# [排序链表](https://leetcode-cn.com/explore/interview/card/bytedance/244/linked-list-and-tree/1040/)

**头条重点**

## 题目

在 O(n log n) 时间复杂度和常数级空间复杂度下，对链表进行排序。

示例 1:  
  
输入: 4->2->1->3  
输出: 1->2->3->4  
示例 2:  
  
输入: -1->5->3->4->0  
输出: -1->0->3->4->5

## 解题思路

1. 通过快慢指针将链表拆分
2. 递归进行拆分，再通过合并两个排序链表的方式进行合并
3. 类似于归并排序

public ListNode sortList(ListNode head) {  
 if (head == null || head.next == null) {  
 return head;  
 }  
  
 ListNode slow = head, fast = head;  
 while (fast.next != null && fast.next.next != null) {  
 fast = fast.next.next;  
 slow = slow.next;  
 }  
  
 ListNode mid = slow.next;  
 slow.next = null;  
  
 ListNode l1 = sortList(head);  
 ListNode l2 = sortList(mid);  
  
 return merge(l1, l2);  
}  
  
private ListNode merge(ListNode l1, ListNode l2) {  
 if (l1 == null) {  
 return l2;  
 }  
  
 if (l2 == null) {  
 return l1;  
 }  
  
 ListNode head,res;  
 if (l1.val > l2.val) {  
 head = l2;  
 l2 = l2.next;  
 } else {  
 head = l1;  
 l1 = l1.next;  
 }  
 res = head;  
// head.next = null;  
  
 while (l1 != null || l2 != null) {  
 if (l1 == null) {  
 head.next = l2;  
 l2 = l2.next;  
 } else if (l2 == null) {  
 head.next = l1;  
 l1 = l1.next;  
 } else {  
 if (l1.val > l2.val) {  
 head.next = l2;  
 l2 = l2.next;  
 } else {  
 head.next = l1;  
 l1 = l1.next;  
 }  
 }  
 head = head.next;  
 }  
  
 return res;  
}