Problem 1171: Remove Zero Sum Consecutive Nodes from Linked List

Problem Information

Difficulty: Medium
Acceptance Rate: 53.00%
Paid Only: No
Tags: Hash Table, Linked List

Problem Description

Given the `head` of a linked list, we repeatedly delete consecutive sequences of nodes that sum to `0` until there are no such sequences.

After doing so, return the head of the final linked list. You may return any such answer.

(Note that in the examples below, all sequences are serializations of `ListNode` objects.)

Example 1:

Input: head = [1,2,-3,3,1] **Output:** [3,1] **Note:** The answer [1,2,1] would also be accepted.

Example 2:

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

Example 3:

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

Constraints:

* The given linked list will contain between `1` and `1000` nodes. * Each node in the linked list has `-1000 <= node.val <= 1000`.

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* removeZeroSumSublists(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 removeZeroSumSublists(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 removeZeroSumSublists(self, head: Optional[ListNode]) ->
Optional[ListNode]:
```