# 题目

给你一个下标从0开始的整数数组nums，如果恰好删除一个元素后，数组严格递增，那么请你返回true，否则返回false。如果数组本身已经是严格递增的，请你也返回 true 。

数组 nums 是 严格递增 的定义为：对于任意下标的 1 <= i < nums.length 都满足 nums[i - 1] < nums[i] 。

示例 1：

输入：nums = [1,2,10,5,7]

输出：true

解释：从 nums 中删除下标 2 处的 10 ，得到 [1,2,5,7] 。

[1,2,5,7] 是严格递增的，所以返回 true 。

示例 2：

输入：nums = [2,3,1,2]

输出：false

解释：

[3,1,2] 是删除下标 0 处元素后得到的结果。

[2,1,2] 是删除下标 1 处元素后得到的结果。

[2,3,2] 是删除下标 2 处元素后得到的结果。

[2,3,1] 是删除下标 3 处元素后得到的结果。

没有任何结果数组是严格递增的，所以返回 false 。

示例 3：

输入：nums = [1,1,1]

输出：false

解释：删除任意元素后的结果都是 [1,1] 。

[1,1] 不是严格递增的，所以返回 false 。

示例 4：

输入：nums = [1,2,3]

输出：true

解释：[1,2,3] 已经是严格递增的，所以返回 true 。

提示：

2 <= nums.length <= 1000

1 <= nums[i] <= 1000

# 分析

## 方法一：模拟

思路：

通过遍历数组，判断是否存在连续两个以上的元素需要删除才能使数组严格递增。

代码：

class Solution {

public:

bool canBeIncreasing(vector<int>& nums) {

int count = 0; // 记录需要删除的元素数量

for (int i = 1; i < nums.size(); ++i) {

if (nums[i] <= nums[i - 1]) {

++count;

if (count > 1) { // 如果删除超过一个元素，则无法使数组严格递增

return false;

}

if (i > 1 && nums[i] <= nums[i - 2]) {

nums[i] = nums[i - 1]; // 删除当前元素，使数组递增

}

}

}

return true;

}

};