煎饼排序

给你一个整数数组 arr ，请使用 **煎饼翻转** 完成对数组的排序。

一次煎饼翻转的执行过程如下：

- 选择一个整数 k ，1 <= k <= arr.length
- 反转子数组 arr[0...k-1]（**下标从 0 开始**）

例如，arr = [3,2,1,4] ，选择 k = 3 进行一次煎饼翻转，反转子数组 [3,2,1] ，得到 arr = [**1,2,3,**4] 。

以数组形式返回能使 arr 有序的煎饼翻转操作所对应的 k 值序列。任何将数组排序且翻转次数在 10 * arr.length 范围内的有效答案都将被判断为正确。

示例 1：

输入：[3,2,4,1]

输出：[4,2,4,3]

解释：

我们执行 4 次煎饼翻转，k 值分别为 4，2，4，和 3。

初始状态 arr = [3, 2, 4, 1]

第一次翻转后（k = 4）：arr = [**1, 4, 2, 3**]

第二次翻转后（k = 2）：arr = [**4, 1,** 2, 3]

第三次翻转后（k = 4）：arr = [**3, 2, 1, 4**]

第四次翻转后（k = 3）：arr = [**1, 2, 3,** 4]，此时已完成排序。

示例 2：

输入：[1,2,3]

输出：[]

解释：

输入已经排序，因此不需要翻转任何内容。

请注意，其他可能的答案，如 [3，3] ，也将被判断为正确。

```cpp
class Solution {
public:
/*
思路与算法

设一个元素的下标是 index，我们可以通过两次煎饼排序将它放到尾部：

第一步选择 k = index+1，然后反转子数组 arr[0...k-1]，此时该元素已经被放到首部。

第二步选择 k = n，其中 n 是数组 arr 的长度，然后反转整个数组，此时该元素已经被放到尾部。

通过以上两步操作，我们可以将当前数组的最大值放到尾部，然后将去掉尾部元素的数组作为新的处理对象，重复以上操作，直到处理对象的长度等于一，此时原数组已经完成排序，且需要的总操作数是 2×(n−1)，符合题目要求。如果最大值已经在尾部，我们可以省略对应的操作。

*/
    vector<int> pancakeSort(vector<int>& arr) {
        vector<int> ret;
        for (int n = arr.size(); n > 1; n--) {
            int index = max_element(arr.begin(), arr.begin() + n) - arr.beg
in();//寻找最大值元素下标索引
             if (index == n - 1) {//当最大值元素位于末尾 则忽略后续翻转操作 直接
缩进范围继续在剩余区间找最大值
                continue;
            }
            reverse(arr.begin(), arr.begin() + index + 1);
            reverse(arr.begin(), arr.begin() + n);
            ret.push_back(index + 1);
            ret.push_back(n);
        }
        return ret;
    }
};
```