是刷过的题耶! 你还知道这道题如何解吗？

**描述**

给定一个链表，删除链表中倒数第n个节点，返回链表的头节点。

Example 1:

Input: list = 1->2->3->4->5->null， n = 2

Output: 1->2->3->5->null

Example 2:

Input: list = 5->4->3->2->1->null, n = 2

Output: 5->4->3->1->null

这道题因为规定了链表节点个数大于等于n，所以就不用考虑越界问题了。然后这道题和之前倒数第n个节点很类似啊，相对来说也不是很难吧。这里我就没用先遍历链表长度然后再删除的方法了。我直接用快慢指针的方法了。

因为如果是链表节点个数直接到n的话，有点不好操作，因为快指针要迭代n+1次，然后的话就可能跌倒到n次时已经是末尾了，这个时候我再迭代1次的话编译器就直接报错，那问题就很大了。

解决办法很简单，和我之前写的删除链表元素那道题的解决方法一样，直接加一个哑节点，这样直接迭代n+1次，也只会迭代到末尾。

具体方法就是快指针先走n+1次。在这之后慢指针再和快指针出发，到快指针达到链表的末尾时，正好慢指针就走到了快指针的倒数第n-1个节点了。

另外，加了一个哑节点不会影响快慢指针的，毕竟快慢指针说到底就是个相对距离的问题。加一个哑节点就是为了解决边界情况。

代码如下：

ListNode \* removeNthFromEnd(ListNode \* head, int n) {

// write your code here

ListNode\* dummyHead = new ListNode(0);

dummyHead->next = head;

ListNode\* p = dummyHead;

ListNode\* q = dummyHead;

for( int i = 0 ; i < n + 1 ; i ++ ){

q = q->next;

}

while(q){

p = p->next;

q = q->next;

}

ListNode\* delNode = p->next;

p->next = delNode->next;

delete delNode;

return dummyHead->next;

}

通过测试如下：

