1. Reverse Linked List II

Medium

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

**Note:** 1 ≤ *m* ≤ *n* ≤ length of list.

**Example:**

Input: 1->2->3->4->5->NULL, m = 2, n = 4  
Output: 1->4->3->2->5->NULL

**Solution**

三指针法反转链表。直接用地址比较作为截至条件会出错

/\*\*  
 \* Definition for singly-linked list.  
 \* struct ListNode {  
 \* int val;  
 \* ListNode \*next;  
 \* ListNode(int x) : val(x), next(NULL) {}  
 \* };  
 \*/  
class Solution {  
public:  
 ListNode\* reverseBetween(ListNode\* head, int m, int n) {  
 ListNode \*tmpHead = new ListNode(0);  
 tmpHead->next = head;  
 ListNode \*p, \*p\_pre, \*q, \*q\_next, \*cur = tmpHead;  
 int i = 0;  
 while(i <= n){  
 if(i == m - 1){  
 p = cur->next;  
 p\_pre = cur;  
 }  
 if(i == n){  
 q = cur;  
 q\_next = cur->next;  
 }  
 i++;  
 cur = cur->next;  
 }  
 p\_pre->next = q;  
 ListNode \*pre = q\_next, \*curr = p, \*post;  
 int pos = m;  
 while(pos <= n){  
 //printf("%x %x\n", curr, curr->next);  
 post = curr->next;  
 curr->next = pre;  
 pre = curr;  
 curr = post;  
 pos++;  
// if(curr == q->next)break;  
 }  
 return tmpHead->next;  
 }  
};