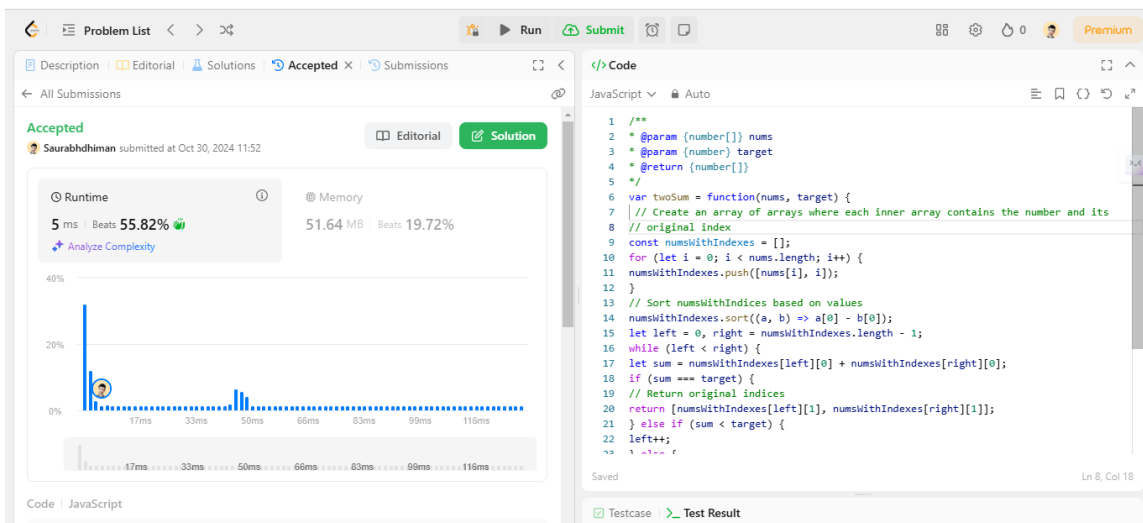


My Leet code profile link - <https://leetcode.com/u/Saurabhdhiman/>

Question No-1

Link - <https://leetcode.com/problems/two-sum/description/>

Solution Link - <https://leetcode.com/problems/two-sum/submissions/1437947202/>



Description -

Time Complexity-

$O(n)$

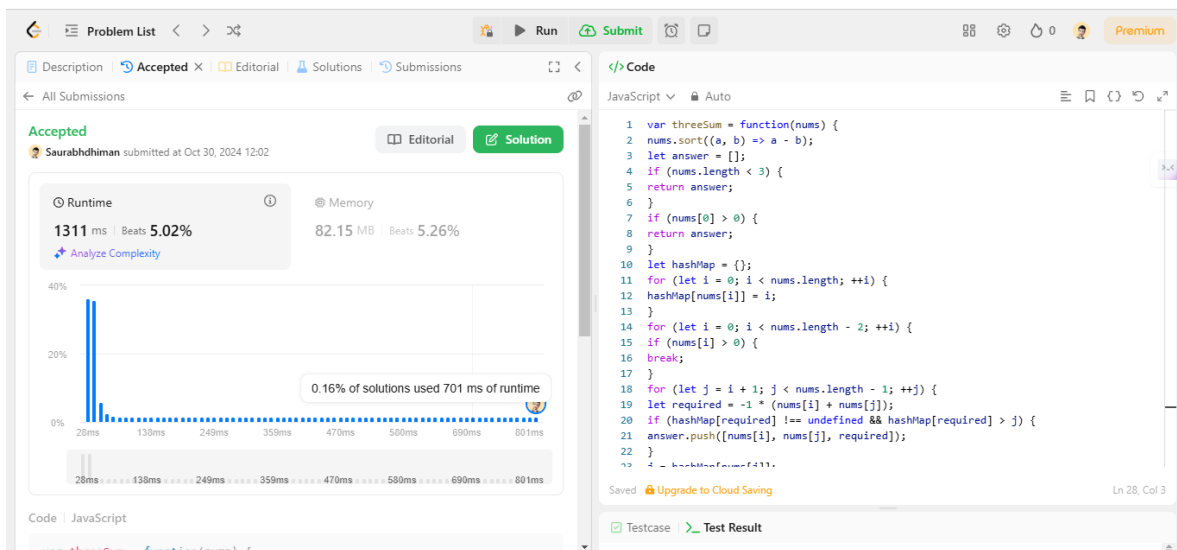
Space Complexity –

$O(n)$

Question No-2

Link - <https://leetcode.com/problems/3sum/description/>

Solution Link - <https://leetcode.com/problems/3sum/submissions/1437954695/>



Description –

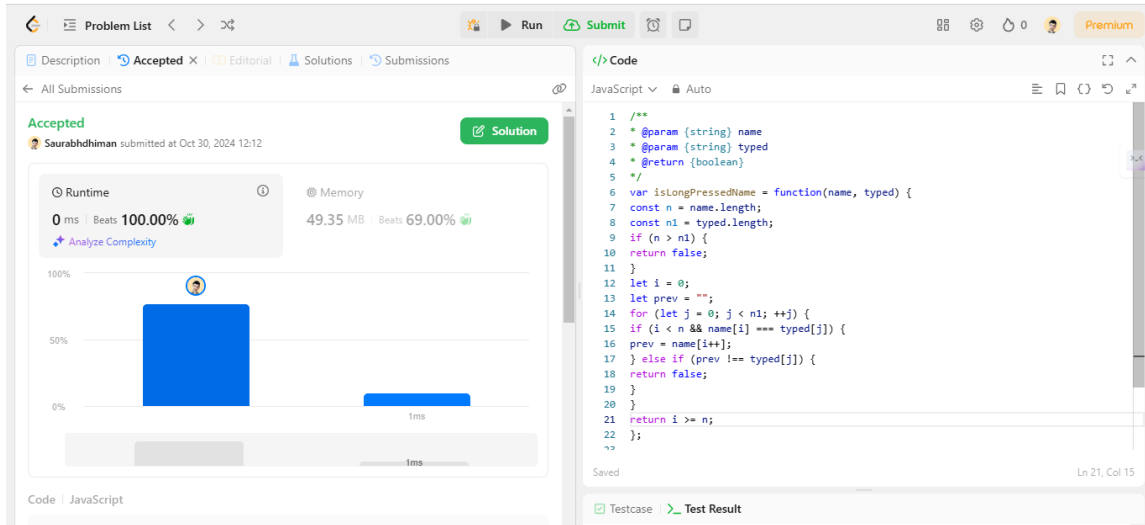
Time Complexity: $O(n^2)$

Space Complexity: $O(n)$.

Question No-3

Link - <https://leetcode.com/problems/long-pressed-name/description/>

solution link - <https://leetcode.com/problems/long-pressed-name/submissions/1437961889/>



Description -

Time Complexity- $O(n)$

Space complexity- $O(1)$

Question No-4

Link - <https://leetcode.com/problems/max-chunks-to-make-sorted/description/>

solution link- <https://leetcode.com/problems/max-chunks-to-make-sorted/submissions/1437966539/>

The screenshot shows the LeetCode interface for the problem "Max Chunks To Make Sorted". The submission is accepted, with a runtime of 0ms and memory usage of 48.63 MB. The code is in JavaScript and implements a function to find the maximum number of chunks that can be made from an array such that each chunk can be sorted individually to form a sorted array.

```
1 var maxChunksToSorted = function(arr) {
2   let chunk = 0;
3   let max = 0;
4   for (let i = 0; i < arr.length; i++) {
5     max = Math.max(max, arr[i]);
6     if (max === i) {
7       chunk++;
8     }
9   }
10  return chunk;
11 };
12
```

Description -

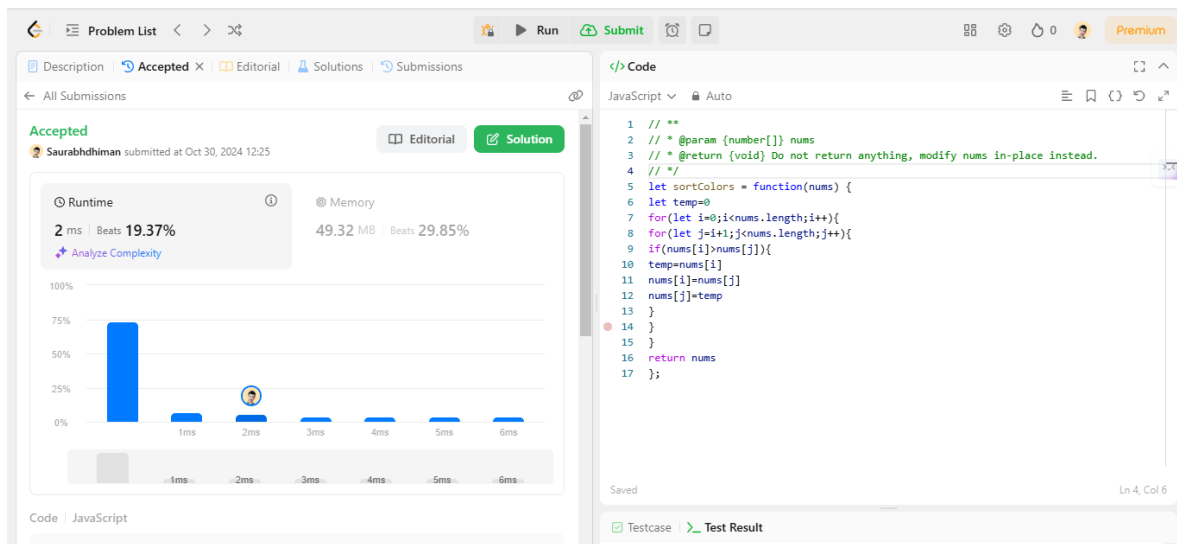
Time Complexity- $O(n)$

Space Complexity- $O(1)$

Question No-5

Link - <https://leetcode.com/problems/sort-colors/description/>

solution link - <https://leetcode.com/problems/sort-colors/submissions/1437971166/>



Description -

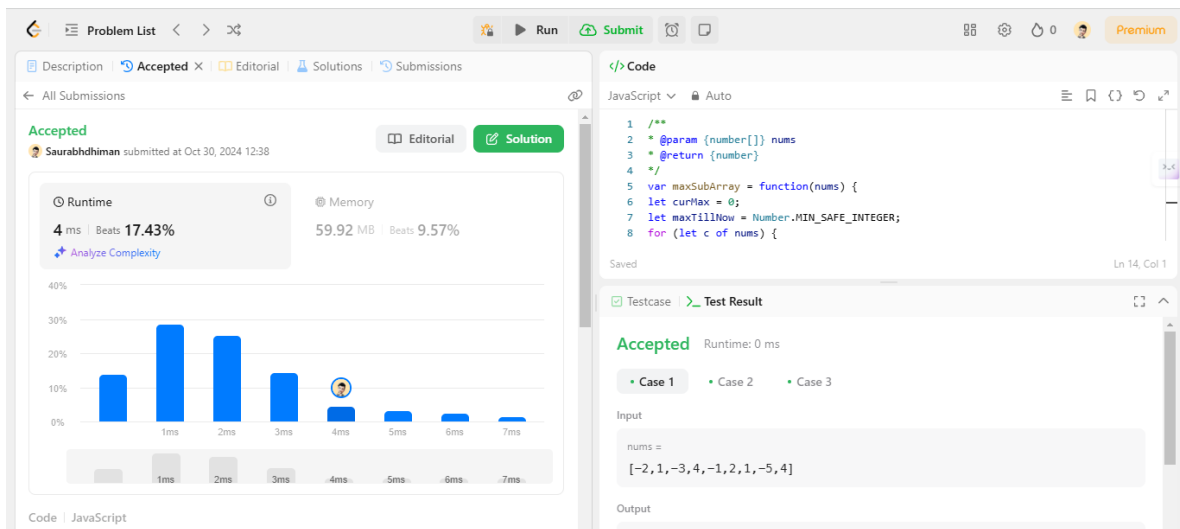
Time Complexity- $O(n^2)$.

Space Complexity - $O(1)$

Question No-6

Link - <https://leetcode.com/problems/maximum-subarray/description/>

Solution Link - <https://leetcode.com/problems/maximum-subarray/submissions/1437980369/>



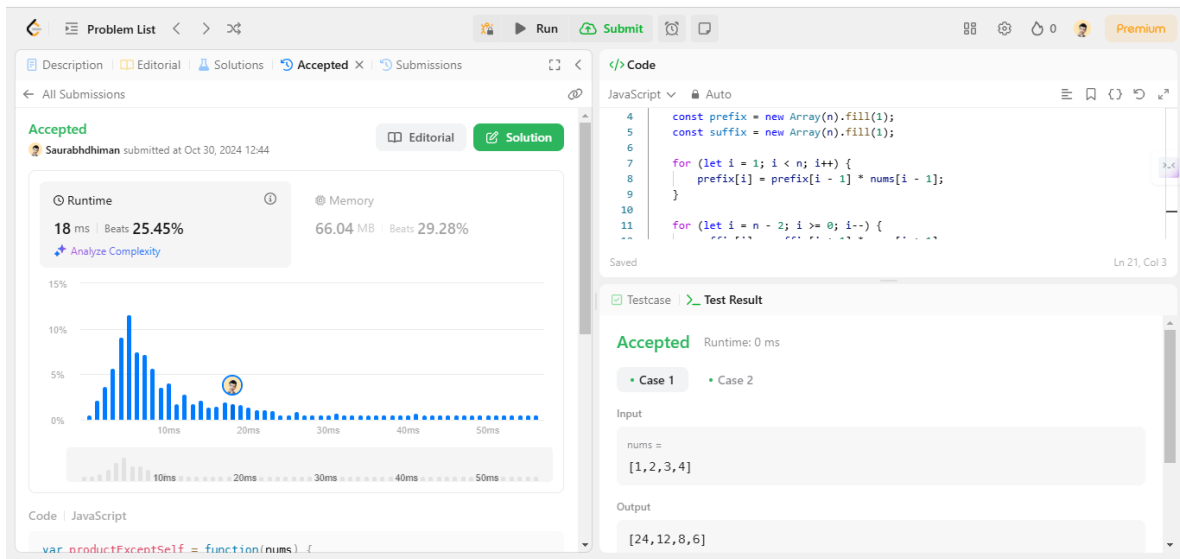
Time Complexity – $O(n)$.

Space Complexity – $O(1)$.

Question No – 7

Link – <https://leetcode.com/problems/product-of-array-except-self/description/>

Solution - <https://leetcode.com/problems/product-of-array-except-self/submissions/1437984231/>



Time Complexity – $O(n)$.

Space Complexity – $O(n)$.

