Description | Solution | Discuss (987) | Submissions

703. Kth Largest Element in a Stream

Easy 2629 1540 Add to List Share

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.

Accepted 224,850 | Submissions 416,108

Seen this question in a real interview before? Yes No

Companies

Related Topics

Similar Questions

Java Autocomplete

```java
class KthLargest {

    static PriorityQueue<Integer> sh = new PriorityQueue<>();
    static int k =0;

    public KthLargest(int k, int[] nums) {
        this.k=k;
        sh.clear();
        for(int i:nums) add(i);
    }


    public static int add(int val) {
        sh.add(val);
        if(sh.size()>k) sh.poll();
        return sh.peek();
    }
}

/**
 * Your KthLargest object will be instantiated and called as such:
 * KthLargest obj = new KthLargest(k, nums);
 * int param_1 = obj.add(val);
 */
```