<https://leetcode.com/problems/top-k-frequent-elements/>

**Top K Frequent Elements**

**Given an integer array nums and an integer k, return the k most frequent elements. You may return the answer in any order.**

Example 1:

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

Output: [1,2]

Example 2:

Input: nums = [1], k = 1

Output: [1]

Constraints:

1 <= nums.length <= 105

-104 <= nums[i] <= 104

k is in the range [1, the number of unique elements in the array].

It is guaranteed that the answer is unique.

**Method 1: ….**

**Method 2: (HashMap method)**

Store element and freq in an unordered\_map

Copy it to a pair vector and sort by second element

Time Complexity: O(nlogn) *[sorting]*

Space Complexity: O(n) *[]*

static bool cmp(pair<int,int>&a, pair<int,int>&b){

        return a.second>b.second;

    }

    vector<int> topKFrequent(vector<int>& nums, int k) {

        vector<pair<int,int>> freq;

        unordered\_map<int,int> mp;

        for(int i=0; i<nums.size(); i++){ //O(n)

            mp[nums[i]]++;

        }

        for(auto x : mp){

            freq.push\_back({x.first, x.second});

        }

        sort(freq.begin(), freq.end(), cmp); //O(dlogd):d=distinct elements; worst case O(nlogn)

        vector<int> res;

        for(int i=0; i<k; i++) //O(k)

            res.push\_back(freq[i].first);

        return res;

    }