Experiment No. - 3

Student Name: Vivek Kumar
Branch: BE-CSE(LEET)
Semester: 6th
Subject Name: Competitive coding - II

UID: 21BCS8129
Section/Group: 20BCS-ST-801/B
Date of Performance: 28/02/2023
Subject Code: 20CSP-351

1. **Aim/Overview of the practical:**

   Q.1 Kth Largest Element in a Stream.

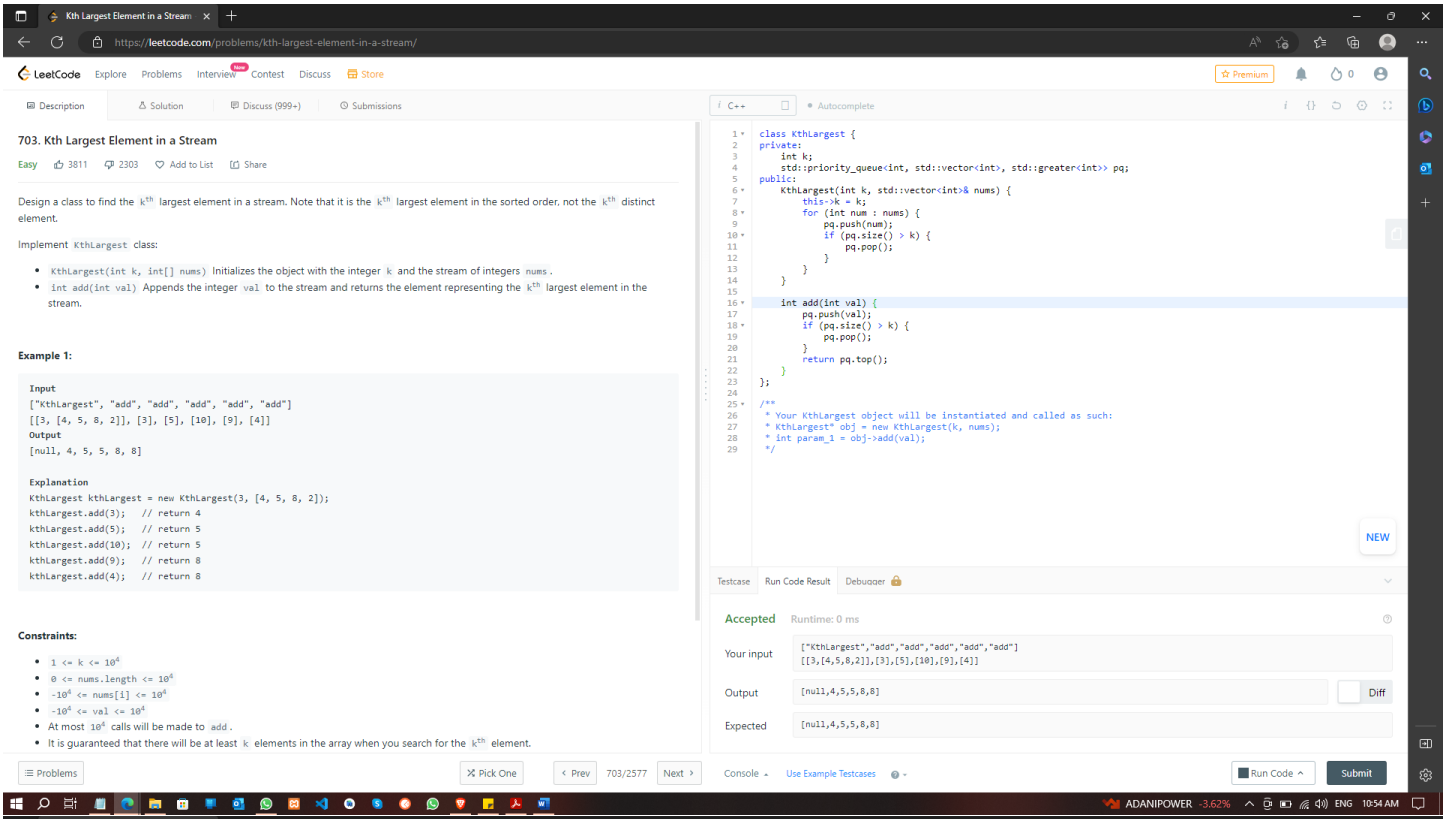
   https://leetcode.com/problems/kth-largest-element-in-a-stream/

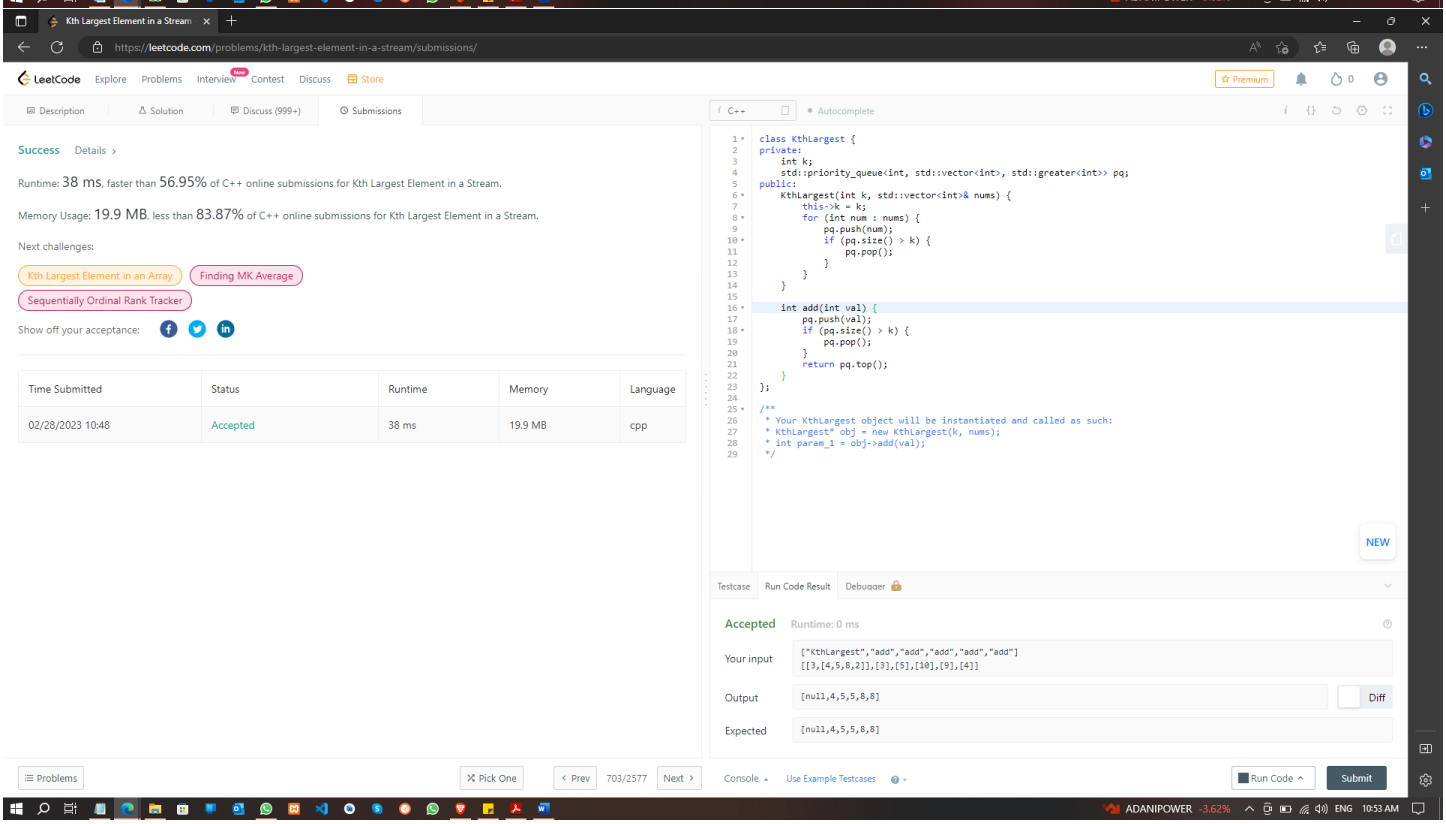
2. **Apparatus / Simulator Used:**
   - Windows 7 or above
   - Google Chrome
3. **Objective:**
   - To understand the concept of Queue
   - To implement the concept of Heap.
4. **Code:**

```
class KthLargest {
private:
    int k;
    std::priority_queue<int, std::vector<int>, std::greater<int>> pq;
public:
    KthLargest(int k, std::vector<int>& nums) {
        this->k = k;
        for (int num : nums) {
            pq.push(num);
            if (pq.size() > k) {
                pq.pop();
            }
        }
    }

    int add(int val) {
        pq.push(val);
        if (pq.size() > k) {
            pq.pop();
        }
        return pq.top();
    }
};
```

5. Result/Output/Writing Summary:





Submitted By: Vivek Kumar

1. **Aim/Overview of the practical:**

   Q.2 Cheapest Flights Within K Stops
   https://leetcode.com/problems/cheapest-flights-within-k-stops/

2. **Apparatus / Simulator Used:**
   - Windows 7 or above
   - Google Chrome

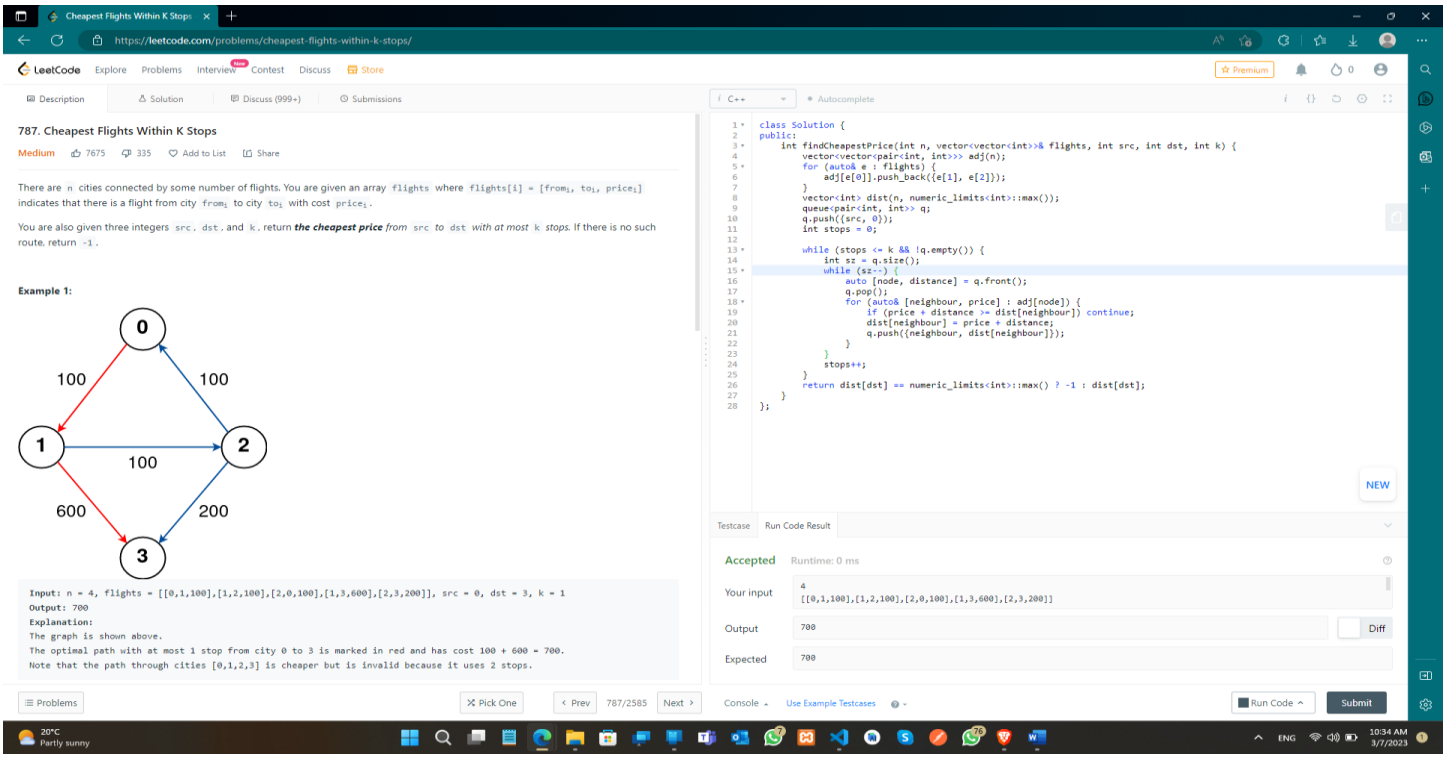
3. **Objective:**
   - To understand the concept of Vector.
   - To implement the concept of Vector iteration.

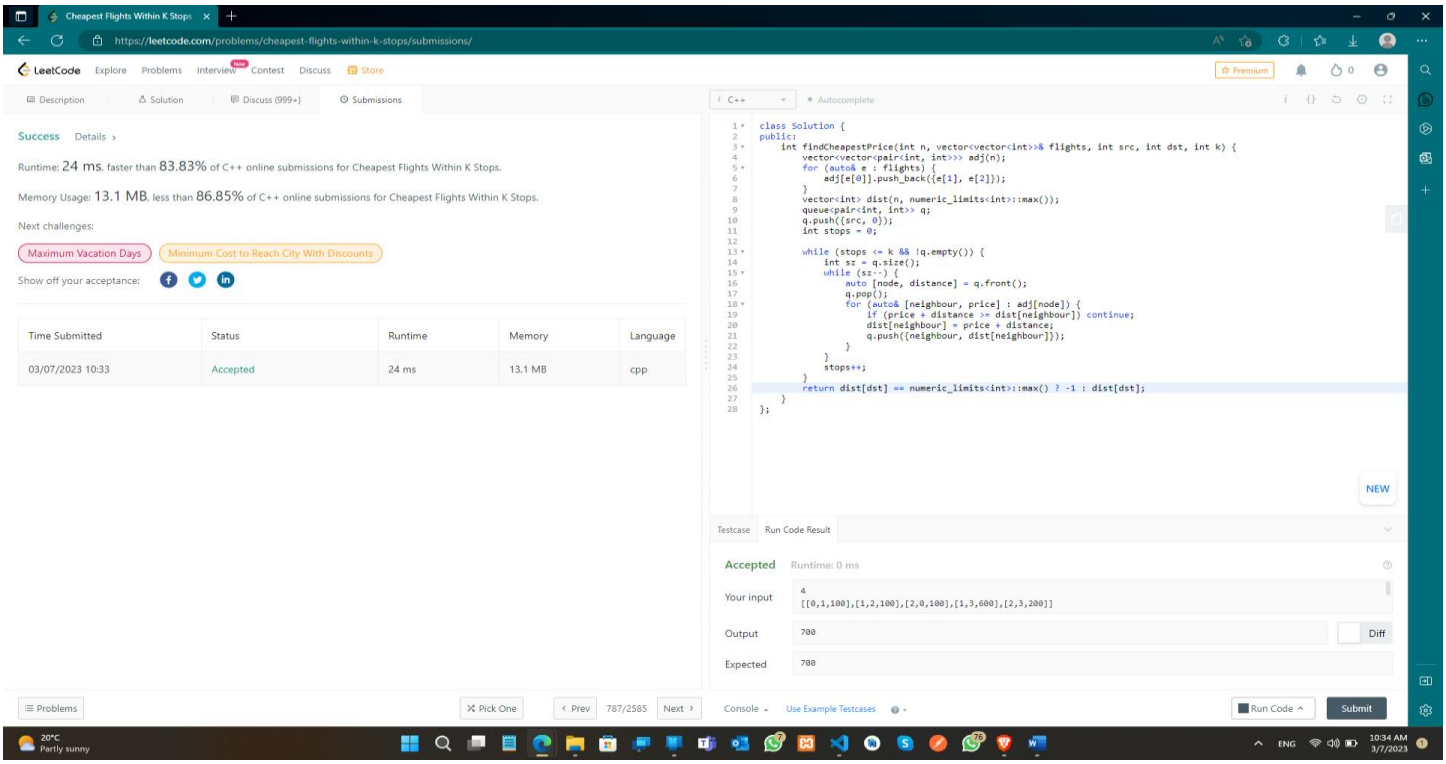
4. **Code:**

```
class Solution {
public:
    int findCheapestPrice(int n, vector<vector<int>>& flights, int src, int dst, int k) {
        vector<vector<pair<int, int>>> adj(n);
        for (auto& e : flights) {
            adj[e[0]].push_back({e[1], e[2]});
        }
        vector<int> dist(n, numeric_limits<int>::max());
        queue<pair<int, int>> q;
        q.push({src, 0});
        int stops = 0;

        while (stops <= k && !q.empty()) {
            int sz = q.size();
            while (sz--) {
                auto [node, distance] = q.front();
                q.pop();
                for (auto& [neighbour, price] : adj[node]) {
                    if (price + distance >= dist[neighbour]) continue;
                    dist[neighbour] = price + distance;
                    q.push({neighbour, dist[neighbour]});
                }
            }
            stops++;
        }
        return dist[dst] == numeric_limits<int>::max() ? -1 : dist[dst];
    }
};
```

5. Result/Output/Writing Summary:





Learning outcomes (What I have learnt):

- Learned the concept of cheapest flights within k stops.
- Learnt about Array in Vector and Its iteration.

Submitted By: Vivek Kumar