

092. Reverse Linked List II

092 Reverse Linked List II

- Linked List

Description

Reverse a linked list from position m to n . Do it in-place and in one-pass.

For example:

Given `1->2->3->4->5->NULL`, $m = 2$ and $n = 4$,

return `1->4->3->2->5->NULL`.

Note:

Given m, n satisfy the following condition:

$1 \leq m \leq n \leq \text{length of list}$.

1. Thought line

2. Linked List

```
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* reverseBetween(ListNode* head, int m, int n) {
12         ListNode* dummyHead = new ListNode(0);
13         dummyHead->next = head;
14         unsigned int size = 0;
15         ListNode* ptr = dummyHead->next;
16         ListNode* ptr_st = dummyHead;
17         ListNode* ptr_ed = dummyHead->next;
18
19         while(ptr!=nullptr){
20             ++size;
21             ListNode* ptr_next = ptr->next;
22             if(size<=m){
23                 ptr_st = (size<m)?ptr_st->next:ptr_st;
24                 ptr_ed = (size<m)?ptr_ed->next:ptr_ed;
25             }
26             else if (size>m && size<=n){
27                 ListNode* ptr_st_next = ptr_st->next;
28                 ptr_st->next = new ListNode(ptr->val);
29                 ptr_st->next->next = ptr_st_next;
30                 ptr_ed->next = ptr->next;
31             }else{
32                 break;
33             }
34             ptr = ptr_next;
35         }
36         return dummyHead->next;
37     }
38 };
```

