1. Remove Nth Node From End of List

Medium

Given a linked list, remove the *n*-th node from the end of list and return its head.

**Example:**

Given linked list: 1->2->3->4->5, and n = 2.  
  
After removing the second node from the end, the linked list becomes 1->2->3->5.

**Note:**

Given *n* will always be valid.

**Follow up:**

Could you do this in one pass?

**解法1**

移除倒数第个节点和移除第个节点的效果一样，为此先求出链表长度，再移除节点

/\*\*  
 \* Definition for singly-linked list.  
 \* struct ListNode {  
 \* int val;  
 \* ListNode \*next;  
 \* ListNode(int x) : val(x), next(NULL) {}  
 \* };  
 \*/  
class Solution {  
public:  
 ListNode\* removeNthFromEnd(ListNode\* head, int n) {  
 ListNode \*dummy = new ListNode(-1);  
 dummy->next = head;  
 ListNode \*p = head;  
 int len = 0;  
 while(p != NULL){  
 len++;  
 p = p->next;  
 }  
 len = len - n;  
 p = dummy;  
 while(len){  
 p = p->next;  
 len--;  
 }  
 p->next = p->next->next;  
 return dummy->next;  
 }  
};

技巧：先设置一个虚拟的头结点，统一各种操作

**解法2**

双指针扫描，两个指针的间隔为

class Solution {  
public:  
 ListNode\* removeNthFromEnd(ListNode\* head, int n) {  
 ListNode \*p = head, \*pre = NULL, \*q = head;  
 int cnt = 0;  
 while(cnt < n){  
 p = p->next;  
 cnt++;  
 }  
 while(p != NULL){  
 pre = q;  
 q = q->next;  
 p = p->next;  
 }  
 if(q == head)head = q->next;  
 else if(q->next != NULL)pre->next = q->next;  
 else pre->next = NULL;  
 return head;  
 }  
};