Priyanshu
22BCS16931
BCS_FL_IOT-603 (A)

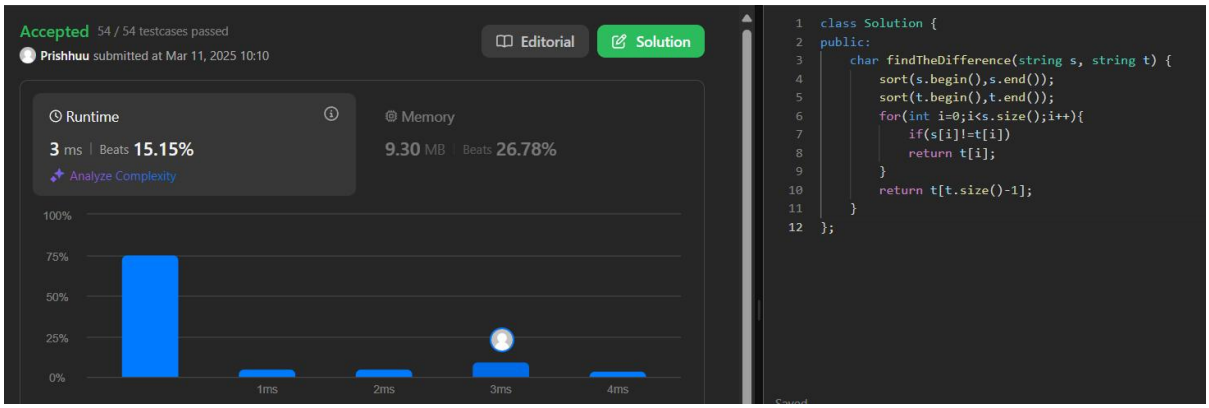
Assignment-5

1. 389. Find the Difference

- **Solution:**

```
class Solution {
public:
    char findTheDifference(string s, string t) {
        sort(s.begin(),s.end());
        sort(t.begin(),t.end());
        for(int i=0;i<s.size();i++){
            if(s[i]!=t[i])
            return t[i];
        }
        return t[t.size()-1];
    }
};
```

- **Screenshot:**



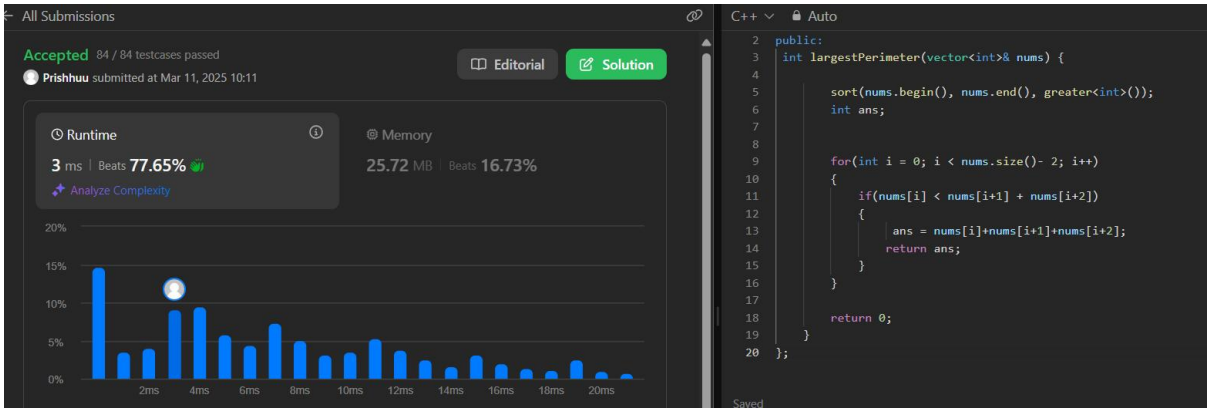
2. 976. Largest Perimeter Triangle

- **Solution:**

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

```
                ans = nums[i]+nums[i+1]+nums[i+2];
                return ans;
            }
        }
        return 0;
    }
};
```

- **Screenshot:**

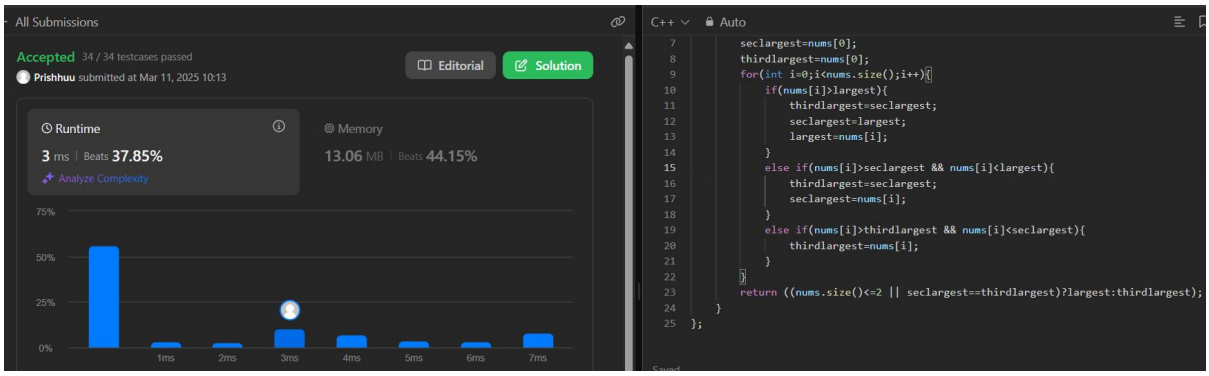


3. 414. Third Maximum Number

- **Solution:**

```
class Solution {
public:
    int thirdMax(vector<int>& nums) {
        sort(nums.begin(),nums.end());
        int largest,seclargest,thirdlargest;
        largest= nums[0];
        seclargest=nums[0];
        thirdlargest=nums[0];

        for(int i=0;i<nums.size();i++){
            if(nums[i]>largest){
                thirdlargest=seclargest;
                seclargest=largest;
                largest=nums[i];
            }
            else if(nums[i]>seclargest && nums[i]<largest)
            {
                thirdlargest=seclargest;
                seclargest=nums[i];
            }
            else if(nums[i]>thirdlargest && nums[i]<seclargest)
            {
                thirdlargest=nums[i];
            }
        }
        return ((nums.size()<=2 || seclargest==thirdlargest)?largest:thirdlargest);
    }
};
```

- **Screenshot:**

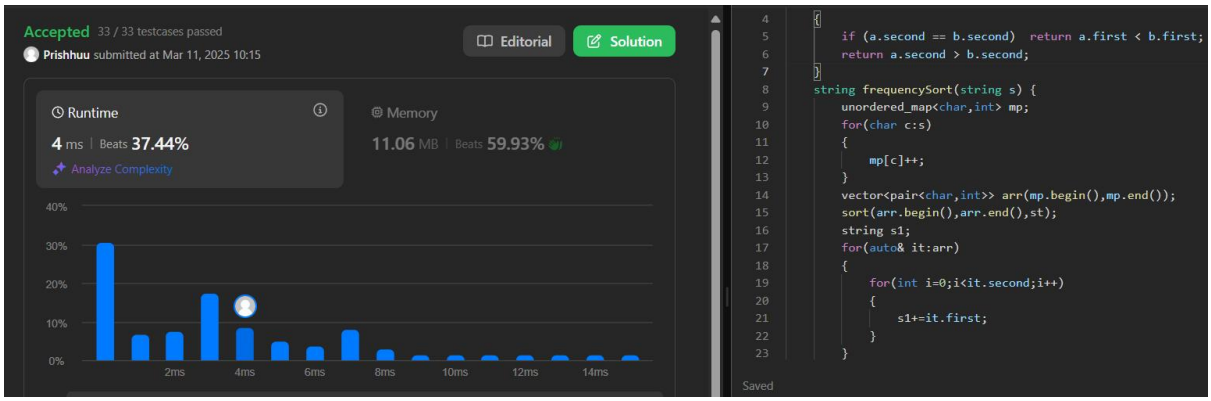


4. 451. Sort Characters By Frequency

- **Solution:**

```
class Solution {

public:
    static bool st(pair<char,int>& a,pair<char,int>& b)
    {
        if (a.second == b.second)  return a.first < b.first;
        return a.second > b.second;
    }

    string frequencySort(string s)
    {
        unordered_map<char,int> mp;

        for(char c:s)
        {
            mp[c]++;
        }

        vector<pair<char,int>> arr(mp.begin(),mp.end());

        sort(arr.begin(),arr.end(),st);

        string s1;
        for(auto& it:arr)
        {
            for(int i=0;i<it.second;i++)
            {
                s1+=it.first;
            }
        }

        return s1;
    }
};
```

- **Screenshot:**

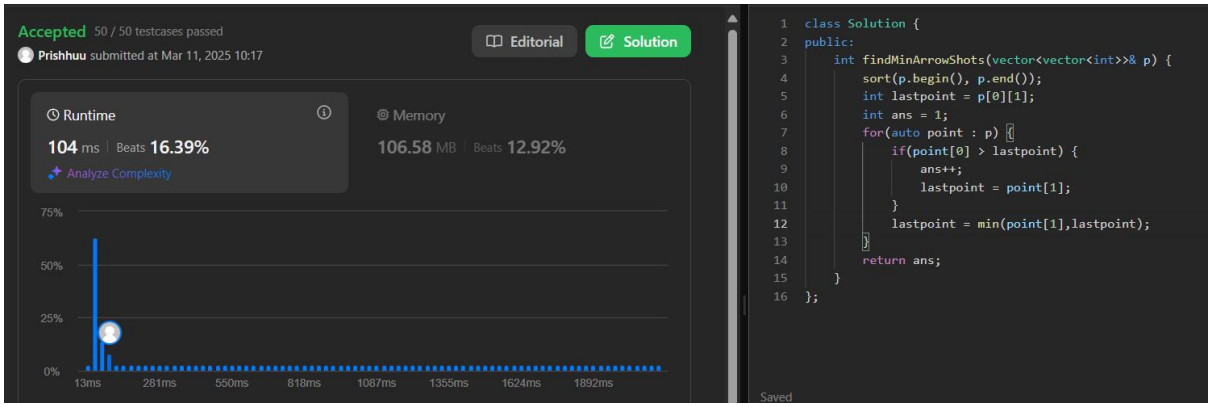


5. *452. Minimum Number of Arrows to Burst Balloons*

- **Solution:**

```
class Solution {
public:
    int findMinArrowShots(vector<vector<int>>& p) {
        sort(p.begin(), p.end());
        int lastpoint = p[0][1];
        int ans = 1;
        for(auto point : p) {
            if(point[0] > lastpoint) {
                ans++;
                lastpoint = point[1];
            }
            lastpoint = min(point[1],lastpoint);
        }
        return ans;
    }
};
```

- **Screenshot:**

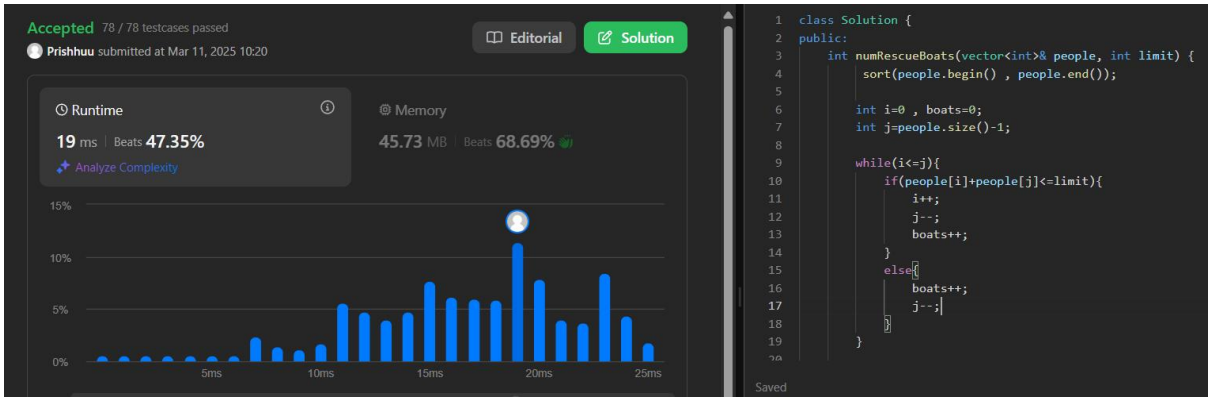


6. 881. Boats to Save People

- **Solution:**

```cpp
class Solution {
public:
    int numRescueBoats(vector<int>& people, int limit)
    {
         sort(people.begin() , people.end());

        int i=0 , boats=0;
        int j=people.size()-1;
        while(i<=j){
            if(people[i]+people[j]<=limit)
            {
                i++;
                j--;
                boats++;
            }
            else
            {
                boats++;
                j--;
            }
        }
        return boats;
    }
};
```

- **Screenshot:**



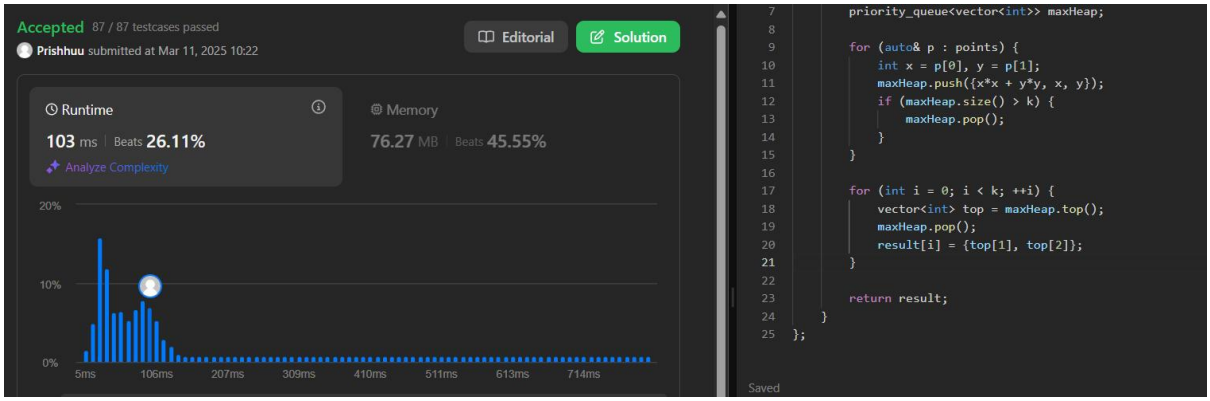
7. 973. K Closest Points to Origin

- **Solution:**

```cpp
class Solution {
public:
    vector<vector<int>> kClosest(vector<vector<int>>& points, int k) {
        vector<vector<int>> result(k);
        priority_queue<vector<int>> maxHeap;
        for (auto& p : points) {
```

```
            int x = p[0], y = p[1];
            maxHeap.push({x*x + y*y, x, y});
            if (maxHeap.size() > k) {
                maxHeap.pop();
            }
        }
        for (int i = 0; i < k; ++i) {
            vector<int> top = maxHeap.top();
            maxHeap.pop();
            result[i] = {top[1], top[2]};
        }
        return result;
    }
};
```

- **Screenshot:**

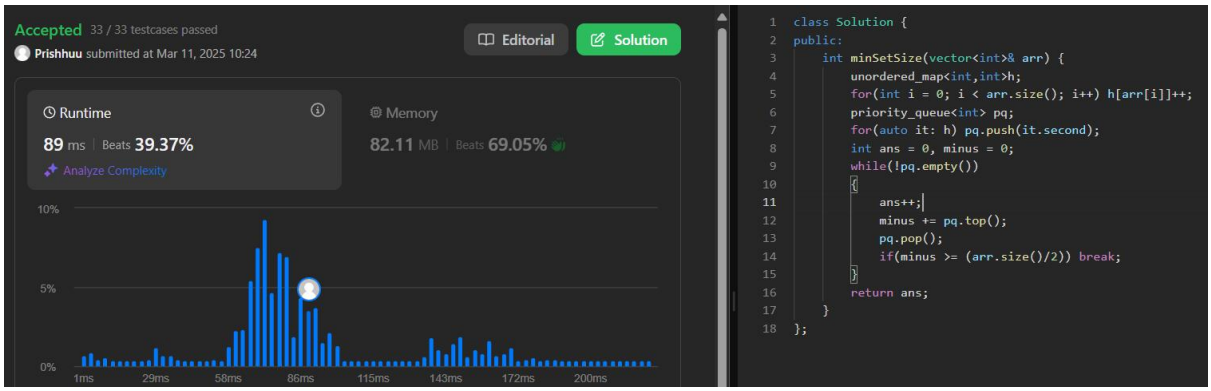


8. *1338. Reduce Array Size to The Half*

- **Solution:**

```
class Solution {
public:
    int minSetSize(vector<int>& arr)
    {
        unordered_map<int,int>h;
        for(int i = 0; i < arr.size(); i++) h[arr[i]]++;
        priority_queue<int> pq;
        for(auto it: h) pq.push(it.second);
        int ans = 0, minus = 0;
        while(!pq.empty())
        {
            ans++;
            minus += pq.top();
            pq.pop();
            if(minus >= (arr.size()/2)) break;
        }
        return ans;
    }
};
```

- **Screenshot:**


Accepted 33 / 33 testcases passed
Prishhuu submitted at Mar 11, 2025 10:24
Editorial
Solution
Runtime
89 ms | Beats 39.37%
Analyze Complexity
Memory
82.11 MB | Beats 69.05%
10%
5%
0%
1ms
29ms
58ms
86ms
115ms
143ms
172ms
200ms
class Solution {
public:
int minSetSize(vector<int>& arr) {
unordered_map<int,int>h;
for(int i = 0; i < arr.size(); i++) h[arr[i]]++;
priority_queue<int> pq;
for(auto it: h) pq.push(it.second);
int ans = 0, minus = 0;
while(!pq.empty())
{
ans++;
minus += pq.top();
pq.pop();
if(minus >= (arr.size()/2)) break;
}
return ans;
}
};