**难度: 简单**

输入两个递增排序的链表，合并这两个链表并使新链表中的节点仍然是递增排序的。

示例1：

输入：1->2->4, 1->3->4

输出：1->1->2->3->4->4

限制：

0 <= 链表长度 <= 1000

来源：力扣（LeetCode）

链接：https://leetcode-cn.com/problems/he-bing-liang-ge-pai-xu-de-lian-biao-lcof

著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。

**思路1:**

1). 先判断两个链表是否其中一个为空,如果是直接返回其中一个.

2). 先把两个链表中,小的值给到合并节点

3). 循环两个链表进行节点合并

4). 在判断l1或者l2是否有不为空的,再进行遍历合并

|  |
| --- |
| public ListNode mergeTwoLists(ListNode l1, ListNode l2) {  //判断链表中,是否有一个为空  if (l1 == null || l2 == null) {  return l1 == null ? l2 : l1;  }  //创建合并节点  ListNode merge;  //比较头个节点的大小  //小就给到合并节点,并且指向下一个节点  if (l1.val < l2.val) {  merge = new ListNode(l1.val);  l1 = l1.next;  }else {  merge = new ListNode(l2.val);  l2 = l2.next;  }  //临时节点,用来合并  ListNode temp = merge;    while(l1 != null && l2 != null) {  //判断大小  if (l1.val < l2.val) {  temp.next = l1;  l1 = l1.next;  }else {  temp.next = l2;  l2 = l2.next;  }    //走向下个节点  temp = temp.next;  }  //防止l1或者l2其中一个先为空  while(l1 != null) {  temp.next = l1;  temp = temp.next;  l1 = l1.next;  }    while(l2 != null) {  temp.next = l2;  temp = temp.next;  l2 = l2.next;  }  return merge;  } |