2058. Find the Minimum and Maximum Number of Nodes Between Critical Points

Solved

Medium | Topics | Companies | Hint

A **critical point** in a linked list is defined as **either** a **local maxima** or a **local minima**.

A node is a **local maxima** if the current node has a value **strictly greater** than the previous node and the next node.

A node is a **local minima** if the current node has a value **strictly smaller** than the previous node and the next node.

Note that a node can only be a local maxima/minima if there exists **both** a previous node and a next node.

Given a linked list `head`, return *an array of length 2 containing* `[minDistance, maxDistance]` *where* `minDistance` *is the* ***minimum distance*** *between* ***any two distinct*** *critical points and* `maxDistance` *is the* ***maximum distance*** *between* ***any two distinct*** *critical points. If there are* ***fewer*** *than two critical points, return* `[-1, -1]`.

```
/**
* Definition for singly-linked list.
* public class ListNode {
*     public var val: Int
*     public var next: ListNode?
*     public init() { self.val = 0; self.next = nil; }
*     public init(_ val: Int) { self.val = val; self.next = nil; }
*     public init(_ val: Int, _ next: ListNode?) { self.val = val;
self.next = next; }
* }
*/
class Solution {
   func nodesBetweenCriticalPoints(_ head: ListNode?) -> [Int] {
       var pts:[Int] = []
       var curr = head
       var prev = curr
       curr = curr?.next
       var a = prev!.val
       var b = curr!.val
       var c = 0
       var index = 2
       while(curr?.next != nil) {
           c = curr!.next!.val
```

```
        if(a < b && b > c){
            // print("maxima")
            // print(a)
            // print(b)
            // print(c)
            pts.append(index)

        }

        if(a > b && b < c) {
            // print("minima")
            // print(a)
            // print(b)
            // print(c)
            pts.append(index)
        }

        prev = curr
        curr = curr?.next
        a = prev!.val
        b = curr!.val
        index = index + 1
    }

    if (pts.count < 2) {
        return [-1,-1]
    }

    var ans:[Int] = []
    var t = pts[1] - pts[0]

    for i in (0..<pts.count-1) {
        t = min(t,(pts[i+1]-pts[i]))
    }
    // print(pts)
    ans.append(t)
    ans.append(pts[pts.count-1] - pts[0])
    return ans
}}
```