21. Merge Two Sorted Lists

Easy 19.9K 1.8K

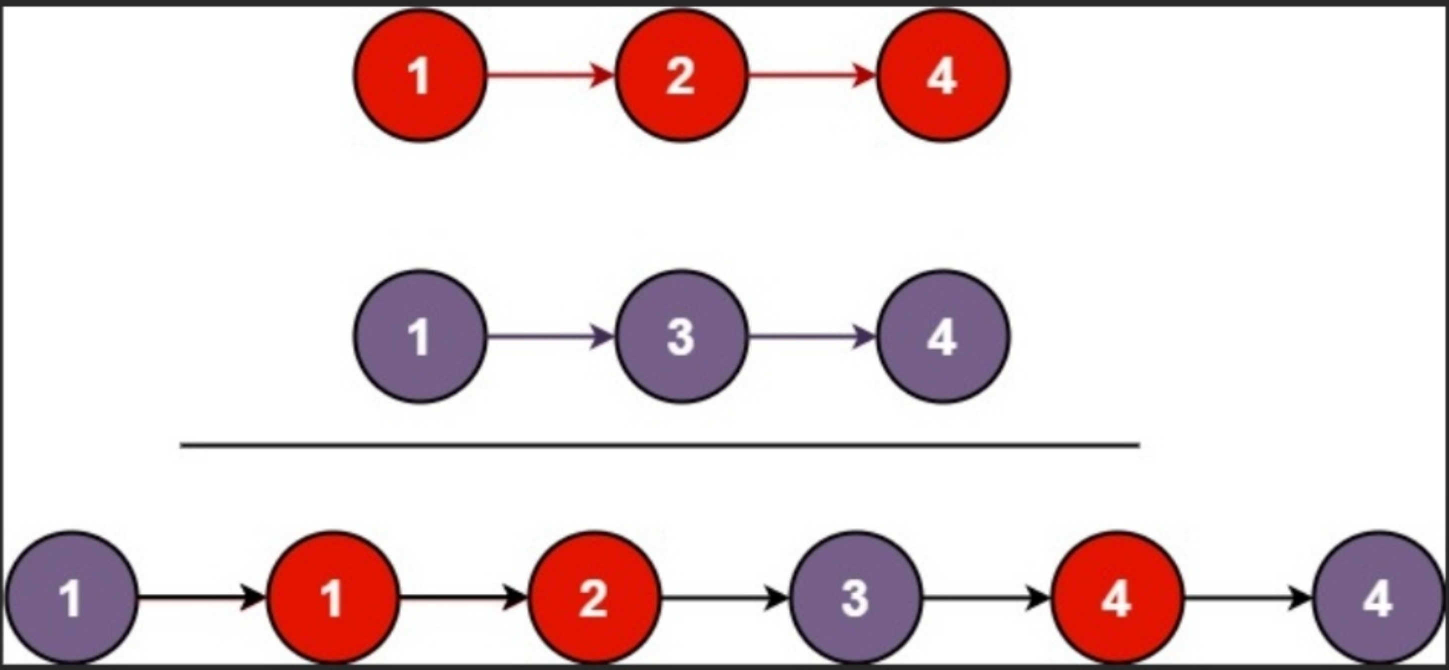
Companies

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:



```
Input: list1 = [1,2,4], list2 = [1,3,4]
Output: [1,1,2,3,4,4]
```

Example 2:

```
Input: list1 = [], list2 = []
Output: []
```

Example 3:

```
Input: list1 = [], list2 = [0]
Output: [0]
```

Constraints:

- The number of nodes in both lists is in the range `[0, 50]`.
- `-100 <= Node.val <= 100`
- Both `list1` and `list2` are sorted in **non-decreasing** order.

Accepted 3.5M | Submissions 5.5M | Acceptance Rate 63.1%

Approach 1: Creating another linked list of size m+n

→ Start iterating over both linked lists and compare their respective values and then create a node for minimum value and add it to ans. The move that respective linked list pointers from where we got the min. [Same as merge sort merging]

→ If either of the heads is null return the non null list.

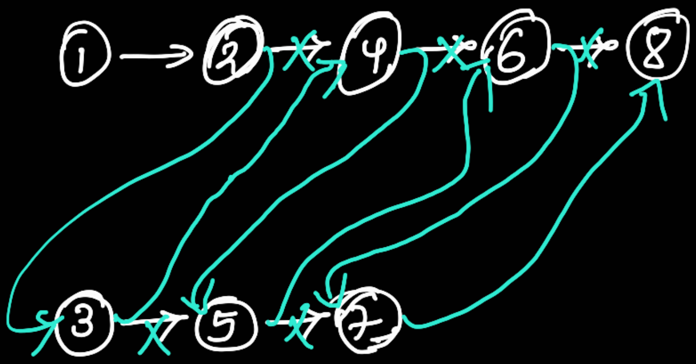
```
→ while( list1 && list 2)
  {
      create node for min among
              list1→val and list2→val
      if(ans == NULL)
              ans = node
      else
              ans→next = node

      if list1 val is min then list1 = list1→next
      else list2 = list2→next
  }
```

If one of the list got exhausted then add remaining elements nodes to ans directly.

T(n) : O(m+n)
S(n) : O(m+n)

Approach 2: By slicing the given linked lists



$T(n) : O(n+m)$

$S(n) : O(1)$