

Reverse Nodes in k-Group

Given a linked list, reverse the nodes of a linked list k at a time and return its modified list. k is a positive integer and is less than or equal to the length of the linked list. If the number of nodes is not a multiple of k then left-out nodes in the end should remain as it is.

Example:

Given this linked list: 1->2->3->4->5

For k = 2, you should return: 2->1->4->3->5

For k = 3, you should return: 3->2->1->4->5

Note:

Only constant extra memory is allowed.

You may not alter the values in the list's nodes, only nodes itself may be changed.

Solution in C++

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode(int x) : val(x), next(NULL) {}
 * };
 */
class Solution {
public:
    ListNode* reverseKGroup(ListNode* head, int k) {
        vector<ListNode*> p(k);
        ListNode *h=head;

        int i=0;
        while(i<k) {
            if (!h) return head;
            p[i++]=h;
            h=h->next;
        }

        i--;
        while(i>0) {
            p[i]->next=p[i-1];
            i--;
        }
        p[0]->next=(h ? reverseKGroup(h, k): NULL);

        return p[k-1];
    }
};
```