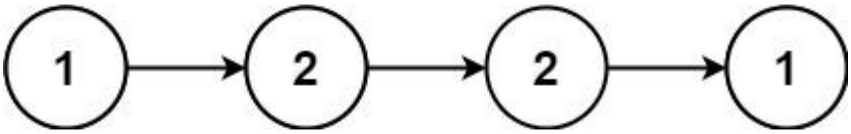
234. Palindrome Linked List

Given the `head` of a singly linked list, return `true` *if it is a*

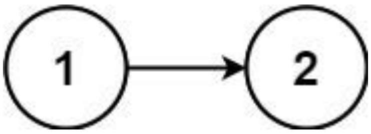
palindrome
or `false` *otherwise*.

Example 1:



```
Input: head = [1,2,2,1]
Output: true
```

Example 2:

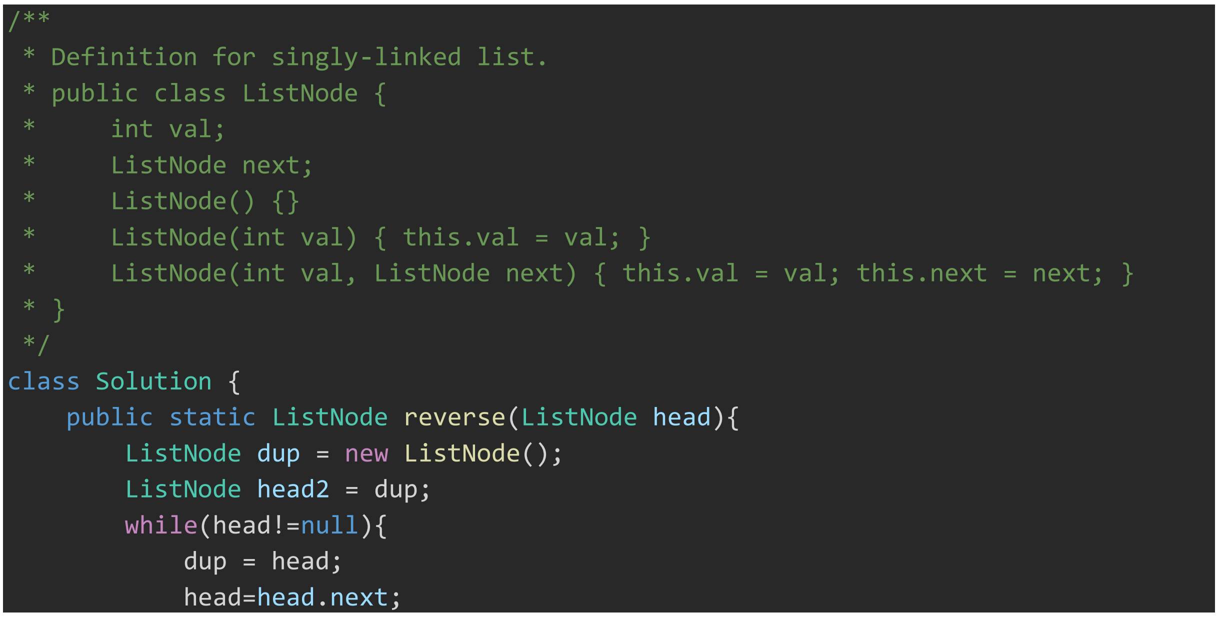


```
Input: head = [1,2]
Output: false
```

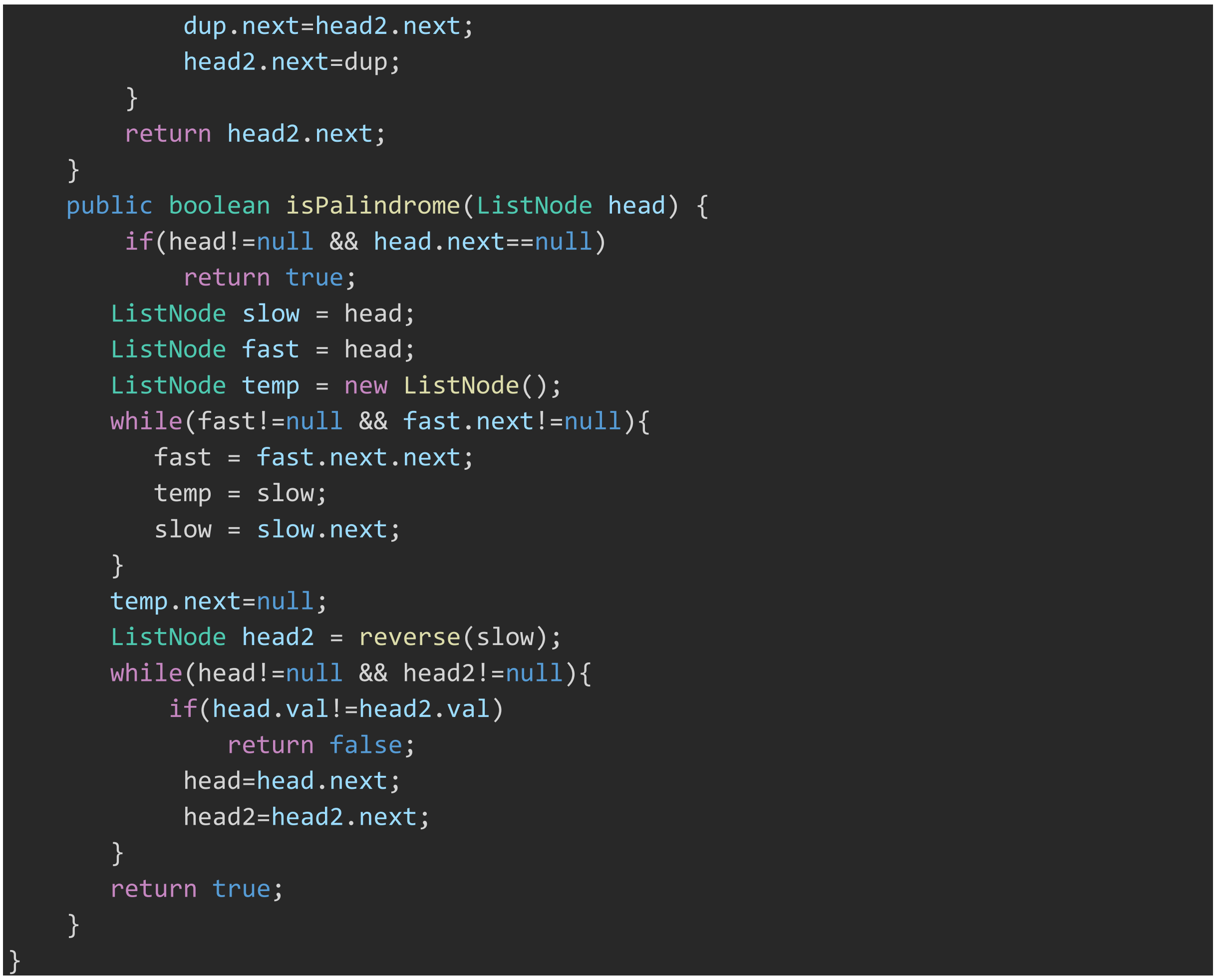
Constraints:

- The number of nodes in the list is in the range `[1, 10^5]`.
- `0 <= Node.val <= 9`

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 static ListNode reverse(ListNode head){
        ListNode dup = new ListNode();
        ListNode head2 = dup;
        while(head!=null){
            dup = head;
            head=head.next;
```



```
            dup.next=head2.next;
            head2.next=dup;
        }
        return head2.next;
    }
    public boolean isPalindrome(ListNode head) {
        if(head!=null && head.next==null)
            return true;
       ListNode slow = head;
       ListNode fast = head;
       ListNode temp = new ListNode();
       while(fast!=null && fast.next!=null){
          fast = fast.next.next;
          temp = slow;
          slow = slow.next;
       }
       temp.next=null;
       ListNode head2 = reverse(slow);
       while(head!=null && head2!=null){
           if(head.val!=head2.val)
               return false;
            head=head.next;
            head2=head2.next;
       }
       return true;
    }
}
```