Problem 4: Given the head of a singly linked list, return true if it is a palindrome.

```
class ListNode:
def __init_(self, val=0, next=None):
    self.val = val
    self.next = next

def isPalindrome(head):
    if head is None or head.next is None:
        return True

    slow = fast = head
    while fast.next and fast.next.next:
        slow = slow.next
        fast = fast.next.next


    second_half = reverse_linked_list(slow.next)
    slow.next = None

    first_half = head
    while first_half and second_half:
        if first_half.val != second_half.val:
            return False
        first_half = first_half.next
        second_half = second_half.next

    return True

def reverse_linked_list(head):
    prev = None
    curr = head

    while curr:
        next_node = curr.next
```

```
        curr.next = prev
        prev = curr
        curr = next_node

    return prev


head = ListNode(1)
head.next = ListNode(2)
head.next.next = ListNode(3)
head.next.next.next = ListNode(3)
head.next.next.next.next = ListNode(2)
head.next.next.next.next.next = ListNode(1)

print(isPalindrome(head))
```