

019. Remove Nth Node From End of List

019 Remove Nth Node From End of List

- Linked List+Two Pointers

Description

Given a linked list, remove the n^{th} node from the end of list and return its head.

For 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.

Try to do this in one pass.

1. Thought Line

2. Linked List+Two Pointers

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     ListNode *next;
6  *     ListNode(int x) : val(x), next(NULL) {}
7  * };
8  */
9 class Solution {
10 public:
11     ListNode* removeNthFromEnd(ListNode* head, int n) {
12         int sizeList = 0;
13
14         // calculate the total size of list
15         ListNode* node = head;
16         while(node!=nullptr){
17             ++sizeList;
18             node = node->next;
19         }
20         if(n<1 || n>sizeList) return head;
21
22         // find the node at N+1 from end (sizeList - N from head)
23         ListNode* dummyHead = new ListNode(0);
24         dummyHead->next = head;
25         int count = 0;
26         ListNode* findTheNodeBeforeDeleteNode = dummyHead;
27         ListNode* findTheNodeOfDeleteNode = head;
28         while(count<sizeList - n){
29             findTheNodeBeforeDeleteNode = findTheNodeBeforeDeleteNode->next;
30             findTheNodeOfDeleteNode = findTheNodeOfDeleteNode->next;
31             ++count;
32         }
33         findTheNodeBeforeDeleteNode->next = findTheNodeOfDeleteNode->next;
34         findTheNodeOfDeleteNode->next = nullptr;
35         return dummyHead->next;
36     }
37 };
```

