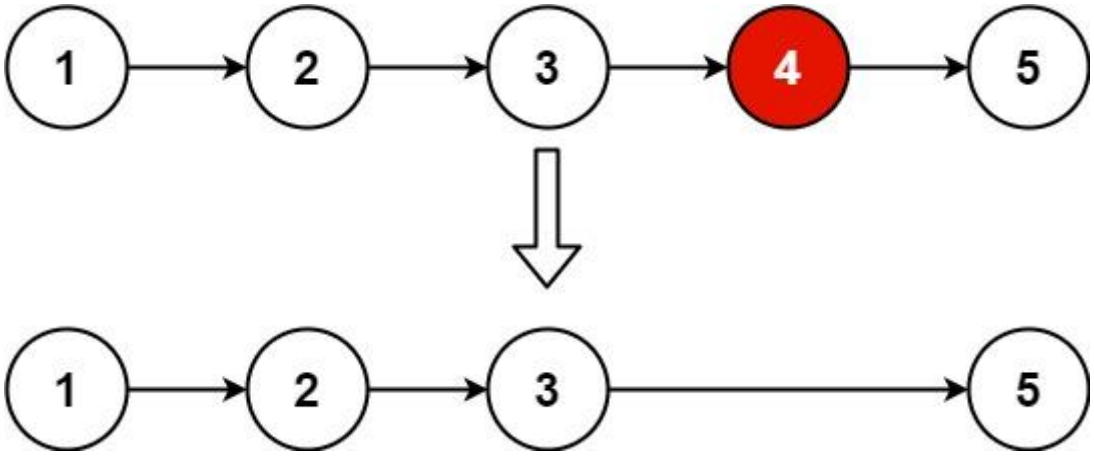
Remove Nth Node From End of List

Question:

https://leetcode.com/problems/remove-nth-node-from-end-of-list/

Given the `head` of a linked list, remove the `nth` node from the end of the list and return its head.

Example 1:




```
Input: head = [1,2,3,4,5], n = 2
Output: [1,2,3,5]
```

Example 2:

```
Input: head = [1], n = 1
Output: []
```

Example 3:

```
Input: head = [1,2], n = 1
Output: [1]
```

Approach:

The first approach was to use two pointers: **fast and slow**. We will give our fast pointer a head start of n steps.

Then we will move both the slow and fast pointers together making. In this way, when the fast pointer reaches the end, the slow pointer will be n nodes from the end.

Now we will just remove the current slow node and we are done.

The problem here is that **if the number of nodes is less than n**, we will get an error while deleting.

To overcome and handle this problem, we check this condition after giving our fast pointer a head start.

Solution:

https://gist.github.com/vermaayush680/cfe5b593f29824ce6719a048333d83a3

```
fast = slow = head
for _ in range(n):
        fast=fast.next
if not fast:
        return head.next
while fast.next:
        fast=fast.next
        slow=slow.next
slow.next=slow.next.next
return head
```