/\*\*

 \* Definition for singly-linked list.

 \* struct ListNode {

 \*     int val;

 \*     struct ListNode \*next;

 \* };

 \*/

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

struct ListNode\* index=head;//取得index現在指標位置

struct ListNode\* pre\_index=index;//取得index前位置

struct ListNode\* next\_index=head->next;//取得index下一個

struct ListNode\* pre\_rm\_index=NULL;//取得rm\_index前一個，先定義為NULL

struct ListNode\* rm\_index=head;//要刪除的node位置

struct ListNode\* next\_rm\_index=rm\_index->next;//要刪除的node的下一個

int count = 1;//計算倒數第幾個

    while(1){

         //printf("== NULL index==%d rm\_index=%d \n",index->val,rm\_index->val);

        if( next\_index == NULL){//當next\_index找到NULL的時候停止迴圈

            //printf("cc== NULL index==%d rm\_index=%d index=%d \n",index->val,rm\_index->val,index->val);

            if( pre\_rm\_index == NULL && next\_rm\_index==NULL ){//如果rm\_index的前後都是NULL就將head給pre\_rm\_index=NULL

                head=pre\_rm\_index;

            }else if( pre\_rm\_index == NULL ){//如只有pre\_rm\_index是NULL就將head給next\_rm\_index的節點

                head=next\_rm\_index;

            }else{

                pre\_rm\_index->next=next\_rm\_index;//一般情況下就將rm\_index的前後相接即可

            }

            break;

        }

        if( count >= n){//當count計算到與n相同大小的時候rm\_index與index都往下找next

            pre\_rm\_index=rm\_index;//rm\_index前中後三個指標往下找

            rm\_index=rm\_index->next;

            next\_rm\_index=rm\_index->next;

            pre\_index=index;//index前中後三個指標往下找

            index=index->next;

            next\_index=index->next;

        }else{//當count還沒有等於n的時候先計算index前中後三個指標往下找

            pre\_index=index;

            index=index->next;

            next\_index=index->next;

        }

        count++;

    }

    return head;

}