Problem 1721: Swapping Nodes in a Linked List

Problem Information

Difficulty: Medium
Acceptance Rate: 68.96%
Paid Only: No
Tags: Linked List, Two Pointers

Problem Description

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 _`kth` _node from the beginning and the_`kth` _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 <= 105` * `0 <= Node.val <= 100`

Code Snippets

C++:

```
/**
* 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* swapNodes(ListNode* head, int k) {

}
};
```

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) {

}
}
```

Python3:

```
# Definition for singly-linked list.
# class ListNode:
# def __init__(self, val=0, next=None):
# self.val = val
# self.next = next
class Solution:
def swapNodes(self, head: Optional[ListNode], k: int) -> Optional[ListNode]:
```