Rotate linked list by k

BRUTE FORCE :

Run a loop for k times and each time pick up the last element and attach to head and point second last element to NULL.

- Time Complexity : O(N*k)
- Space Complexity : O(1)

Optimised Approach

Find len-kth node and the end node of the list. connect the last node to head and make the len-k +1 node to be the head and len-kth node point to NULL.

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* rotateRight(ListNode* head, int k) {
        if(head==NULL || head->next==NULL)\
            return head;
        ListNode* slow=head;
        ListNode* fast=head;
        int cnt=0;
        while(slow!=NULL)
        {
            cnt++;
            slow=slow->next;
        }
        k=k%cnt;
        slow=head;
        for(int i=0;i<k;i++)
        {
            fast=fast->next;
        }
        while(fast->next!=NULL)
        {
            slow=slow->next;
            fast=fast->next;
        }
        fast->next=head;
        head=slow->next;
        slow->next=NULL;
        return head;

    }
};
```

- Time Complexity : O(N)
- Space Complexity : O(1)