**WEB TECHNOLOGY**

**Lab Assignment 4**

**Name: Atharva Kavade**

**Div: SY-AIDS-B**

**Roll No.: 12**

**PRN No: 12310269**

**1. Continuous Subarray Sum: Given an unsorted array of non-negative integers and a target sum, find a continuous subarray that sums up to the target sum and return the left and right indices (1-based indexing) of that subarray.**

**JavaScript code**

*class* Solution {

    // Function to find a continuous sub-array which adds up to a given number.

    subarraySum(arr, n, s) {

        // your code here

       let start = 0;

       let current\_sum = arr[0];

*for* (let end = 1; end <= n; end++) {

*while* (current\_sum > s && start < end - 1) {

               current\_sum -= arr[start];

               start++;

           }

*if* (current\_sum === s) {

               return [start + 1, end];

           }

*if* (end < n) {

               current\_sum += arr[end];

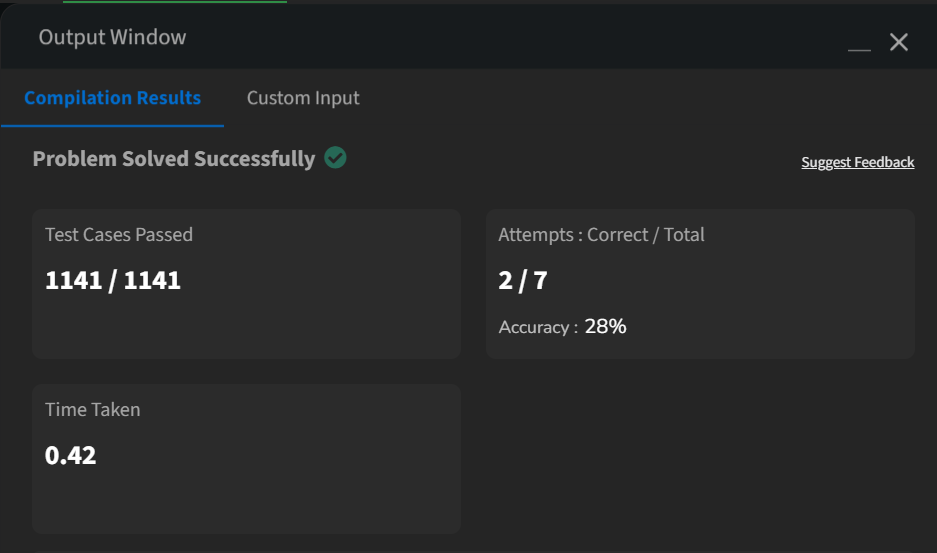
           }

       }

       return [-1];

    }

}



**2. Missing Number: Given an array of size N-1 containing distinct integers in the range of 1 to N, find the missing element.**

*class* Solution {

    missingNumber(n, arr) {

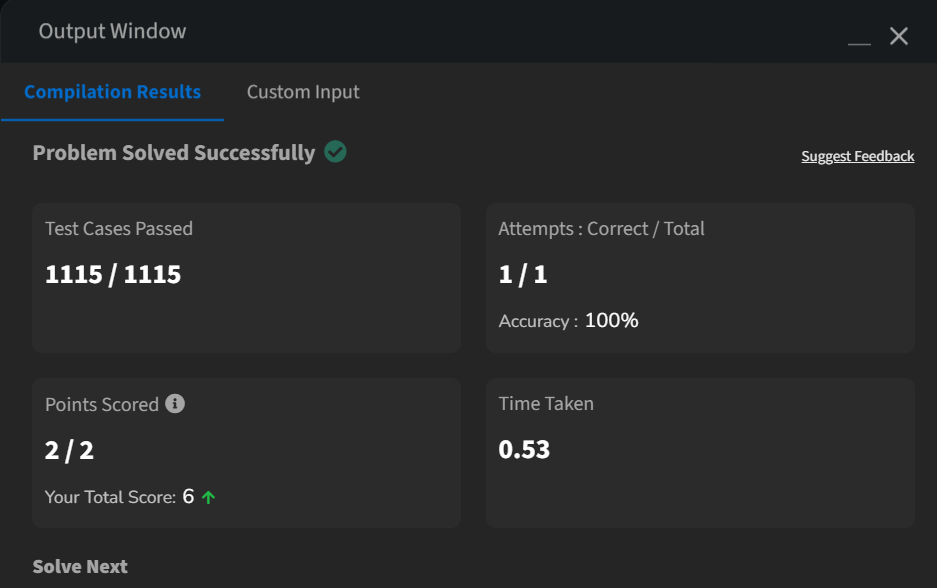
        let sum\_N = n \* (n + 1) / 2;

        let sum\_arr = arr.reduce((acc, num) => acc + num, 0);

*return* sum\_N - sum\_arr;

    }

}



**3. Maximum Subarray Sum: Given an integer array, find the contiguous subarray (containing at least one number) with the largest sum and return its sum.**

var maxSubArray = function(nums) {

    let max\_current = nums[0];

    let max\_global = nums[0];

    for (let i = 1; i < nums.length; i++) {

        max\_current = Math.max(nums[i], max\_current + nums[i]);

        if (max\_current > max\_global) {

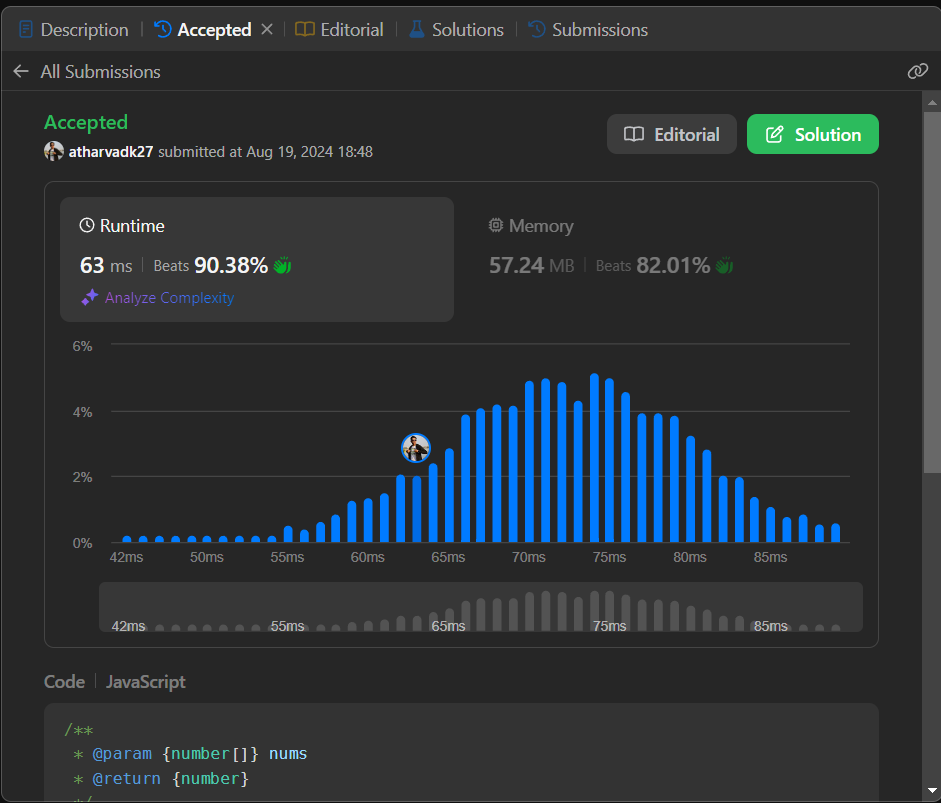
            max\_global = max\_current;

        }

    }

    return max\_global;

};



**4. Longest Consecutive Sequence: Given an unsorted array of integers, find the length of the longest consecutive elements sequence.**

var longestConsecutive = function(nums) {

    if (nums.length === 0) return 0;

    let numSet = new Set(nums);

    let maxLength = 0;

    for (let num of nums) {

        if (!numSet.has(num - 1)) {

            let currentNum = num;

            let currentLength = 1;

            while (numSet.has(currentNum + 1)) {

                currentNum += 1;

                currentLength += 1;

            }

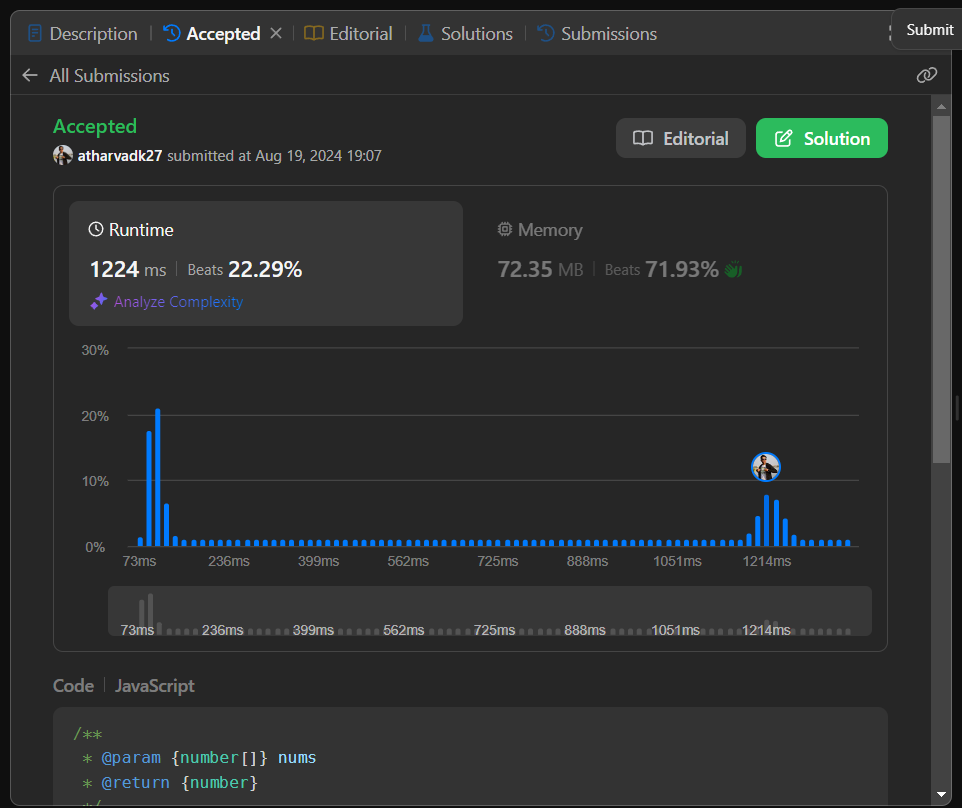
            maxLength = Math.max(maxLength, currentLength);

        }

    }

    return maxLength;

};

****