Bumpy sequence

1. **Define the entries of your table in words. E.g. T(i) or T(i, j) is ...**
   T(i, 0) is the maximum length of a bumpy sequence for a[1]…a[i] that the last element has a positive difference with its previous element;
   T(i, 1) is the maximum length of a bumpy sequence for a[1]…a[i] that the last element has a negative difference with its previous element;

2. **State a recurrence for the entries of your table in terms of smaller subproblems. Don't forget your base case(s).**
   Base case:
   for i = 1, T(i, 0) = 1, T(i, 1) = 1

   Recurrence:
   for 2<=i<=n:
   T(i, 0) = T(i-1,1) + 1, T(i,1) = T(i-1,1), if a[i-1]<a[i]
   T(i, 1) = T(i-1,0) + 1, T(i,0) = T(i-1,0), if a[i-1]>a[i]
   T(i, 1) = T(i-1,1), T(i,0) = T(i-1,0), if a[i-1]=a[i]

3. **Write pseudocode for your algorithm to solve this problem.**

```
when i = 1:
    T(i, 0) = 1
    T(i, 1) = 1
for i = 2 to n:
    if a[i] – a[i-1] > 0:
        T(i, 0) = T(i-1, 1) +1
        T(i,1) = T(i-1,1)
    else if a[i] – a[i-1] < 0:
        T(i, 1) = T(i-1,0) + 1
        T(i,0) = T(i-1,0)
    else:
        T(i, 1) = T(i-1,1)
        T(i,0) = T(i-1,0)
return max{T(n, 0), T(n, 1)}
```

4. **State and analyze the running time of your algorithm.**
   when i = 1: O(1)
   for i = 2 to n: O(n)
   return max{T(n, 0), T(n, 1)}: O(2)

   The running time can be represented by: O(1) + O(n) + O(2) ➔ O(n)