**ASSIGNMENT – 5**

**Name – Ritvik Jindal**

**UID – 22BCS10235**

**CLASS – FL\_IOT\_604-A**

**389.**[**Find the diffrence**](https://leetcode.com/problems/find-the-difference/description/)

class Solution {

public:

char findTheDifference(string s, string t) {

int s1=0,s2=0;

for(auto i: s){

s1+= (int)i;

}

for(auto i: t){

s2+= (int)i;

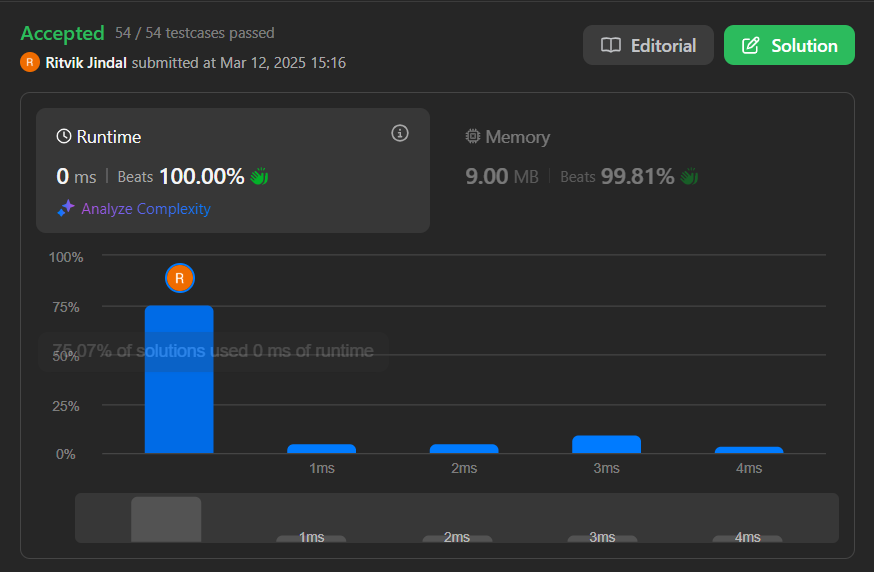
}

char res = abs(s1-s2);

return res;

}

};



**976.**[**Largest Perimeter Triangle**](https://leetcode.com/problems/largest-perimeter-triangle/description/)

class Solution {

public:

int largestPerimeter(vector<int>& nums) {

sort(nums.begin(),nums.end());

int mxpara = 0;

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 && b+c>a && c+a>b){

mxpara = max(mxpara,a+b+c);

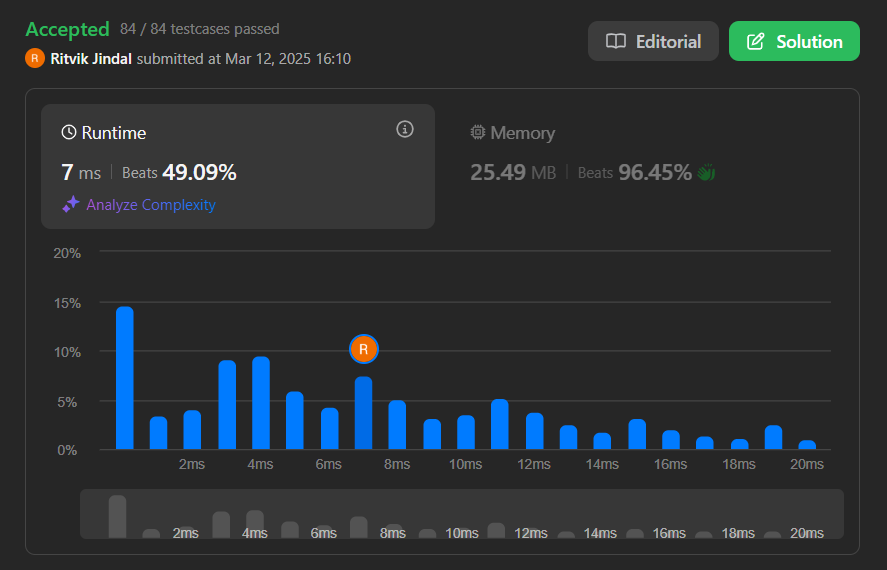
}

}

return mxpara;

}

};



**414.**[**Third Maximum Number**](https://leetcode.com/problems/third-maximum-number/description/)

class Solution {

public:

int thirdMax(vector<int>& nums) {

set<int> s;

for(auto i: nums){

s.insert(i);

}

priority\_queue<int> pq;

for(auto i: s){

pq.push(i);

}

if(pq.size()>2){

pq.pop();

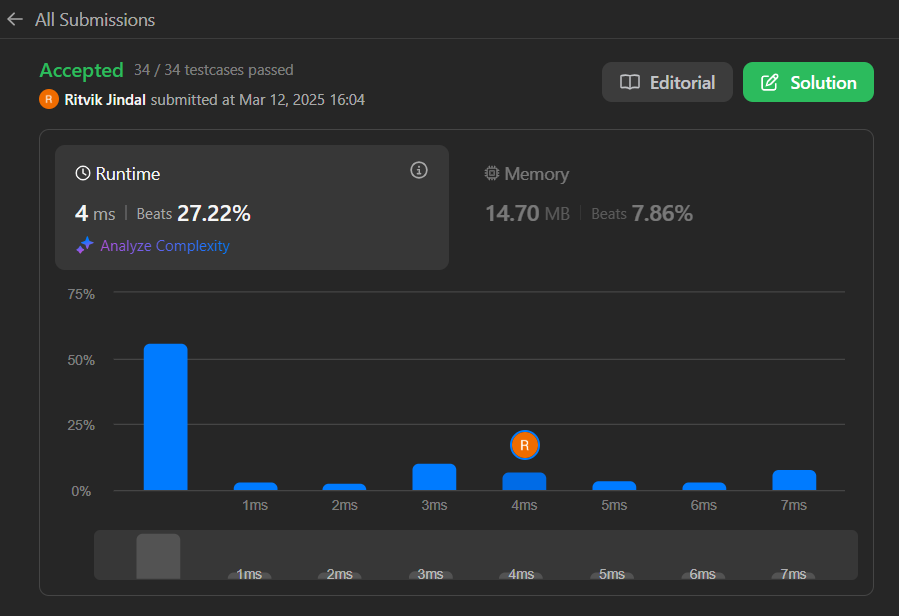
pq.pop();

}

return pq.top();

}

};



**451.**[**Sort Characters By Frequency**](https://leetcode.com/problems/sort-characters-by-frequency/description/)

class Solution {

public:

string frequencySort(string s) {

unordered\_map<char,int>mapi;

for(auto i :s){

mapi[i]++;

}

vector<pair<int,char>> vec;

for(auto i:mapi){

vec.push\_back({i.second,i.first});

}

sort(vec.rbegin(),vec.rend());

string res;

for(auto i:vec){

int temp = i.first;

while(temp>0){

res.push\_back(i.second);

temp--;

}

}

return res;

}

};

**452.**[**Minimum Number of Arrows to Burst Balloons**](https://leetcode.com/problems/minimum-number-of-arrows-to-burst-balloons/)

class Solution {

public:

int findMinArrowShots(vector<vector<int>>& points) {

// Sort the balloons based on their end coordinates

sort(points.begin(), points.end(), [](const vector<int>& a, const vector<int>& b) {

return a[1] < b[1];

});

int arrows = 1;

int prevEnd = points[0][1];

// Count the number of non-overlapping intervals

for (int i = 1; i < points.size(); ++i) {

if (points[i][0] > prevEnd) {

arrows++;

prevEnd = points[i][1];

}

}

return arrows;

}

};

**881.**[**Boats to Save People**](https://leetcode.com/problems/boats-to-save-people/description/)

class Solution {

public:

int numRescueBoats(vector<int>& people, int limit) {

int boatCount = 0;

sort(people.begin(), people.end());

int left = 0;

int right = people.size() - 1;

while(left <= right){

int sum = people[left] + people[right];

if(sum <= limit){

boatCount++;

left++;

right--;

}

else{

boatCount++;

right--;

}

}

return boatCount;

}

};

