Problem 206: Reverse Linked List

Problem Information

Difficulty: Easy
Acceptance Rate: 79.91%
Paid Only: No
Tags: Linked List, Recursion

Problem Description

Given the `head` of a singly linked list, reverse the list, and return _the reversed list_.

Example 1:

![](https://assets.leetcode.com/uploads/2021/02/19/rev1ex1.jpg)

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

Example 2:

![](https://assets.leetcode.com/uploads/2021/02/19/rev1ex2.jpg)

Input: head = [1,2] **Output:** [2,1]

Example 3:

Input: head = [] **Output:** []

Constraints:

* The number of nodes in the list is the range `[0, 5000]`. * `-5000 <= Node.val <= 5000`

Follow up: A linked list can be reversed either iteratively or recursively. Could you implement both?

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* reverseList(ListNode* head) {

}
};
```

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 reverseList(ListNode head) {

}
}
```

Python3:

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