## 6 → Merge Two Sorted Lists -

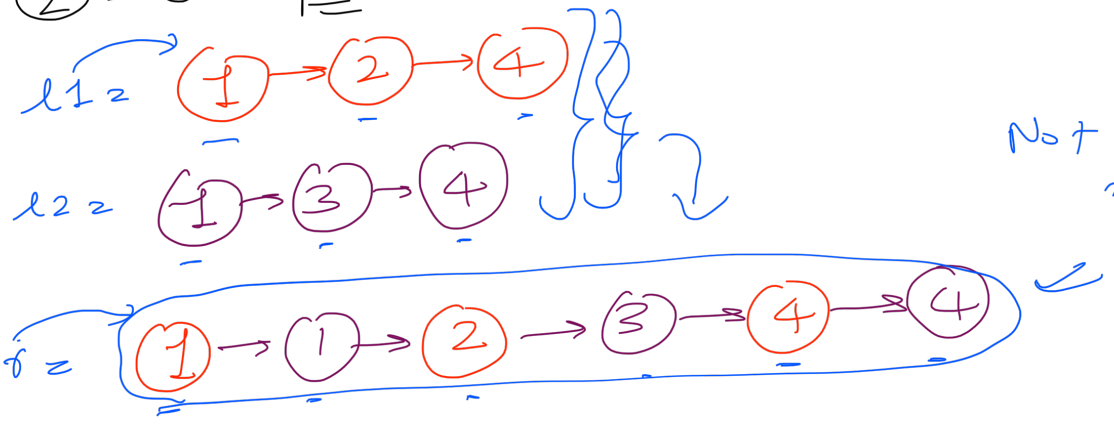
### 1 → Question

Given the heads of two sorted linked lists **list1** and **list2**

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

Return the head of merged linked list.

### 2 → Examples -




list1 = [1, 2, 4]

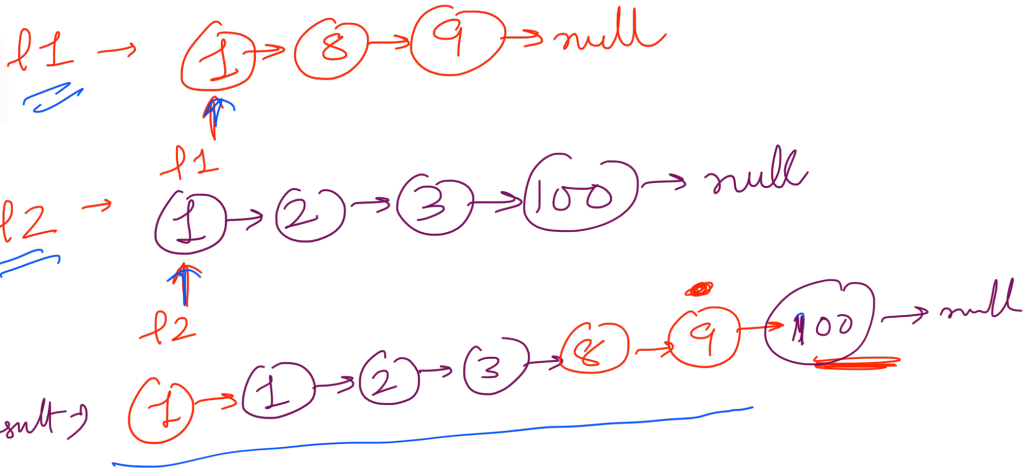
list2 = [1, 3, 4]

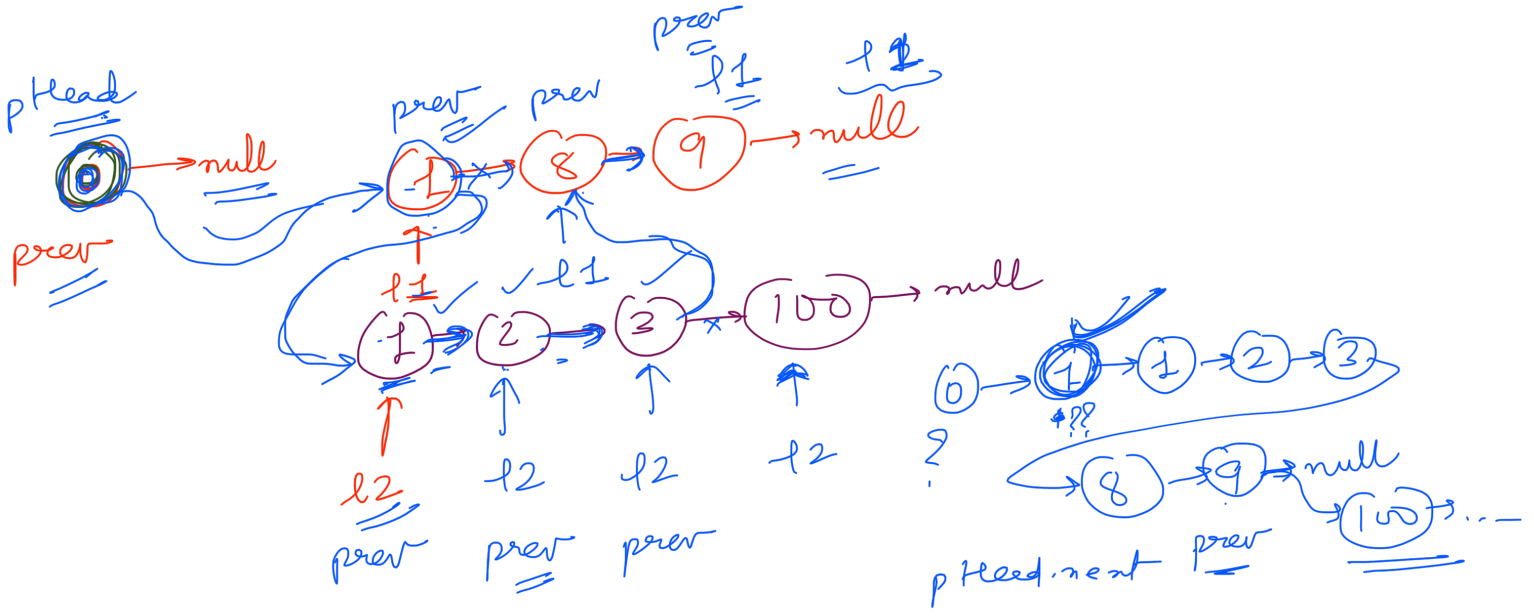
→ [1, 1, 2, 3, 4, 4]

### 3 → Intuition ? -

$$lR = \begin{cases} l1[0] + \langle l1[1\ldots], l2 \rangle & \text{if } l1[0] <= l2[0] \\ l2[0] + \langle l1, l2[1\ldots] \rangle & \text{else} \end{cases}$$

### 4 → Optimized Solution -



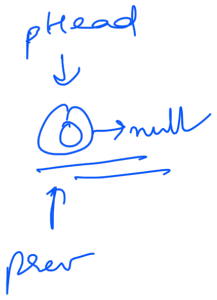

(5) → Time & space complexity-

$T = O(len(l1) + len(l2))$ $O(l(l1)+l(l2))$

$S = O(1)$

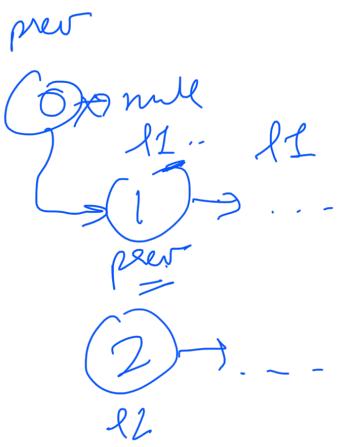
(6) → Code walkthrough -

```
ListNode mergeTwoLists(ListNode l1, ListNode l2) {
    LN pHead = new LN(0);
    LN prev = pHead;
    while (l1 != null && l2 != null) {
```

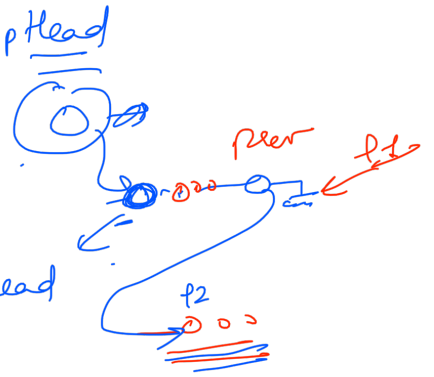



```
class LN {
    int val;
    LN next;
    LN() {}
    LN(val) {
        this.val = val;
    }
    LN(val, next) {
```

this.val = val;
this.next = next;




```
if( l1.val <=
    l2.val ) {
    prev.next = l1;
    l1 = l1.next;
} else {
    prev.next = l2;
    l2 = l2.next;
}
prev = prev.next;
}
if (l1 == null ) {
    prev.next = l2;
} else {
    prev.next = l1;
}
return pHead.next;
}
```




head == pHead.next