STS Digital Assignment

-Amitabha Nath(22BCE8048)

Q) Largest Number At Least Twice of Others

You are given an integer array `nums` where the largest integer is **unique**.

Determine whether the largest element in the array is **at least twice** as much as every other number in the array. If it is, return *the **index** of the largest element, or return* `-1` *otherwise*.

Example 1:

```
Input: nums = [3,6,1,0]
Output: 1
Explanation: 6 is the largest integer.
For every other number in the array x, 6 is at least twice
as big as x.
The index of value 6 is 1, so we return 1.
```

Example 2:

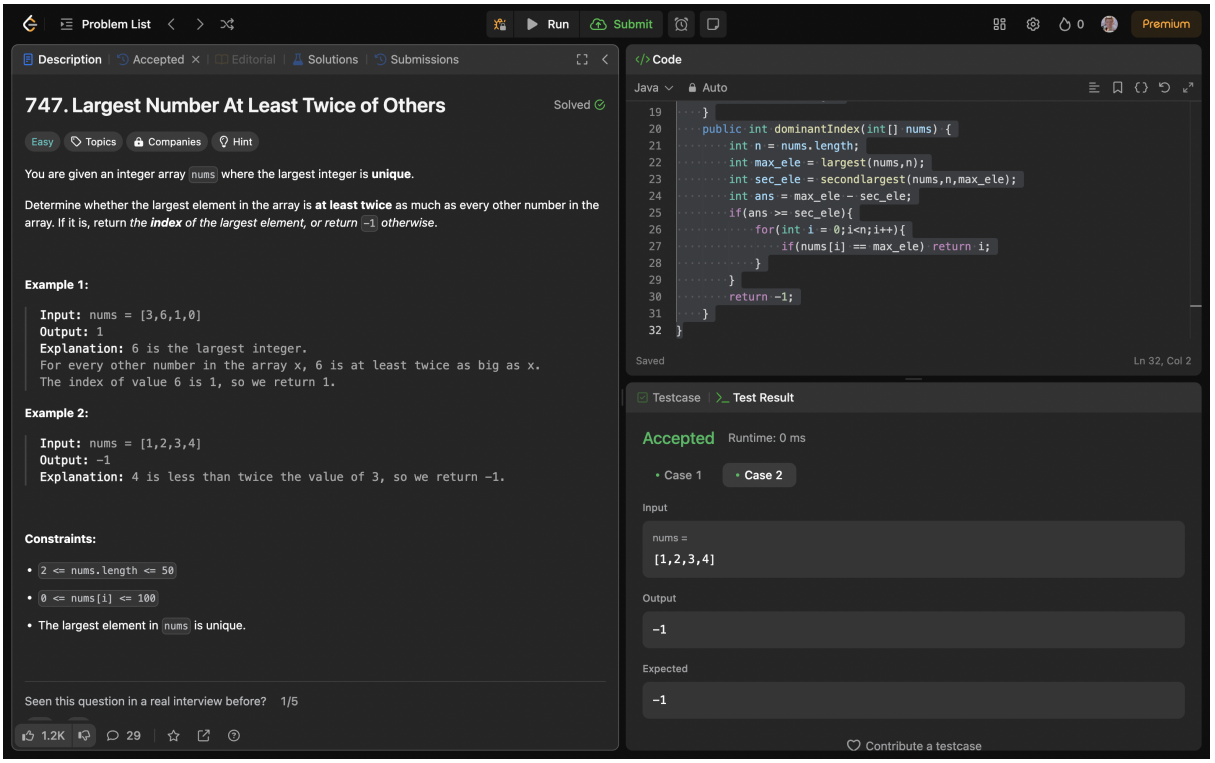
```
Input: nums = [1,2,3,4]
Output: -1
```

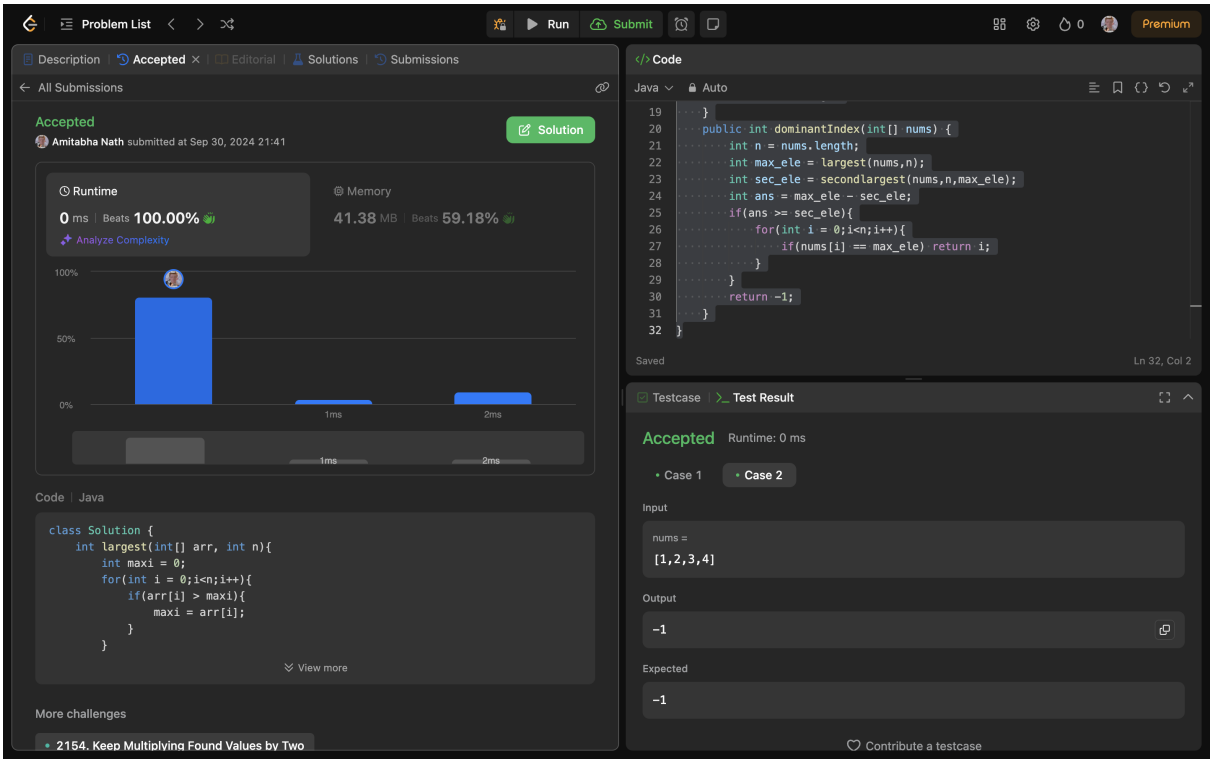
```
Explanation: 4 is less than twice the value of 3, so we ret
urn -1.
```

Constraints:

- `2 <= nums.length <= 50`
- `0 <= nums[i] <= 100`
- The largest element in `nums` is unique.

ScreenShots:





Code for this Question

```
class Solution {
    int largest(int[] arr, int n){
        int maxi = 0;
        for(int i = 0;i<n;i++){
            if(arr[i] > maxi){
                maxi = arr[i];
            }
        }
        return maxi;
    }
    int secondlargest(int []arr, int n, int x){
        int sec_large = 0;
        for(int i = 0;i<n;i++){
            if(arr[i] > sec_large && arr[i] < x){
                sec_large = arr[i];
            }
        }
        return sec_large;
    }
    public int dominantIndex(int[] nums) {
```

```
        int n = nums.length;
        int max_ele = largest(nums,n);
        int sec_ele = secondlargest(nums,n,max_ele);
        int ans = sec_ele*2;
        if(ans <= max_ele){
            for(int i = 0;i<n;i++){
                if(nums[i] == max_ele) return i;
            }
        }
        return -1;
    }
}
```

Logic

- This code compares the largest value with the second largest value to determine the result.
- If twice the second largest value is greater than the largest value, it returns -1. Otherwise, it returns the index of the largest value.
- The time complexity of this code is O(N).
- The space complexity of this code is O(1).

Handwritten Note:

```
class Solution {

    int largest(int[] arr, int n) {
        int maxi = 0;
        for(int i = 0; i < n; i++) {
            if(arr[i] > maxi) maxi = arr[i];
        }
        return maxi;
    }

    int secondlargest(int arr[], int n, int x) {
        int sec_large = 0;
        for(int i = 0; i < n; i++) {
            if(arr[i] > sec_large && arr[i] < x)
                sec_large = arr[i];
        }
        return sec_large;
    }

    public int dominantIndex(int[] nums) {
        int n = nums.length;
        int max_ele = largest(nums, n);
        int sec_ele = secondlargest(nums, n, max_ele);
        int ans = sec_ele * 2;
        if(ans <= max_ele) {
            for(int i = 0; i < n; i++) {
                if(nums[i] == max_ele) return i;
            }
        }
        return -1;
    }
}
```