Leetcode\_376\_WiggleSubsequence\_摇摆序列\_Medium

# Leetcode\_376\_WiggleSubsequence\_摇摆序列\_Medium

## 题目介绍

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Leetcode\_376\_WiggleSubsequence\_Medium\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 难度：Medium

\* https://leetcode.com/problems/wiggle-subsequence/description/

\* 题目介绍：

\* A sequence of numbers is called a wiggle sequence if the differences

\* between successive numbers strictly alternate between positive and negative.

\* The first difference (if one exists) may be either positive or negative.

\* A sequence with fewer than two elements is trivially a wiggle sequence.

\*

\* For example, [1,7,4,9,2,5] is a wiggle sequence because the differences (6,-3,5,-7,3)

\* are alternately positive and negative. In contrast, [1,4,7,2,5] and [1,7,4,5,5] are

\* not wiggle sequences, the first because its first two differences are positive and

\* the second because its last difference is zero.

\*

\* Given a sequence of integers, return the length of the longest subsequence

\* that is a wiggle sequence. A subsequence is obtained by deleting some number of elements (eventually, also zero)

\* from the original sequence, leaving the remaining elements in their original order.

\*

\* Examples:

\* Input: [1,7,4,9,2,5] Output: 6

\* The entire sequence is a wiggle sequence.

\*

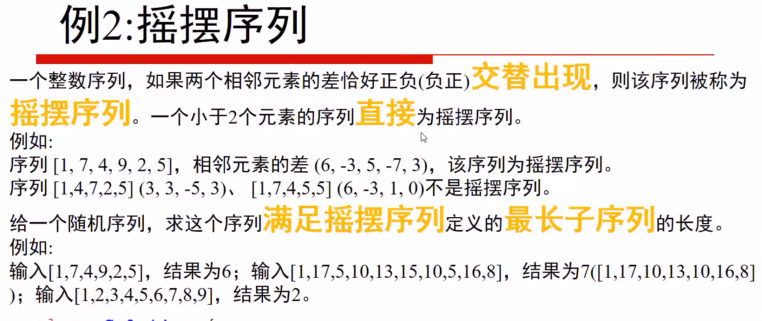
\* Input: [1,17,5,10,13,15,10,5,16,8] Output: 7

\* There are several subsequences that achieve this length. One is [1,17,10,13,10,16,8].

\*

\* Input: [1,2,3,4,5,6,7,8,9] Output: 2

\* Follow up: Can you do it in O(n) time? 遍历一遍



## 思路分析

\* 思路分析：方法利用状态转换环。每次都是贪心算法。

\* 三个状态：START、UP、DOWN

\* 初始时处于START状态，当遇到下一个数比第一个数大，则状态变为UP；

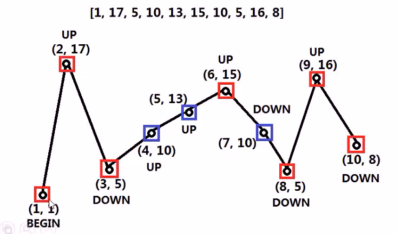
\* 若遇到下一个数比第一个数小，则状态变为DOWN；若相等，则continue；

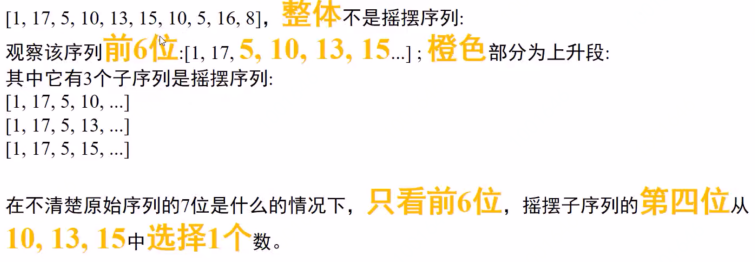
\* 处于UP状态，当遇到的数比上一个数小，则改为DOWN状态，计数加1；否则，continue；

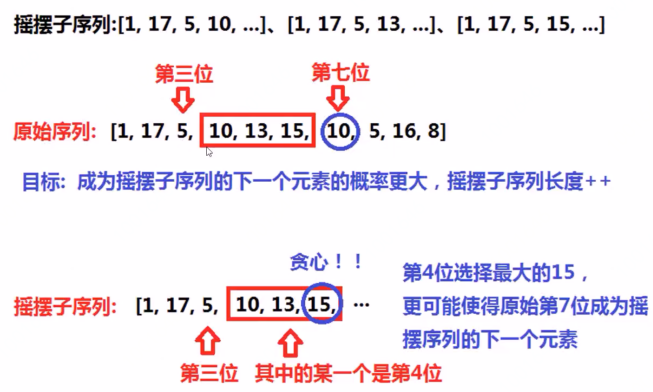
\* 处于DOWN状态，当遇到的数比上一个数大，则改为UP状态，计数加1；否则，continue.

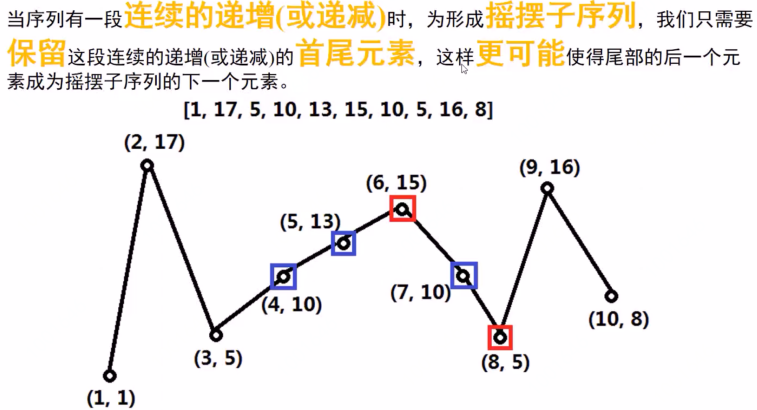
总的来说就是：**在上升阶段找下降的拐点，在下降阶段找上升拐点。**

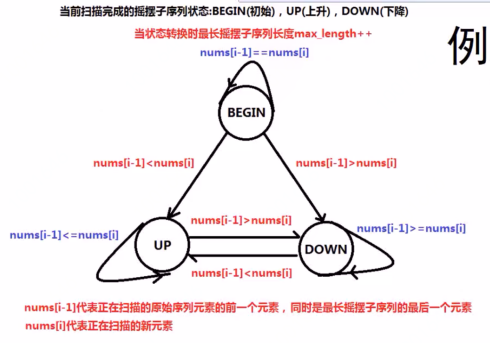
\* 注意：是相邻数i与i-1相比，**不是和上一个拐点相比**。一开始，定义一个preNum表示上一个拐点，每个数与这个拐点相比较，是错误的。











## Java代码

public int **wiggleMaxLength**(int[] nums) {

if(nums == null||nums.length == 0) return 0;

if(nums.length == 1) return 1;

int START = 0,UP = 1,DOWN = 2;

int STATE = START;//初始状态为START

int result = 1;

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

if(STATE == START){//可以利用switch实现

if(nums[i] == nums[i-1]) continue;//直接退出

if(nums[i] > nums[i-1])

STATE = UP;

else

STATE = DOWN;

result++;

}else if(STATE == UP){

if(nums[i] >= nums[i-1]) continue;

result++;

STATE = DOWN;

}else {//DOWN

if (nums[i] <= nums[i-1]) continue;

result++;

STATE = UP;

}

}

return result;

}

