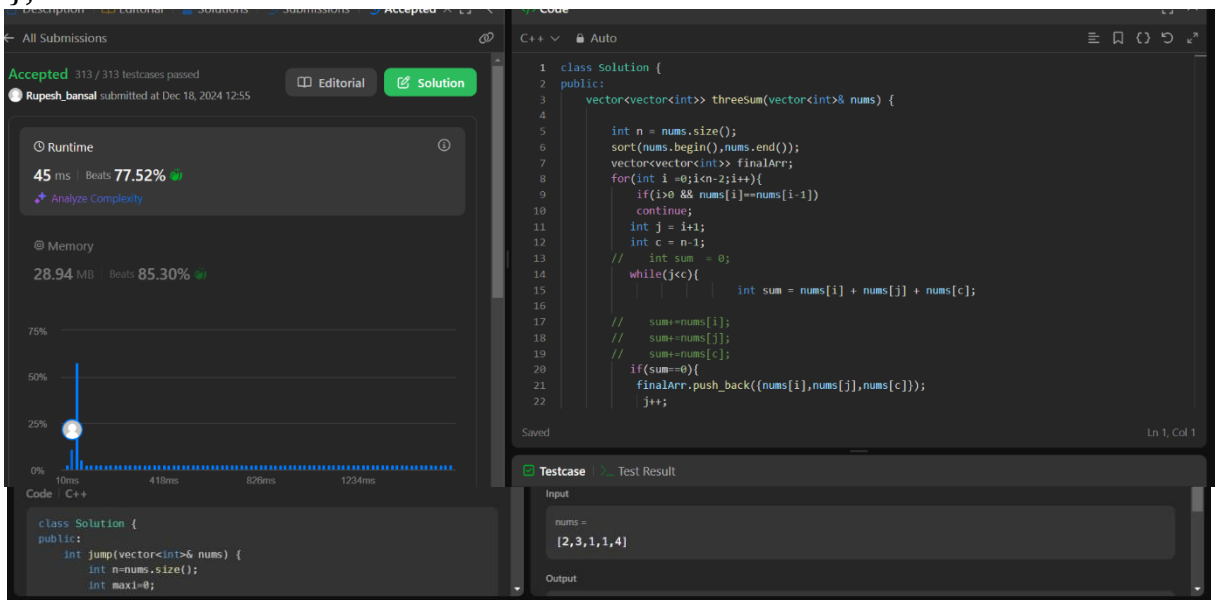
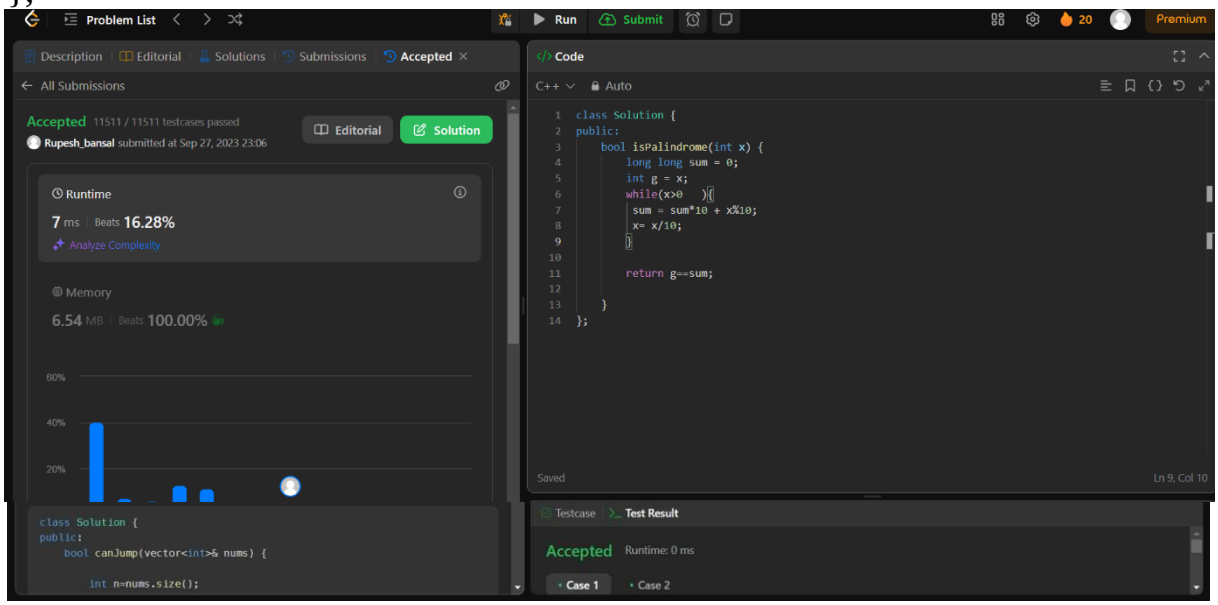
1. Jump Game II

```
class Solution {
public:
    int jump(vector<int>& nums) {
        int n=nums.size();
        int maxi=0;
        int current=0;
        int count=0;
        for(int i=0;i<n-1;i++){
            maxi=max(maxi,i+nums[i]);
            if(i==current){
                count++;
                current=maxi;
                if(current>=n-1)break;
            }
        }
        return count;
    }
};
```



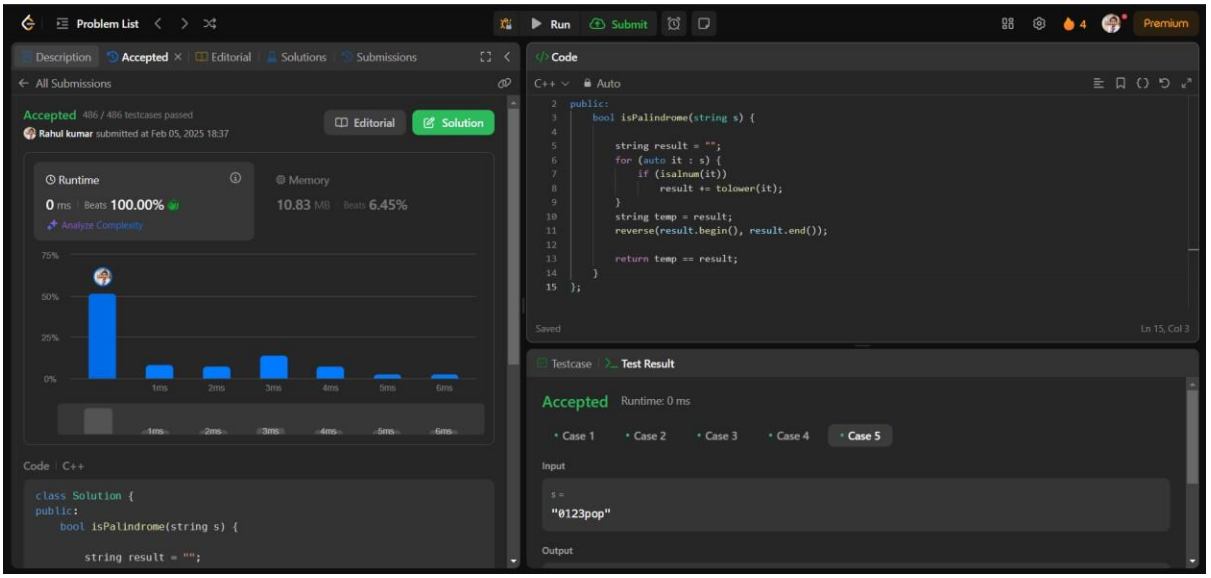
2. Jump Game

```
class Solution {
public:
    bool canJump(vector<int>& nums) {

        int n=nums.size();
        int maxJump=0;
        for(auto i=0;i<n;++i){
            if(i>maxJump)return false;
            maxJump=max(maxJump,i+nums[i]);
        }
        return true;
    }
};
```



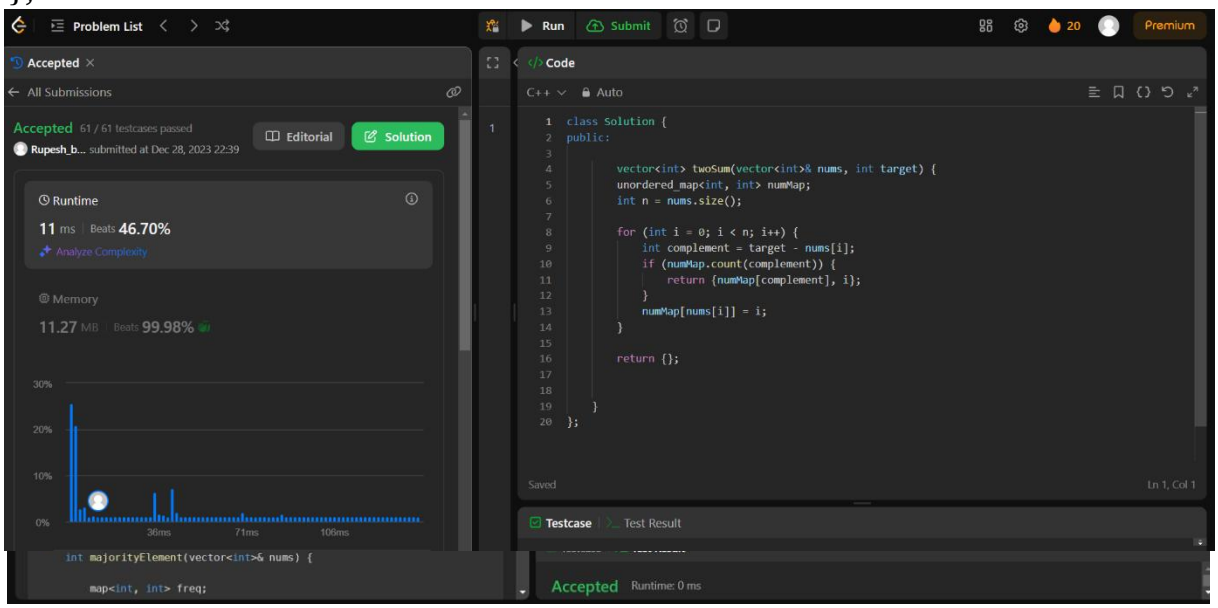
4. Valid Palindrome

```
class Solution {
public:
    bool isPalindrome(string s) {

        string result = "";
        for (auto it : s) {
            if (isalnum(it))
                result += tolower(it);
        }
        string temp = result;
        reverse(result.begin(), result.end());

        return temp == result;
    }
};
```



6. Majority Element

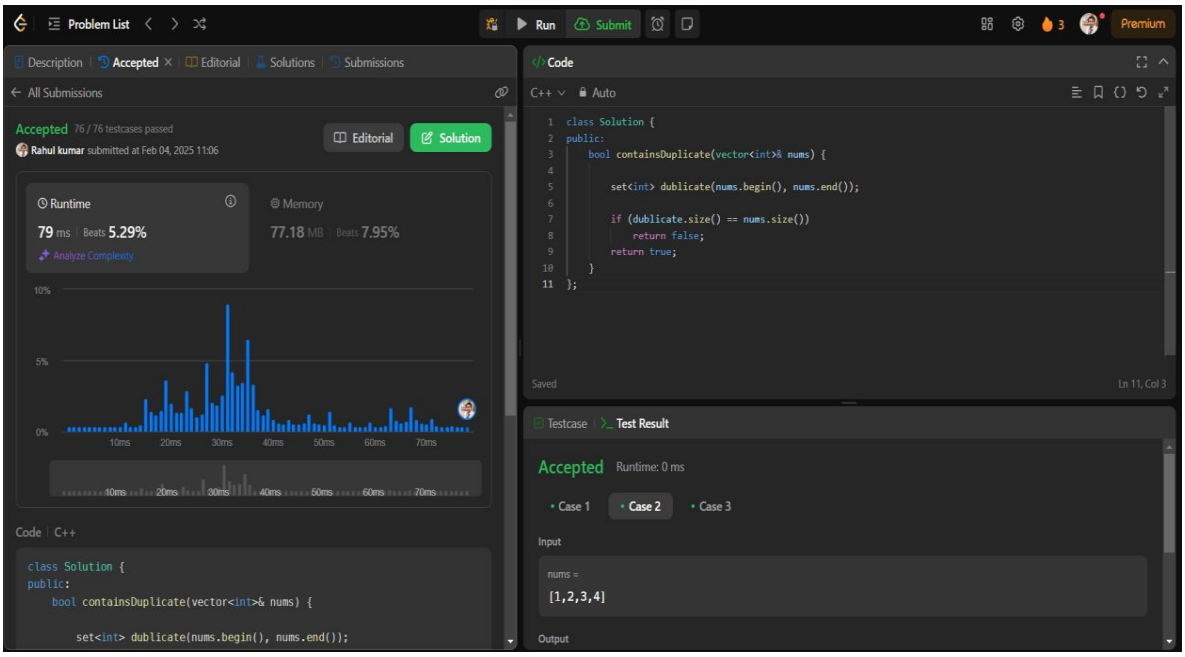
```
class Solution {
public:
    int majorityElement(vector<int>& nums) {
        map<int, int> freq;
        int n = nums.size();
        int majority = 0;

        for (auto it : nums) {
            freq[it]++;
        }
        for (auto it : freq) {
            if (it.second > n / 2)
                majority = it.first;
        }
        return majority;
    }
};
```



7. Contains Duplicate

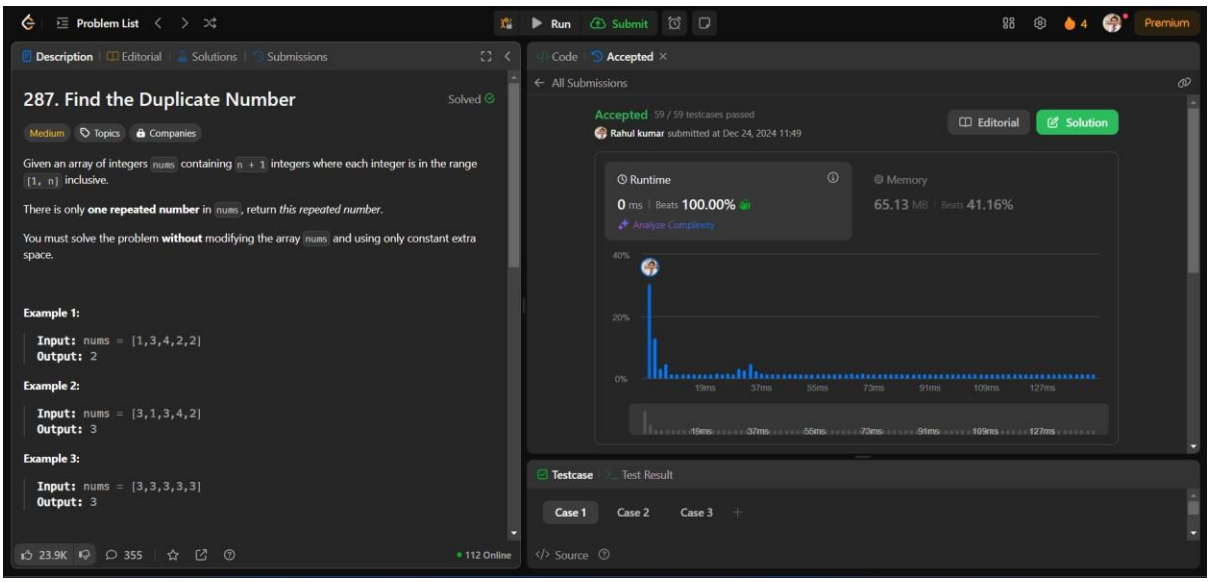
// code

```
class Solution {
public:
    bool containsDuplicate(vector<int>& nums) {
        set<int> dublicate(nums.begin(), nums.end());


        if (dublicate.size() == nums.size())
            return false;
        return true;
    }
};
```



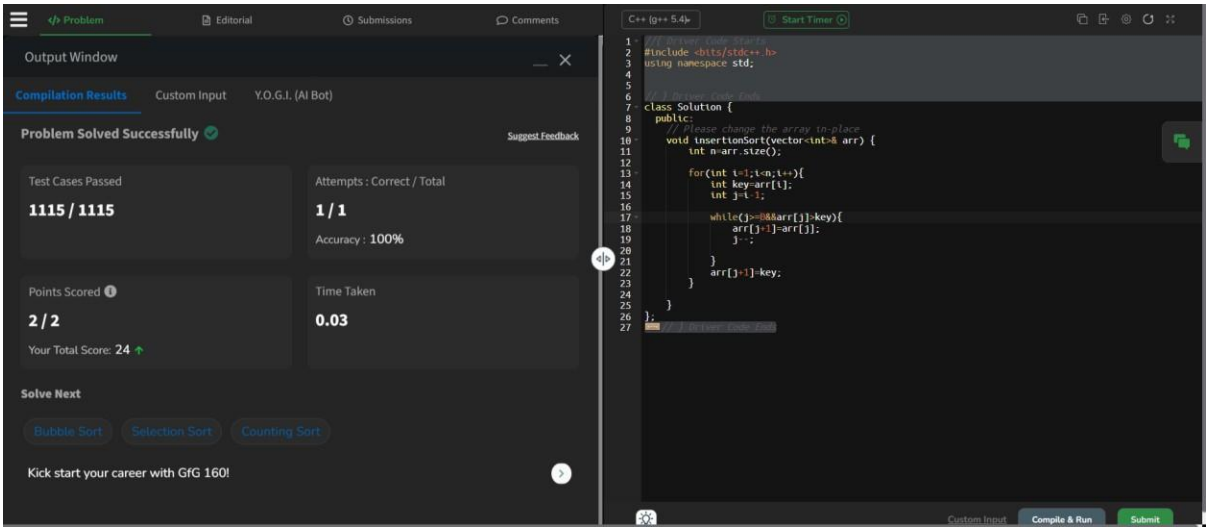
8. Find the Duplicate Number

```
class Solution {
public:
    int findDuplicate(vector<int>& nums) {
        int slow = nums[0];
        int fast = nums[0];
        do {
            slow = nums[slow];
            fast = nums[nums[fast]];
        } while (slow != fast);
        slow = nums[0];
        while (slow != fast) {
            slow = nums[slow];
            fast = nums[fast];
        }
        return slow;
    }
};
```



9. Insertion Sort

```
// code
class Solution {
  public:
    void insertionSort(vector<int>& arr) {
        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--;
            }
            arr[j+1]=key;
        }
    }
};
```



10. Two Sum

```
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        vector<pair<int, int>> indexedNums;
        for (int i = 0; i < nums.size(); i++) {
            indexedNums.push_back({nums[i], i});
        }
        sort(indexedNums.begin(), indexedNums.end());
        int left = 0, right = nums.size() - 1;
        while (left < right) {
            int sum = indexedNums[left].first + indexedNums[right].first;
            if (sum == target) {
                return {indexedNums[left].second, indexedNums[right].second};
            } else if (sum < target) {
                left++;
            } else {
                right--;
            }
        }
        return {};
    }
};
```

All Submissions

Accepted 313 / 313 testcases passed

Rupesh_bansal submitted at Dec 18, 2024 12:55

Editorial | Solution

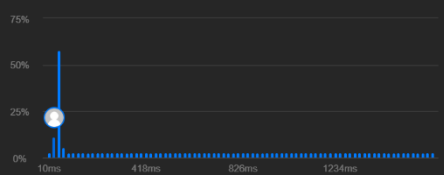
Runtime

45 ms | Beats 77.52%

Analyze Complexity

Memory

28.94 MB | Beats 85.30%



C++ | Auto

```cpp
class Solution {
public:
    vector<vector<int>> threeSum(vector<int>& nums) {

        int n = nums.size();
        sort(nums.begin(),nums.end());
        vector<vector<int>> finalArr;
        for(int i =0;i<n-2;i++){
            if(i>0 && nums[i]==nums[i-1])
            continue;
           int j = i+1;
           int c = n-1;
        //     int sum  = 0;
           while(j<c){
                                int sum = nums[i] + nums[j] + nums[c];

        //     sum+=nums[i];
        //     sum+=nums[j];
        //     sum+=nums[c];
           if(sum==0){
            finalArr.push_back({nums[i],nums[j],nums[c]});
             j++;
```

Saved

Ln 1, Col 1

Testcase | Test Result