Description Solution Discuss (999+) Submissions

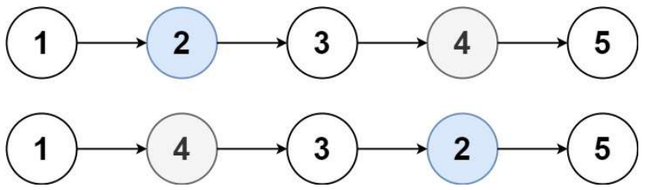
1721. Swapping Nodes in a Linked List

Medium 1816 77 Add to List Share

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]
```

Constraints:

- The number of nodes in the list is `n`.
- $1 <= k <= n <= 10^5$
- `0 <= Node.val <= 100`

Accepted 107,853 Submissions 163,511

Seen this question in a real interview before? Yes No

Companies

Java Autocomplete

```java
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
class Solution {
    public ListNode swapNodes(ListNode head, int k) {

        ListNode slowPointer = head;
        ListNode fastPointer = head;
        ListNode tempPointer = head;

        for(int i = 1; i < k; i++)
            fastPointer = fastPointer.next;

        tempPointer = fastPointer;

        while(fastPointer != null && fastPointer.next != null){
            fastPointer = fastPointer.next;
            slowPointer = slowPointer.next;
        }


        int temp = slowPointer.val;
        slowPointer.val = tempPointer.val;
        tempPointer.val = temp;

        return head;
    }
}
```

Your previous code was restored from your local storage. Reset to default