

```

# Definition for singly-linked list.
class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

def is_palindrome(head: ListNode) -> bool:
    # Helper function to reverse a linked list
    def reverse_list(head: ListNode) -> ListNode:
        prev = None
        current = head
        while current:
            next_node = current.next
            current.next = prev
            prev = current
            current = next_node
        return prev

    # Helper function to find the middle of the list
    def find_middle(head: ListNode) -> ListNode:
        slow = head
        fast = head
        while fast and fast.next:
            slow = slow.next
            fast = fast.next.next
        return slow

    if not head or not head.next:
        return True

    # Step 1: Find the middle of the linked list
    middle = find_middle(head)

    # Step 2: Reverse the second half of the list
    second_half_start = reverse_list(middle)

    # Step 3: Compare the first and the second half
    first_half_start = head
    second_half_copy = second_half_start # To restore the list later
    is_palindrome_flag = True

    while second_half_start:
        if first_half_start.val != second_half_start.val:
            is_palindrome_flag = False
            break
        first_half_start = first_half_start.next
        second_half_start = second_half_start.next

    # Step 4 (Optional): Restore the original list by reversing the second
half again
    reverse_list(second_half_copy)

    return is_palindrome_flag

# Helper function to convert a list into a linked list
def list_to_linkedlist(elements):
    head = ListNode(elements[0])
    current = head
    for element in elements[1:]:
        current.next = ListNode(element)
        current = current.next

```

```
        return head

# Helper function to print the linked list
def print_linkedlist(head):
    current = head
    result = []
    while current:
        result.append(current.val)
        current = current.next
    print(result)

# Example 1
head = list_to_linkedlist([1, 2, 2, 1])
print(is_palindrome(head))  # Output: True

# Example 2
head = list_to_linkedlist([1, 2])
print(is_palindrome(head))  # Output: False
```

Output:

True

False