347. Top K Frequent Elements

Hint:

1.新建hashmap来记录每个数字以及其出现次数

2.将数字和频率组合成新的数据结构

3.通过priorityqueue大顶堆进行排序输出

```
class Solution {
    class Cordinator{
        int a;
        int b;
        Cordinator(int a, int b){
            this.a = a;
            this.b = b;
        }

    }
    public List<Integer> topKFrequent(int[] nums, int k) {
        List<Integer> list = new ArrayList<>();
        PriorityQueue<Cordinator> pq = new PriorityQueue<>(
            new Comparator<Cordinator>(){
                public int compare(Cordinator c1, Cordinator c2){
                return c2.b - c1.b;
            }
        });
        Map<Integer, Integer> map = new HashMap<>();
        for(int i = 0; i < nums.length; i++){
            map.put(nums[i], map.getOrDefault(nums[i], 0) + 1);
        }
        for(Integer temp: map.keySet()){
            pq.add(new Cordinator(temp, map.get(temp)));
        }
        for(int i = 0; i < k; i++){
            list.add(pq.poll().a);
        }
        return list;
    }
}
```

973. K Closest Points to Origin

```
class Solution {
    public int[][] kClosest(int[][] points, int K) {
        int[][] ans = new int [K][2];
        PriorityQueue<int[]> pq = new PriorityQueue<>(
            new Comparator<int[]>(){
                public int compare(int[] a1, int[] a2){
                    return a1[0] * a1[0] + a1[1] * a1[1] - a2[0] * a2[0] - a2[1] * a2[1];
                }
            }
        );
        for(int i = 0; i < points.length; i++){
            pq.add(points[i]);
        }
        for(int i = 0; i < K; i++){
            int[] temp = pq.poll();
            ans[i][0] = temp[0];
            ans[i][1] = temp[1];
        }
        return ans;
    }
}
```

287. Find the Duplicate Number

```
class Solution {
    public int findDuplicate(int[] nums) {
        int tortoise = nums[0];
        int hare = nums[0];
        do {
            tortoise = nums[tortoise];
            hare = nums[nums[hare]];
        } while (tortoise != hare);
        tortoise = nums[0];
        while(tortoise != hare){
            tortoise = nums[tortoise];
            hare = nums[hare];
        }
        return tortoise;
    }
}
```