Assingment -5

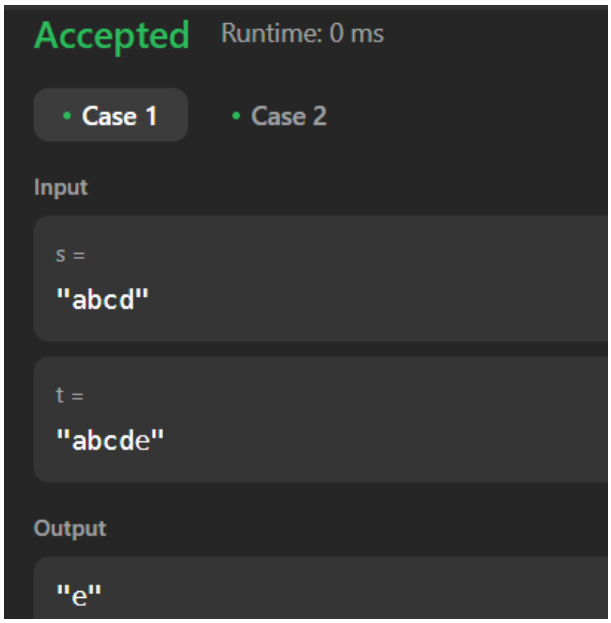
Name- Akul UID-22bcs10330 Section- 605-B

389. Find the Difference

Code:

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

Output:



976. Largest Perimeter Triangle

Code:

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

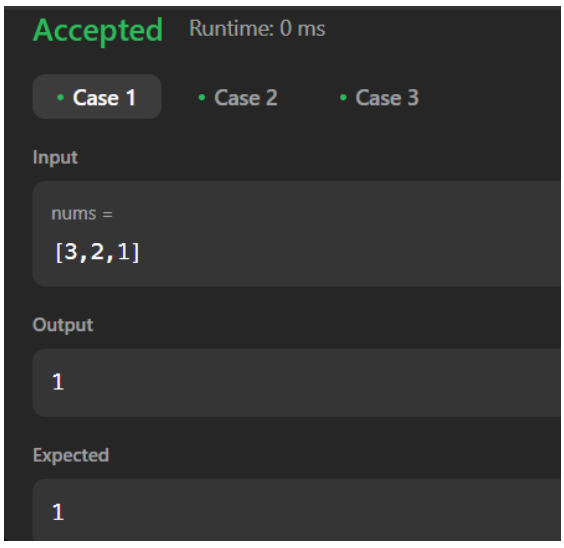
Output:

414. Third maximum number

Code:

```
class Solution {
public:
 int thirdMax(vector<int>& nums) {
           set<int>s;
        for(int i=0;i<nums.size();i++){
            s.insert(nums[i]);
        }
        if(s.size()>=3){
            int Third_index_from_last=s.size()-3;
            auto third_maximum=next(s.begin(),Third_index_from_last);
            return *third_maximum;
        }
            return *--s.end();}
};
```

OUTPUT:

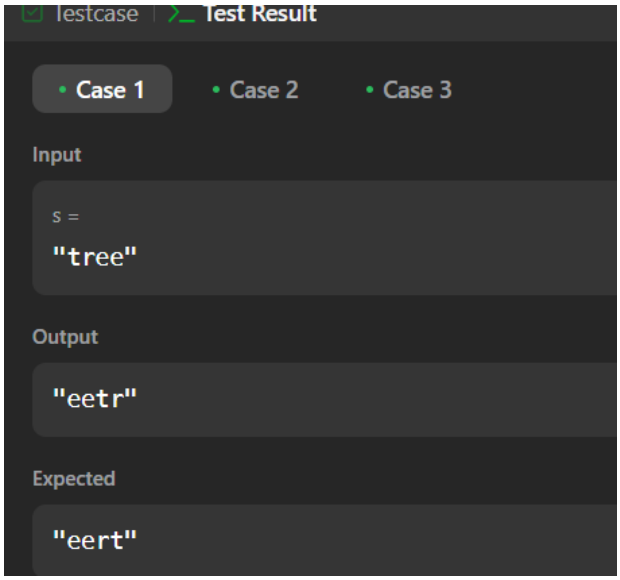


451. Sort characters by frequency

Code:

```cpp
class Solution {
public:
    string frequencySort(string s) {
        string ans="";
        map<char,int> mp;
        for(auto x:s){
            mp[x]++;
        }
        vector<pair<int,char>> res;
        for(auto x:mp){
            res.push_back({x.second,x.first});
        }
        sort(res.rbegin(),res.rend());
        for(auto x:res){
            int l=x.first;
            while(l--){
                ans+= x.second;
            }
        }
        return ans;
    }
};
```

OUTPUT:

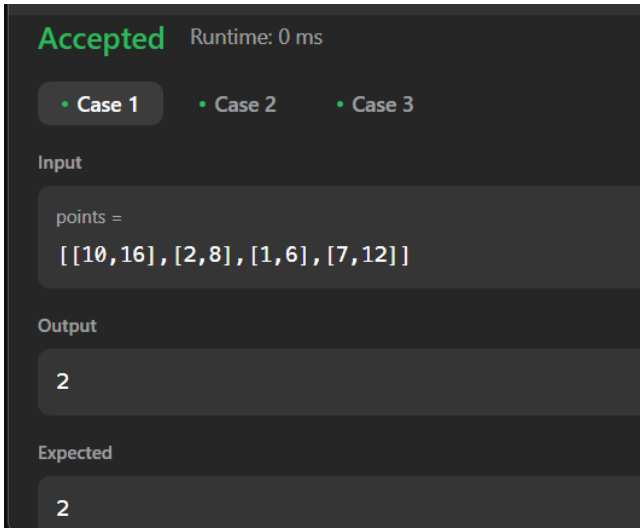


452. Minimum number of arrows to burst ballons Code:

```cpp
class Solution {
public:
    int findMinArrowShots(vector<vector<int>>& points) {
        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];
        for (int i = 1; i < points.size(); ++i) {
            if (points[i][0] > prevEnd) {
                arrows++;
                prevEnd = points[i][1];
            }
        }
        return arrows;
    }
};
```

Output:

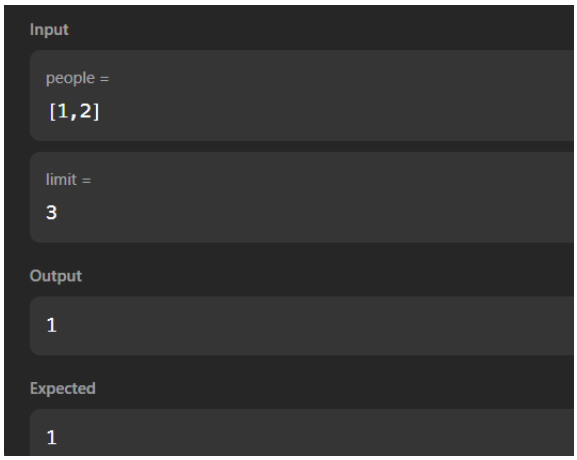


881.Boats to save people Code:

```
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;
```

```
    }
};
```

Output:

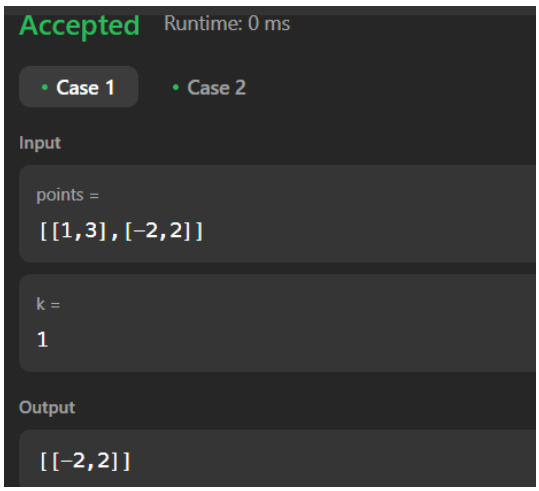


973. k closest points to origin

Code:

```
class Solution {
public:
    vector<vector<int>> kClosest(vector<vector<int>>& points, int k) {
        priority_queue<pair<int, vector<int>>> maxHeap;
        for (auto& point : points) {
            int distance = point[0] * point[0] + point[1] * point[1];
            maxHeap.push({distance, point});
            if (maxHeap.size() > k) maxHeap.pop();
        }
        vector<vector<int>> ans;
        while (!maxHeap.empty()) {
            ans.push_back(maxHeap.top().second);
            maxHeap.pop();
        }
        return ans;}
};
```

Output:



1338. Reduce array size to half

```cpp
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;
    }
};
```

Output: