

25. Reverse Nodes in k-Group

[My Submissions](#)

QuestionEditorial Solution

Total Accepted: **62870** Total Submissions: **224326** Difficulty: **Hard**

Given a linked list, reverse the nodes of a linked list k at a time and return its modified list.

If the number of nodes is not a multiple of k then left-out nodes in the end should remain as it is.

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

Only constant memory is allowed.

For 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

[Subscribe](#) to see which companies asked this question

```
/**
 * 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) {
        if( head == NULL || head->next == NULL || k<2) return head;
        ListNode dummy(-1);
        dummy.next = head;
        ListNode *prev = &dummy;
        ListNode *end = head;
        while(end)
        {
            for(int i=1;i<k && end;i++)
            {
                end = end->next;
            }
            if(end == NULL) break;
            prev = reverse(prev,prev->next,end);
            end = prev->next;
        }
        return dummy.next;
    }

    ListNode *reverse(ListNode *prev,ListNode *begin, ListNode *end)
    {
        ListNode *end_next = end->next;
```

```
ListNode *p = begin;
ListNode *cur = p->next;
ListNode *post = cur->next;
while(cur!=end_next)
{
    post = cur->next;
    cur->next = p;
    p = cur;
    cur = post;
}

begin->next = end_next;
prev->next = end;
return begin;
}
};
```