**Day 1**

**Student Name:** Prabhnoor **UID:** 22BCS15639

**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

**Arrays**

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

**Code:**

class Solution {

public:

int removeDuplicates(vector<int>& nums) {

int n = nums.size();

int i=0;

int j = i+1;

while(j<n){

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

j++;

}

else{

swap(nums[i+1], nums[j]);

i++;

j++;

}

}

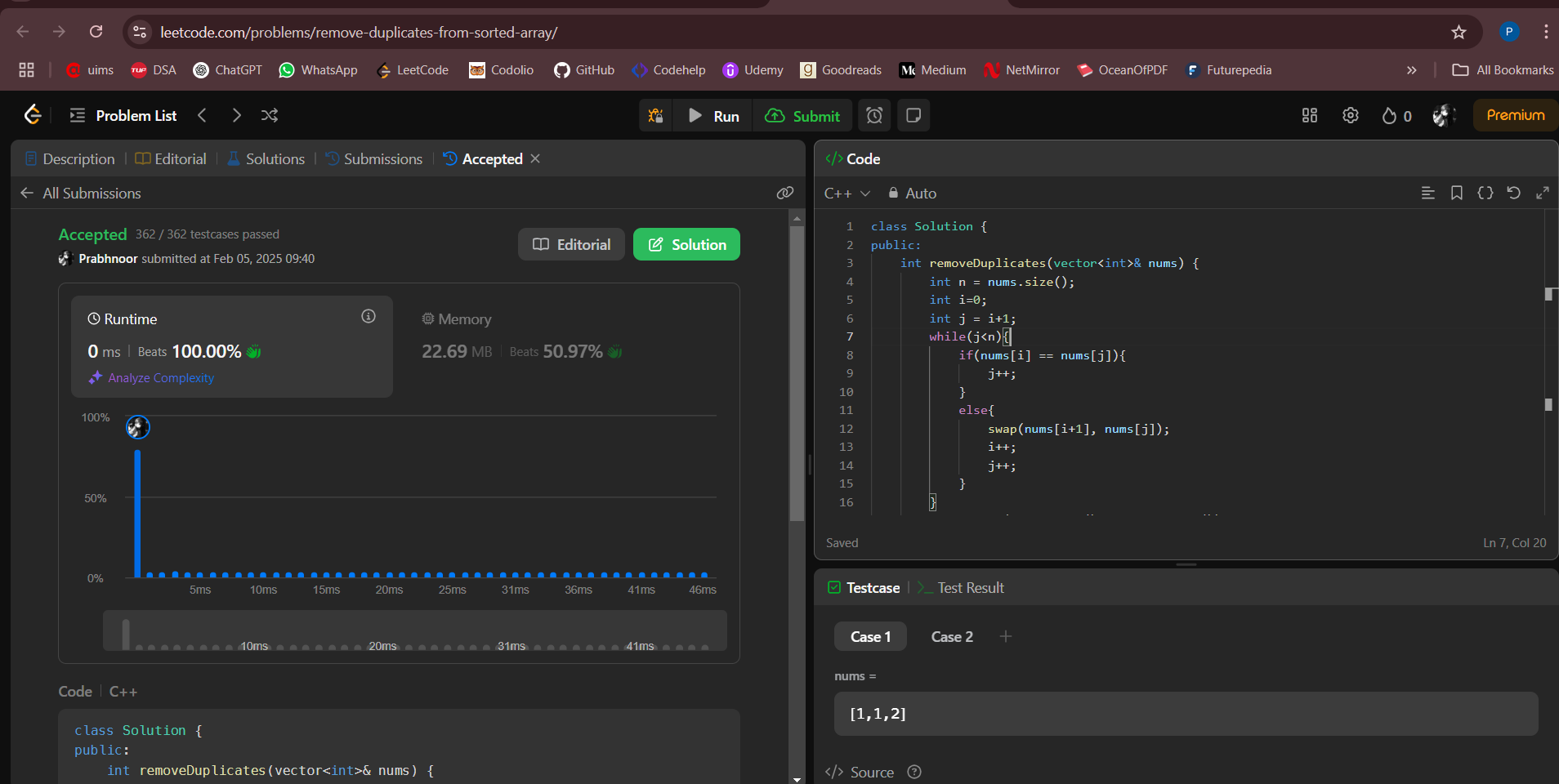
nums.erase(nums.begin()+i+1, nums.end());

return i+1;

}

};

**Acceptance Screenshot:**

****

**Problem 2:** [Implementing insertion sort](https://www.geeksforgeeks.org/problems/insertion-sort/1)

**Code:**

class solution{

public:

void insertionSort(vector<int>& arr) {

int n = arr.size();

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

for(int j=i; j>0; j--){

if(arr[j] < arr[j-1]){

swap(arr[j], arr[j-1]);

}

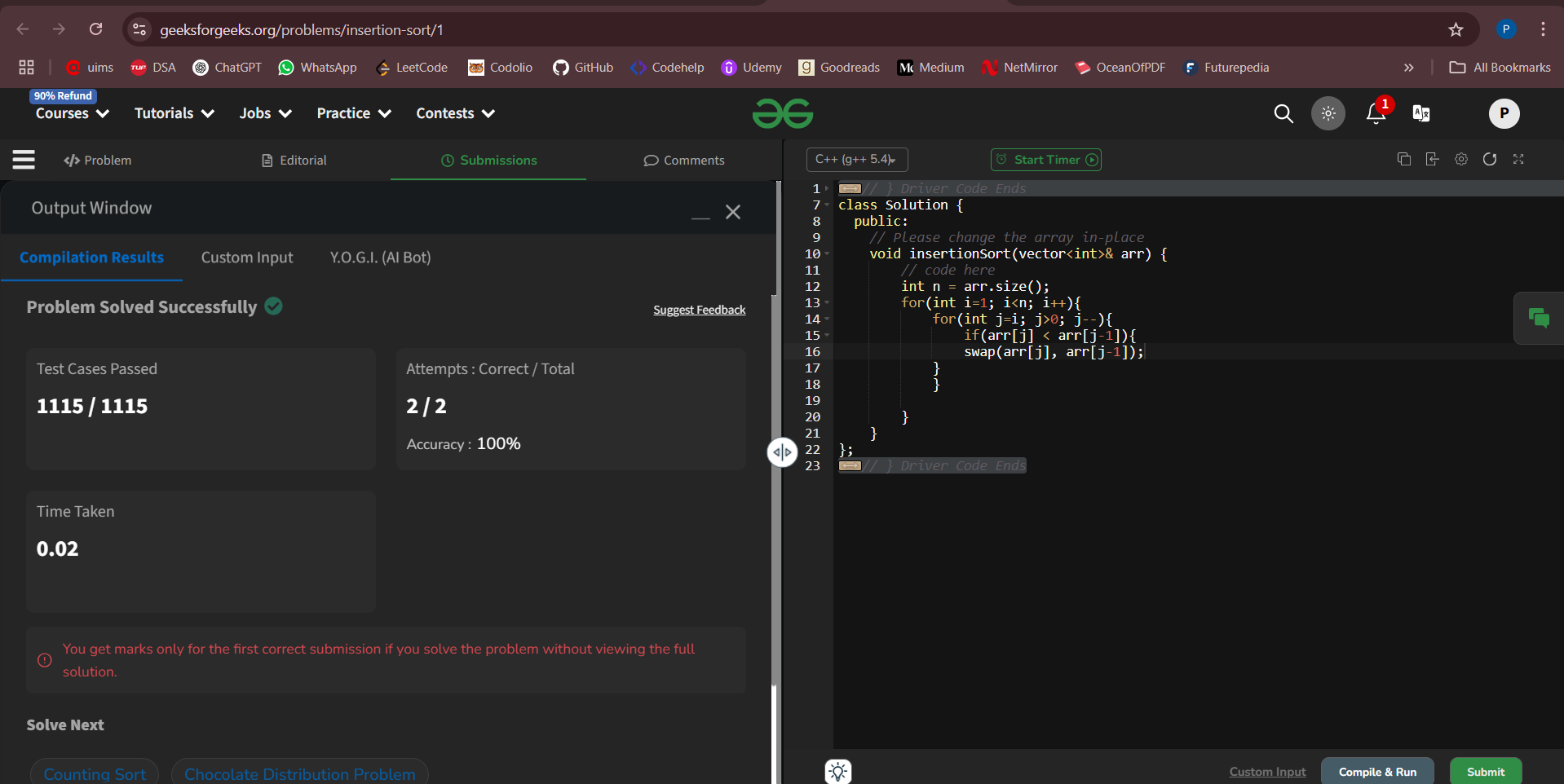
}

}

}

};

**Acceptance** **Screenshot:**



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

**Code:**

class Solution {

public:

bool containsDuplicate(vector<int>& nums) {

unordered\_map<int,int> mp;

for(auto i: nums){

mp[i]++;

}

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

if(mp[nums[i]]>1){

return true;

}

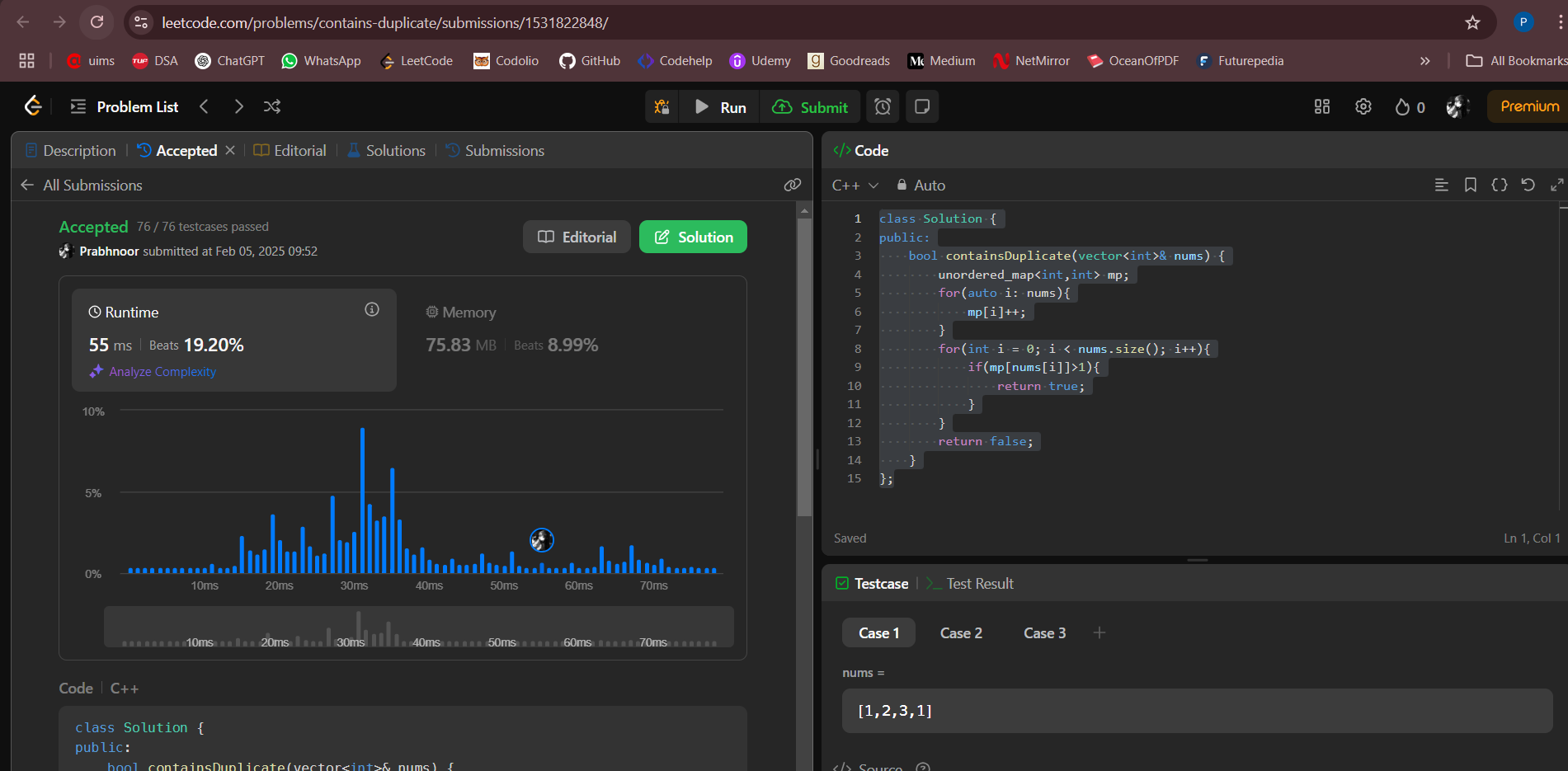
}

return false;

}

};

**Acceptance Screenshot:**



**Problem 4:** [Two Sum](https://leetcode.com/problems/two-sum/description/)

**Code:**

class Solution {

public:

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

int n = nums.size();

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

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

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

return {i,j};

}

}

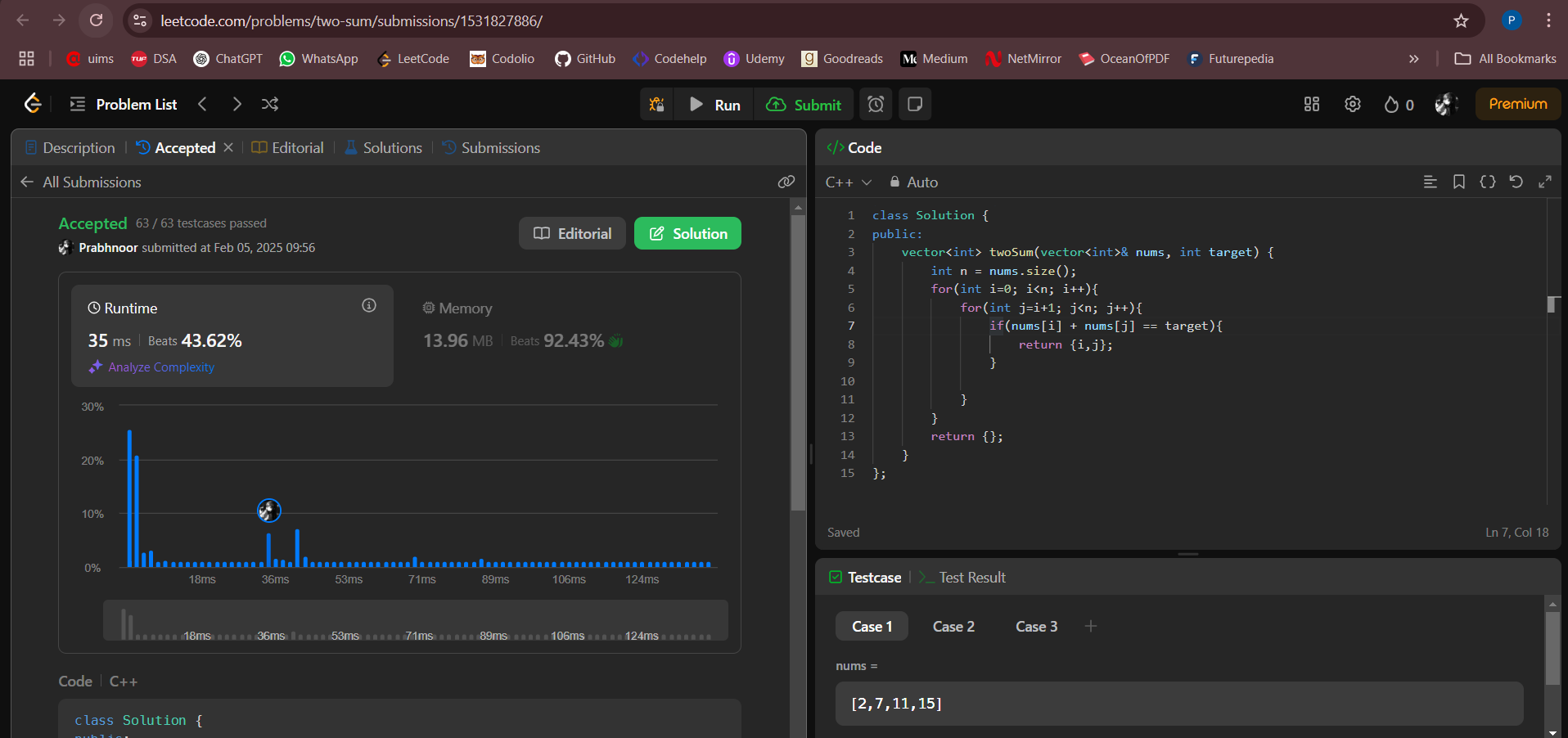
}

return {};

}

};

**Acceptance Screenshot:**

****

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

**Code:**

class Solution {

public:

bool canJump(vector<int>& nums) {

int n = nums.size();

int maxPoint = 0;

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

if(i>maxPoint){

return false;

}

maxPoint = max(maxPoint, nums[i] + i);

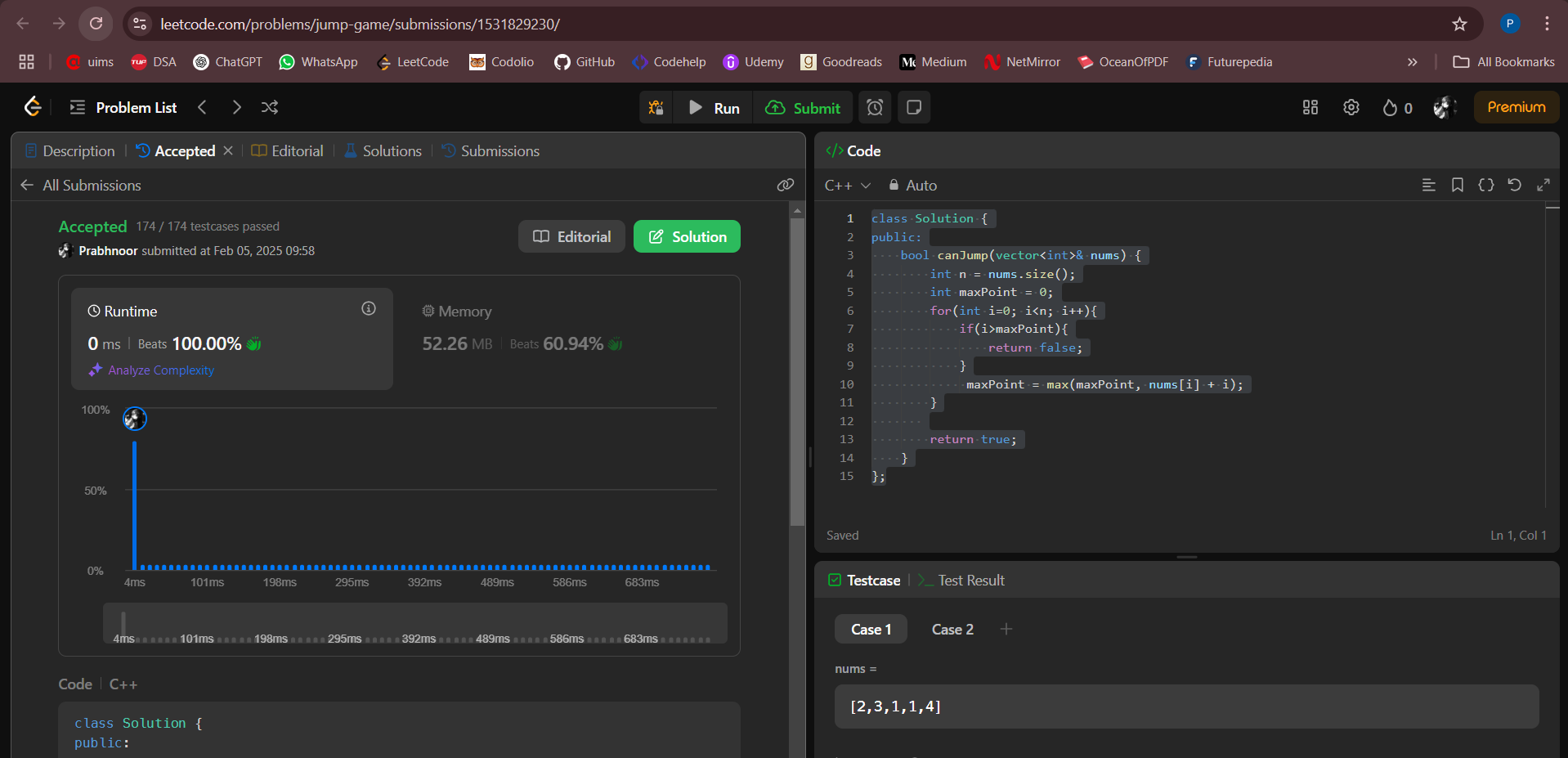
}

return true;

}

};

**Acceptance Screenshot:**

****

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

**Code:**

class Solution {

public:

int majorityElement(vector<int>& nums) {

int n = nums.size();

unordered\_map<int,int> mp;

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

mp[nums[i]]++;

}

for(auto it: mp){

if(it.second>n/2){

return it.first;

}

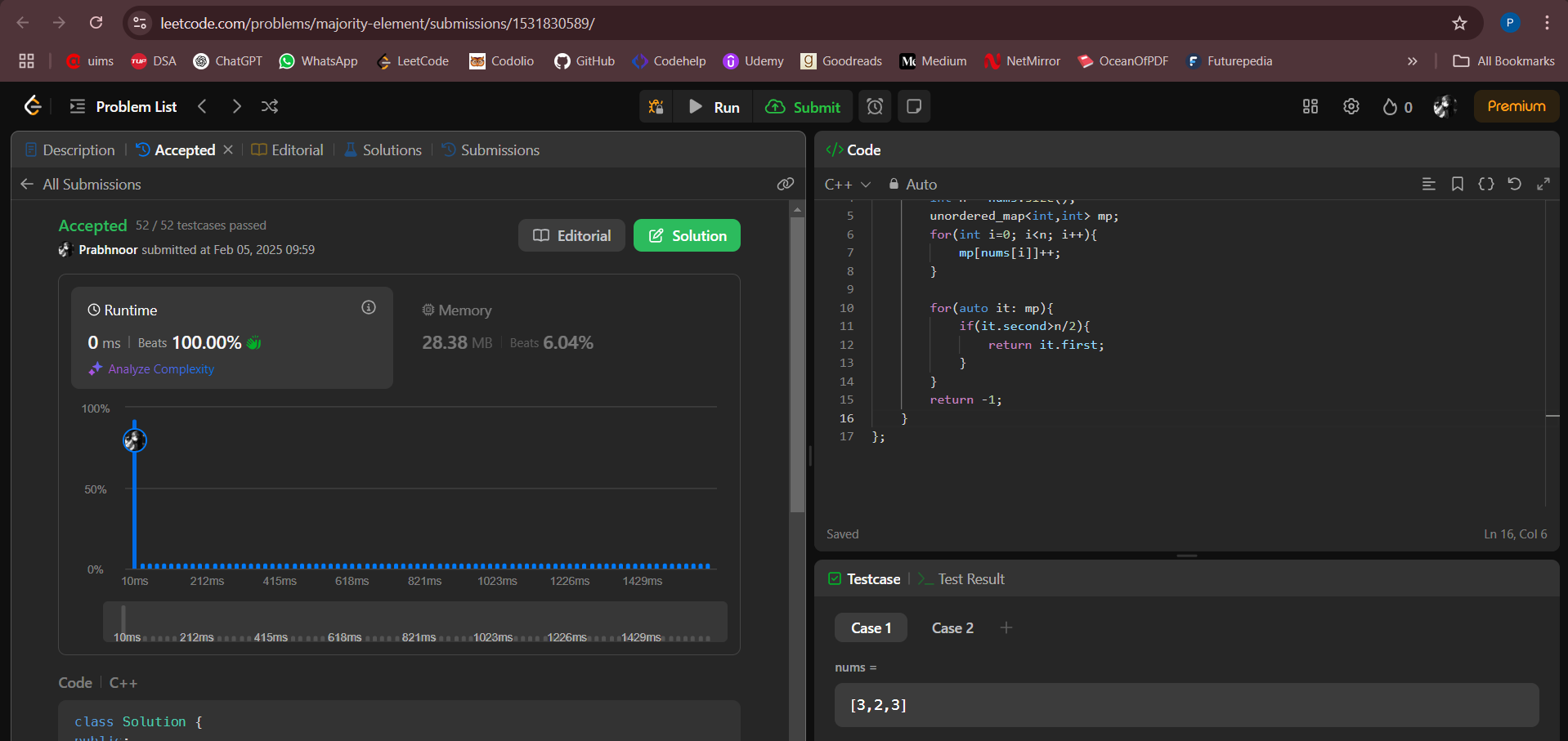
}

return -1;

}

};

**Acceptance Screenshot:**

****

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

**Code:**

class Solution {

public:

bool isPalindrome(string s) {

int i = 0;

int j = s.length()-1;

while(i<j){

while(i<j && !isalnum(s[i])) i++;

while(i<j && !isalnum(s[j])) j--;

if(tolower(s[i]) != tolower(s[j])) return false;

i++;

j--;

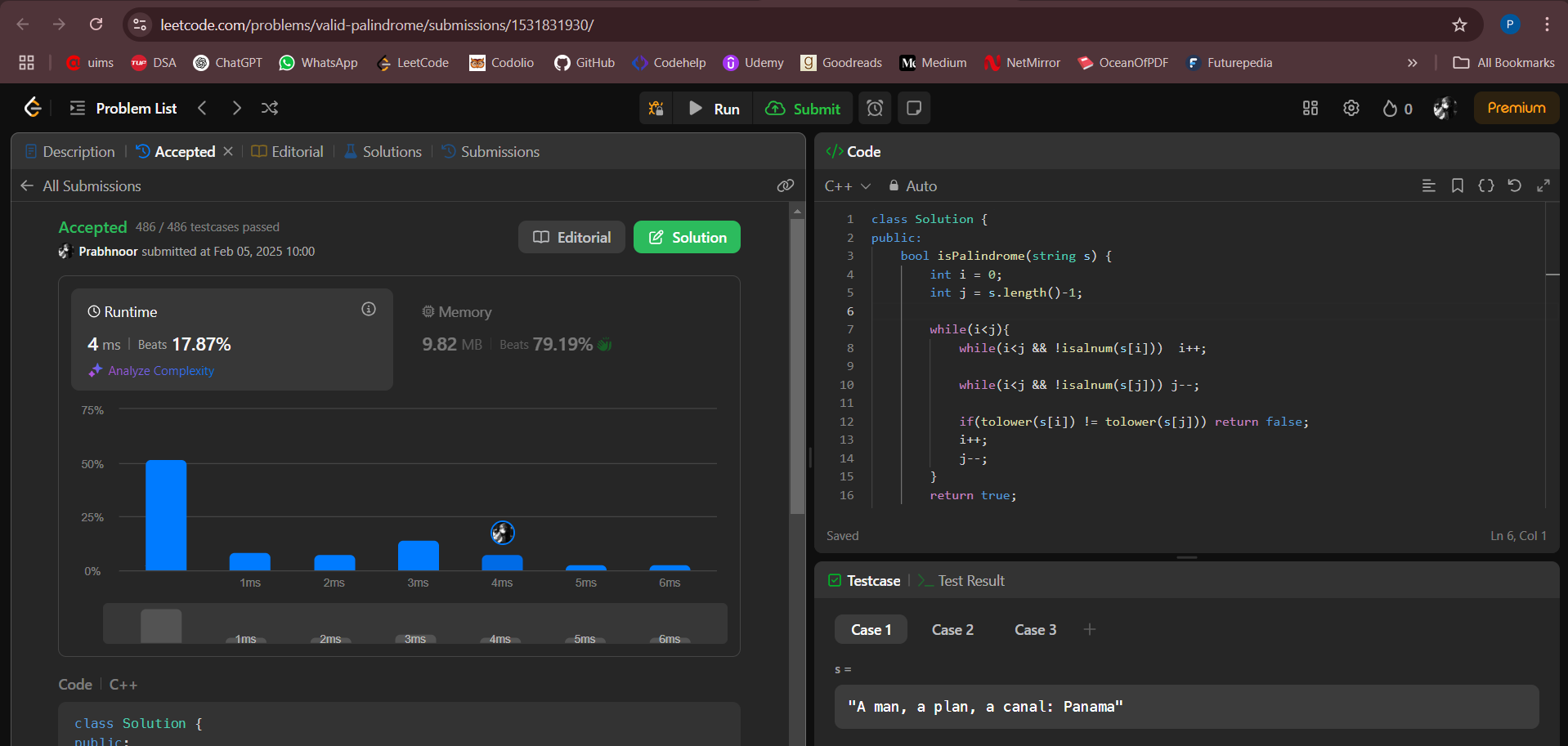
}

return true;

}

};

**Acceptance Screenshot:**

****

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

**Code:**

class Solution {

public:

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

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

vector<vector<int>> ans;

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

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

int j=i+1;

int k=nums.size()-1;

while(j<k){

int sum=nums[i]+nums[j]+nums[k];

if(sum<0){

j++;

}

else if(sum>0){

k--;

}

else{

vector<int> temp={nums[i],nums[j],nums[k]};

ans.push\_back(temp);

j++;

k--;

while(j<k && nums[j]==nums[j-1]) j++;

while(j<k && nums[k]==nums[k+1]) k--;

}

}

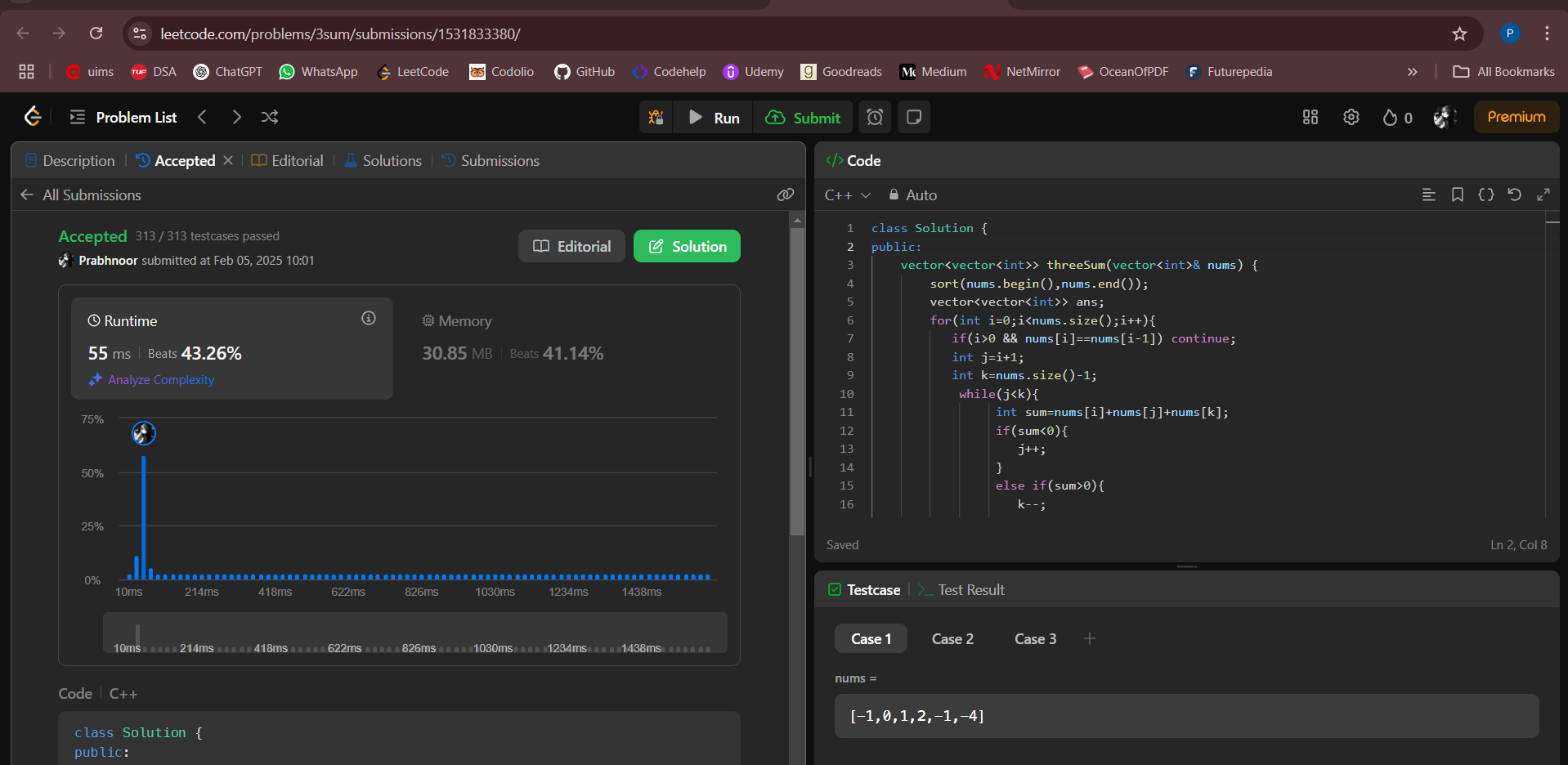
}

return ans;

}

};

**Acceptance Screenshot:**

****

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

**Code:**

class Solution {

public:

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

int m = matrix.size();

int n = matrix[0].size();

vector <int> row(m,0);

vector<int> column(n,0);

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

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

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

row[i] = 1;

column[j] = 1;

}

}

}

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

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

if(row[i] == 1 || column[j] == 1 ){

matrix[i][j] = 0;

}

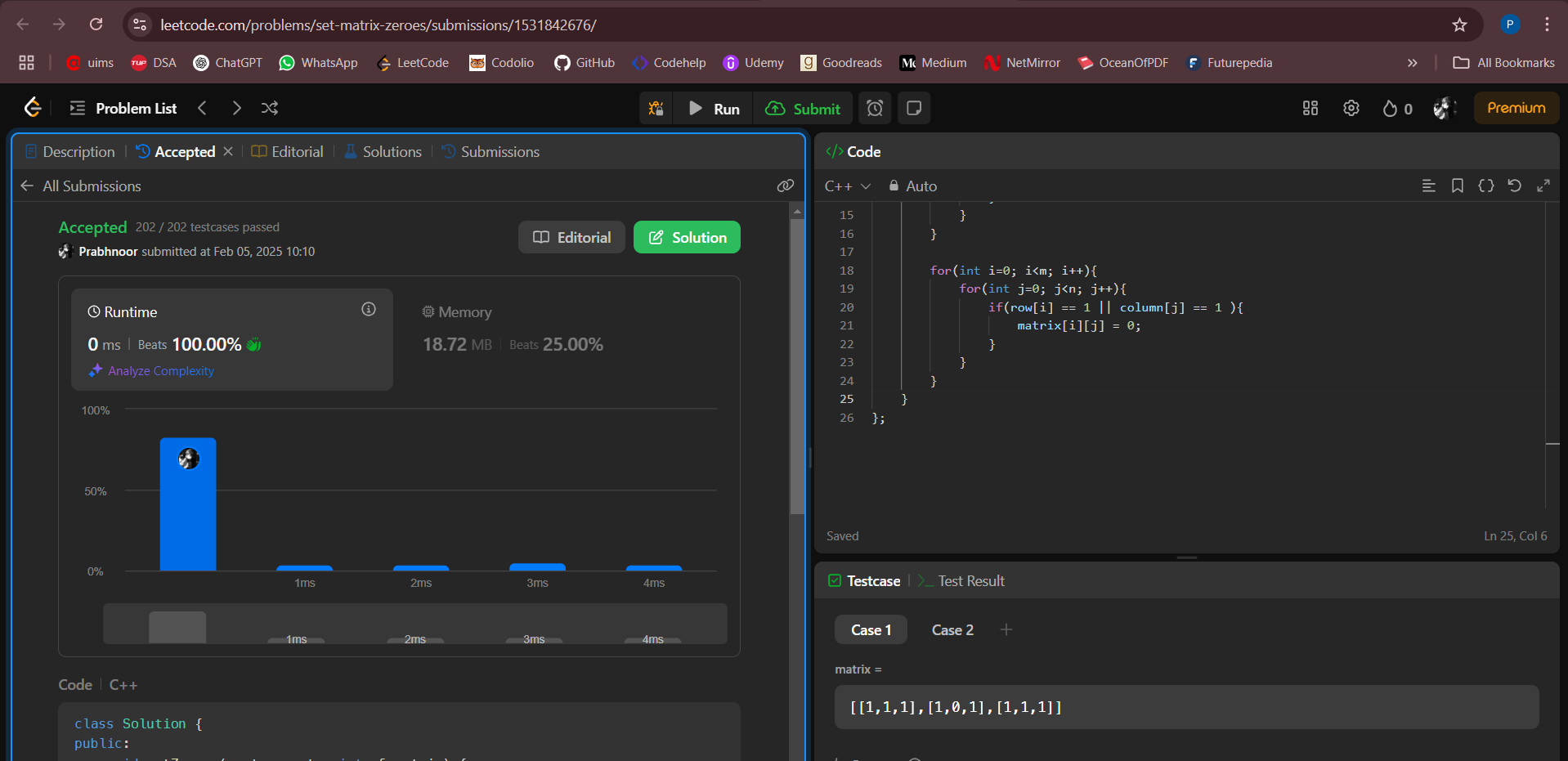
}

}

}

};

**Acceptance Screenshot:**

****

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

**Code:**

class Solution {

public:

int lengthOfLongestSubstring(string s) {

int answer = 0;

int j = -1;

vector<int> lastPosition(256,-1);

for(int i=0; i<s.length(); i++){

j=max(j,lastPosition[s[i]]);

answer = max(answer, i-j);

lastPosition[s[i]] = i;

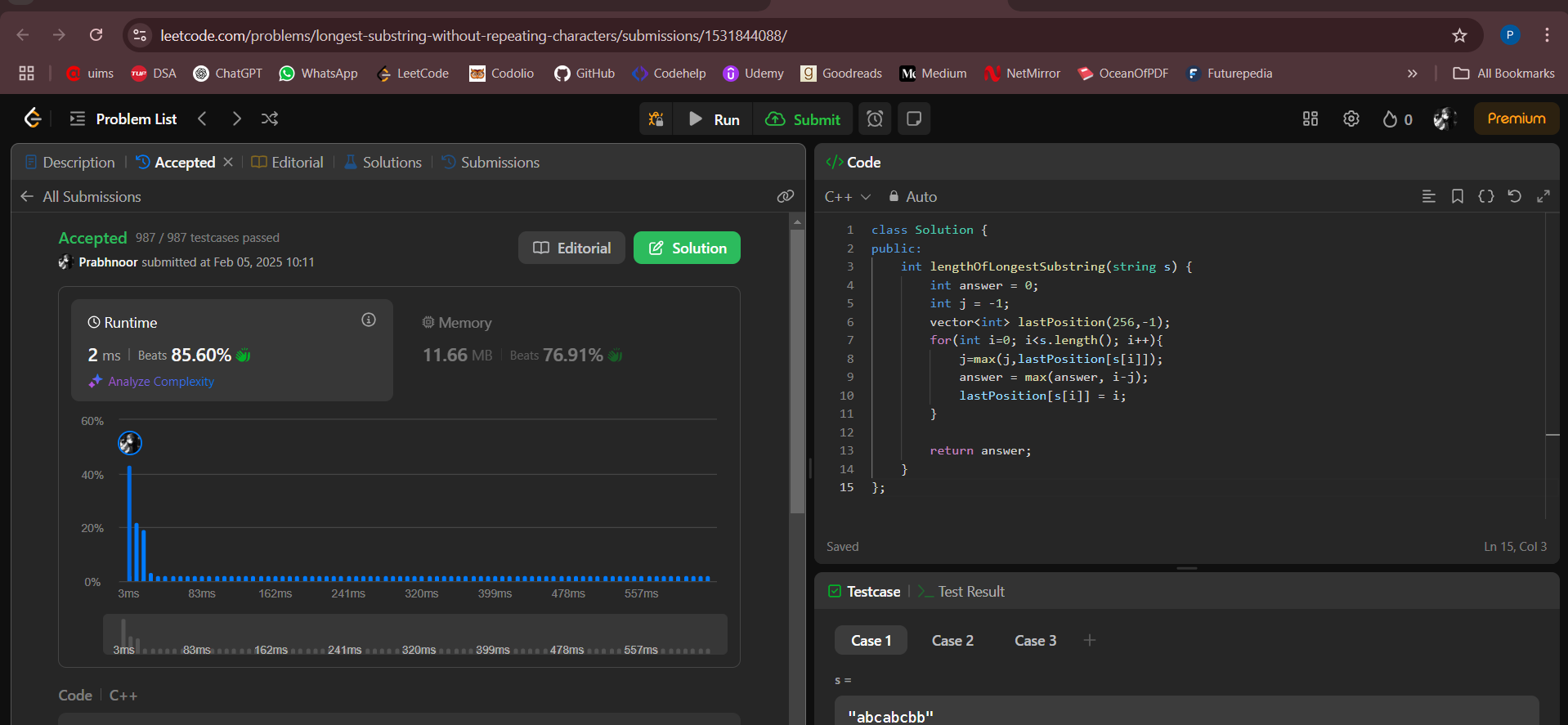
}

return answer;

}

};

**Acceptance Screenshot:**

****

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

**Code:**

class Solution {

public:

int findDuplicate(vector<int>& nums) {

unordered\_set<int> duplicate;

for(int i: nums){

if(duplicate.count(i)){

return i;

}

duplicate.insert(i);

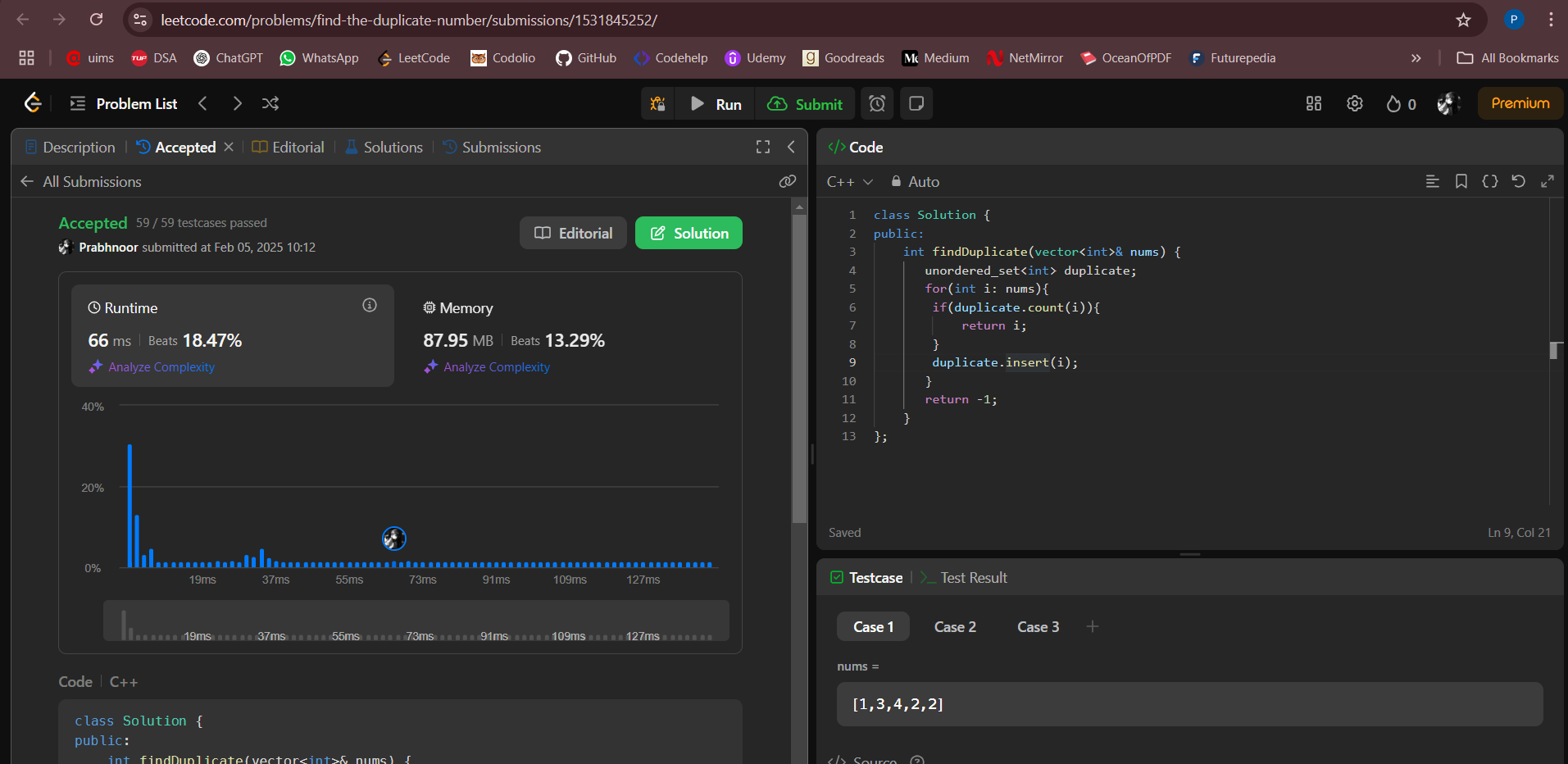
}

return -1;

}

};

**Acceptance Screenshot:**

****

**Problem 12:** [Jump Game 2](https://leetcode.com/problems/jump-game-ii/description/)

**Code:**

class Solution {

public:

int jump(vector<int>& nums) {

int n = nums.size();

if (n == 1) return 0;

int count = 0;

int index = 0;

int maxi = 0;

for (int i = 0; i < n - 1; i++) {

maxi = max(maxi, i + nums[i]);

if (i == index) {

count++;

index = maxi;

if (index >= n - 1) break;

}

}

return count;

}

};

**Acceptance Screenshot:**

