[19. 删除链表的倒数第N个节点](https://leetcode-cn.com/problems/remove-nth-node-from-end-of-list/)

难度中等797收藏分享切换为英文关注反馈

给定一个链表，删除链表的倒数第 *n*个节点，并且返回链表的头结点。

**示例：**

给定一个链表: **1->2->3->4->5**, 和 ***n* = 2**.

当删除了倒数第二个节点后，链表变为 **1->2->3->5**.

**说明：**

给定的 *n* 保证是有效的。

**进阶：**

你能尝试使用一趟扫描实现吗？

我通过的：

/\*\*

 \* Definition for singly-linked list.

 \* public class ListNode {

 \*     public int val;

 \*     public ListNode next;

 \*     public ListNode(int x) { val = x; }

 \* }

 \*/

public class Solution {

    public ListNode RemoveNthFromEnd(ListNode head, int n) {

        // 声明快慢指针

        ListNode fast = head, slow = head;

        // fast指针走n步

        while(n > 0){

            fast = fast.next;

            n--;

        }

        // 判断需要删除的节点是否为第一个节点

        if(fast == null){

            return head.next;

        }

        // 同时移动快慢指针

        while(fast.next != null){

            fast = fast.next;

            slow = slow.next;

        }

        // 删除节点

        slow.next = slow.next.next;

        return head;

    }

}

参考：<https://www.jianshu.com/p/be3fe87f099f>

/\*\*

\* Definition for singly-linked list.

\* public class ListNode {

\* int val;

\* ListNode next;

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

\* }

\*/

class Solution {

public ListNode removeNthFromEnd(ListNode head, int n) {

// 声明快慢指针

ListNode fast = head, slow = head;

// fast指针走n步

while(n > 0){

fast = fast.next;

n--;

}

// 判断需要删除的节点是否为第一个节点

if(fast == null){

return head.next;

}

// 同时移动快慢指针

while(fast.next != null){

fast = fast.next;

slow = slow.next;

}

// 删除节点

slow.next = slow.next.next;

return head;

}

}