PW SKILLS

Heaps

Lecture 3

Today's checklist

1. Questions on heaps

Minheap / Maxheap →

pop() → $O(\log n)$

push() → $O(\log n)$

top() → $O(1)$

find(x) → $O(n)$

remove(x) → $O(n) + \underbrace{O(\log n)}_{\text{neglect}}$

Ques:

Q1 : Find Median from Data Stream.

6, 1, 2, 4, 2, 1, 3, 8, 1

sort →

1, 1, 1, 2, 2, 3, 4, 6, 8

↓

median

6 1 2 4 2 8

sort →

1 2 2 4 6 8

↓

3.0

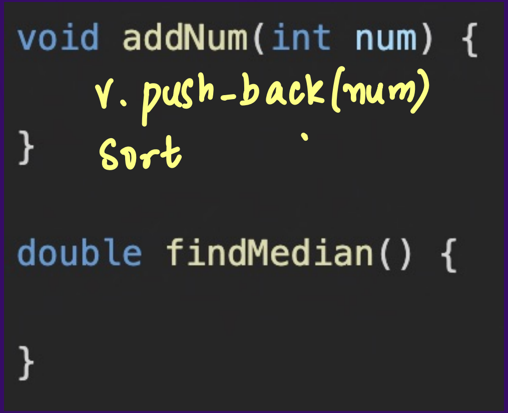
[Leetcode 295]

Ques:

6, 1, 2, 4, 2, 1, 3, 8, 1

Q1 : Find Median from Data Stream.

Stream	median
6	6
6 1	3.5
6 1 2	2
6 1 2 4	3



```
void addNum(int num) {
    v.push-back(num)
}   Sort

double findMedian() {

}
```

v = 1 2 6 4

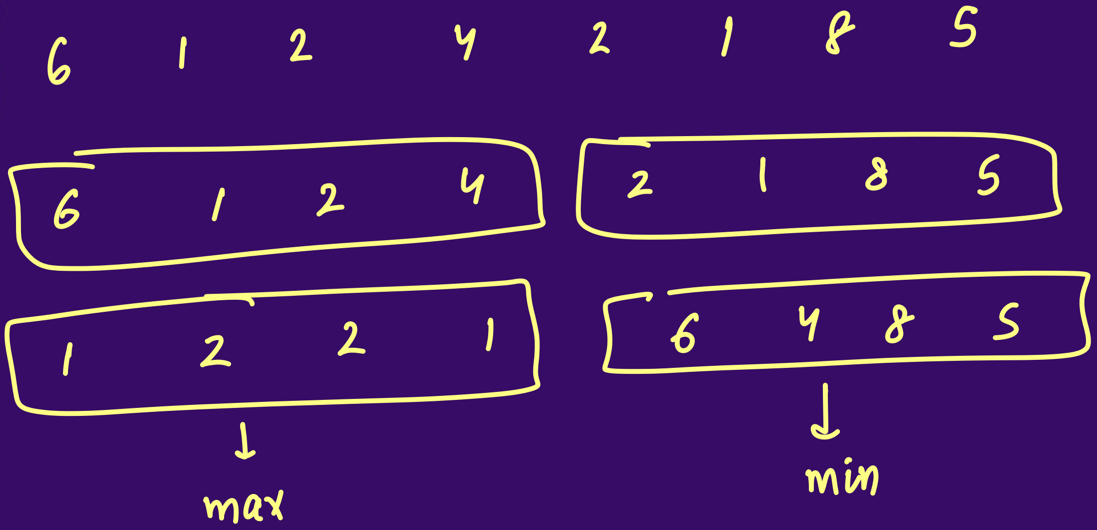
[Leetcode 295]

Ques:

2 heaps, 1 maxheap, 1 minheap

PW SKILLS

Q1 : Find Median from Data Stream.



$$\frac{max + min}{2} = median$$

[Leetcode 295]

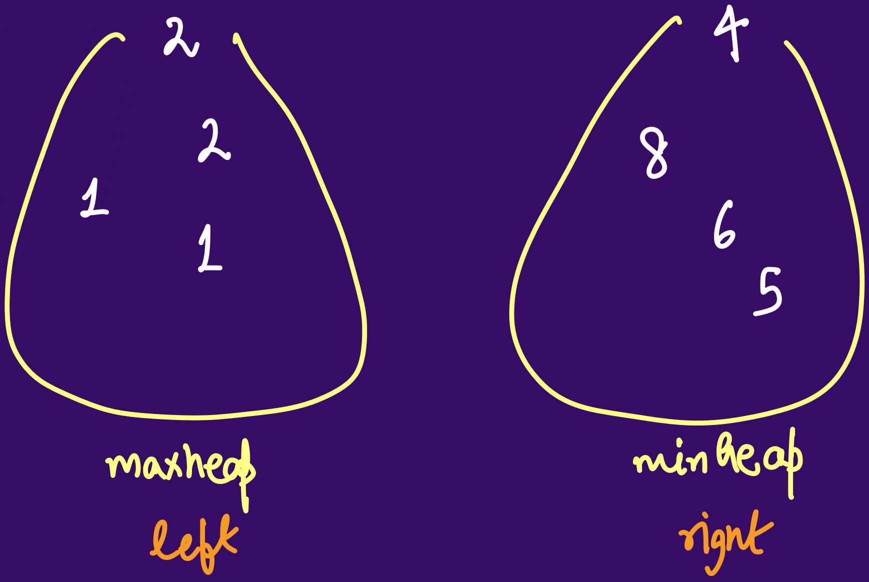
Ques:

Q1 : Find Median from Data Stream.

$\boxed{3}$

6 1 2 4 2 1 8 5

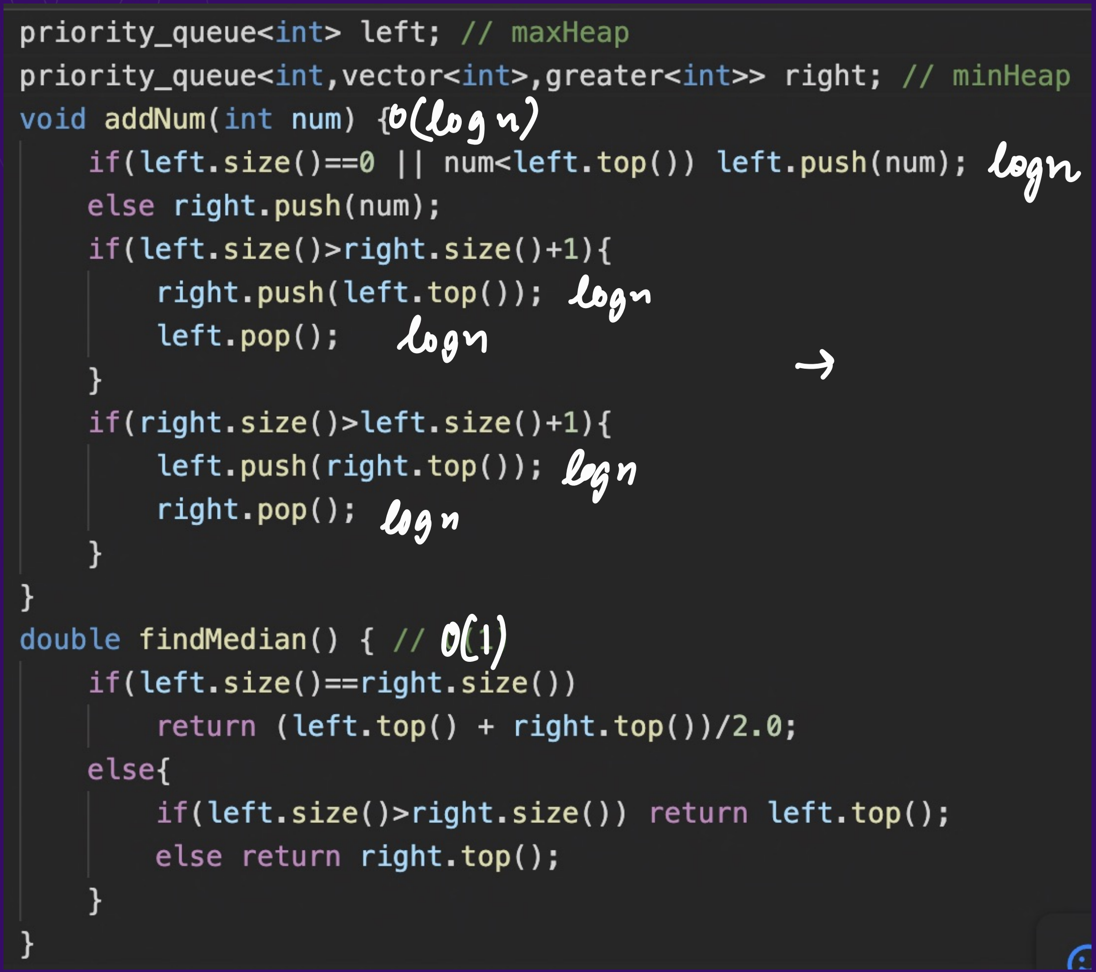
6 3.5 2 3 2 2 2 3



1) all elements of left <= all elements of right

2) Size of left & right should be same or the diff should be 1

[Leetcode 295]



```
priority_queue<int> left; // maxHeap
priority_queue<int,vector<int>,greater<int>> right; // minHeap
void addNum(int num) {O(log n)
    if(left.size()==0 || num<left.top()) left.push(num); logn
    else right.push(num);
    if(left.size()>right.size()+1){
        right.push(left.top()); logn
        left.pop();    logn                              →
    }
    if(right.size()>left.size()+1){
        left.push(right.top()); logn
        right.pop(); logn
    }
}
double findMedian() { // O(1)
    if(left.size()==right.size())
        return (left.top() + right.top())/2.0;
    else{
        if(left.size()>right.size()) return left.top();
        else return right.top();
    }
}
```

[Leetcode 295]

Ques:

Q2 : Smallest Range covering elements from K Lists

{4, 10, 15, 24, 26}

{0, 9, 12, 20}

{5, 18, 22, 30}

$\boxed{K=3}$

[4, 10]

[2, 9]

[2, 9] is a larger range than [4, 10]

[0, 30] 30   [10, 18] 8

[0.5] 5   [4, 9] 5

[Leetcode 632]

Ques:

k=3

Heap element

num, row, col

Q2 : Smallest Range covering elements from K Lists

$\{4, 10, 15, \overset{\downarrow}{21}, 26\}$

$\{0, 9, 12, \overset{\downarrow}{19}\}$ α

$\{5, 18, \overset{\downarrow}{22}, 30\}$

K elements (One from each list)

22

21	22

minheap

mx = ~~5~~ ~~9~~ ~~10~~ ~~18~~ ~~19~~ ~~21~~ 22

0	12
4	15
5	18
9	19
10	

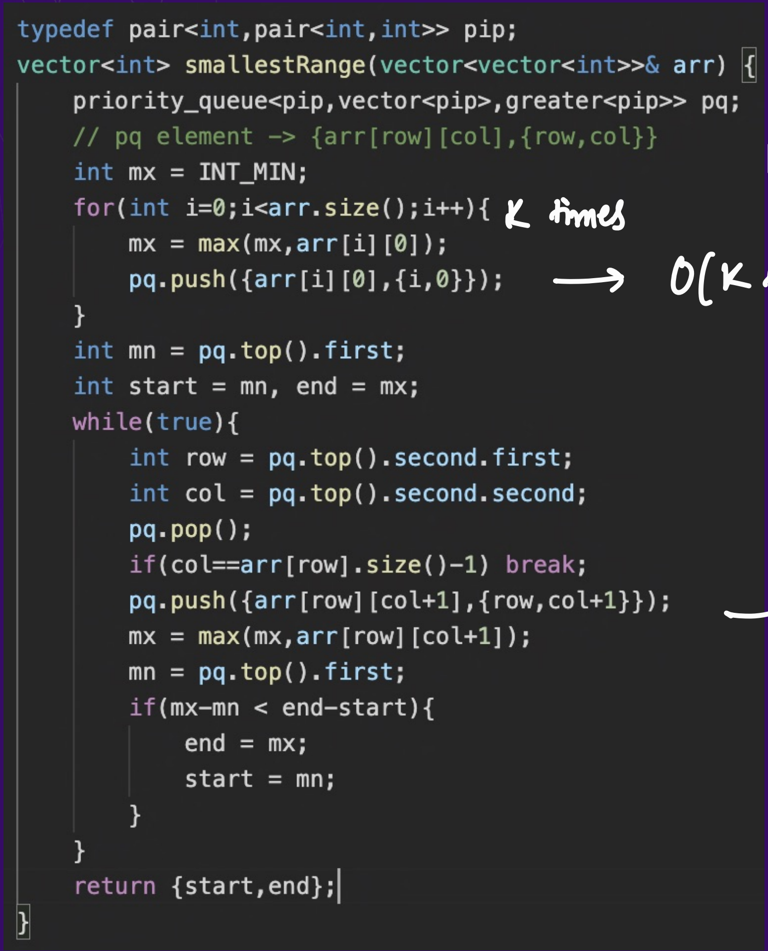
Start	end
~~0~~	~~5~~
~~15~~	~~19~~
18	21

pair<int, pair<int,int>>

[Leetcode 632]

K lists each of size 'n'
on an average

ng elements from K Lists



```
typedef pair<int,pair<int,int>> pip;
vector<int> smallestRange(vector<vector<int>>& arr) {
    priority_queue<pip,vector<pip>,greater<pip>> pq;
    // pq element -> {arr[row][col],{row,col}}
    int mx = INT_MIN;
    for(int i=0;i<arr.size();i++){
        mx = max(mx,arr[i][0]);
        pq.push({arr[i][0],{i,0}});
    }
    int mn = pq.top().first;
    int start = mn, end = mx;
    while(true){
        int row = pq.top().second.first;
        int col = pq.top().second.second;
        pq.pop();
        if(col==arr[row].size()-1) break;
        pq.push({arr[row][col+1],{row,col+1}});
        mx = max(mx,arr[row][col+1]);
        mn = pq.top().first;
        if(mx-mn < end-start){
            end = mx;
            start = mn;
        }
    }
    return {start,end};
}
```

K times

→ $O(K\log K)$

→ $O(nK\log K)$

$$T.C. = O(nk\log k)$$

$$= O(m\log k)$$

↓

m is the total no. of elements

[Leetcode 632]

```
pair< int , pair<int,int>> p = { 0, {1, 4}}
```

p.second.second

p.first p.second.first

THANK YOU