Name:Atul Uid:22BCS15834

Q1. Find the Difference

```
char findTheDifference(string s, string t) {
    int hash[26] = {0};
    for (char ch : s) hash[ch - 'a']++;
    for (char ch : t) {
        hash[ch - 'a']--;
        if (hash[ch - 'a'] < 0) return ch;
    }
    return 0;
}
```

Q2. Largest Perimeter Triangle

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

Q3. Third Maximum Number

```
int thirdMax(vector<int>& nums) {
    set<int> s(nums.begin(), nums.end());
```

```
    if (s.size() < 3) return *s.rbegin();
    auto it = s.rbegin();
    advance(it, 2);
    return *it;
}
```

Q4. Sort Characters By Frequency

```
string frequencySort(string s) {
    unordered_map<char, int> freq;
    for (char ch : s) freq[ch]++;
    vector<pair<int, char>> v;
    for (auto it : freq) v.push_back({it.second, it.first});
    sort(v.rbegin(), v.rend());
    string res = "";
    for (auto p : v) res += string(p.first, p.second);
    return res;
}
```

Q5. Minimum Number of Arrows to Burst Balloons

```
int findMinArrowShots(vector<vector<int>>& points) {
    sort(points.begin(), points.end(), [](vector<int>& a, vector<int>& b) { return a[1] < b[1]; });
    int arrows = 1, end = points[0][1];
    for (int i = 1; i < points.size(); i++) {
        if (points[i][0] > end) {
            arrows++;
            end = points[i][1];
        }
    }
    return arrows;
```

```
}
```

Q6. Boats to Save People

```
int numRescueBoats(vector<int>& people, int limit) {
    sort(people.begin(), people.end());
    int i = 0, j = people.size() - 1, boats = 0;
    while (i <= j) {
        if (people[i] + people[j] <= limit) i++;
        j--;
        boats++;
    }
    return boats;
}
```

Q7. K Closest Points to Origin

```
vector<vector<int>> kClosest(vector<vector<int>>& points, int k) {
    priority_queue<pair<int, vector<int>>> pq;
    for (auto& p : points) {
        int dist = p[0] * p[0] + p[1] * p[1];
        pq.push({dist, p});
        if (pq.size() > k) pq.pop();
    }
    vector<vector<int>> res;
    while (!pq.empty()) {
        res.push_back(pq.top().second);
        pq.pop();
    }
    return res;
}
```

Q8. Reduce Array Size to The Half

```
  int minSetSize(vector<int>& arr) {
    unordered_map<int, int> freq;
    for (int num : arr) freq[num]++;
    vector<int> counts;
    for (auto it : freq) counts.push_back(it.second);
    sort(counts.rbegin(), counts.rend());
    int removed = 0, half = arr.size() / 2, sets = 0;
    for (int count : counts) {
      removed += count;
      sets++;
      if (removed >= half) break;
    }
    return sets;
  }
};
```