347. Top K Frequent Elements

Given a non-empty array of integers, return the k most frequent elements.

Example 1:

```
Input: nums = [1,1,1,2,2,3], k = 2
Output: [1,2]
```

```
class Solution {
    public List<Integer> topKFrequent(int[] nums, int k) {

        // heap
        Map<Integer, Integer> count = new HashMap<>();

        for (int n : nums) {
            count.put(n, count.getOrDefault(n, 0) + 1);
        }

        PriorityQueue<Integer> heap =
            new PriorityQueue<Integer>((n1, n2) ->
            count.get(n1) - count.get(n2));

        for (int n : count.keySet()) {
        //returns a set having the keys of the hash map
            heap.add(n);
            if (heap.size() > k)
                heap.poll();
                // Removing the top priority element (or head)
                // root is the smallest?
        }

        List<Integer> topK = new LinkedList();
        while (!heap.isEmpty()) {
            topK.add(heap.poll());
        }

        Collections.reverse(topK);
        return topK;
    }
}
```