题目描述：

给你一个数组 nums 和一个整数 K，请你返回出现频率前 K 高的元素。

如 111223，K=2

结果集：[1, 2]

大顶堆、小顶堆

问题：不是很明白 这个 priority_queue 是对 key 排序还是对 value 排序

```
//升序队列  小顶堆 great 小到大
priority_queue <int,vector<int>,greater<int> > pri_que;
//降序队列  大顶堆 less  大到小 默认
priority_queue <int,vector<int>,less<int> > pri_que;
```

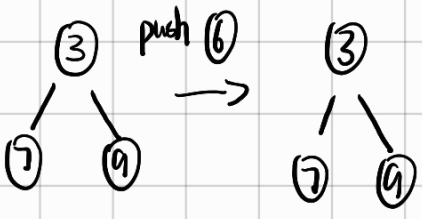
2、优先级队列内置函数

```
q.size();//返回q里元素个数
q.empty();//返回q是否为空，空则返回1，否则返回0
q.push(k);//在q的末尾插入k
q.pop();//删掉q的第一个元素
q.top();//返回q的第一个元素
```

思路：

遍历元素将其放入 map 中，Key 为数值，value 为计数值

再将其放入小顶堆中，用自定义的小顶堆操作符将较小的数据当作堆头




这样遍历过 map 之后会只剩下最大的数

于是我们得到了最高频的 K 个数

代码：

```
//先存进map中
for (int i=0; i < nums.size(); i++) {
        map[nums[i]]++;
}

//对map的值进行排序
Priority_queue<int,int>  que;

for (map :it) {
    que.push(it)
    if(que.size() > K) que.pop();
}
vector<int> result;
for( int i= k-1; i>=0;i--) {
     result[i] = que.top().first();
     que.pop();
}
```

注意代码中省略了自定义的比较方式

这是略省写法，要传入类型和容器、比较方式