# 题目

给你一个整数数组nums和一个整数k。你需要找到nums中长度为k的子序列，且这个子序列的和最大 。

请你返回任意一个长度为k的整数子序列。

子序列定义为从一个数组里删除一些元素后，不改变剩下元素的顺序得到的数组。

示例 1：

输入：nums = [2,1,3,3], k = 2

输出：[3,3]

解释：

子序列有最大和：3 + 3 = 6 。

示例 2：

输入：nums = [-1,-2,3,4], k = 3

输出：[-1,3,4]

解释：

子序列有最大和：-1 + 3 + 4 = 6 。

示例 3：

输入：nums = [3,4,3,3], k = 2

输出：[3,4]

解释：

子序列有最大和：3 + 4 = 7 。

另一个可行的子序列为 [4, 3] 。

提示：

1 <= nums.length <= 1000

-105 <= nums[i] <= 105

1 <= k <= nums.length

# 分析

## 方法一：排序

思路：

首先构建一个辅助数组 vals，其中每个元素是一个 pair，包含原始数组中的值和对应的下标。然后，对 vals 数组按值进行降序排序，这样得到的数组在值相同的情况下，下标较小的元素会排在前面。接着，从排序后的数组中取出前 k 个元素，并按照它们在原始数组中的下标升序排序，最终得到的就是符合题意的长度为 k 的子序列。

代码：

class Solution {

public:

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

vector<pair<int, int>> vals; // 辅助数组

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

vals.emplace\_back(nums[i], i);

}

// 按照数值降序排序

sort(vals.rbegin(), vals.rend());

// 取前 k 个并按照下标升序排序

sort(vals.begin(), vals.begin() + k, [](auto& x1, auto& x2) {

return x1.second < x2.second;

});

vector<int> res; // 目标子序列

for (int i = 0; i < k; ++i) {

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

}

return res;

}

};

## 方法二：堆/优先队列