703. Kth Largest Element in a Stream

08 April 2022 05:16 PM

Design a class to find the `kth` largest element in a stream. Note that it is the `kth` largest element in the sorted order, not the `kth` distinct element.

Implement `KthLargest` class:

- `KthLargest(int k, int[] nums)` Initializes the object with the integer `k` and the stream of integers `nums`.
- `int add(int val)` Appends the integer `val` to the stream and returns the element representing the `kth` largest element in the stream.

Example 1:

```
Input
["KthLargest", "add", "add", "add", "add", "add"]
[[3, [4, 5, 8, 2]], [3], [5], [10], [9], [4]]
Output
[null, 4, 5, 5, 8, 8]

Explanation
KthLargest kthLargest = new KthLargest(3, [4, 5,
8, 2]);
kthLargest.add(3);   // return 4
kthLargest.add(5);   // return 5
kthLargest.add(10);  // return 5
kthLargest.add(9);   // return 8
kthLargest.add(4);   // return 8
```

Constraints:

- `1 <= k <= 10^4`
- `0 <= nums.length <= 10^4`
- `-10^4 <= nums[i] <= 10^4`
- `-10^4 <= val <= 10^4`
- At most `10^4` calls will be made to `add`.
- It is guaranteed that there will be at least `k` elements in the array when you search for the `kth` element.

[Kth largest, add, add, add, add, add]

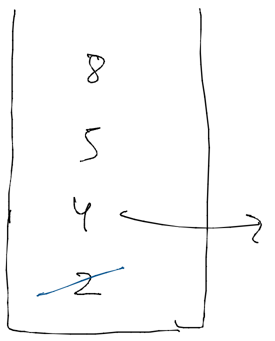
[[3, [4, 5, 8, 2]], [3], [5], [10], [9], [4]]

output ⇒ [4, 5, 5, 8, 8]

$\boxed{k = 3}$

[2, 8, 5, 4]

$$\begin{tikzcd} & \text{minheap} \arrow[r] & \text{smallest} \\ \boxed{PQ} \arrow[ur] \arrow[dr] & & \\ & \text{maxheap} \arrow[r] & \text{largest} \end{tikzcd}$$

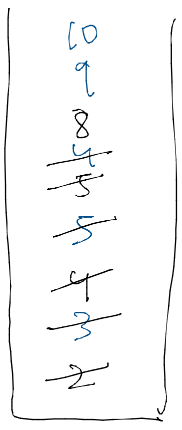


pq (minHeap)

size = 4 , k = 3

* Once you make $\boxed{k = = size}$ you can return kth largest element it can be done by removing minimum element

[2, 8, 5, 4]



add(4)
add(9)
add(10)
add(5)
add(3)

k = 3 , current size = ~~8~~ ~~4~~ ~~8~~ ~~4~~ ~~3~~ ~~4~~ 3
~~4~~ ~~3~~ ~~4~~ 3

we need to do $\underline{\underline{CS == k}}$, so remove mini element

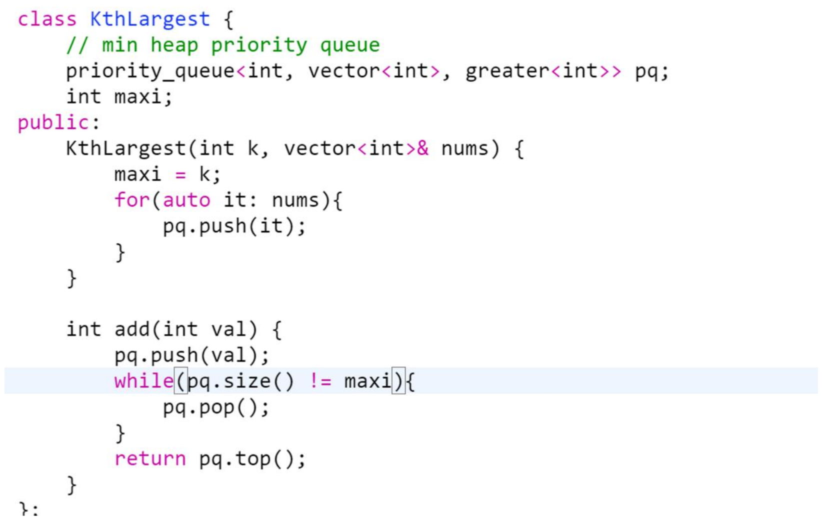
initialize ⇒ null
so for add(3) ans ⇒ 4
so for add(5) ans ⇒ 5
add(10) ans ⇒ 5
add(9) ans ⇒ 8
add(4) ans ⇒ 8
} ans

We use priority queue , but by default priority queue is maxheap

```
pq <int, vector<int>, greater<int>> pq; // mini heap

int maxElement.

kth largest (k, nums)

    maxElement = k

    for (auto it : nums)

        pq.push(it)

int add(int val)

    pq.push(val)

    while (k != pq.size()) {

        pq.pop()

    return pq.top()
```

$T.C = O(N\log N) + M\log k$

$S.C = O(N)$



```
class KthLargest {
    // min heap priority queue
    priority_queue<int, vector<int>, greater<int>> pq;
    int maxi;
public:
    KthLargest(int k, vector<int>& nums) {
        maxi = k;
        for(auto it: nums){
            pq.push(it);
        }
    }

    int add(int val) {
        pq.push(val);
        while(pq.size() != maxi){
            pq.pop();
        }
        return pq.top();
    }
};
```

$\boxed{\text{Check 215}} \Rightarrow$ same