703. Kth Largest Element in a Stream

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

703. Kth Largest Element in a Stream

```
/*
    Sorting (TLE)
    q: number of add queries
    n: size of stream
    Time complexity: O(n+qnlogn)=O(qnlogn)
    Space complexity:O(n+logn)
*/
class KthLargest{
    public:
        std::vector<int> A;
        int k;
    public:
        KthLargest(int k, vector<int>& nums) {
            this->k=k;
            A=nums;
        }

        int add(int val) {
            A.push_back(val);
            std::sort(A.begin(),A.end());
            return A[A.size()-k];
        }
    };
```

703. Kth Largest Element in a Stream

```
/*
    Multiset (AC)
    q: number of add queries
    n: size of stream
    Time complexity: O(n+q(logn+k))=O(q(logn+k))
    Space complexity:O(n)
*/
class KthLargest{
    public:
        std::multiset<int> s;
        int k;
    public:
        KthLargest(int k, vector<int>& nums) {
            this->k=k;
            for(auto& e: nums) s.insert(e);
        }

        int add(int val) {
            s.insert(val);

            auto it=s.end();
            for(int i=1;i<=k;++i) it--;

            return *it;
        }
    };
```

703. Kth Largest Element in a Stream

```
/*
    Min heap
    q: number of add queries
    n: size of stream
    Time complexity: O(nlogk+qlogk)=O((n+k)logk)
    Space complexity:O(k)
*/
class KthLargest{
    public:
        std::priority_queue<int,std::vector<int>,std::greater<int>>
min_heap;
        int k;
    public:
        KthLargest(int k, vector<int>& nums) {
            this->k=k;
            for(auto& e: nums) {
                if(min_heap.size()<k || e>min_heap.top())
                                  min_heap.push(e);

                if(min_heap.size()>k) min_heap.pop();
            }
        }

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