376-摆动序列

题述

376. 摆动序列

难度 中等 585 收藏 分享 切换为英文 接收动态 反馈

如果连续数字之间的差严格地在正数和负数之间交替，则数字序列称为 **摆动序列** 。第一个差（如果存在的话）可能是正数或负数。仅有一个元素或者含两个不等元素的序列也视作摆动序列。

- 例如， `[1, 7, 4, 9, 2, 5]` 是一个 **摆动序列** ，因为差值 `(6, -3, 5, -7, 3)` 是正负交替出现的。
- 相反，`[1, 4, 7, 2, 5]` 和 `[1, 7, 4, 5, 5]` 不是摆动序列，第一个序列是因为它的前两个差值都是正数，第二个序列是因为它的最后一个差值为零。

子序列 可以通过从原始序列中删除一些（也可以不删除）元素来获得，剩下的元素保持其原始顺序。

给你一个整数数组 `nums` ，返回 `nums` 中作为 **摆动序列** 的 **最长子序列的长度** 。

示例 1：

输入： nums = [1,7,4,9,2,5]
输出： 6
解释： 整个序列均为摆动序列，各元素之间的差值为 (6, -3, 5, -7, 3) 。

示例 2：

输入： nums = [1,17,5,10,13,15,10,5,16,8]
输出： 7
解释： 这个序列包含几个长度为 7 摆动序列。
其中一个是 [1, 17, 10, 13, 10, 16, 8] ，各元素之间的差值为 (16, -7, 3, -3, 6, -8) 。

示例 3：

输入： nums = [1,2,3,4,5,6,7,8,9]
输出： 2

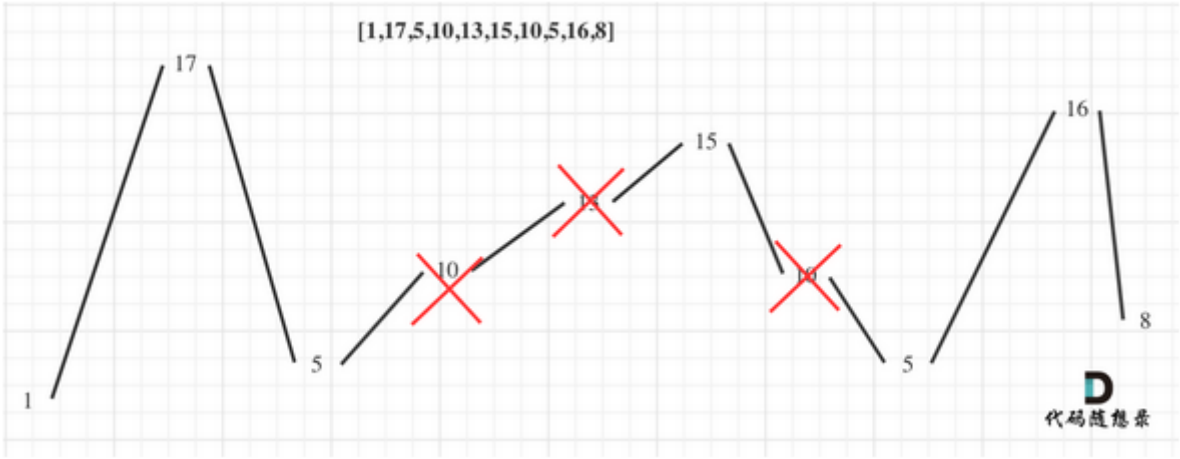
思路

贪心

本题要求通过从原始序列中删除或者不删除一些元素来获得子序列，剩下的保持原序。

摆动序列就是差值正负交替出现的序列

这道题可以使用贪心思想来解决。



局部最优：删除单调坡度上的节点（不包括单调坡度两端的节点），那么这个坡度就可以有两个局部峰值。

整体最优：整个序列有最多的局部峰值，从而达到最长摆动序列。

局部最优=》全局最优！

代码角度：

不需要执行删除操作，题目只要求返回最长摆动子序列的长度，那么我们只需要统计数组的峰值数量就可以了。

本题代码实现中，还有一些技巧，例如统计峰值的时候，数组最左面和最右面是最不好统计的。

例如序列[2,5]，它的峰值数量是2，如果靠统计差值来计算峰值个数就需要考虑数组最左面和最右面的特殊情况。

所以可以针对序列[2,5]，可以假设为[2,2,5]，这样它就有坡度了即preDiff = 0

题解

贪心

```
class Solution {
public:
    int wiggleMaxLength(vector<int>& nums)
    {
        //贪心   局部峰值个数
        if(nums.size()<=1)
        {
            return nums.size();
        }

        int curDiff=0;  //当前的两数之差
        int preDiff=0;  //上一个两数之差
        int result=1;   //峰值个数 默认最右边有一个峰值
        for(int i=0;i < nums.size()-1;i++)
        {
            //因为默认最右边有一个峰值 所以是 i < nums.size()-1
            curDiff=nums[i+1]-nums[i];  //计算新差值
            if((curDiff > 0 && preDiff <= 0)||(preDiff >= 0&& curDiff < 0))
            {
                //之所以要加上=0的情况是为了考虑一开始的情况
                result++;
```

```
                preDiff=curDiff;
            }
        }
        return result;
    }
};
```

提交结果	执行用时	内存消耗	语言	提交时间	备注
通过	0 ms	6.7 MB	C++	2022/02/06 09:44	添加备注
通过	0 ms	6.8 MB	C++	2022/02/06 09:43	添加备注
解答错误	N/A	N/A	C++	2022/02/06 09:42	添加备注
通过	0 ms	6.8 MB	C++	2022/02/06 09:42	添加备注
通过	0 ms	7 MB	C++	2022/02/06 09:42	添加备注
通过	0 ms	6.8 MB	C++	2022/02/06 09:41	添加备注

思考

局部最优=》全局最优