



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment 1.3

Student Name: Saumyamani Bhardwaz

UID: 20BCS1682

Branch: BE-CSE

Section/Group: 701_A

Semester: 6th

Date of Performance: 21/03/2023

Subject Name: Competitive Coding-II

Subject Code: 20CSP-351

Aim: To demonstrate the concept of Heap model

1. Kth Largest Element in a Stream

```
class KthLargest {
public:
    priority_queue<int, vector<int>, greater<int>>> pq;
    int K;
    KthLargest(int k, vector<int>& nums) {
        K = k;
        for(int x : nums) {
            pq.push(x);
            if(pq.size() > k) {
                pq.pop();
            }
        }
        int add(int val) {
            pq.push(val);
            if(pq.size() > K) {
                pq.pop();
            }
        }
        cout<<"Saumyamani Bhardwaz_20BCS1682"<<endl;
        return pq.top();
    }
};
```

Output



The screenshot shows a coding platform interface. At the top, there are tabs for 'Testcase' and 'Result'. The 'Result' tab is active, displaying 'Accepted' in green text and 'Runtime: 0 ms'. Below this, there is a section for 'Case 1'. Under 'Input', there are two lines of input: ["KthLargest","add","add","add","add","add"] and [[3],[4,5,8,2]],[3],[5],[10],[9],[4]]. Under 'Stdout', the output is 'Saumyamani Bhardwaz_20BCS1682'. At the bottom, there is a 'Console' dropdown menu, a 'Run' button, and a 'Submit' button.

2. Last Stone Wiegth

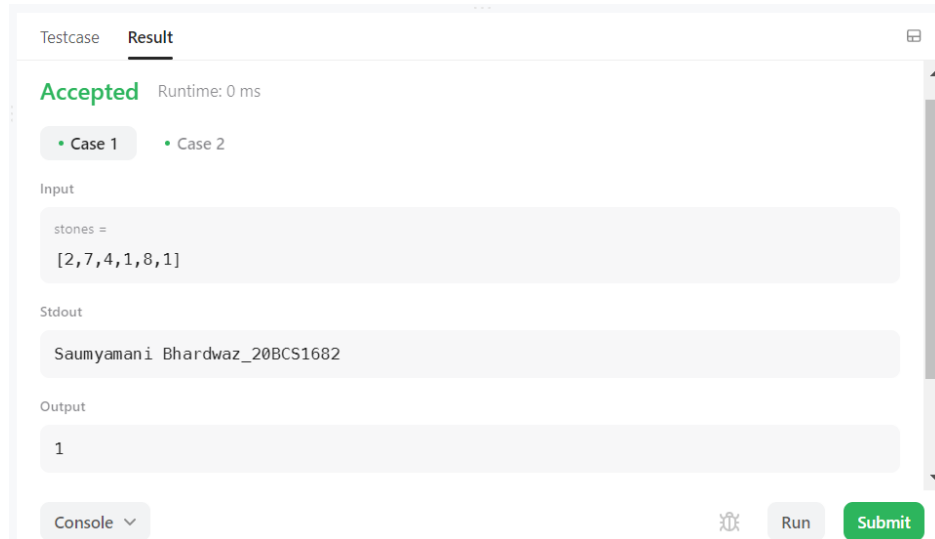
```
class Solution {
public:
    int lastStoneWeight(vector<int>& s) {
        make_heap(s.begin(), s.end());

        while (s.size() > 1)
        {
            int x = s.front();
            pop_heap(s.begin(), s.end());
            s.pop_back();

            int y = s.front();
            pop_heap(s.begin(), s.end());
            s.pop_back();

            if (x == y)
                continue;
            s.push_back(x - y);
            push_heap(s.begin(), s.end());
        }
        cout<<"Saumyamani Bhardwaz_20BCS1682";
        return s.size() ? s.front() : 0;
    }
};
```

Output:



3. Cheapest Flights Within K Stops

```
class Solution {
public:
    int findCheapestPrice(int n, vector<vector<int>>& flights, int src, int dst, int k) {
        vector<pair<int,int>> adj[n];
        for(auto it : flights){
            adj[it[0]].push_back({it[1],it[2]});
        }
        queue<pair<int,pair<int,int>>> pn;
        pn.push({0,{src,0}});
        vector<int> dist(n,1e9);
        dist[src] = 0;
        while(!pn.empty()){
            auto front = pn.front();
            pn.pop();
            int stops = front.first;
            int node = front.second.first;
            int distance = front.second.second;
            if(stops>k)continue;
            for(auto it:adj[node]){
                int adjnode = it.first;
                int d = it.second;
                if(distance + d<dist[adjnode]&&stops<=k){
                    dist[adjnode] = distance + d;
                    pn.push({stops+1,{adjnode,distance+d}});
                }
            }
        }
        if(dist[dst]==1e9)return -1;
        return dist[dst];
    }
};
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

};

Output:

787. Cheapest Flights Within K Stops

Medium7.7K336

Companies

There are `n` cities connected by some number of flights. You are given an array `flights` where `flights[i] = [fromi, toi, pricei]` indicates that there is a flight from city `fromi` to city `toi` with cost `pricei`.

Testcase

Result

Accepted

Runtime: 0 ms

Case 1

Case 2

Case 3

Input

n =
4

flights =
[[0,1,100],[1,2,100],[2,0,100],[1,3,600],[2,3,200]]

src =
0

dst =
3

k =
1

Stdout

Saumyamani Bhardwaz_20BCS1682

Output

700

Expected

700

Console

Run

Submit