[079]206. Reverse Linked List I

- Date:2020-12-07(Mon)
- easy
- Related topic:linkedList

> Reverse a singly linked list.

Example:

- Input: 1->2->3->4->5->NULL
- Output: 5->4->3->2->1->NULL

Follow up:

A linked list can be reversed either iteratively or recursively. Could you implement both?

Think process:

這題是linkedList的基本題，後面會有很多相關延伸， 關鍵套路在於，用pre, cur, temp三個pointer做listNode的置換，總共有四個step，前兩個就是移動單次的箭頭指向，後兩個是做指標的歸位，以便下次再做一次。

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 reverseList(ListNode head) {
        if(head==null || head.next ==null) return head;
        ListNode pre = null;
        ListNode cur = head;
        while(cur!=null){
            ListNode nextTemp = cur.next; //mark temp pointer to 2
            cur.next = pre; // from 1->2 to "1->null"
            pre = cur; // mark pre pointer to (1->2)
            cur = nextTemp; //
        }
        return pre;
    }

}
```

Complexity analysis

- Time complexity : O(n). Assume that nn is the list's length, the time complexity is O(n).

- Space complexity : O(1)

- Runtime: 0 ms, faster than 100.00% of Java online submissions for Reverse Linked List.

- Memory Usage: 38.2 MB, less than 99.45% of Java online submissions for Reverse Linked List.

Next challenges:

Binary Tree Upside Down