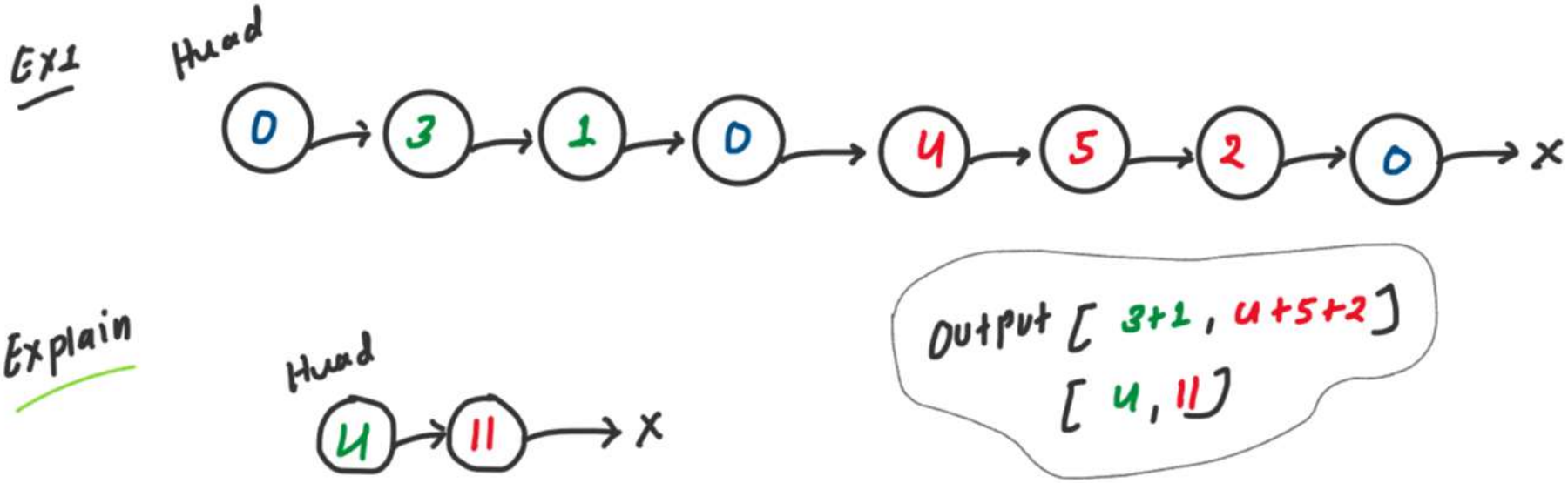
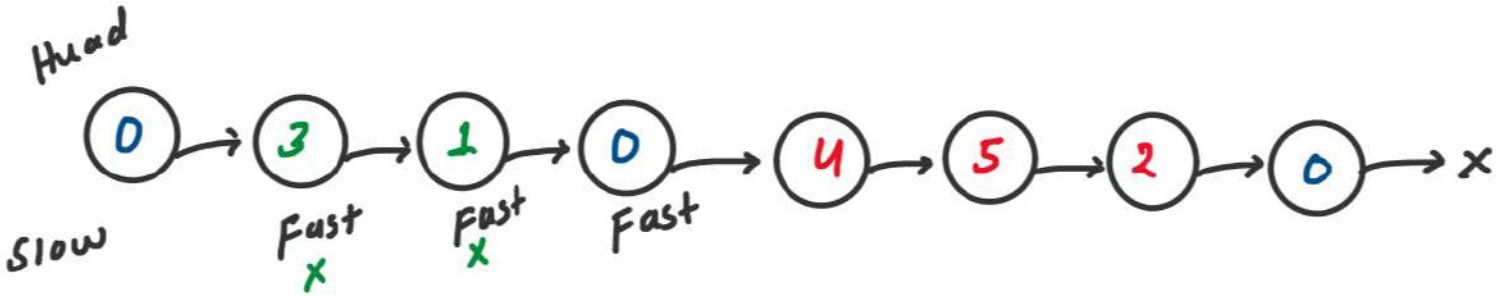
HW 11: Merge Nodes in between Zeros (Leetcode-2181)

Example 1:
Input: head = [0,3,1,0,4,5,2,0]
Output: [4,11]

Example 2:
Input: head = [0,1,0,3,0,2,2,0]
Output: [1,3,4]



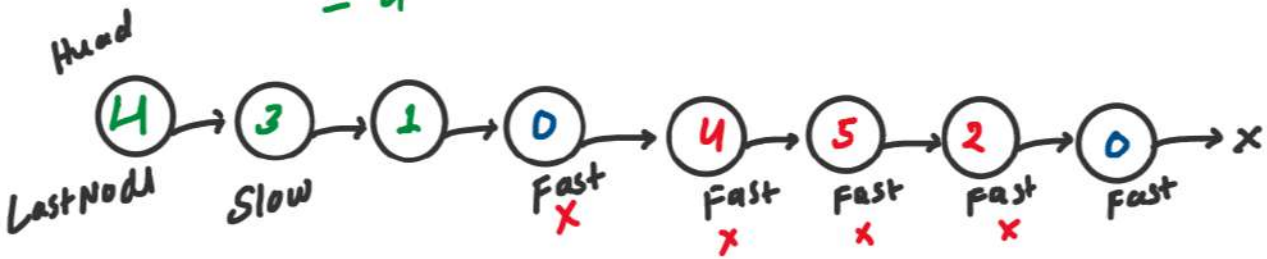
DRY RUN




```
Node* slow = head;
Node* fast = head->next;
Node* newLastNode = NULL;
int sum = 0
```

Sum kab tak karna hai - jab tak fast->val == 0 na ho

$$Sum = 0 + 3 + 1$$
$$= 4$$

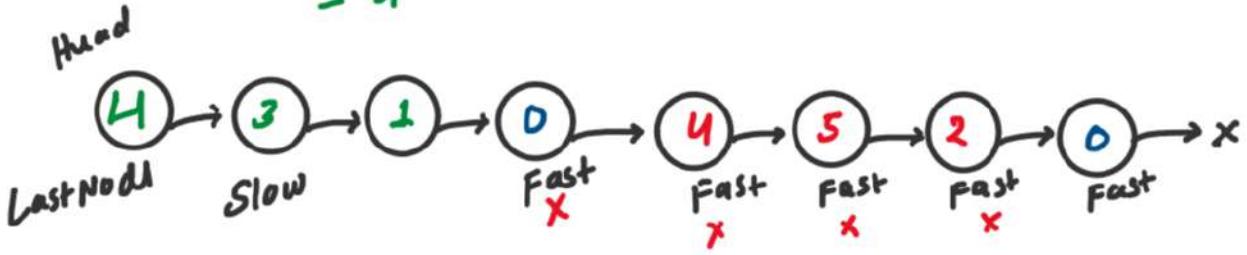


$$Sum = 0 + 4 + 5 + 2$$
$$= 11$$

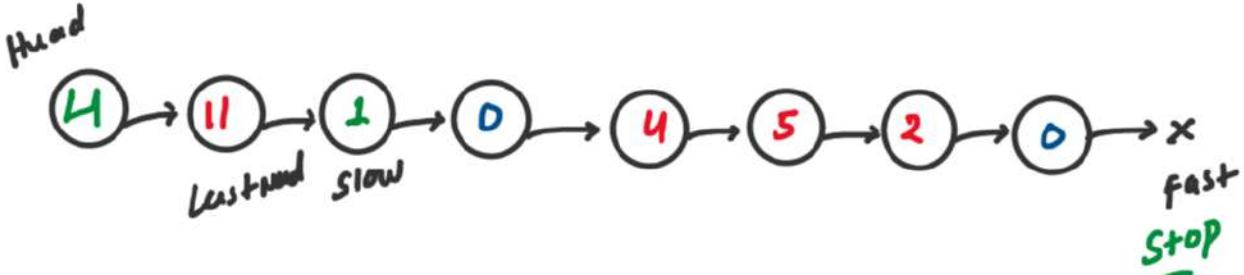
```
Node* slow = head;
Node* fast = head->next;
Node* newLastNode = NULL;
int sum = 0

While(fast){
    if(fast->val != 0){
        sum += fast->val;
    }
    else if(fast->val == 0){
        slow->val = sum;
        lastNode = slow;
        slow = slow->next;
        sum = 0;
    }
    fast = fast->next
}
```

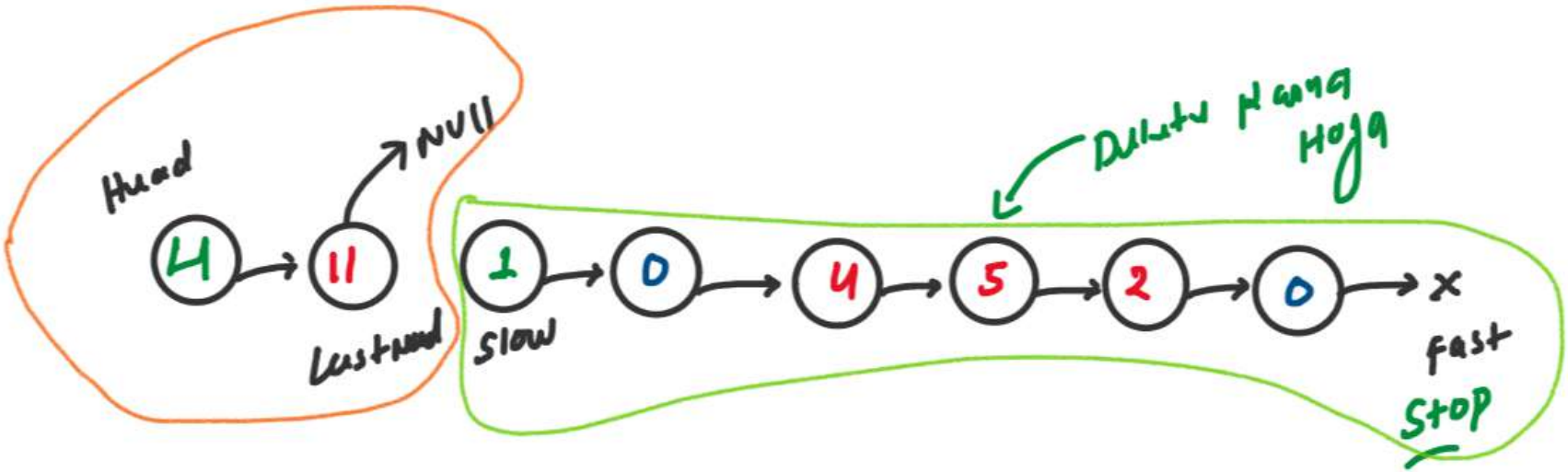
Sum kab tak karna hai - jab tak fast->val == 0 na ho

Sum = 0 + 3 + 1
= 4



Sum = 0 + 4 + 5 + 2
= 11






```
{ node* temp = slow;
  Lastnode->next = NULL;
  delete temp;
}
return head;
```

```cpp
// HW 11: Merge Nodes in between Zeros (Leetcode-2181)

class Solution {
public:
    ListNode* mergeNodes(ListNode* head) {
        ListNode* slow = head;
        ListNode* fast = head->next;
        ListNode* newLastNode = NULL;
        int sum = 0;

        while(fast){
            if(fast->val != 0){
                sum += fast->val;
            }
            else{
                // fast->val == 0
                slow->val = sum;
                newLastNode = slow;
                slow = slow->next;
                sum = 0;
            }
            fast = fast->next;
        }

        ListNode* temp = slow;

        // Just formed new list
        newLastNode->next = NULL;

        // Deleting old list
        delete temp;

        return head;
    }
};
```

Time complexity: O(N),
Where N is number of nodes of the linked list

Space complexity: O(1),
Where no extra space used