

■ Merge Two Sorted Lists

Weave two sorted chains into one — the merge step of merge sort

● Linked List

■ EASY

■ Intuition First

Two sorted piles of playing cards, face-up. Pick the smaller top card from either pile and add it to your result pile. Repeat until both piles are empty. The dummy head trick avoids special-casing the first node — it's a placeholder that makes the code clean.

```
list1: 1 → 2 → 4
list2: 1 → 3 → 4

dummy → ?
Compare 1 vs 1: take list1[1] → dummy→1
Compare 2 vs 1: take list2[1] → dummy→1→1
Compare 2 vs 3: take list1[2] → dummy→1→1→2
Compare 4 vs 3: take list2[3] → dummy→1→1→2→3
Compare 4 vs 4: take list1[4] → dummy→1→1→2→3→4
list1 empty, attach list2[4] → ...→4→4

Result: 1→1→2→3→4→4 ✓
```

■ Java Code

```
1  class Solution {
2  public ListNode mergeTwoLists(ListNode l1, ListNode l2) {
3      ListNode dummy = new ListNode(0); // Placeholder head
4      ListNode curr = dummy;
5
6      while (l1 != null && l2 != null) {
7          if (l1.val <= l2.val) {
8              curr.next = l1;
9              l1 = l1.next;
10         } else {
11             curr.next = l2;
12             l2 = l2.next;
13         }
14         curr = curr.next;
15     }
16     // Attach remaining nodes
17     curr.next = (l1 != null) ? l1 : l2;
18     return dummy.next; // Skip the dummy placeholder
19 }
```

■ Complexity

Type	Value	Meaning
Time	$O(m+n)$	Visit every node in both lists exactly once
Space	$O(1)$	Only pointers used — no new nodes created

■ Dummy Head Trick

The dummy node eliminates the need to special-case 'first node insertion'. Always start merge/construction problems with a dummy head — it simplifies the code dramatically.

■ Quick Revision

✓	Create dummy head node as placeholder
✓	Compare heads of both lists, take the smaller
✓	Advance the pointer from the chosen list
✓	Attach remaining non-empty list at end
✓	Return dummy.next (skip placeholder)