347. Top K Frequent Elements (Medium)

Given an integer array `nums` and an integer `k`, return the `k` most frequent elements within the array.

The test cases are generated such that the answer is always unique.

You may return the output in any order.

Example 1:

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

Output: [2,3]
```

Example 2:

```
Input: nums = [7,7], k = 1

Output: [7]
```

Constraints:

- `1 <= nums.length <= 10^4`.
- `1000 <= nums[i] <= 1000`
- `1 <= k <= number of distinct elements in nums`.

▼ 思路:

作法一：hash紀錄次數後sort比對

T:O(NlogN), S:O(N)

作法二：根據出現次數作為index搜尋top k字母

T:O(N), S:O(N)

作法一

```
class Solution {
public:
    vector<int> topKFrequent(vector<int>& nums, int k) {
        unordered_map<int,int> tpk;
        for(int num: nums){
            tpk[num]++;
        }
        vector<pair<int,int>> sortk;
        for(const auto &a : tpk){
            sortk.push_back({a.second,a.first}); //sort first of pair
        }
        sort(sortk.rbegin(),sortk.rend()); //descending order
        vector<int> ans;
        for(int i=0;i<k;i++){
            ans.push_back({sortk[i].second});
        }
        return ans;
    }
};
```

作法二

```
class Solution {
public:
    vector<int> topKFrequent(vector<int>& nums, int k) {
        unordered_map<int,int> tpk;
        for(int num: nums){
            tpk[num]++;
        }
        vector<vector<int>> sortk(nums.size()+1);
        for(const auto &a : tpk){
            sortk[a.second].push_back(a.first);
            //cout << a.second << ": " << a.first<<endl;
        }
        vector<int> ans;
```

```
        int s = nums.size();
        while(k>0){
            if(!sortk[s].empty()){
                for(int tmp: sortk[s]){
                    ans.push_back(tmp);
                    k--;
                }
            }
            s--;
        }
        return ans;
    }
};
```