DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Assignment - 5

Student Name: Anirudh Gautam **UID: 22BCS12994**

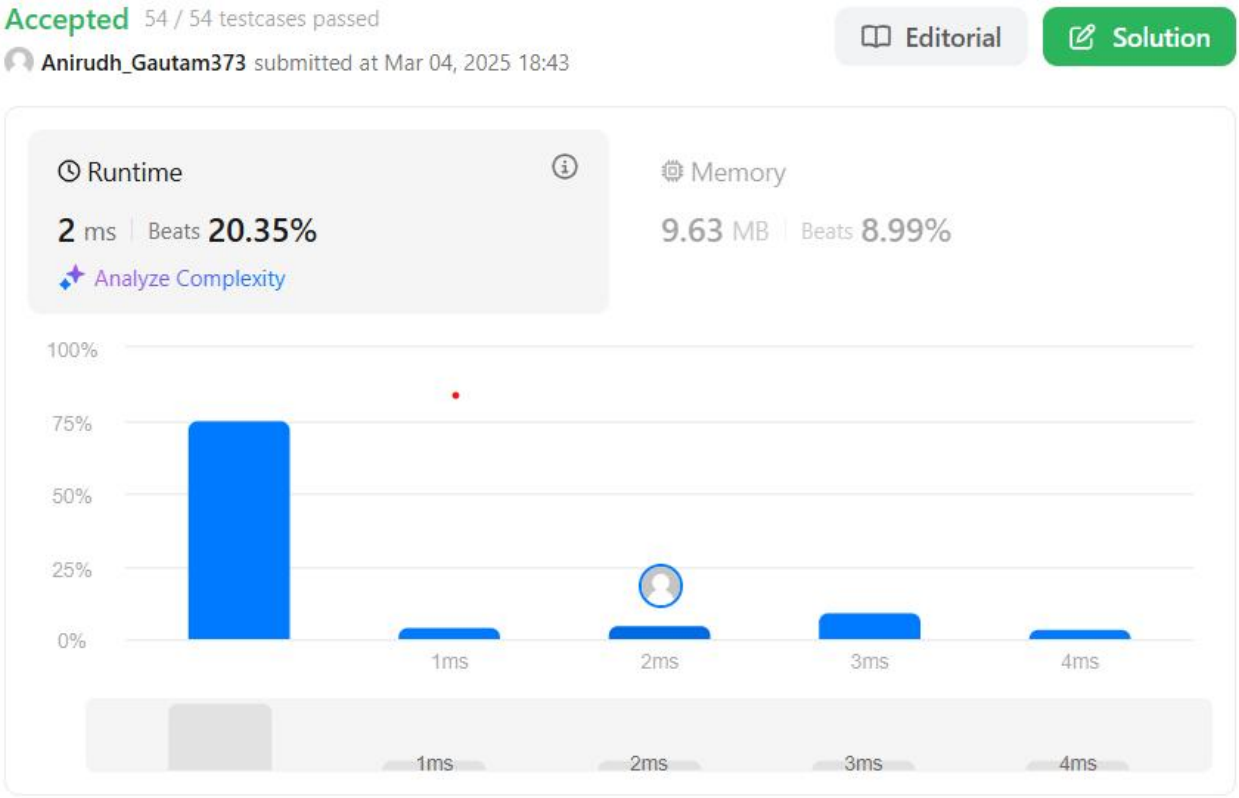
Branch: CSE **Section/Group:602-A**

Semester: 6 **Date of Performance:04-03-2025**

Subject Name: Advanced Programming Lab **Subject Code: 22CSH-359**

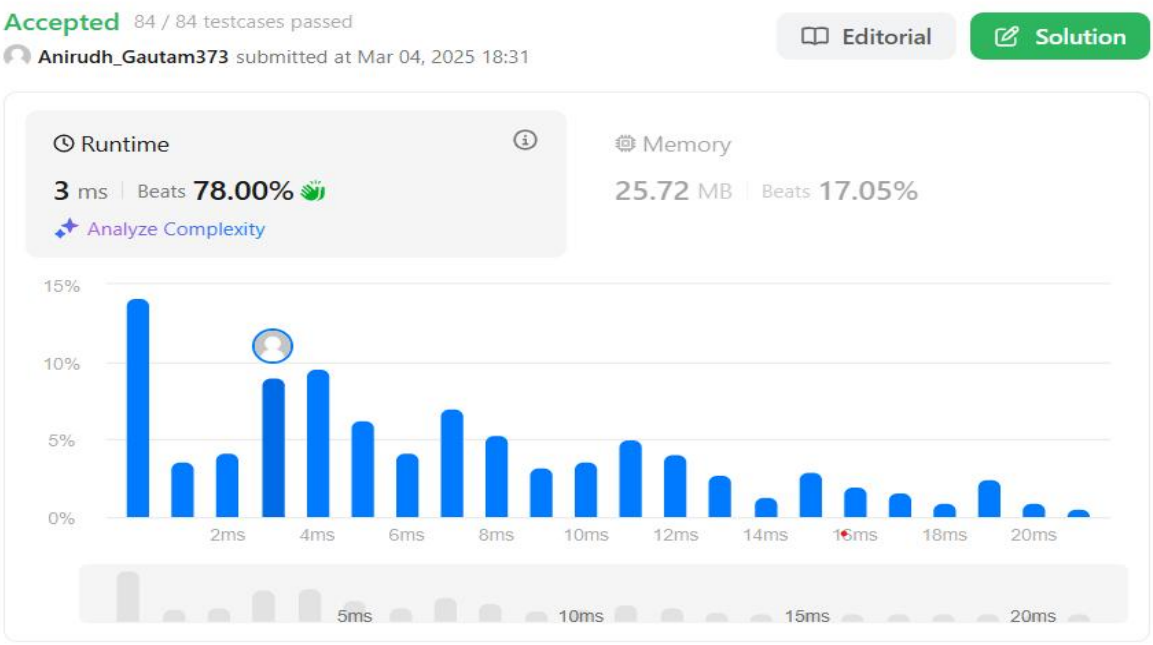
1.Find the difference

```
class Solution {
public:
    char findTheDifference(string s, string t) {
        std::unordered_map<char, int> count;

        for (char c : t) {
            count[c]++;
        }

        for (char c : s) {
            count[c]--;
            if (count[c] == 0) {
                count.erase(c);
            }
        }

        return count.begin()->first;

    }
};
```



2. Largest Perimeter Triangle

```
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;
   }
};
```

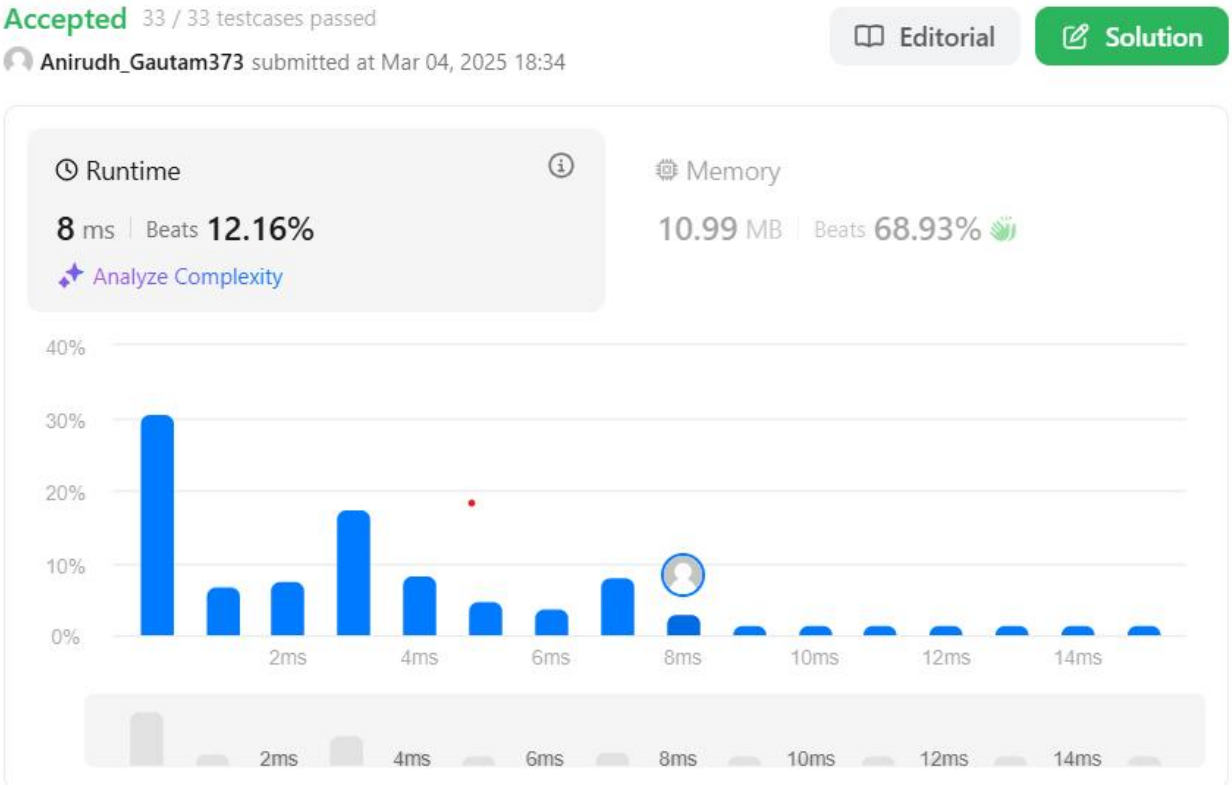


3.Third maximum Number

```
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);

    }
};
```

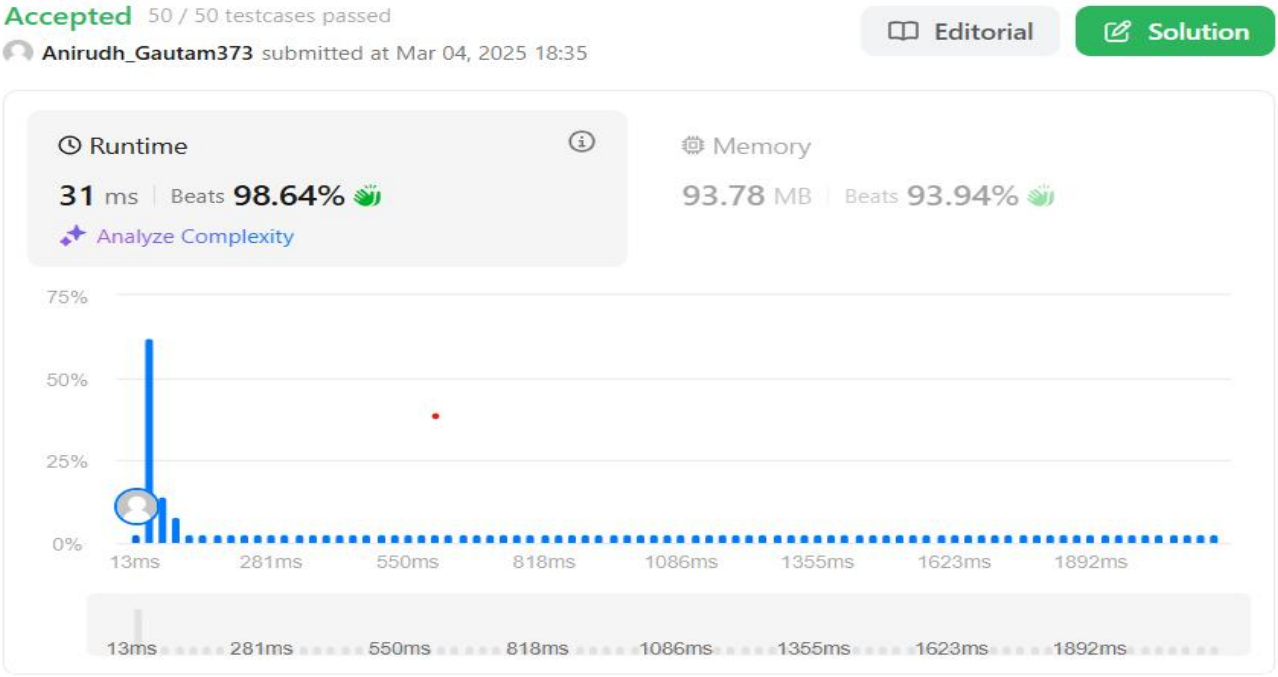
4.Sort Character by Frequency

```
class Solution {
public:
    string frequencySort(string s) {
        priority_queue<pair<int, char>, vector<pair<int, char>>> pq;

        unordered_map<char,int> m;
        for(int i = 0;i<s.length();i++){
            m[s[i]]++;
        }
        for(auto elem:m){
            int first = elem.first;
            int second = elem.second;
            pq.push({second,first});
        }
        string ans = "";
        while(!pq.empty()){
            pair<int,int> p = pq.top();
            for(int i = 0;i<p.first;i++){
                ans+=p.second;
            }
            pq.pop();
        }
        return ans;
    }
};
```



5.Minimum Number of Arrows to Burst Baloons

```
class Solution {
public:
    int findMinArrowShots(vector<vector<int>>& points) {
        sort(points.begin(), points.end(), [](const auto& a, const auto& b) {
            return a[0] < b[0];
        });

        int arrows = 1;
        int end = points[0][1];

        for (size_t i = 1; i < points.size(); ++i) {
            if (points[i][0] > end) {
                arrows++;
                end = points[i][1];
            } else {
                end = min(end, points[i][1]);
            }
        }

        return arrows;
    }
};
```

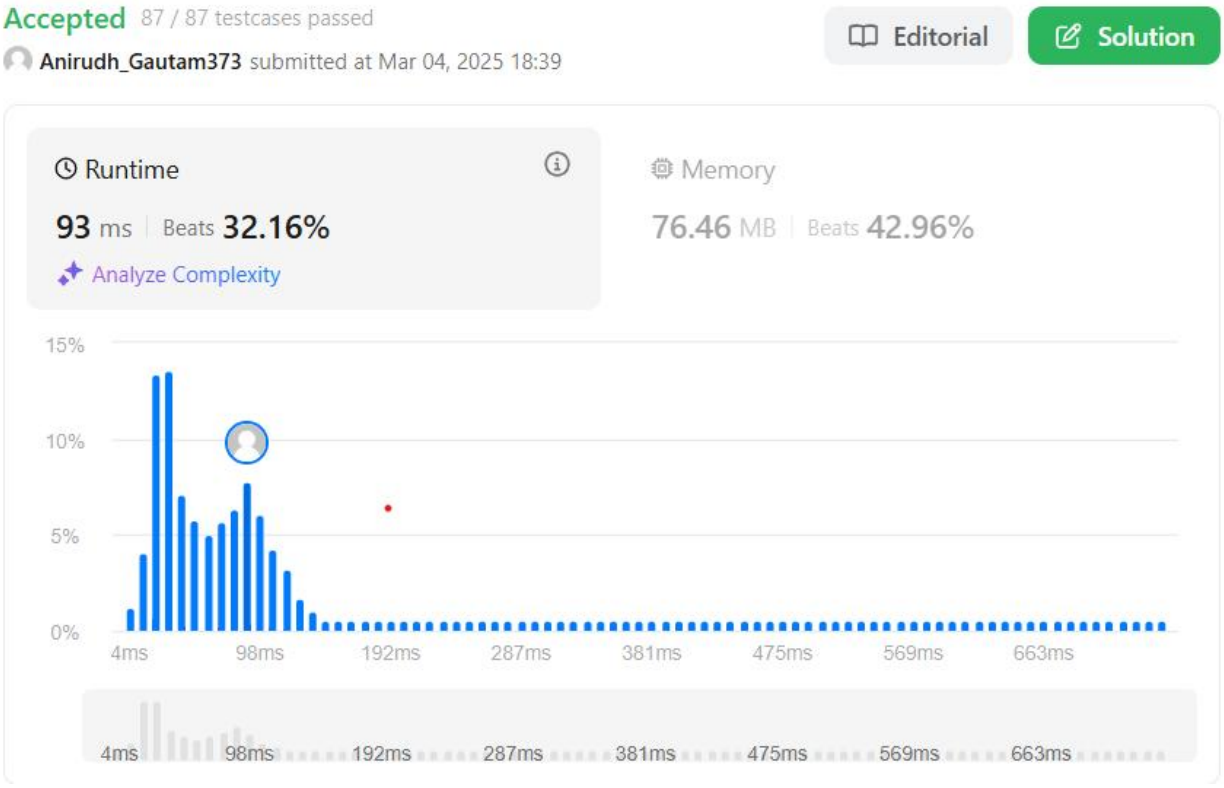


6. Boats to Save People

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

        int i = 0, j = people.size() - 1,cnt = 0;

        while(i <= j)
        {

            if(people[i] + people[j] <= limit)
            {
                ++i;
                --j;
            }

            else
                --j;

            ++cnt;
        }

        return cnt;
    }
};
```

7. K closest point To Origin

```
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;
    }
};
```



8. Reduce Array Size to Half

```
class Solution {
public:
    int minSetSize(vector<int>& arr) {
        map<int,int>mp;
        for(auto val:arr) mp[val]++;

        priority_queue<int>pq;
        for(auto [val, cnt]:mp)
            pq.push(cnt);

        int ans = 0, need = arr.size()/2;
        while(need > 0)
        {
          ans++;
          need -= pq.top(); pq.pop();
        }
        return ans;
    }
};
```

Accepted 33 / 33 testcases passed

Anirudh_Gautam373 submitted at Mar 04, 2025 18:40

Editorial | Solution

Runtime

151 ms | Beats **20.45%**

Analyze Complexity

Memory

83.94 MB | Beats 39.03%