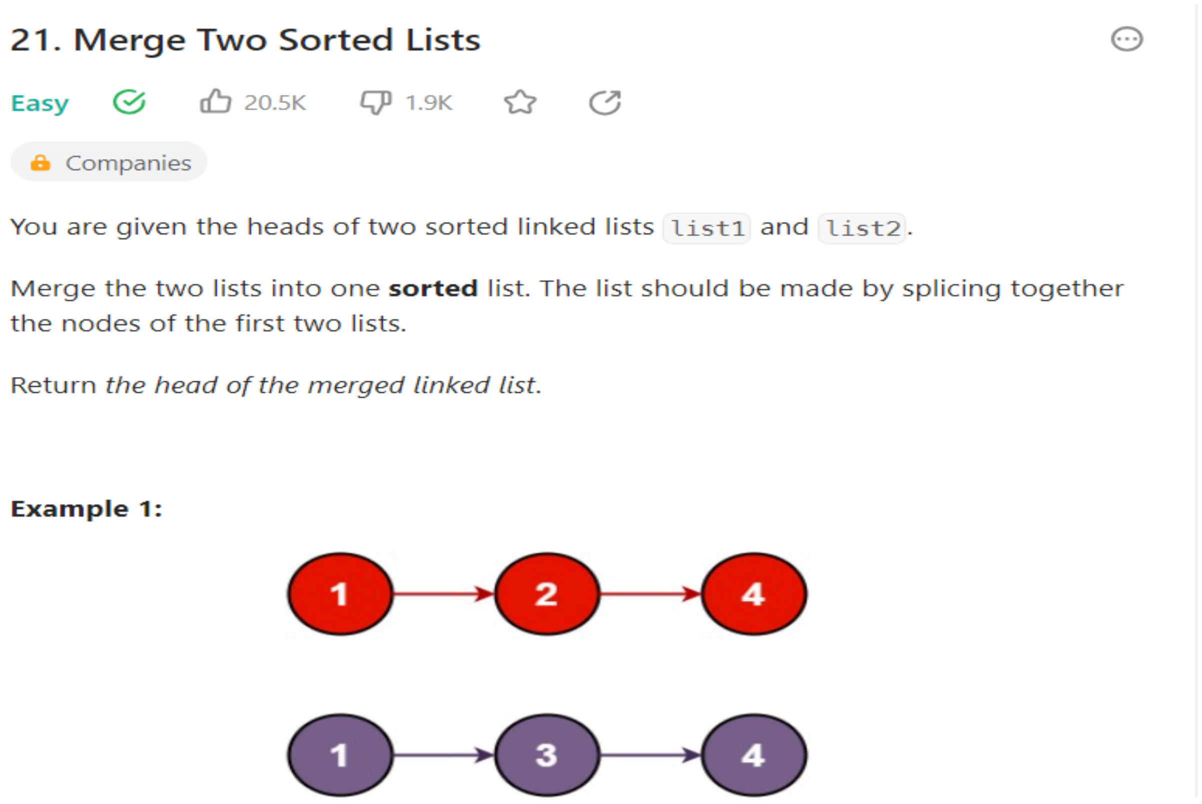

21. Merge Two Sorted Lists

Easy 20.5K 1.9K

Companies

You are given the heads of two sorted linked lists `list1` and `list2`.

Merge the two lists into one **sorted** list. The list should be made by splicing together the nodes of the first two lists.

Return *the head of the merged linked list.*

Example 1:

Code

```
/**
 * 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 {
// Recursive
        func mergeTwoLists(_ list1: ListNode?, _ list2:
ListNode?) -> ListNode? {
            if l1 == nil || l2 == nil { return l1 == nil ? l2 :
l1 }
```

```
        if l1!.val <= l2!.val {
            l1?.next = mergeTwoLists(l1?.next, l2)
            return l1
        } else {
            l2?.next = mergeTwoLists(l1, l2?.next)
            return l2
        }
        }

// Iterative
    func mergeTwoLists(_ list1: ListNode?, _ list2: ListNode?)
-> ListNode? {
        var list1 = list1
        var list2 = list2
        var prev = ListNode(0)
        let head = prev
        while(list1 != nil && list2 != nil) {
            if(list1!.val >= list2!.val) {
                let curr = ListNode(list2!.val)
                prev.next = curr
                prev = curr
                list2 = list2!.next
            } else {
                let curr = ListNode(list1!.val)
                prev.next = curr
                prev = curr
                list1 = list1!.next
            }
        }
        if(list1 != nil) {
            prev.next = list1
        } else {
            prev.next = list2
        }

        return head.next
    }}
```