**Day 1**

**Student Name:** Arun Thakur **UID:** 22BCS14972

**Branch:** BE-CSE **Section/Group:** 22BCS\_FL\_IOT-601/B

**Semester:** 6TH  **Date of Performance:** 03/02/2025

**Subject Name:** Advanced Programming Lab - 2  **Subject Code:** 22CSH-352

**Problem 1:** [Remove Duplicates from Sorted Array - LeetCode](https://leetcode.com/problems/remove-duplicates-from-sorted-array/description/)

**Code:**

class Solution {

public:

    int removeDuplicates(vector<int>& nums) {

        int j = 1;

        for(int i = 1; i < nums.size(); i++){

            if(nums[i] != nums[i - 1]){

                nums[j] = nums[i];

                j++;

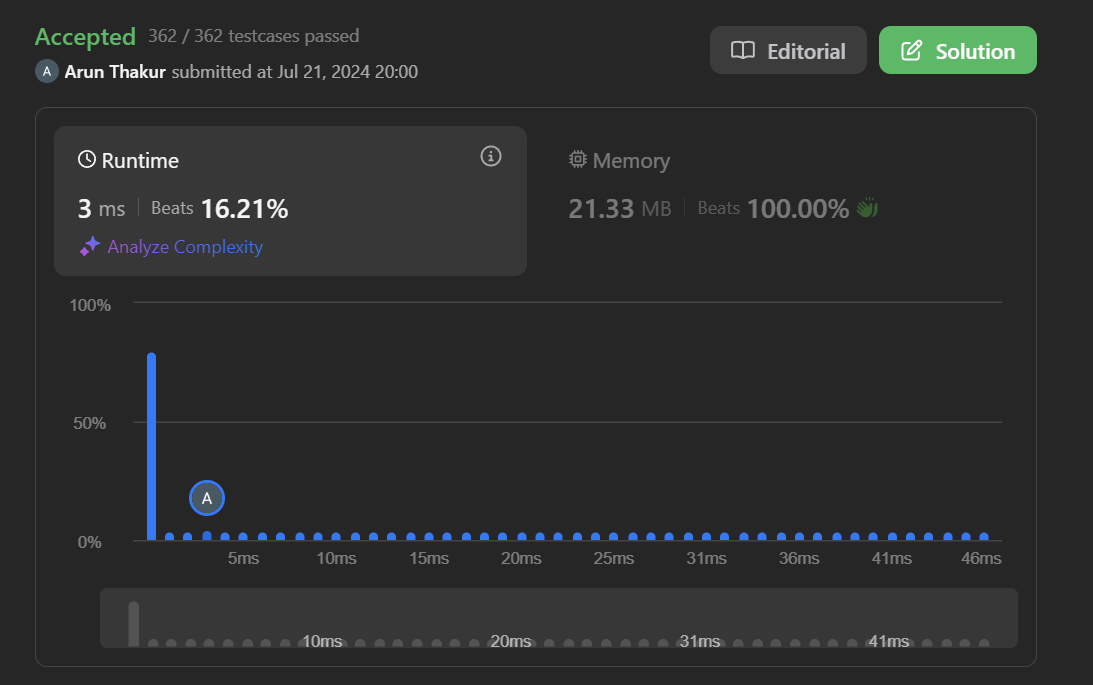
            }

        }

        return j;

    }

};

**Screenshot:**

**Problem 2:** [Insertion Sort | Practice | GeeksforGeeks](https://www.geeksforgeeks.org/problems/insertion-sort/1)

**Code:**

class Solution {

public:

// Please change the array in-place

void insertionSort(vector<int>& arr) {

// code here

int n = arr.size();

for (int i = 1; i < n; ++i) {

int key = arr[i];

int j = i - 1;

while (j >= 0 && arr[j] > key) {

arr[j + 1] = arr[j];

j = j - 1;

}

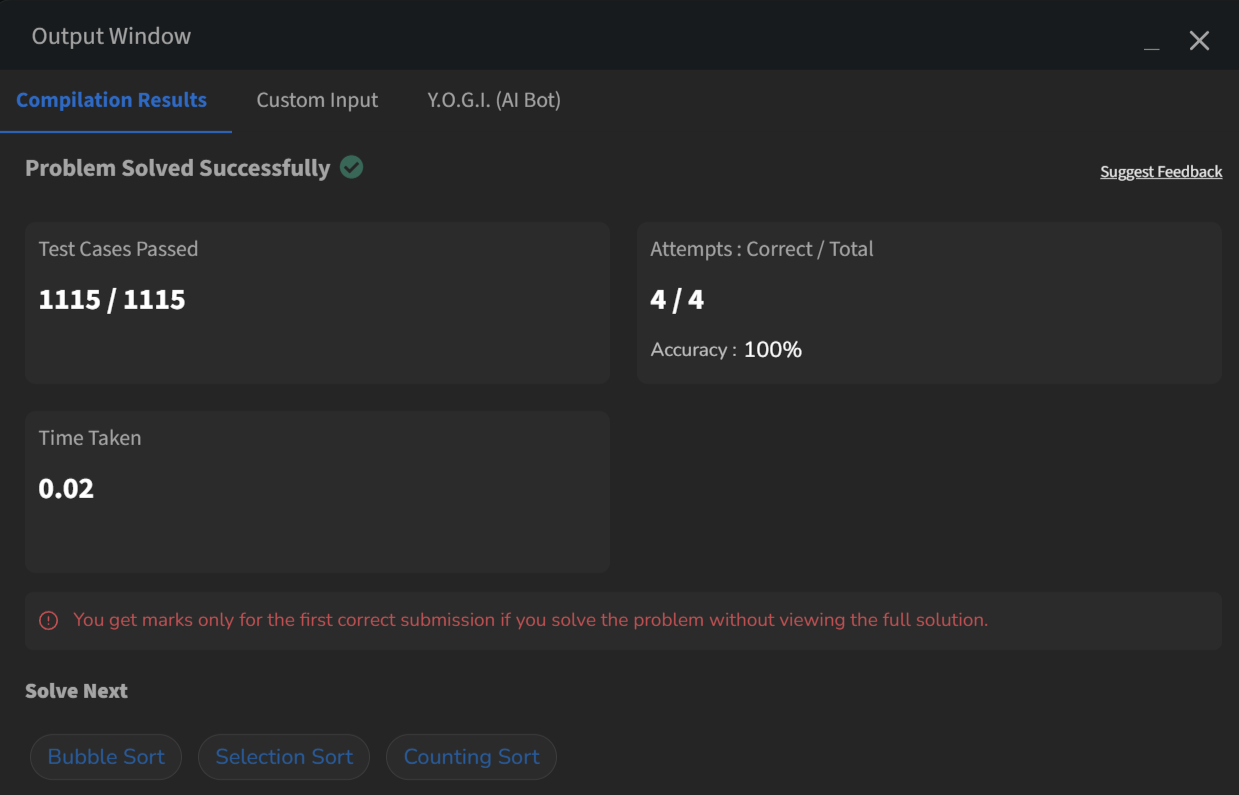
arr[j + 1] = key;

}

}

};

**Screenshot:**

****

**Problem 3:** [Contains Duplicate - LeetCode](https://leetcode.com/problems/contains-duplicate/description/)

**Code:**

class Solution {

public:

    bool containsDuplicate(vector<int>& nums) {

        unordered\_map <int,int> mp;

        for(int i =0; i<nums.size();i++){

            mp[nums[i]]++;

            if(mp[nums[i]]>1){

                return true;

            }

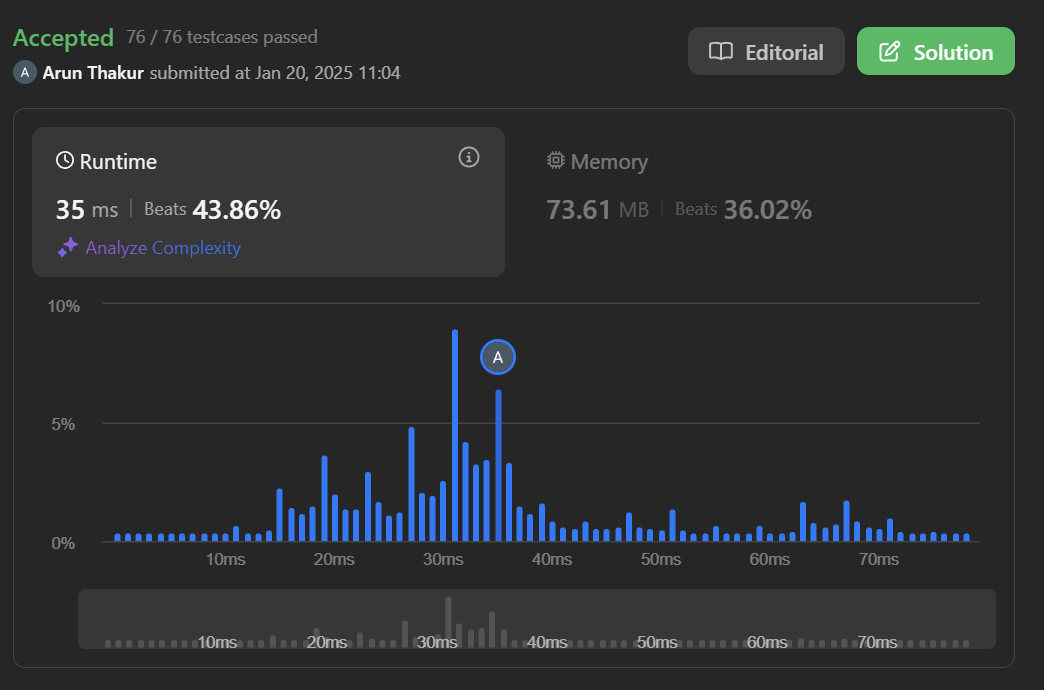
        }

        return false;

    }

};

**Screenshot:**

****

**Problem 4:** <https://leetcode.com/problems/two-sum/>

**Code:**

class Solution {

public:

    vector<int> twoSum(vector<int>& nums, int target) {

        int n = nums.size();

        vector <int> ind;

        int sum =0;

        for(int i =0; i< n; i++){

            for(int j =i+1; j<n;j++){

                if(nums[i]+nums[j]==target){

                return ind = {i,j};

            }

            }

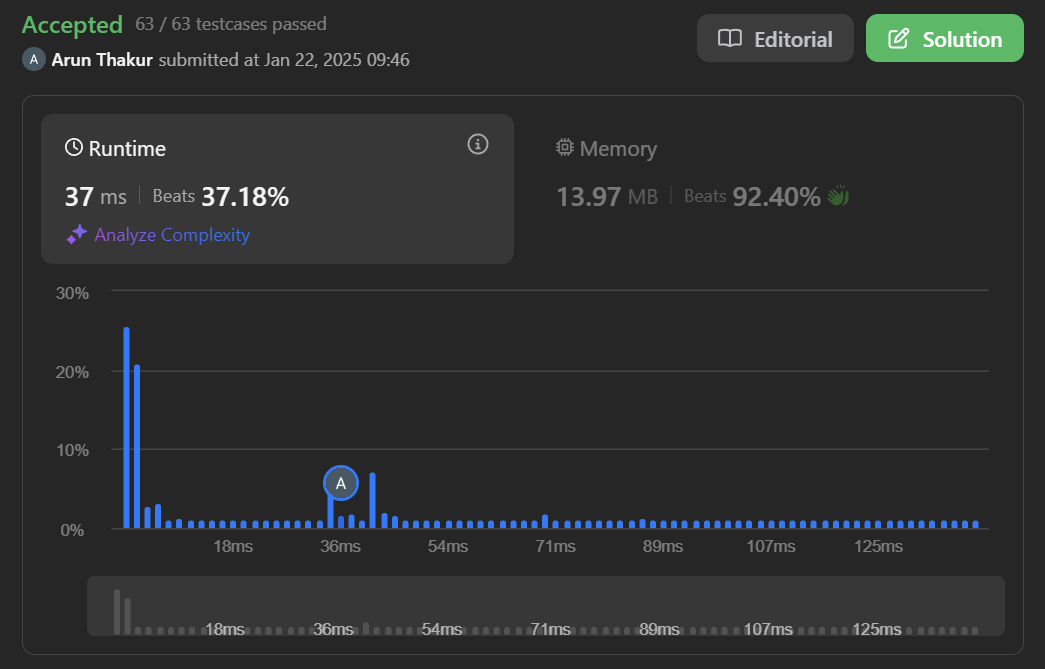
        }

        return ind;

    }

};

**Screenshot:**

****

**Problem 5:** [Jump Game - LeetCode](https://leetcode.com/problems/jump-game/description/)

**Code:**

class Solution {

public:

    bool canJump(vector<int>& nums) {

        int max = nums.size() - 1;

        for (int i = nums.size() - 1; i >= 0; i--) {

            if (i + nums[i] >= max) {

                max = i;

            }

        }

        return max == 0;

    }

};

**Screenshot:**

****

**Problem 6:** [Majority Element - LeetCode](https://leetcode.com/problems/majority-element/)

**Code:**

class Solution {

public:

    int majorityElement(vector<int>& nums) {

        int n = nums.size();

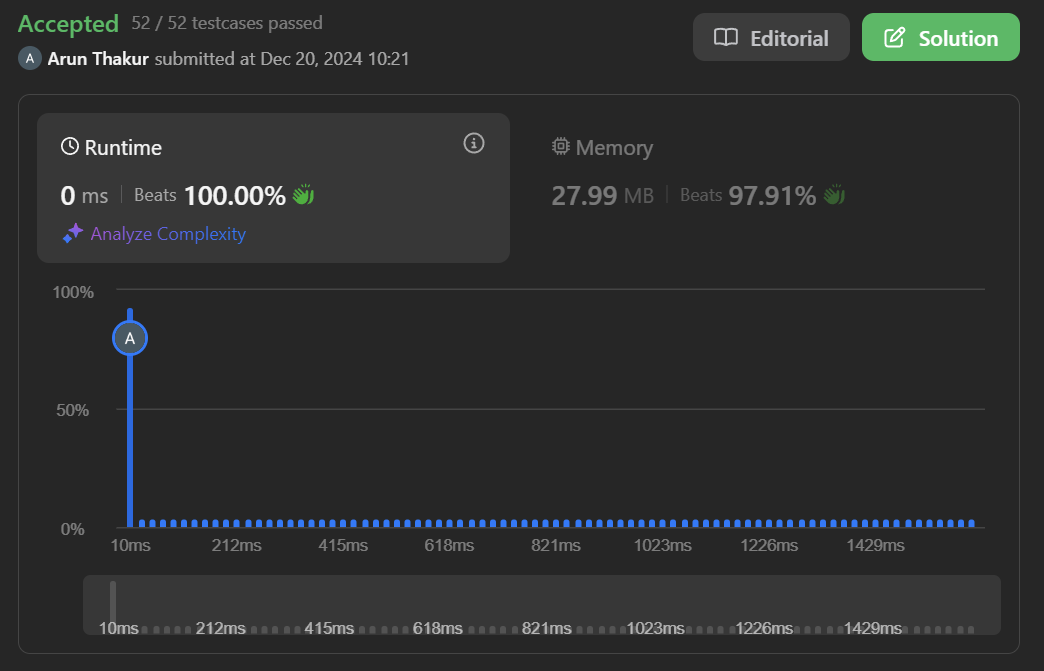
        sort(nums.begin(), nums.end());

        return nums[n/2];

    }

};

**Screenshot:**

****

**Problem 7:** [Valid Palindrome - LeetCode](https://leetcode.com/problems/valid-palindrome/)

**Code:**

class Solution {

public:

    bool isPalindrome(string s) {

        vector<char> arr;

        for(char c : s){

            if(!isalnum(c)){

                continue;

            }

            else{

                arr.push\_back(tolower(c));

                cout<<c;

            }

        }

        vector <char> n = arr;

        reverse(n.begin(),n.end());

        if(n==arr){

            return true;

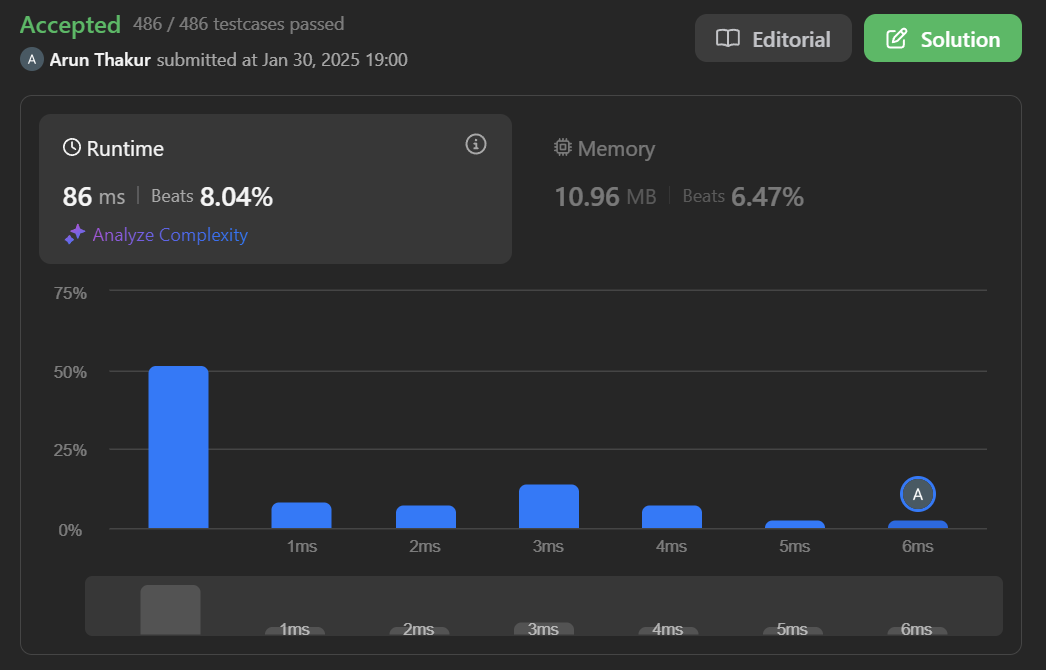
        }

        return false;

    }

};

**Screenshot:**



**Problem 8:** [3Sum - LeetCode](https://leetcode.com/problems/3sum/description/)

**Code:**

class Solution {

public:

    vector<vector<int>> threeSum(vector<int>& nums) {

        std::vector<std::vector<int>> result;

        int n = nums.size();

        std::sort(nums.begin(), nums.end());

        for (int i = 0; i < n - 2; ++i) {

            if (i > 0 && nums[i] == nums[i - 1]) continue;

            int left = i + 1, right = n - 1;

            while (left < right) {

                int sum = nums[i] + nums[left] + nums[right];

                if (sum == 0) {

                    result.push\_back({nums[i], nums[left], nums[right]});

                    ++left;

                    --right;

                    while (left < right && nums[left] == nums[left - 1]) ++left;

                    while (left < right && nums[right] == nums[right + 1]) --right;

                } else if (sum < 0) {

                    ++left;

                } else {

                    --right;

                }

            }

        }

        return result;

    }

};

**Screenshot:**

****

**Problem 9:** [Set Matrix Zeroes - LeetCode](https://leetcode.com/problems/set-matrix-zeroes/)

**Code:**

class Solution {

public:

    void setZeroes(vector<vector<int>>& matrix) {

        int rows = matrix.size();

        int cols = matrix[0].size();

        bool firstRowZero = false, firstColZero = false;

        for (int j = 0; j < cols; j++) {

            if (matrix[0][j] == 0) {

                firstRowZero = true;

                break;

            }

        }

        for (int i = 0; i < rows; i++) {

            if (matrix[i][0] == 0) {

                firstColZero = true;

                break;

            }

        }

        for (int i = 1; i < rows; i++) {

            for (int j = 1; j < cols; j++) {

                if (matrix[i][j] == 0) {

                    matrix[i][0] = 0;

                    matrix[0][j] = 0;

                }

            }

        }

        for (int i = 1; i < rows; i++) {

            for (int j = 1; j < cols; j++) {

                if (matrix[i][0] == 0 || matrix[0][j] == 0) {

                    matrix[i][j] = 0;

                }

            }

        }

        if (firstRowZero) {

            for (int j = 0; j < cols; j++) {

                matrix[0][j] = 0;

            }

        }

        if (firstColZero) {

            for (int i = 0; i < rows; i++) {

                matrix[i][0] = 0;

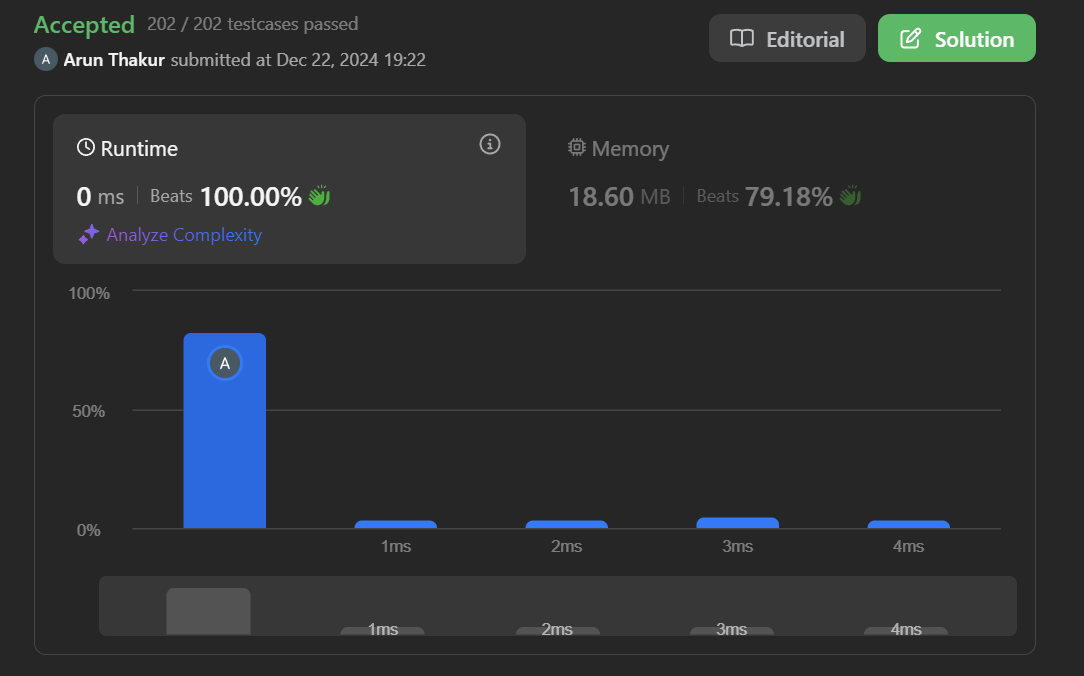
            }

        }

    }

};

**Screenshot:**

****

**Problem 10:** [Longest Substring Without Repeating Characters - LeetCode](https://leetcode.com/problems/longest-substring-without-repeating-characters/description/)

**Code:**

class Solution {

public:

    int lengthOfLongestSubstring(string s) {

        unordered\_map<char, int> mp;

        int left = 0, maxLen = 0;

        for (int right = 0; right < s.size(); right++) {

            mp[s[right]]++;

            while (mp[s[right]] > 1) {

                mp[s[left]]--;

                left++;

            }

            maxLen = max(maxLen, right - left + 1);

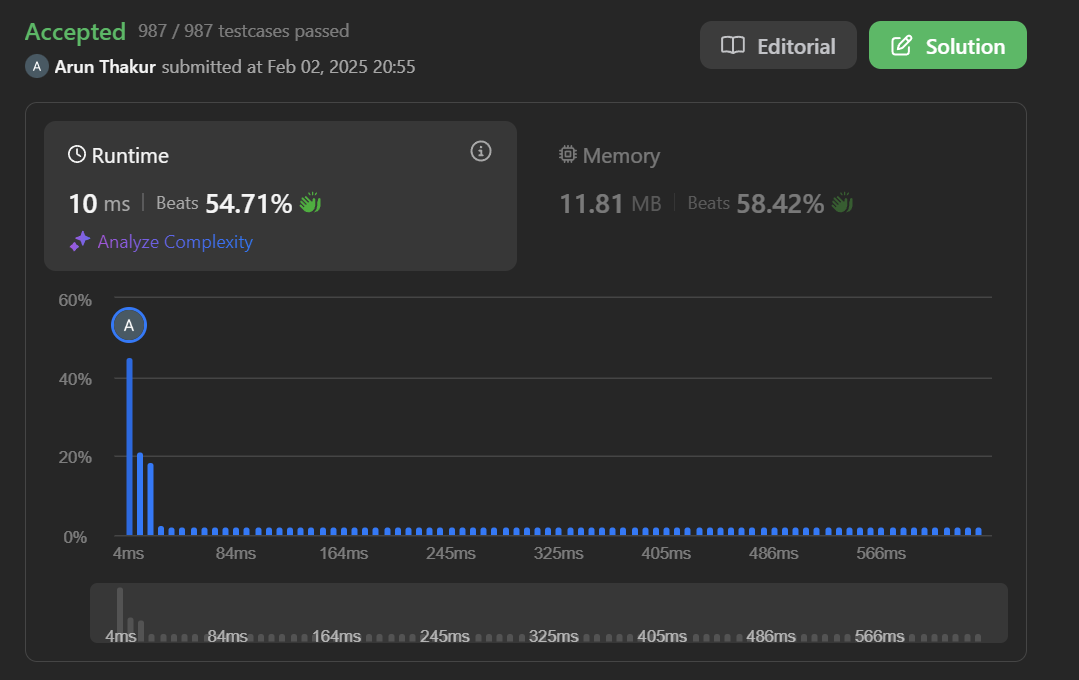
        }

        return maxLen;

    }

};

**Screenshot:**

****

**Problem 11:** [Find the Duplicate Number - LeetCode](https://leetcode.com/problems/find-the-duplicate-number/)

**Code:**

class Solution {

public:

    int findDuplicate(vector<int>& nums) {

        unordered\_map <int,int> mp;

        for(int i : nums){

            mp[i]++;

            if(mp[i]>1){return i;}

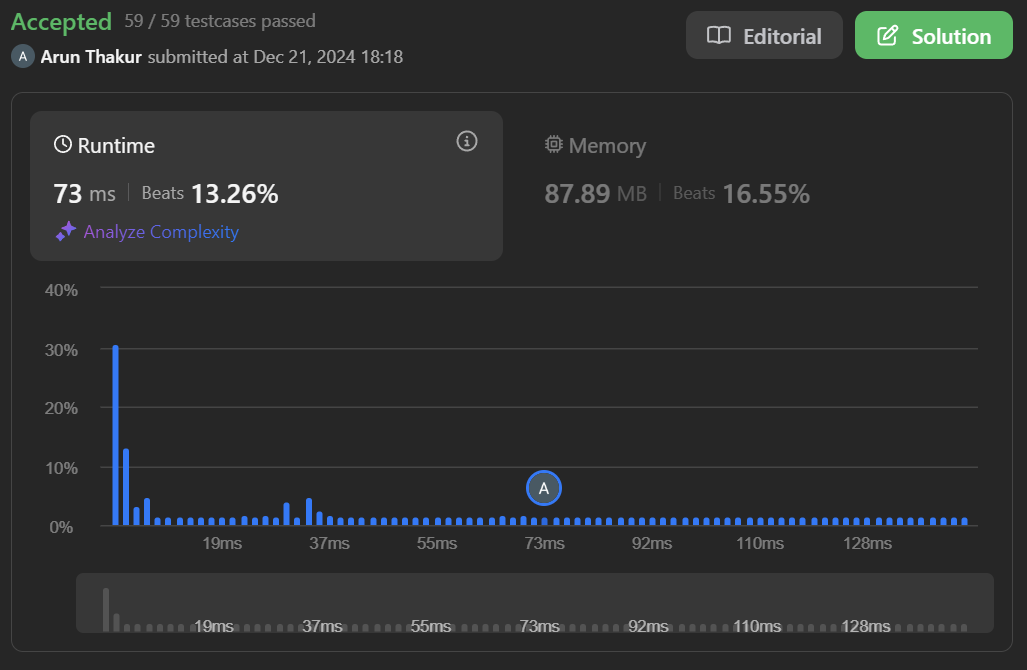
        }

        return 0;

    }

};

**Screenshot:**

****