Reverse Linked List in groups of Size K

In this article we will solve a very popular question Reverse Linked List in groups of Size K.

Problem Statement: Given the head of a linked list, reverse the nodes of the list k at a time, and return *the 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.

Examples:

Example 1:

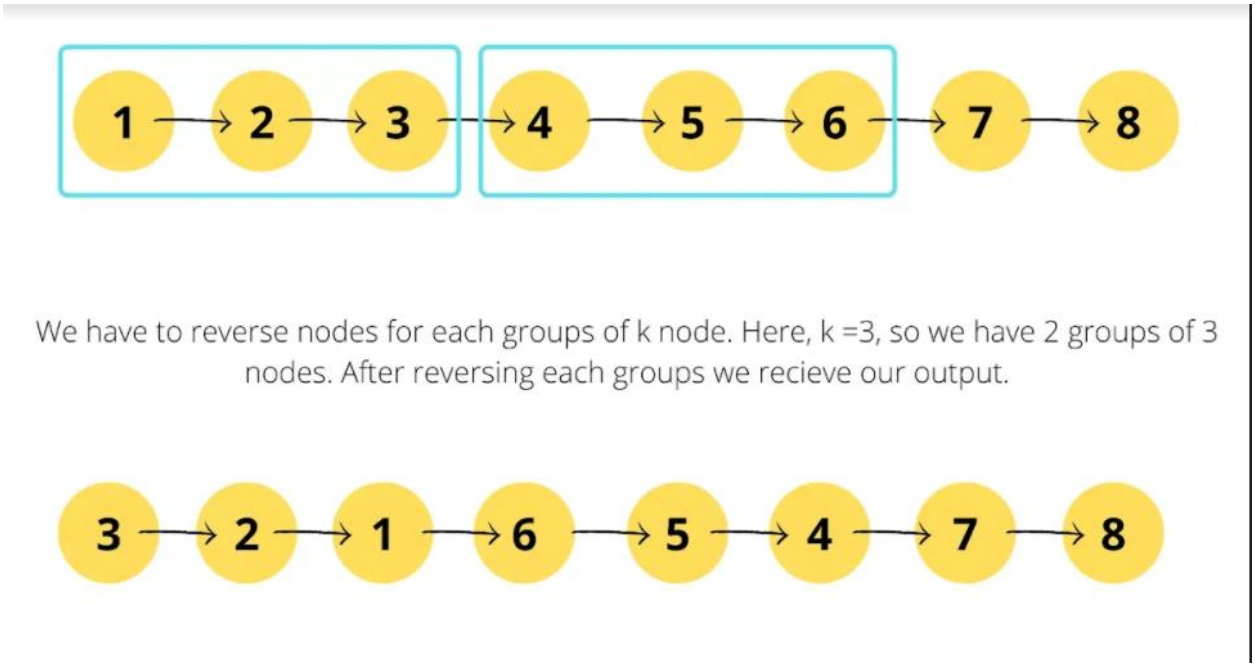
Input:

```
head = [1,2,3,4,5,6,7,8] k=3
```

Output:

```
head = [3,2,1,6,5,4,7,8]
```

Explanation:



Example 2:

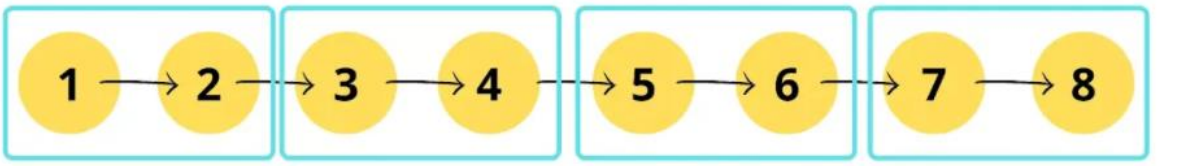
Input:

```
head = [1,2,3,4,5,6,7,8] k=2
```

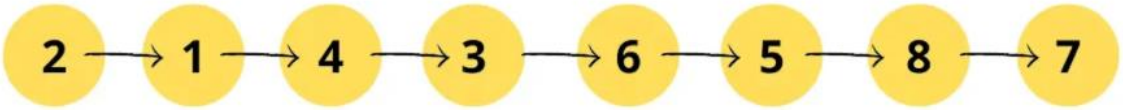
Output:

```
head = [2,1,4,3,6,5,8,7]
```

Explanation:



We have to reverse nodes for each groups of k node. Here, k =2, so we have 3 groups of 2 nodes. After reversing each groups we recieve our output.



Solution:

Approach:

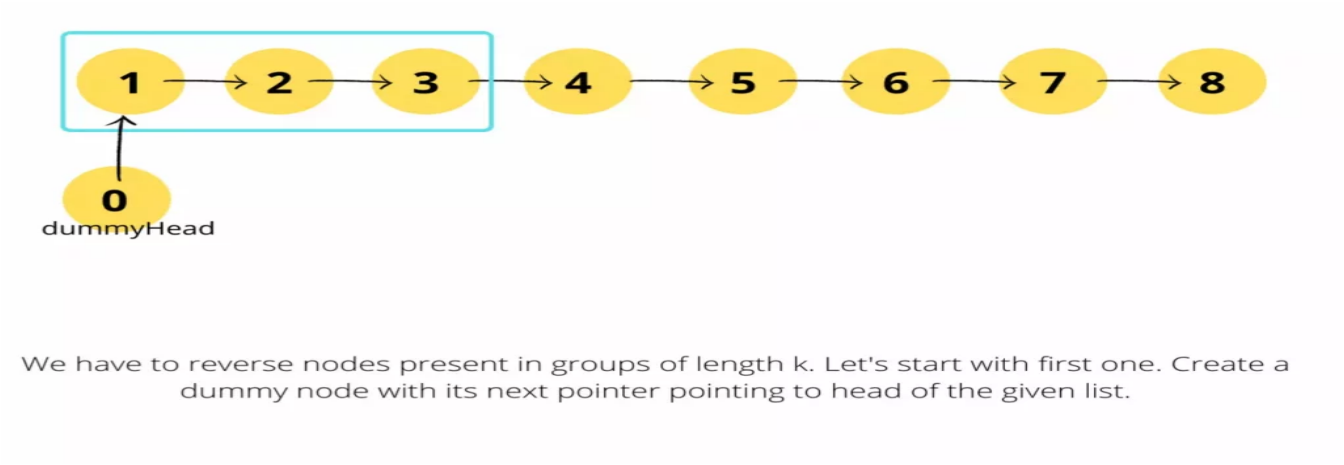
The following steps are needed to arrive at the desired output.

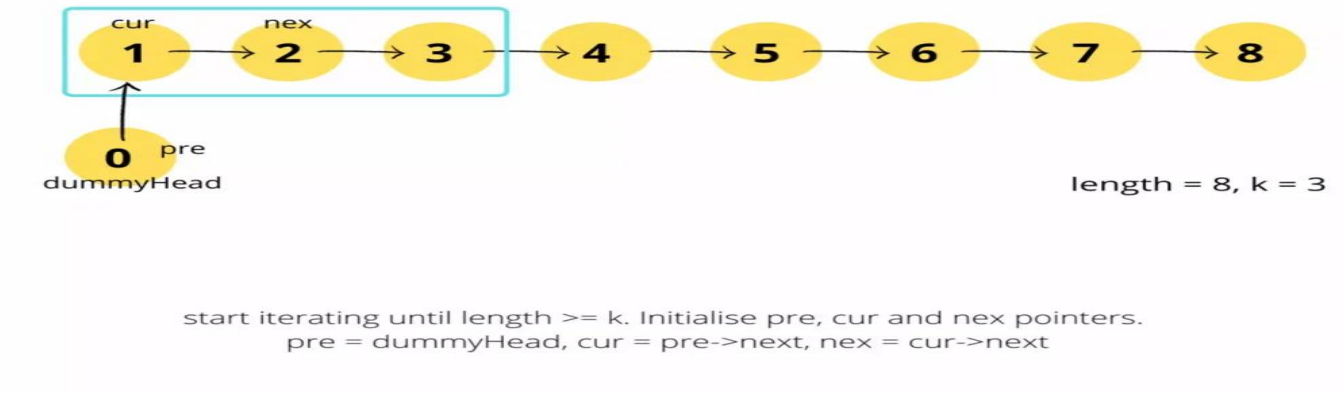
- Create a dummy node. Point next to this node to head of the linked list provided.
- Iterate through the given linked list to get the length of the list.
- Create three pointer pre,cur and nex to reverse each group. Why? If we observe output, we can see that we have to reverse each group, apart from modifying links of group.
- Iterate through the linked list until the length of list to be processed is greater than and equal to given k.
- For each iteration, point cur to pre->next and nex to nex->next.
- Start nested iteration for length of k.
- For each iteration, modify links as following steps:-

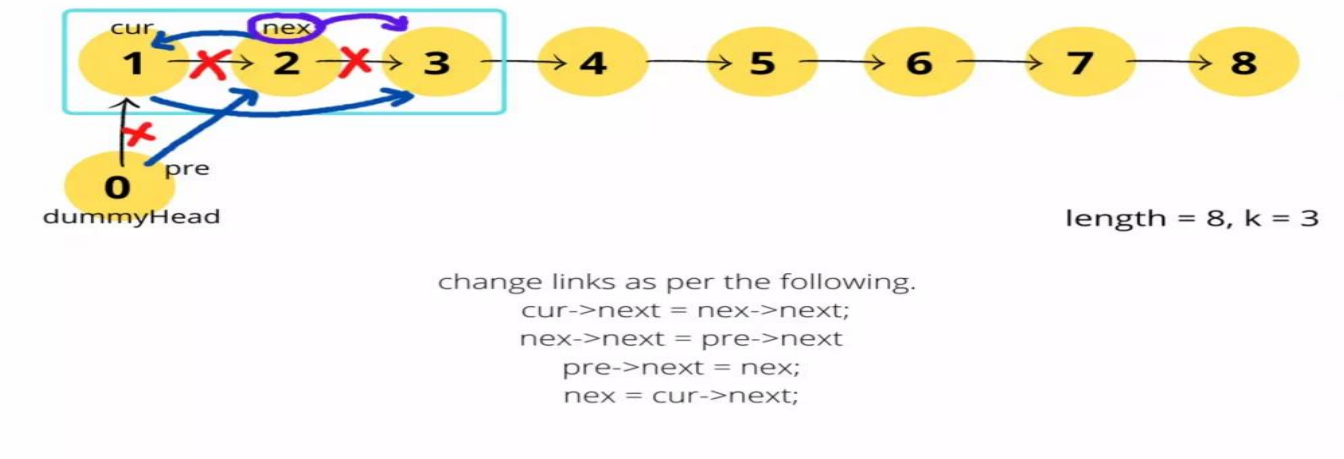
- cur->next = nex->next

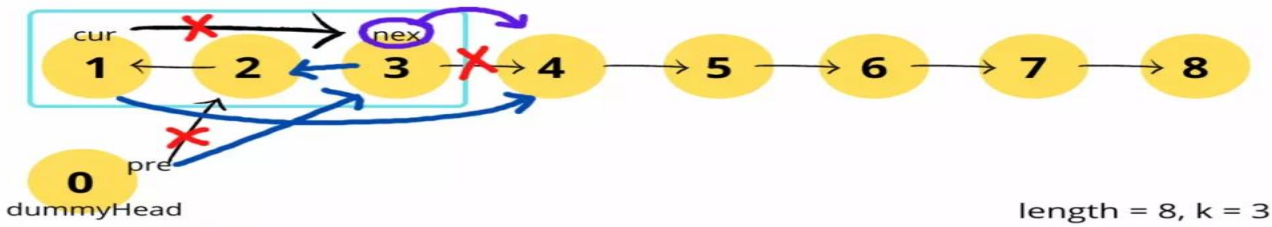
- nex->next = pre->next
- pre->next = nex
- nex = cur->next
- Move pre to cur and reduce length by k.

Dry Run:



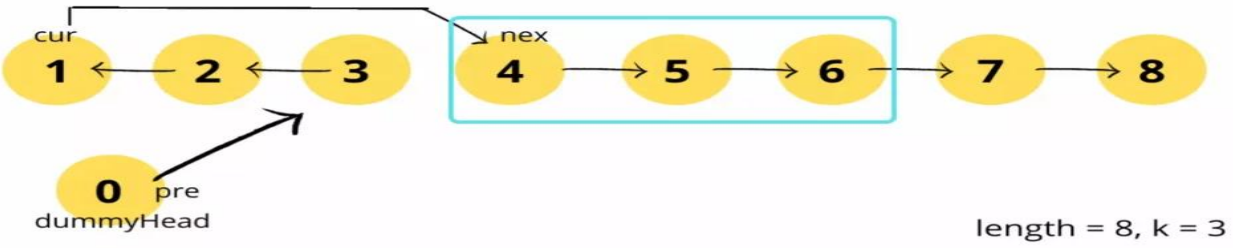






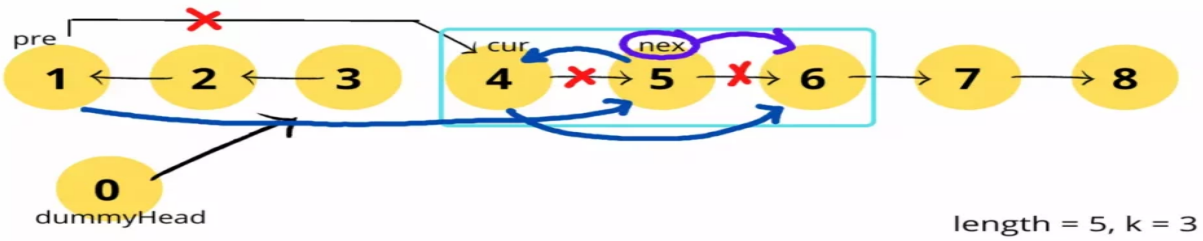
Now, we have to point 3 -> 2. So, perform same steps again.

```
cur->next = nex->next;
nex->next = pre->next
pre->next = nex;
nex = cur->next;
```



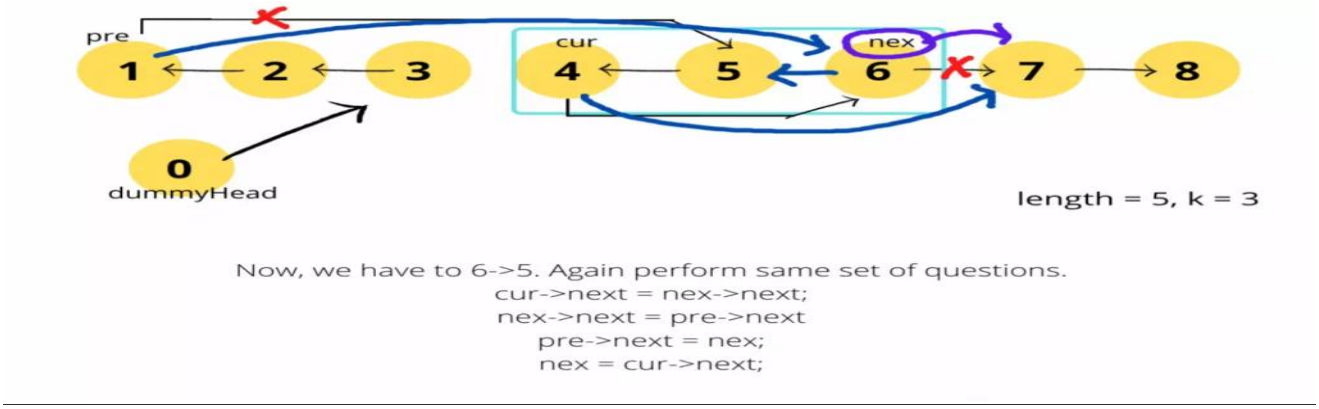
Our first group is reversed. Now, we start reversing second group. Move pre to curr and reduce length by k

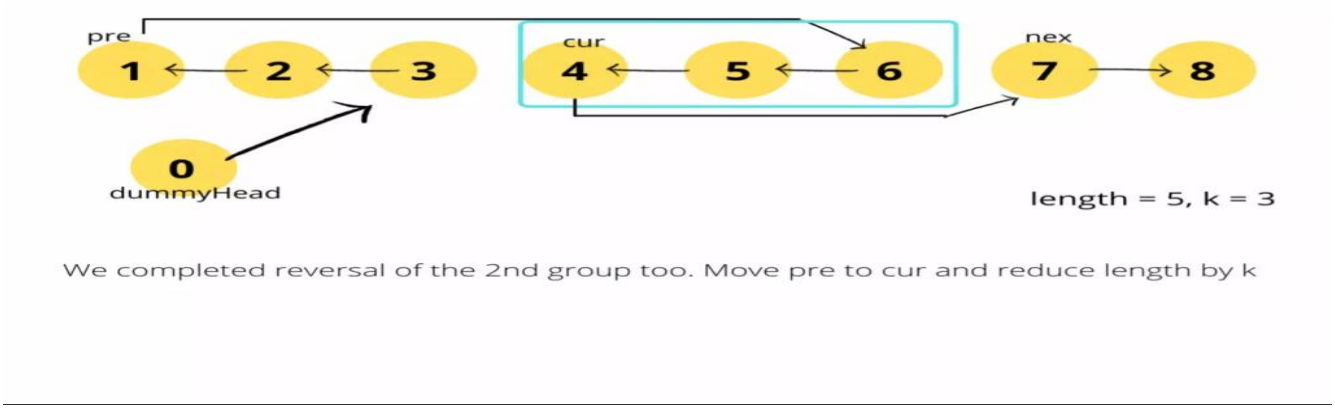
---

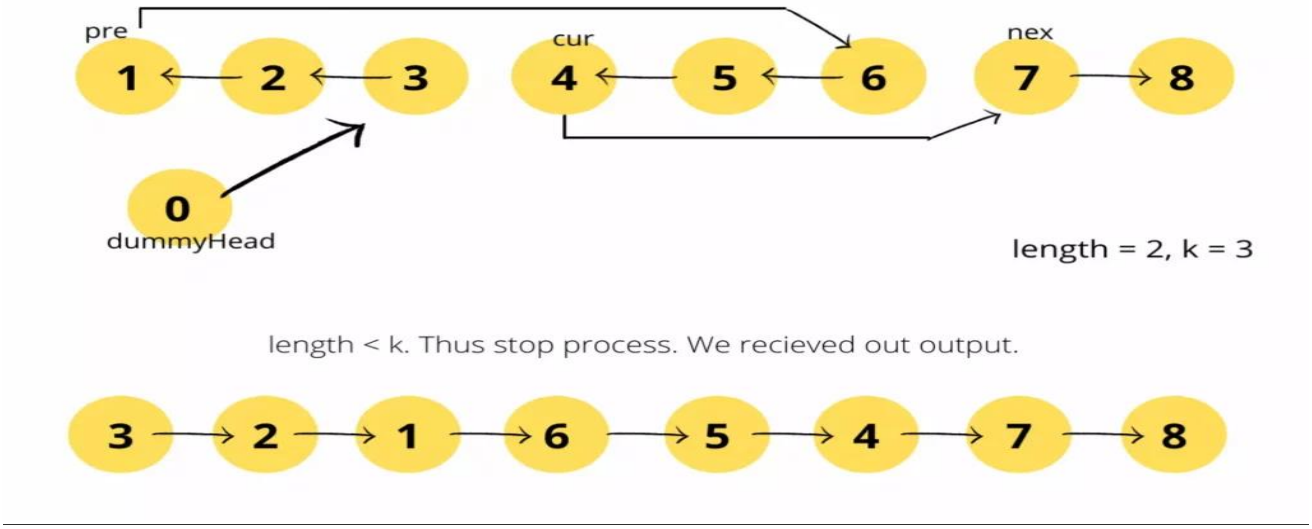


curr moves to pre->next and nex to nex->next. Again perform same set of questions.

```
cur->next = nex->next;
nex->next = pre->next
pre->next = nex;
nex = cur->next;
```







Code:

Iterative:

```
static int lengthOfLinkedList(Node head) {
    int length = 0;
```

```
    while(head != null) {
        ++length;
        head = head.next;
    }
    return length;
}
//utility function to reverse k nodes in the list
static Node reverseKNodes(Node head,int k) {
    if(head == null||head.next == null) return head;

    int length = lengthOfLinkedList(head);

    Node dummyHead = new Node(0);
    dummyHead.next = head;

    Node pre = dummyHead;
    Node cur;
    Node nex;

    while(length >= k) {
        cur = pre.next;
        nex = cur.next;
        for(int i=1;i<k;i++) {
            cur.next = nex.next;
            nex.next = pre.next;
            pre.next = nex;
            nex = cur.next;
```

```
        }
        pre = cur;
        length -= k;
    }
    return dummyHead.next;
}
```