Problem 92: Reverse Linked List II

Problem Information

Difficulty: Medium
Acceptance Rate: 50.51%
Paid Only: No
Tags: Linked List

Problem Description

Given the `head` of a singly linked list and two integers `left` and `right` where `left <= right`, reverse the nodes of the list from position `left` to position `right`, and return _the reversed list_.

Example 1:

Input: head = [1,2,3,4,5], left = 2, right = 4 **Output:** [1,4,3,2,5]

Example 2:

Input: head = [5], left = 1, right = 1 **Output:** [5]

Constraints:

* The number of nodes in the list is `n`. * `1 <= n <= 500` * `-500 <= Node.val <= 500` * `1 <= left <= right <= n`

Follow up: Could you do it in one pass?

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* reverseBetween(ListNode* head, int left, int right) {

}
};
```

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 reverseBetween(ListNode head, int left, int right) {

}
}
```

Python3:

```
# Definition for singly-linked list.
# class ListNode:
# def __init__(self, val=0, next=None):
# self.val = val
# self.next = next
class Solution:
```

```
def reverseBetween(self, head: Optional[ListNode], left: int, right: int) ->
Optional[ListNode]:
```