Approach

1. Traverse the ll to know the mid index of list
2. Traverse the ll once again, but only till middle node, now reverse the LL from mid node till last. Keep this as a separate ll
3. Traverse both LL simultaneously and if any mismatch in values of nodes, return False
4. Return True if both LL’s become empty at the end of traversal

CODE:  
 def reverse(self, head):

prev = None

while head:

temp = head.next

head.next = prev

prev = head

head = temp

return prev

def isPalindrome(self, head: Optional[ListNode]) -> bool:

# use a stack

# Traverse the LL from start to mid , reverse from mid to last, keep comparing

temp = head

n = 0

while temp:

temp = temp.next

n+=1

# go to middle node

mid = n//2

temp = head

while mid:

temp = temp.next

mid -= 1

#reverse LL from mid to last

head2 = self.reverse(temp)

while head and head2:

if head.val != head2.val:

return False

head = head.next

head2 = head2.next

return not (head and head2)