# 题目

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？

# 分析

/\*\*

\* 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(INT\_MAX); //哑结点

dummy->next = head;

ListNode \*p = dummy;

ListNode \*q = dummy; //快慢指针

ListNode \*tmp;

for(int i=0;i<n;i++)

{

p = p->next;

}

while(p->next)

{

p = p->next;

q = q->next;

}

tmp = q->next;

q->next = tmp->next;

delete tmp;

return dummy->next;

}

};

**复杂度分析：**

时间复杂度：O(L)，其中 L 是链表的长度。

空间复杂度：O(1)。

或：

/\*\*

\* 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) {

if(!head || !head->next)

return NULL;

ListNode \*p = new ListNode(0);

p->next = head;

ListNode \*f = p, \*s = p;

for(int i = 0; i <= n; i++)

f = f->next;

while(f)

{

f = f->next;

s = s->next;

}

s->next = s->next->next;

return p->next;

}

};