**234. Palindrome Linked List**

Easy

3209361Add to ListShare

Given a singly linked list, determine if it is a palindrome.

**Example 1:**

**Input:** 1->2

**Output:** false

**Example 2:**

**Input:** 1->2->2->1

**Output:** true

**Follow up:**  
Could you do it in O(n) time and O(1) space?

怎么判定是不是对称，同时不用stack//stack就太简单了，直接倒着堆就完事儿了

首先使用快慢指针，

让快指针是慢指针速度的两倍，这样快指针指到null的时候，我们能利用慢指针找到中间点，

然后颠倒慢指针后面的LinkedList, 例如1221变成1212

这时快指针指向head，每次走一格，一样就是true不一样就是false;

class Solution {

public boolean isPalindrome(ListNode head) {

ListNode fast=head;

ListNode slow=head;

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

fast=fast.next.next;

slow=slow.next;

} //前面的好理解，找中点

slow=reversed(slow);

fast=head;

while(slow!=null){

if(slow.val!=fast.val) return false;

slow=slow.next;

fast=fast.next;

}

return true;

}

难点在于颠倒LinkedList

public ListNode reversed(ListNode head){

ListNode prev = null;

while(head != null){

ListNode next = head.next; //在这一步·,提前在改变head next前存储head.next

head.next=prev; //从左到右改变箭头，让他指向储藏的前一个node

prev=head; //prev向右走一格

head=next; //head向右走一格

}

return prev； **//因为已经颠倒过来了，所以最后3才是head**

}

}

例如0123

next= 1

0 next=null //因为他是尾巴

prev=0

head=1

next=2

1的next=0

prev=1

head=2