**Remove Zero Sum Consecutive**

**Nodes from Linked List**

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.

class Solution {

public:

    ListNode\* removeZeroSumSublists(ListNode\* head) {

        ListNode\* dummy = new ListNode(0);

        dummy->next = head;

        int prefix\_sum = 0;

        std::unordered\_map<int, ListNode\*> prefix\_sums;

        prefix\_sums[0] = dummy;

        ListNode\* current = head;

        while (current) {

            prefix\_sum += current->val;

            if (prefix\_sums.find(prefix\_sum) != prefix\_sums.end()) {

                ListNode\* to\_delete = prefix\_sums[prefix\_sum]->next;

                int temp\_sum = prefix\_sum + to\_delete->val;

                while (to\_delete != current) {

                    prefix\_sums.erase(temp\_sum);

                    to\_delete = to\_delete->next;

                    temp\_sum += to\_delete->val;

                }

                prefix\_sums[prefix\_sum]->next = current->next;

            } else {

                prefix\_sums[prefix\_sum] = current;

            }

            current = current->next;

        }

        return dummy->next;

    }

};

Link : <https://leetcode.com/problems/remove-zero-sum-consecutive-nodes-from-linked-list/?envType=daily-question&envId=2024-03-12>