376. Wiggle Subsequence

题目描述：https://leetcode.com/problems/wiggle-subsequence/

> 给定一个序列，判读满足如下要求的最长子串。
> [1,7,4,9,2,5] 这样的差值为 [6,-3,5,-7,3] 一正一负的。
> 例如：
>
> ```
> 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
> ```

解题思路：

其实贪心就可以。。。。
dp也行 f[i] = max(f[k] + 1) if(num[k]和num[i]满足要求)

代码dp：

```
class Solution {
public:
    int wiggleMaxLength(vector<int>& nums) {
        int len = nums.size();
        if(len == 0 || len == 1 || len == 2)
            return len;
        vector<int> v(len, 0);
        vector<int> f(len, 0);
        // vector<int> d(len, -1);
        f[0] = 1;
        for(int i = 1; i < len; i++) {
            f[i] = 1;
            for(int k = 0; k < i; k++) {
                int tmp = nums[i] - nums[k] > 0 ? 1 : -1;
                if(nums[i] - nums[k] != 0 && (v[k] + tmp == 0 || v[k] == 0)) {
                    f[i] = max(f[i], f[k]+1);
                    if(f[i] == f[k] + 1) {
                        v[i] = tmp;
                        // d[i] = k;
                    }
                }
            }
        }
        int res = f[0];
        int k = 0;
        for(int i = 1; i < len; i++) {
            res = max(res, f[i]);
            if(res == f[i]) {
                k = i;
            }
        }
        // cout << k << ":" << nums[k] << endl;
        // for(int i = k; i >= 0; ) {
        //     cout << d[i] << ":" << nums[d[i]] << endl;
        //     i = d[i];
        // }
        // cout << "********" << endl;
        return res;
    }
};
```

代码Greedy：

```
class Solution {
public:
    int wiggleMaxLength(vector<int>& nums) {
        int len = nums.size();
        if(len <= 2) {
            return len;
        }
        int flag = 0, count = 1;
        for(int i = 1; i < len; i++) {
            if(nums[i-1] > nums[i] && (flag == -1 || flag == 0)) {
                flag = 1; count++;
            }
            else if(nums[i-1] < nums[i] && (flag == 1 || flag == 0)) {
                flag = -1; count++;
            }
        }
        return count;
    }
};
```