[091]148.Sort List

```
- Date:2020-12-19(Sat)
- medium
- Related topic:linked list
```

problem link:https://leetcode.com/problems/sort-list/

> Given the head of a linked list, return the list after sorting it in ascending order.

Follow up:

- Can you sort the linked list in O(n logn) time and O(1) memory (i.e. constant space)?

Exmaple1:

- Input: head = [4,2,1,3]
- Output: [1,2,3,4]

Example2:

- Input: head = [-1,5,3,4,0]
- Output: [-1,0,3,4,5]

Example 3:

- Input: head = []
- Output: []

Constraints:

- The number of nodes in the list is in the range [0, 5 * 10^4].
- -105 <= Node.val <= 105

Think process:

- 這題本質是divid and conquer，把Linked List分成兩階段，
  - 第一是分割，4->2->1，直到最小單元單個ListNode，
  - 第二就是“組合”，merge(ListNode left, ListNode)，左右各1合併為2，左右各2合併為4

code

```
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next;
 * }
 */
class Solution {
    public ListNode sortList(ListNode head) {
        if(head==null || head.next==null) return head;
        ListNode mid = getMid(head);
        ListNode secondHalfFirst = mid.next;
        mid.next = null;
        ListNode l = sortList(head);
        ListNode r = sortList(secondHalfFirst);
        return merge(l,r);
    }

    public ListNode merge(ListNode l1, ListNode l2){
        ListNode dummy = new ListNode(0);
        ListNode curMerge = dummy;
        while(l1!=null && l2!=null){
            if(l1.val>l2.val){
                curMerge.next = l2;
                l2 = l2.next;
                curMerge = curMerge.next;
            }
            else{
                curMerge.next = l1;
                l1 = l1.next;
                curMerge = curMerge.next;
            }
        }
        if(l1!=null){
            curMerge.next = l1;
            //l1 = l1.next;
            //curMerge = curMerge.next;
```

```
        }
        if(l2!=null){
            curMerge.next = l2;
        }
        return dummy.next;
    }

    public ListNode getMid(ListNode head){
        ListNode slow = head;
        ListNode fast = slow;
        while(fast.next !=null && fast.next.next!=null){
            slow = slow.next;
            fast = fast.next.next;
        }
        return slow;
    }
}
```

這裡的merge跟[027]21. Merge Two Sorted Lists不同，後者比較複雜，因為listNode兩邊的長度不一，而本題就是兩個相同長度1:1, 1:0, 0:1，三種可能性，另外指標也會簡化。

- Runtime: 5 ms, faster than 98.74% of Java online submissions for Sort List.
- Memory Usage: 47.5 MB, less than 42.07% of Java online submissions for Sort List.

Not meet follow-up

- Time:O(n*logn)
- Space:O(logn) rather than O(1)

Next challenges:

Insertion Sort List