

Name: Krishna Sharma

UID: 22BCS11885

Section: FL-IOT-641-A

389. Find the Difference

```
class Solution {
public:
    char findTheDifference(string s, string t) {
        map<char,int> mp,mp1;
        for(auto i:s) mp[i]++;
        for(auto j:t) mp1[j]++;
        char ans;
        for(auto i:mp1){
            if(mp[i.first]!=mp1[i.first]){
                ans=i.first;
            }
        }
        return ans;
    }
};
```

The screenshot displays a coding platform interface with a dark theme. On the left, the 'All Submissions' tab is active, showing a submission status of 'Accepted' with 54/54 testcases passed. The submission was made by 'K...' on Aug 31, 2024 at 05:44. Performance metrics are shown: Runtime is 8 ms (Beats 2.38%) and Memory is 9.79 MB (Beats 8.70%). A bar chart at the bottom shows the distribution of runtime performance across different percentiles. On the right, the 'Code' editor shows the C++ solution for the problem. The code defines a class 'Solution' with a public method 'findTheDifference' that uses two maps to count character frequencies in strings 's' and 't', and returns the first character that differs. The bottom section shows a 'Testcase' tab with 'Case 1' selected, and a 'Test Result' area with input 's ='.

Accepted 54 / 54 testcases passed
K... submitted at Aug 31, 2024 05:44

Runtime
8 ms | Beats 2.38%
[Analyze Complexity](#)

Memory
9.79 MB | Beats 8.70%

100%
75%
50%
25%
0%

1ms 2ms 3ms 4ms

Code
C++ Auto

```
1 class Solution {
2 public:
3     char findTheDifference(string s, string t) {
4         map<char,int> mp,mp1;
5         for(auto i:s) mp[i]++;
6         for(auto j:t) mp1[j]++;
7         char ans;
8         for(auto i:mp1){
9             if(mp[i.first]!=mp1[i.first]){
10                 ans=i.first;
11             }
12         }
13         return ans;
14     }
15 };
```

Saved

Testcase Test Result

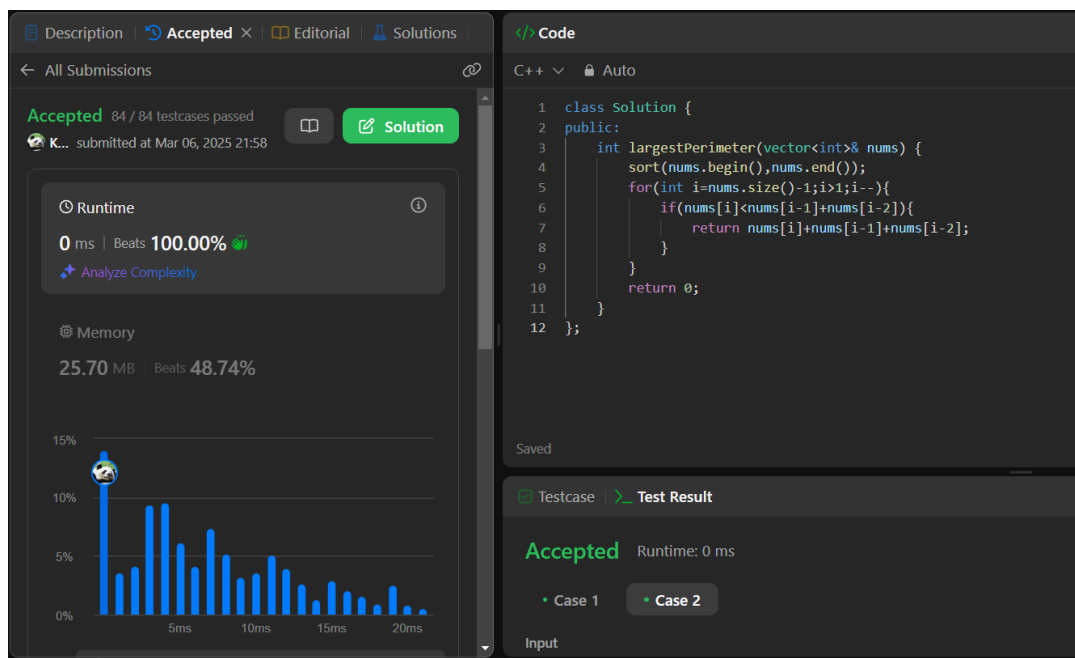
Case 1 Case 2 +

s =

</> Source ?

976. Largest Perimeter Triangle

```
class Solution {
public:
    int largestPerimeter(vector<int>& nums) {
        sort(nums.begin(),nums.end());
        for(int i=nums.size()-1;i>1;i--){
            if(nums[i]<nums[i-1]+nums[i-2]){
                return nums[i]+nums[i-1]+nums[i-2];
            }
        }
        return 0;
    }
};
```



414. Third Maximum Number

```
class Solution {
public:
    int thirdMax(vector<int>& nums) {
        set<int>st(nums.begin(),nums.end());
        if(st.size() < 3) {
            return *st.rbegin();
        }
        auto i = st.rbegin();
        advance(i,2);
        return *i;
    }
};
```

};

Accepted 34 / 34 testcases passed

Runtime: 8 ms | Beats 5.80%

Memory: 11.04 MB | Beats 100.00%

```
1 class Solution {
2 public:
3     int thirdMax(vector<int>& nums) {
4         set<int> st(nums.begin(), nums.end());
5         if(st.size() < 3) {
6             return *st.rbegin();
7         }
8         auto i = st.rbegin();
9         advance(i, 2);
10        return *i;
11    }
12};
```

451. Sort Characters By Frequency

```
class Solution {
public:
    string frequencySort(string s) {
        int n = s.size();
        unordered_map<char, int> mp;
        for(auto i: s) {
            mp[i]++;
        }
        sort(s.begin(), s.end(), [&](char a, char b) {
            if(mp[a] == mp[b]) return a < b;
            return mp[a] > mp[b];
        });
        return s;
    }
};
```

Accepted 33 / 33 testcases passed

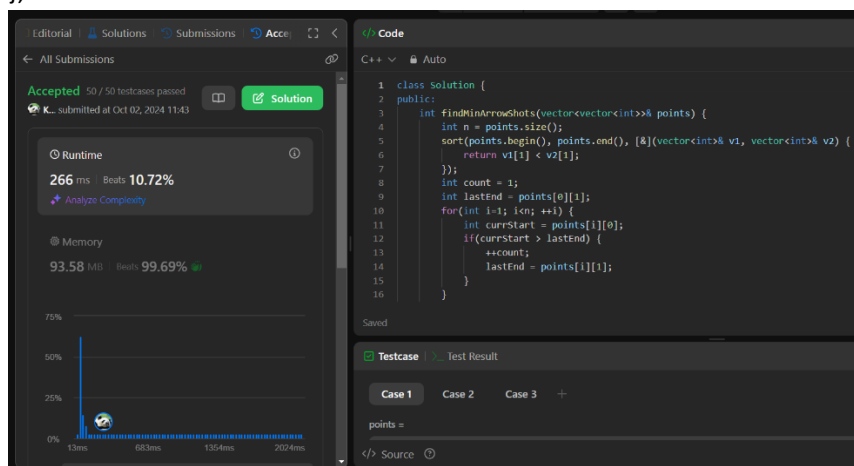
Runtime: 148 ms | Beats 5.00%

Memory: 10.58 MB | Beats 91.81%

```
1 class Solution {
2 public:
3     string frequencySort(string s) {
4         int n = s.size();
5         unordered_map<char, int> mp;
6         for(auto i: s) {
7             mp[i]++;
8         }
9         sort(s.begin(), s.end(), [&](char a, char b) {
10             if(mp[a] == mp[b]) return a < b;
11             return mp[a] > mp[b];
12         });
13         return s;
14     }
15};
```

452. Minimum Number of Arrows to Burst Balloons

```
class Solution {
public:
    int findMinArrowShots(vector<vector<int>>& points) {
        int n = points.size();
        sort(points.begin(), points.end(), [&](vector<int>& v1, vector<int>& v2) {
            return v1[1] < v2[1];
        });
        int count = 1;
        int lastEnd = points[0][1];
        for(int i=1; i<n; ++i) {
            int currStart = points[i][0];
            if(currStart > lastEnd) {
                ++count;
                lastEnd = points[i][1];
            }
        }
        return count;
    }
};
```



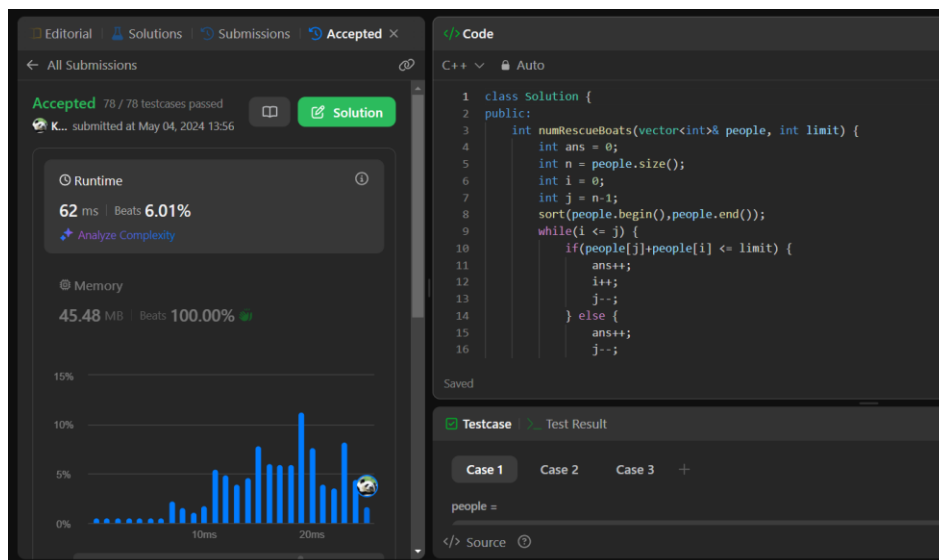
881. Boats to Save People

```
class Solution {
public:
    int numRescueBoats(vector<int>& people, int limit) {
        int ans = 0;
        int n = people.size();
        int i = 0;
        int j = n-1;
        sort(people.begin(), people.end());
        while(i <= j) {
            if(people[j]+people[i] <= limit) {
                ans++;
            }
        }
    }
};
```

```

        i++;
        j--;
    } else {
        ans++;
        j--;
    }
}
return ans;
}
};

```

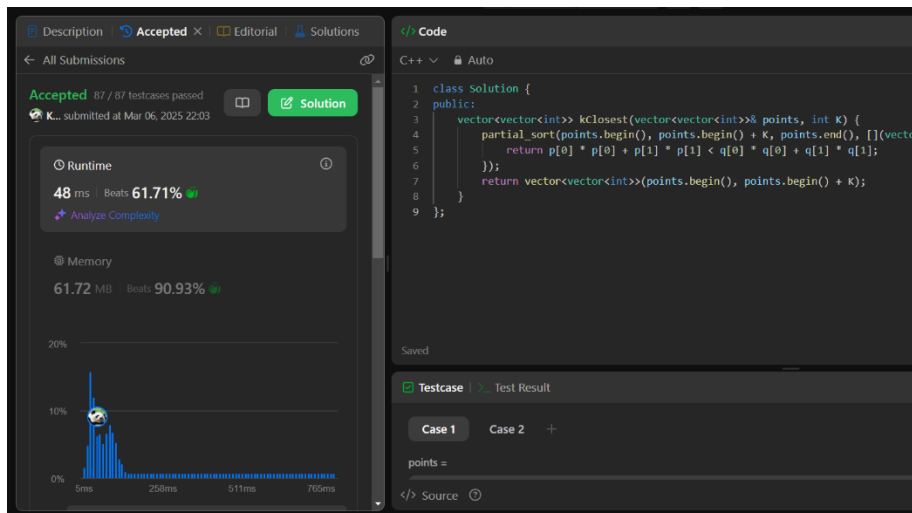


973. K Closest Points to Origin

```

class Solution {
public:
    vector<vector<int>> kClosest(vector<vector<int>>& points, int K) {
        partial_sort(points.begin(), points.begin() + K, points.end(), [](vector<int>& p, vector<int>& q) {
            return p[0] * p[0] + p[1] * p[1] < q[0] * q[0] + q[1] * q[1];
        });
        return vector<vector<int>>(points.begin(), points.begin() + K);
    }
};

```



1338. Reduce Array Size to The Half

```
class Solution {
public:
    int minSetSize(vector<int>& arr) {
        int n = arr.size();
        unordered_map<int, int> cnt;
        for (int x : arr) ++cnt[x];
        vector<int> counting(n + 1);
        for (auto [it, freq] : cnt) ++counting[freq];
        int ans = 0, removed = 0, half = n / 2, freq = n;
        while (removed < half) {
            ans += 1;
            while (counting[freq] == 0) --freq;
            removed += freq;
            --counting[freq];
        }
        return ans;
    }
};
```

escription

Editorial

Solutions

Accepted

All Submissions

Accepted

33 / 33 testcases passed

K... submitted at Mar 06, 2025 22:04

Solution

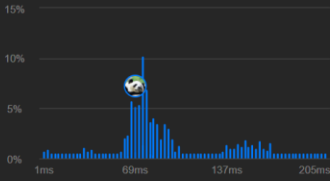
Runtime

71 ms | Beats 79.26%

Analyze Complexity

Memory

81.90 MB | Beats 88.52%



Code

C++

Auto

```
1 class Solution {
2 public:
3     int minSetSize(vector<int>& arr) {
4         int n = arr.size();
5         unordered_map<int, int> cnt;
6         for (int x : arr) ++cnt[x];
7         vector<int> counting(n + 1);
8         for (auto [it, freq] : cnt) ++counting[freq];
9         int ans = 0, removed = 0, half = n / 2, freq = n;
10        while (removed < half) {
11            ans += 1;
12            while (counting[freq] == 0) --freq;
13            removed += freq;
14            --counting[freq];
15        }
16        return ans;
17    }
```

Saved

Testcase

Test Result

Case 1

Case 2

+

err =

Source