这道题与L147的区别是这个只能用O(nlog(n))的时间复杂度。所以本题考虑用归并排序的方法。

148. Sort List

Description | Hints | Submissions | Discuss | Solution

Pick One

Sort a linked list in *O*(*n* log *n*) time using constant space complexity.

Example 1:

```
Input: 4->2->1->3
Output: 1->2->3->4
```

Example 2:

```
Input: -1->5->3->4->0
Output: -1->0->3->4->5
```

```java
public class L148 {

    class ListNode {
        int val;
        ListNode next;

        ListNode(int x) {
           val = x;
        }
    }

    public ListNode sortList(ListNode head) {
        if(head == null || head.next == null)
            return head;
        //pre的目的是获得前半部分，然后最后一个元素置空，slow是获得后半部分的第一个元素
        ListNode slow = head, fast = head, pre = head;
        while (fast != null && fast.next != null) {
            pre = slow;
            slow = slow.next;
            fast = fast.next.next;
        }
        pre.next = null;
        return merge(sortList(head), sortList(slow));
    }

    public ListNode merge(ListNode l1, ListNode l2) {
        if (l1 == null) {
            return l2;
        }
        if(l2 == null)
            return l1;
        if(l1.val < l2.val) {
            l1.next = merge(l1.next, l2);
            return l1;
        }else {
            l2.next = merge(l1, l2.next);
            return l2;
        }
    }

}
```

merge.

~~slow~~ head　　slow

[1] → [3]　　[2] → [4]

L_1　　L_2

1.

[1] < [2]

⇒ [1]

$L_1.next = merge(L_1.next, L_2)$

2. 같다 [3] > [2]

$L_1.next = L_2$

[1] → [2]　　$L_2.next = merge(L_1, L_2.$

3. [3] < [4]

[1] → [2] → [3] → [4]

☆.