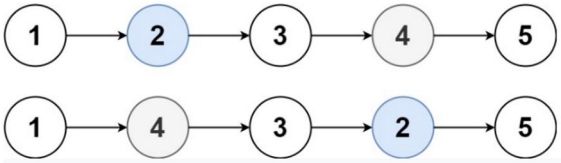
1721. Swapping Nodes in a Linked List

04 April 2022 05:23 PM

You are given the `head` of a linked list, and an integer `k`.

Return *the head of the linked list after **swapping** the values of the* k^{th} *node from the beginning and the* k^{th} *node from the end (the list is **1-indexed**).*

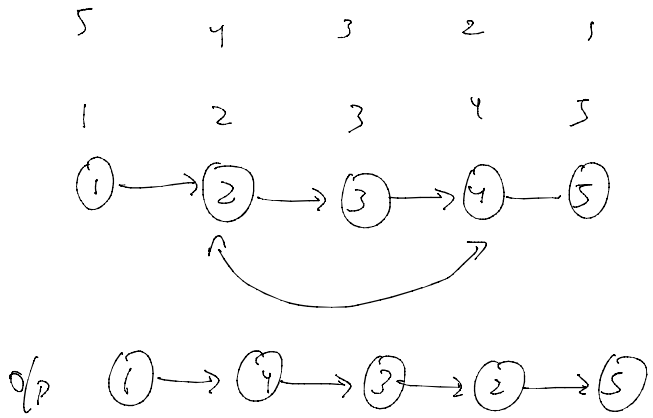
Example 1:




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

Example 2:

```
Input: head = [7,9,6,6,7,8,3,0,9,5], k = 5
Output: [7,9,6,6,8,7,3,0,9,5]
```



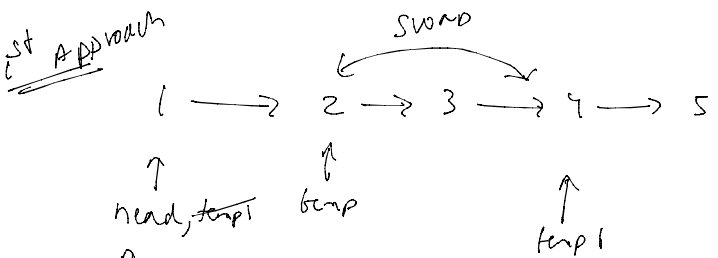
k = 2

k^{th} node from beginning

k^{th} node from end

swap

1st Approach



k = 2

temp * count = 0, when count == k, we swap

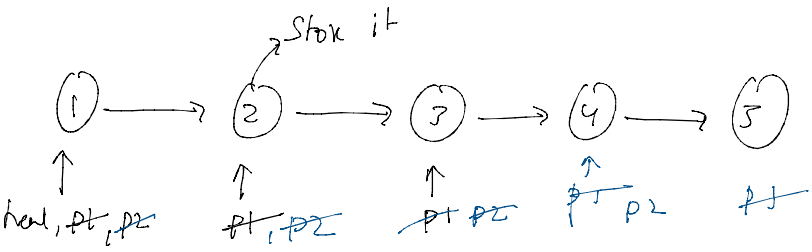
* from end len = 5 − 2 + 1 = 4 temp1

We are doing two traversal

→ k^{th} node from beginning

→ k^{th} node from end (we find length)

2nd Approach :



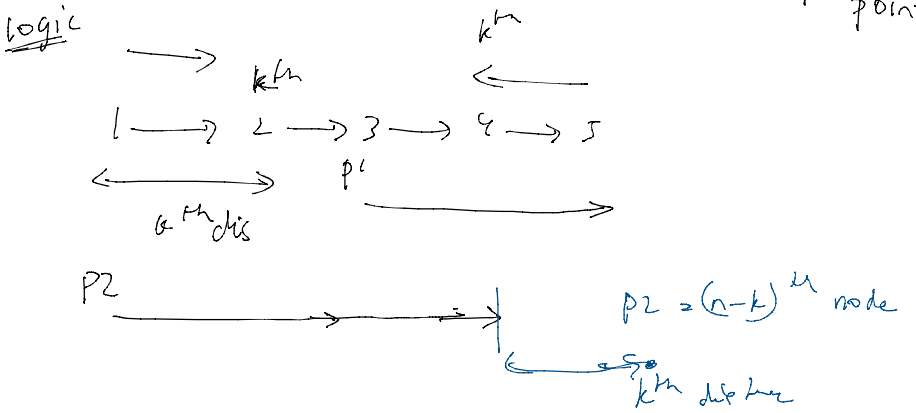
p1 = head

p2 = head

k^{th} node from beginning

p2 = head
p1 = 3
→ now keep increasing until p1 → null

when p1 reach null, p2 will point to k^{th} node of end.



C++ • Autocomplete

```cpp
class Solution {
public:
    ListNode* swapNodes(ListNode* head, int k) {
        ListNode *ptr1 = head, *ptr2 = head, *kth = NULL;
        while (--k)
            ptr1 = ptr1->next;

        kth = ptr1;
        ptr1 = ptr1->next;

        while (ptr1) {
            ptr1 = ptr1->next;
            ptr2 = ptr2->next;
        }
        swap(ptr2->val, kth->val);
        return head;
    }
};
```

Java Autocomplete

```java

class Solution
{
    public ListNode swapNodes(ListNode head, int k)
    {
        ListNode curr = head;
        ListNode pointer1 = head;
        ListNode pointer2= head;
        int count = 1;

        while( curr != null )
        {
            if( count < k )
            {
                pointer1 = pointer1.next;
            }
        if( count > k )
        {
            pointer2 = pointer2.next;
        }
        curr = curr.next;
        count++;
    }

    int temp = pointer1.val;
    pointer1.val = pointer2.val;
    pointer2.val = temp;

    return head;
  }
}
```