Assignment – 5

Name: Raj Singh **UID: 22BCS10624**

Section/ Group: FL-IOT-604/A **Subject: Advance Programming**

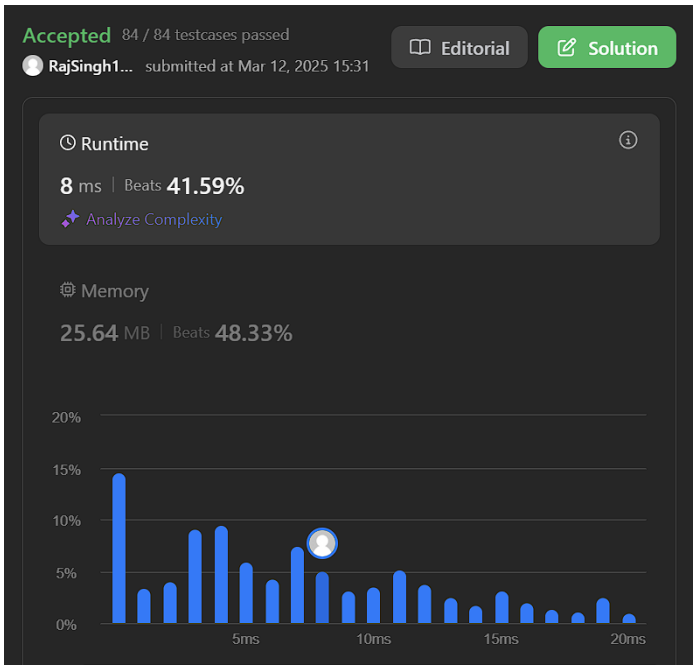
1. **Find the Difference**

```
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];

   }

};
```

2. **Largest Perimeter Triangle**

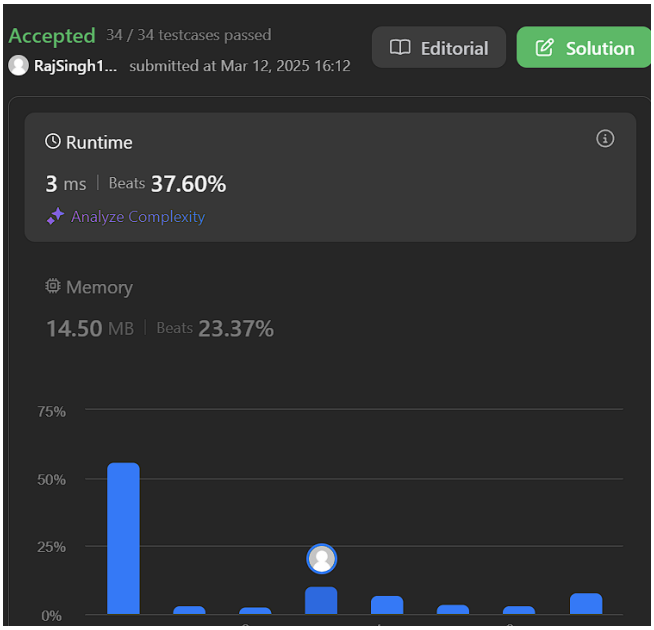
```
class Solution {
public:
   int largestPerimeter(vector<int>& nums) {
      int para=0;
      int mxpara = INT_MIN;
      sort(nums.begin(), nums.end());
      for(int i=2 ;i<nums.size(); i++){
```

```
            int a = nums[i];
            int b = nums[i-1];
            int c = nums[i-2];
            if(a+b >c && a+c > b && b+c >a){
                para = a+b+c;
            }
            mxpara = (mxpara,para);
        }
        (para>0)?  mxpara = mxpara : mxpara = 0;
        return mxpara;
    }
};
```



3. Third Maximum Number

```
class Solution {
public:
    int thirdMax(vector<int>& nums) {
        set<int> st;
        for(int i=0 ;i<nums.size(); i++){
            st.insert(nums[i]);
        }
        if(st.size() < 3) return *st.rbegin();
        else{
            st.erase(--st.end());
            st.erase(--st.end());
        }
        int it ;
        if(!st.empty()) it = *st.rbegin();
        return it;
    }
};
```

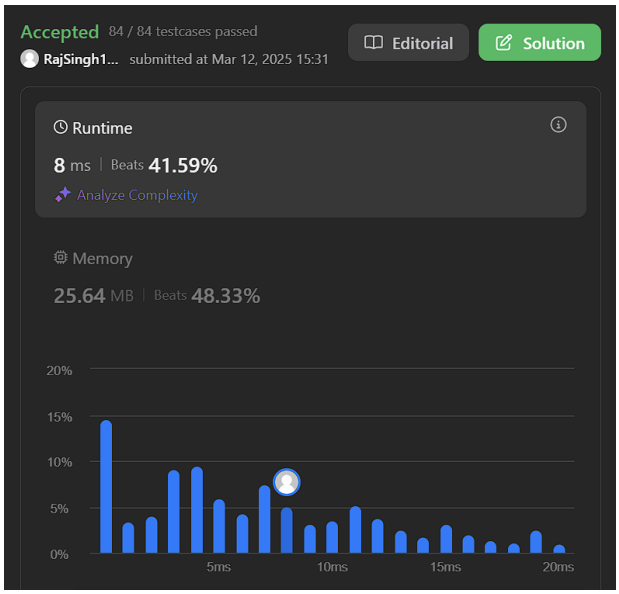


4. [Sort Characters By Frequency]

```
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;
    }
};
```

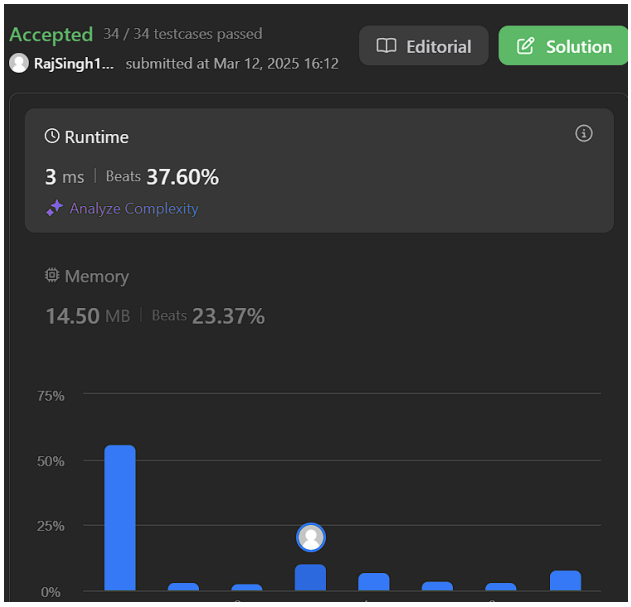
5. [Minimum Number of Arrows to Burst Balloons]

```
bool cmp(vector<int>& a, vector<int>& b) {return a[1] < b[1];}
class Solution {
public:
    int findMinArrowShots(vector<vector<int>>& segments) {
        sort(segments.begin(), segments.end(), cmp);
        int ans = 0, arrow = 0;
        for (int i = 0; i < segments.size(); i ++) {
            if (ans == 0 || segments[i][0] > arrow) {
                ans ++;
                arrow = segments[i][1];
            }
        }
        return ans;
    }
};
```



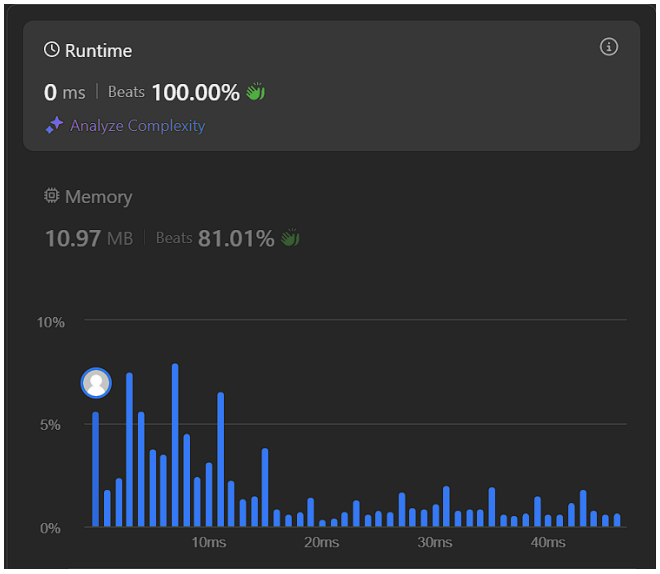
6. **Boats to Save People**

```
int numRescueBoats(vector<int> people, int limit) {
    int i, j;
    sort(people.rbegin(), people.rend());
    for (i = 0, j = people.size() - 1; i <= j; ++i)
        if (people[i] + people[j] <= limit) j--;
   return i;
}
```



7. **K Closest Points to Origin**

```
vector<vector<int>> kClosest(vector<vector<int>>& A, int k) {
     sort(A.begin(), A.end(), [](vector<int>& a, vector<int>& b) {
     return a[0] * a[0] + a[1] * a[1] < b[0] * b[0] + b[1] * b[1];
});
     return vector<vector<int>>(A.begin(), A.begin() + k);
}
```



8. 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 [_, 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;
    }
};
```