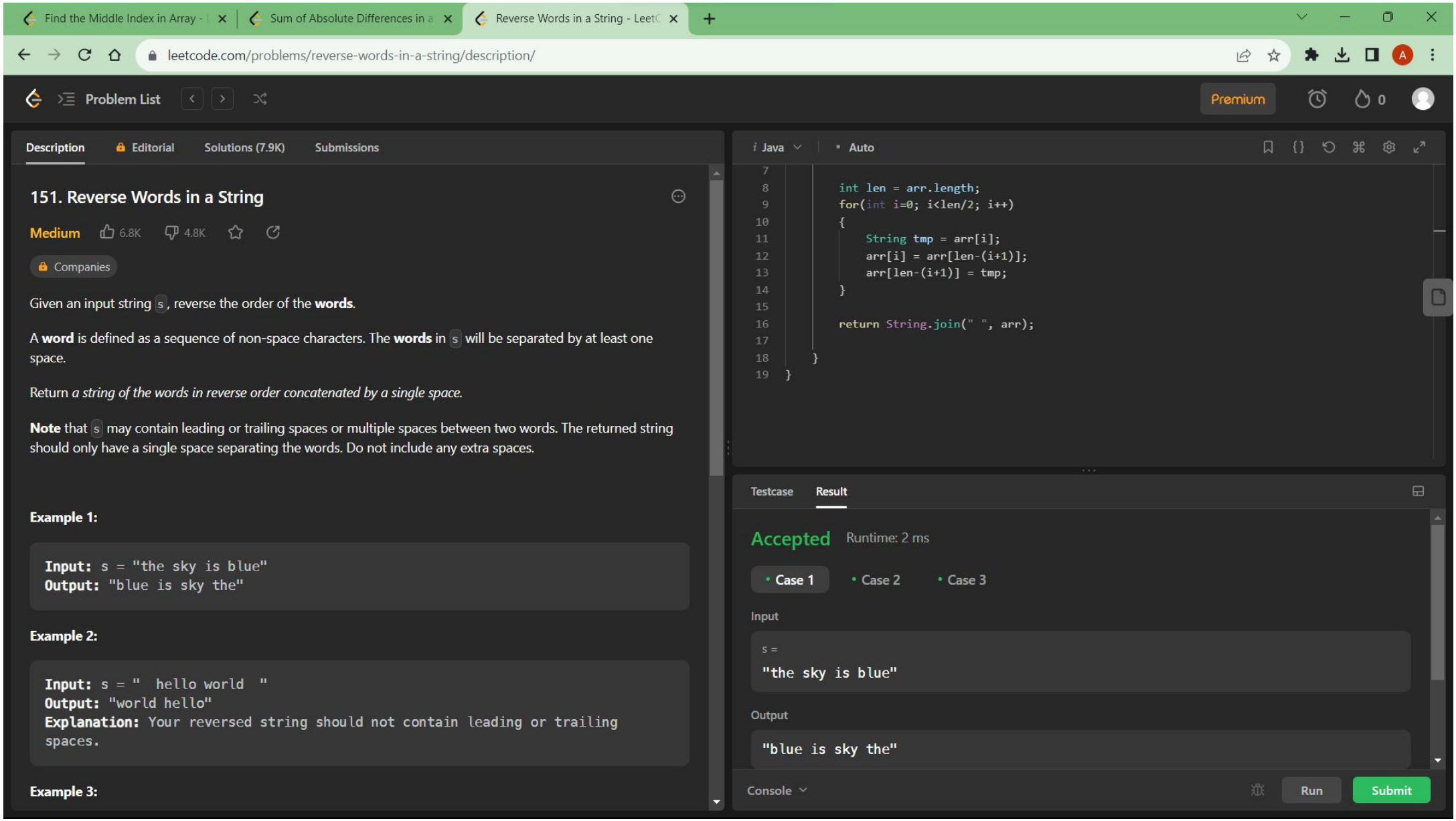

Find the Highest Altitude - LeetC
Trie and Ternary search tree
leetcode.com/problems/find-the-highest-altitude/description/
Problem List
Premium
Description
Editorial
Solutions (3.8K)
Submissions
1732. Find the Highest Altitude
Hint
Easy
2.4K
216
Companies
There is a biker going on a road trip. The road trip consists of n + 1 points at different altitudes. The biker starts his trip on point 0 with altitude equal 0.
You are given an integer array gain of length n where gain[i] is the net gain in altitude between points i and i + 1 for all (0 <= i < n). Return the highest altitude of a point.
Example 1:
Input: gain = [-5,1,5,0,-7]
Output: 1
Explanation: The altitudes are [0,-5,-4,1,1,-6]. The highest is 1.
Example 2:
Input: gain = [-4,-3,-2,-1,4,3,2]
Output: 0
Explanation: The altitudes are [0,-4,-7,-9,-10,-6,-3,-1]. The highest is 0.
Constraints:
n == gain.length
Java
Auto
class Solution {
public int largestAltitude(int[] gain) {
int max=0;
int sum=0;
for(int i=0;i<gain.length;i++){
sum+=gain[i];
max=Math.max(max,sum);
}
return max;
}
}
Testcase
Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
gain =
[-5,1,5,0,-7]
Output
1
Console
Run
Submit


Find the Middle Index in Array -
Trie and Ternary search tree
leetcode.com/problems/find-the-middle-index-in-array/description/
Problem List
Premium
Description
Editorial
Solutions (1.1K)
Submissions
1991. Find the Middle Index in Array
Hint
Easy
1.2K
49
Companies
Given a 0-indexed integer array nums, find the leftmost middleIndex (i.e., the smallest amongst all the possible ones).
A middleIndex is an index where nums[0] + nums[1] + ... + nums[middleIndex-1] == nums[middleIndex+1] + nums[middleIndex+2] + ... + nums[nums.length-1].
If middleIndex == 0, the left side sum is considered to be 0. Similarly, if middleIndex == nums.length - 1, the right side sum is considered to be 0.
Return the leftmost middleIndex that satisfies the condition, or -1 if there is no such index.
Example 1:
Input: nums = [2,3,-1,8,4]
Output: 3
Explanation: The sum of the numbers before index 3 is: 2 + 3 + -1 = 4
The sum of the numbers after index 3 is: 4 = 4
Example 2:
Input: nums = [1,-1,4]
Output: 2
Explanation: The sum of the numbers before index 2 is: 1 + -1 = 0
The sum of the numbers after index 2 is: 0
Java
Auto
class Solution {
public int findMiddleIndex(int[] nums) {
int sum = 0;
for(int i: nums) sum += i;
int left = 0;
for(int i=0;i<nums.length;i++){
int curr = sum - nums[i];
if(curr==left) return i;
left += nums[i];
sum -= nums[i];
}
return -1;
}
}
Testcase
Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
nums =
[2,3,-1,8,4]
Output
3
Console
Run
Submit


Find the Middle Index in Array -
Find the Pivot Integer - LeetCode
leetcode.com/problems/find-the-pivot-integer/description/
Problem List
Premium
Description
Editorial
Solutions (938)
Submissions
2485. Find the Pivot Integer
Hint
Easy
509
8
Companies
Given a positive integer n, find the pivot integer x such that:
The sum of all elements between 1 and x inclusively equals the sum of all elements between x and n inclusively.
Return the pivot integer x. If no such integer exists, return -1. It is guaranteed that there will be at most one pivot index for the given input.
Example 1:
Input: n = 8
Output: 6
Explanation: 6 is the pivot integer since: 1 + 2 + 3 + 4 + 5 + 6 = 6 + 7 + 8 = 21.
Example 2:
Input: n = 1
Output: 1
Explanation: 1 is the pivot integer since: 1 = 1.
Example 3:
Java
Auto
class Solution {
public int pivotInteger(int n) {
int fsum=1,bsum=(n*(n+1))/2;
if(n==1) return 1;
for(int i=2;i<=n;i++)
{
fsum=fsum+i;
bsum=bsum-(i-1);
if(fsum==bsum)
return i;
}
return -1;
}
}
Testcase
Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
n =
8
Output
6
Console
Run
Submit


Find the Middle Index in Array -
Left and Right Sum Differences -
leetcode.com/problems/left-and-right-sum-differences/description/
Problem List
Premium
Description
Editorial
Solutions (1.3K)
Submissions
2574. Left and Right Sum Differences
Hint
Easy
678
50
Companies
Given a 0-indexed integer array nums, find a 0-indexed integer array answer where:
answer.length == nums.length.
answer[i] = |leftSum[i] - rightSum[i]|.
Where:
leftSum[i] is the sum of elements to the left of the index i in the array nums. If there is no such element, leftSum[i] = 0.
rightSum[i] is the sum of elements to the right of the index i in the array nums. If there is no such element, rightSum[i] = 0.
Return the array answer.
Example 1:
Input: nums = [10,4,8,3]
Output: [15,1,11,22]
Explanation: The array leftSum is [0,10,14,22] and the array rightSum is [15,11,3,0].
The array answer is [|0 - 15|,|10 - 11|,|14 - 3|,|22 - 0|] = [15,1,11,22].
Example 2:
Java
Auto
18
19 }
20 temp=0;
21 for(int i=0;i<n;i++){
22 temp=(rightarr[i]-leftarr[i]);
23 if(temp<0){
24 newarr[i]=temp*-1;
25 }
26 else{
27 newarr[i]=temp;
28 }
29
30 }
31 return newarr;
32 }
33 }
Testcase
Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
nums =
[10,4,8,3]
Output
[15,1,11,22]
Console
Run
Submit


Find the Middle Index in Array -
Sum of Absolute Differences in a
Left and Right Sum Differences -
leetcode.com/problems/sum-of-absolute-differences-in-a-sorted-array/description/
Problem List
Premium
0
Description
Editorial
Solutions (574)
Submissions
1685. Sum of Absolute Differences in a Sorted Array
Hint
Medium
1.1K
32
Companies
You are given an integer array nums sorted in non-decreasing order.
Build and return an integer array result with the same length as nums such that result[i] is equal to the summation of absolute differences between nums[i] and all the other elements in the array.
In other words, result[i] is equal to sum(|nums[i]-nums[j]|) where 0 <= j < nums.length and j != i (0-indexed).
Example 1:
Input: nums = [2,3,5]
Output: [4,3,5]
Explanation: Assuming the arrays are 0-indexed, then
result[0] = |2-2| + |2-3| + |2-5| = 0 + 1 + 3 = 4,
result[1] = |3-2| + |3-3| + |3-5| = 1 + 0 + 2 = 3,
result[2] = |5-2| + |5-3| + |5-5| = 3 + 2 + 0 = 5.
Example 2:
Input: nums = [1,4,6,8,10]
Output: [24,15,13,15,21]
Java
Auto
1 class Solution {
2 public int[] getSumAbsoluteDifferences(int[] nums) {
3 int pre = 0;
4 int sum = Arrays.stream(nums).sum();
5 int res[] = new int[nums.length];
6 for (int i = 0; i < nums.length; i++) {
7 res[i] = (i * (nums[i]) - pre) + ((sum - pre) - (nums.length - i) * nums[i]);
8 pre += nums[i];
9 }
10 return res;
11 }
12 }
Testcase
Result
Accepted
Runtime: 1 ms
Case 1
Case 2
Input
nums =
[2,3,5]
Output
[4,3,5]
Console
Run
Submit


Find the Middle Index in Array -
Sum of Absolute Differences in a
Reverse Words in a String - LeetC
leetcode.com/problems/reverse-words-in-a-string/description/
Problem List
Premium
0
Description
Editorial
Solutions (7.9K)
Submissions
151. Reverse Words in a String
Medium
6.8K
4.8K
Companies
Given an input string s, reverse the order of the words.
A word is defined as a sequence of non-space characters. The words in s will be separated by at least one space.
Return a string of the words in reverse order concatenated by a single space.
Note that s may contain leading or trailing spaces or multiple spaces between two words. The returned string should only have a single space separating the words. Do not include any extra spaces.
Example 1:
Input: s = "the sky is blue"
Output: "blue is sky the"
Example 2:
Input: s = " hello world "
Output: "world hello"
Explanation: Your reversed string should not contain leading or trailing spaces.
Example 3:
Java
Auto
int len = arr.length;
for(int i=0; i<len/2; i++)
{
String tmp = arr[i];
arr[i] = arr[len-(i+1)];
arr[len-(i+1)] = tmp;
}
return String.join(" ", arr);
}
}
Testcase
Result
Accepted
Runtime: 2 ms
Case 1
Case 2
Case 3
Input
s =
"the sky is blue"
Output
"blue is sky the"
Console
Run
Submit