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

**Example 1:**



**Input:** head = [1,2,2,1]

**Output:** true

**Example 2:**



**Input:** head = [1,2]

**Output:** false

Solution:

/\*\*

\* 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 boolean isPalindrome(ListNode head) {

if (head == null || head.next == null)

return true;

// step 1. cut the original list to two halves

ListNode prev = null, slow = head, fast = head, l1 = head;

while (fast != null && fast.next != null) {

prev = slow;

slow = slow.next;

fast = fast.next.next;

}

prev.next = null;

// step 2. reverse the 2nd half

ListNode l2 = (fast == null) ? reverse(slow) : reverse(slow.next);

// step 3. compare the new two halves

while (l1 != null && l2 != null) {

if (l1.val != l2.val)

return false;

l1 = l1.next;

l2 = l2.next;

}

return true;

}

// helper function: reverse a list

ListNode reverse(ListNode head) {

ListNode prev = null, curr = head, next = null;

while (curr != null) {

next = curr.next;

curr.next = prev;

prev = curr;

curr = next;

}

return prev;

}

}