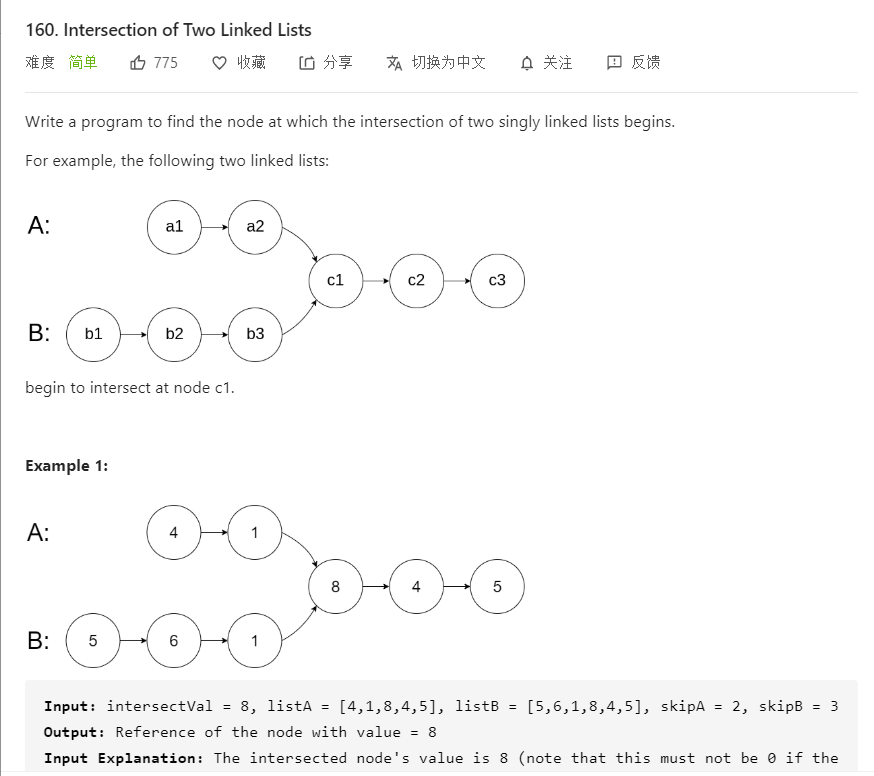
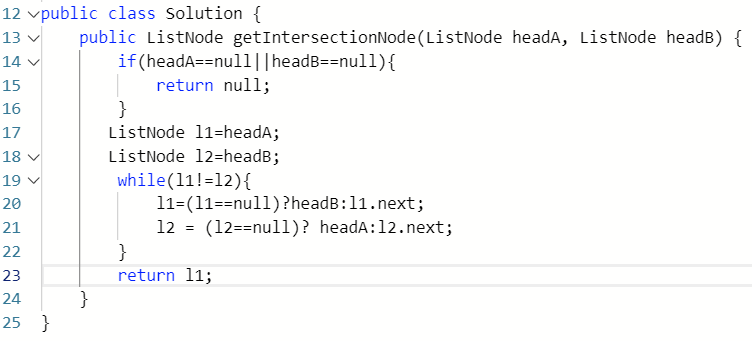
LinkedList

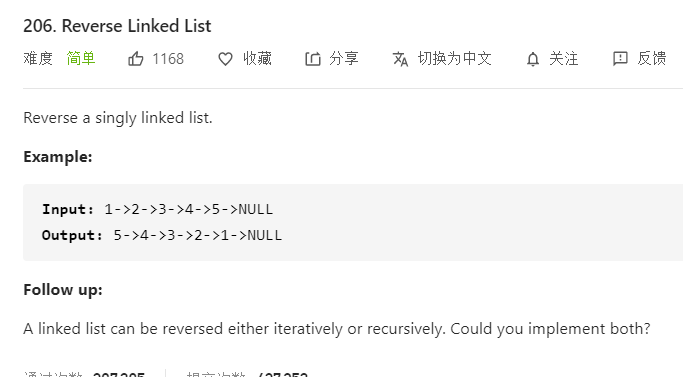
1.



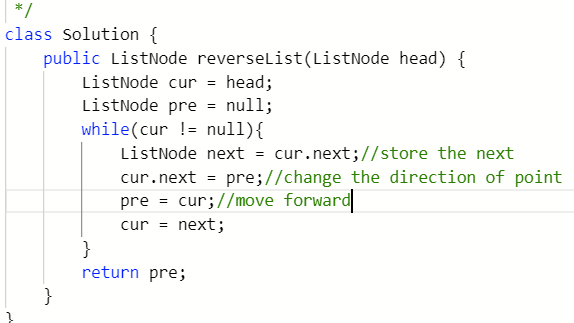
解法：在当两个节点不相同的时候，一直往后遍历，但如果下个是null则开始遍历另一个链表，当一个链表完成一轮遍历两链表节点数量不同则能抹除长度差（长的链表会从剩下的eg 3个node开始遍历因为另一个链表的长度为3），如果两个链表长度一样则最后是null==null



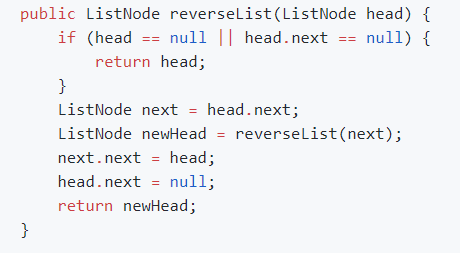
2.



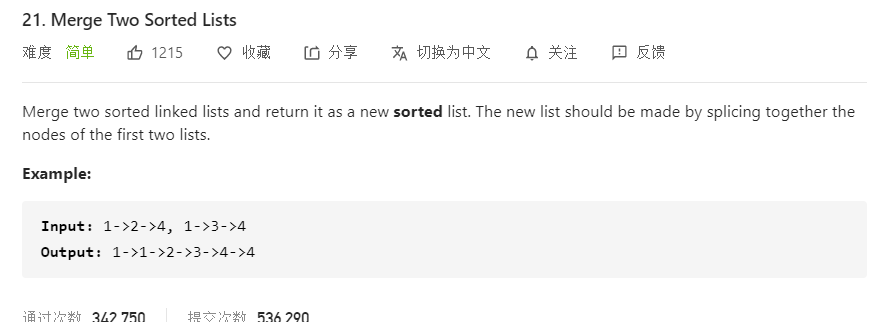
解法1：用pre cur next三个指针每次都把cur的指针指向pre，这样就反转了链表 直到cur==null



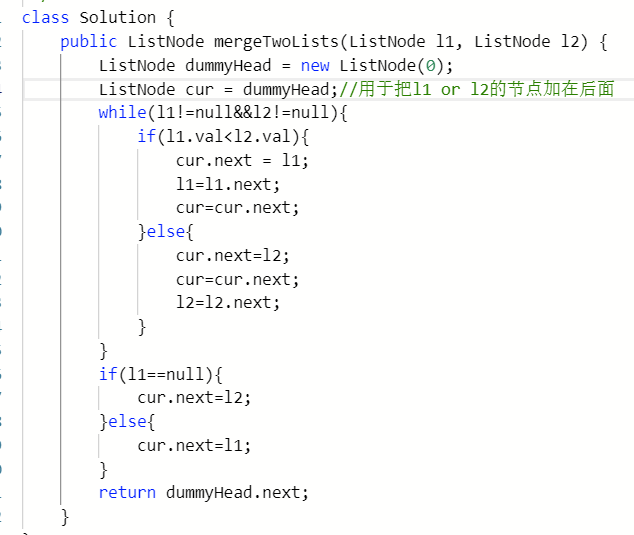
解法2：递归



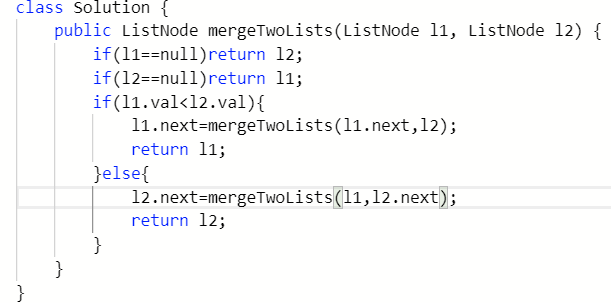
3.



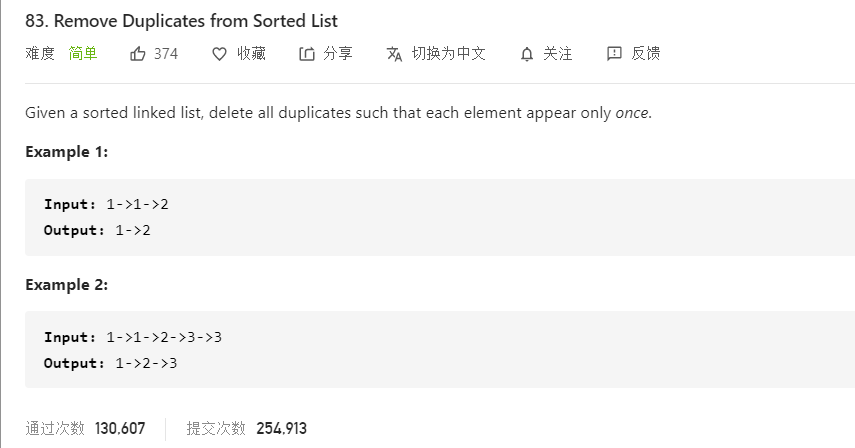
解法1：创建一个链表，head是dummyHead然后比较l1 l2的两个的节点，小的放到创建的链表的下个节点，每放进去一个节点，l1 or l2要更新成下个节点，如果一方全放了，要把另一个链接的剩下节点放入，直接用.next=l2连接剩下的节点



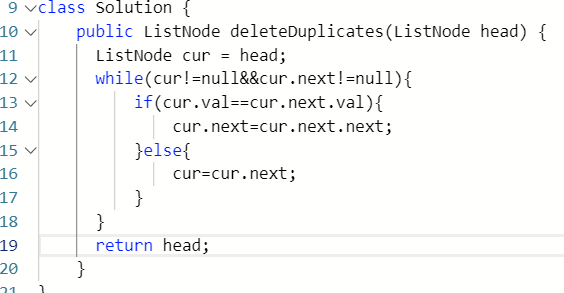
解法2：递归



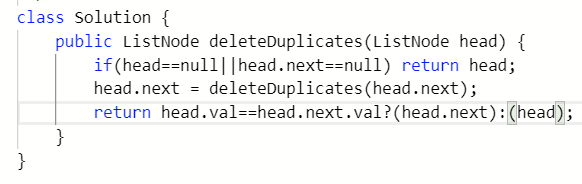
4.



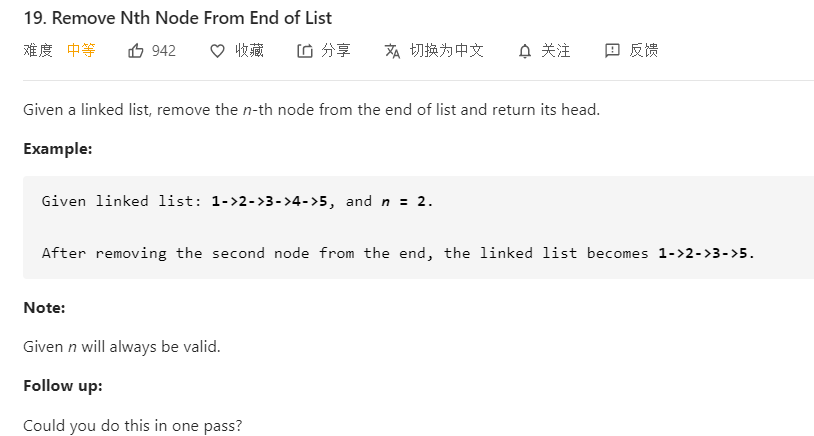
解法1：用cur去作为head，当next的value等于自己value时候直接去等于next的next否则就往后运动就行，cur就是用来去除检查和去除重复元素



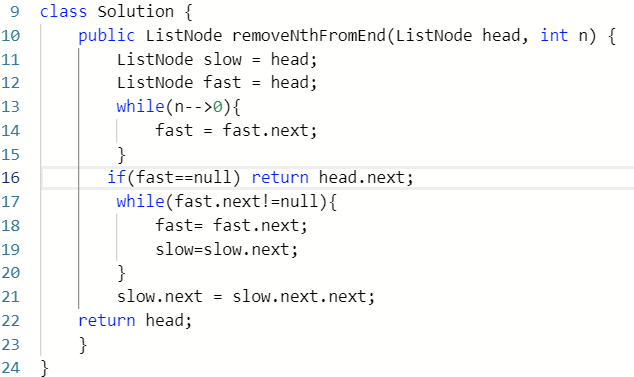
解法2：用递归 如果自己value和next的value一样，则head的next等于head的next



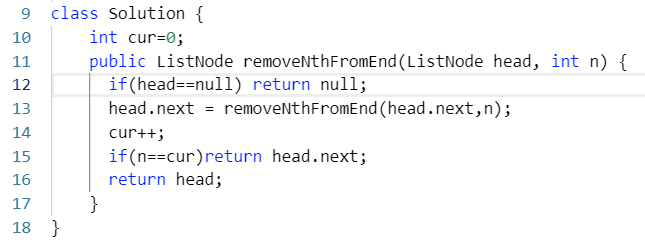
5.



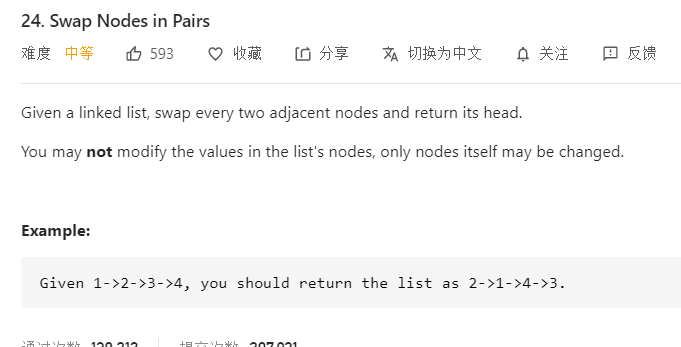
解法1：用快慢指针用快指针先从左到右走n个元素的距离，剩下的距离就是慢指针要从头走到倒数第n个元素的距离，所以等快指针到了从左到右的第n个元素，慢指针开始移动直到快指针走完，慢指针也到对应位置



解法2：递归

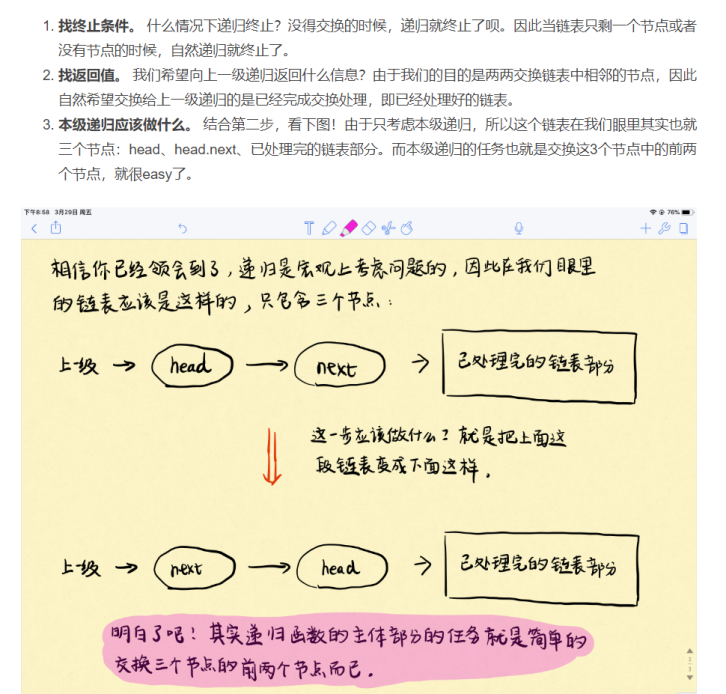


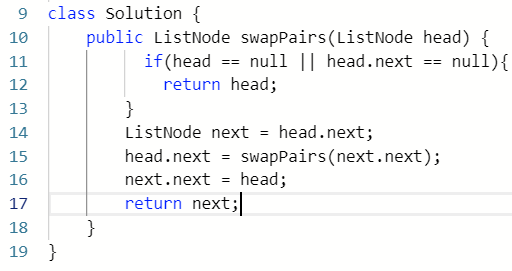
6.



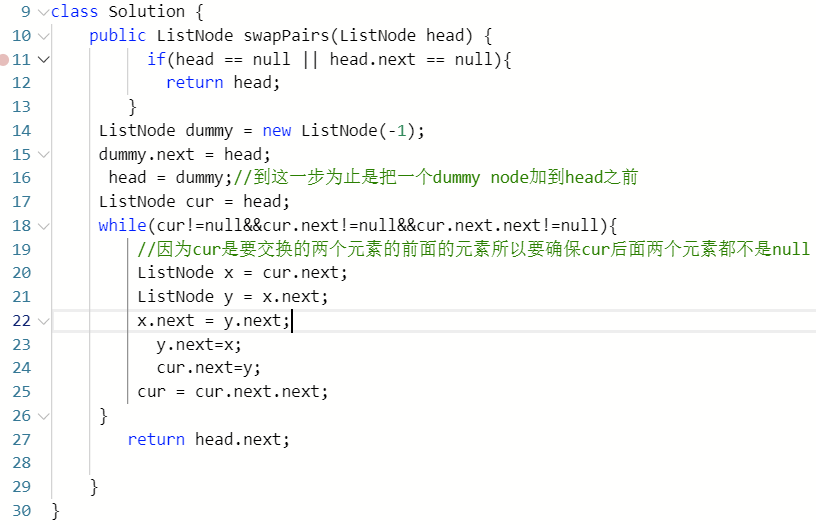
解法1：递归

1. 找终止条件：本题终止条件很明显，当递归到链表为空或者链表只剩一个元素的时候，没得交换了，自然就终止了。
2. 找返回值：返回给上一层递归的值应该是已经交换完成后的子链表。
3. 单次的过程：因为递归是重复做一样的事情，所以从宏观上考虑，只用考虑某一步是怎么完成的。我们假设待交换的俩节点分别为head和next，next的应该接受上一级返回的子链表(参考第2步)。就相当于是一个含三个节点的链表交换前两个节点，就很简单了，想不明白的画画图就ok。

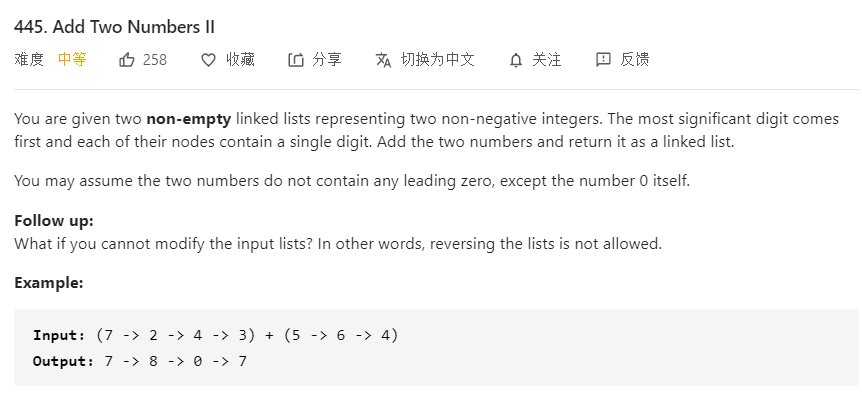




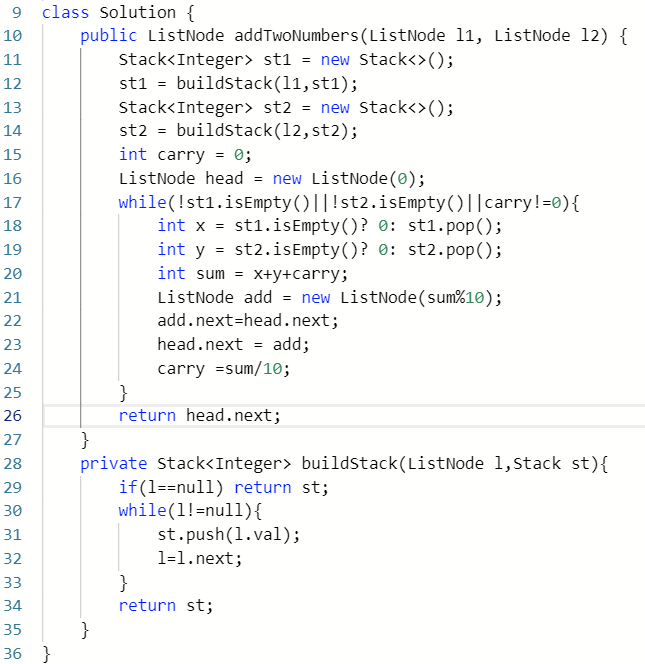
解法2：用一个虚拟头节点（dummy node）去当原来head的前面的节点，之后用两个变量存dummy node后面第一个和第二个节点去交换第一和第二个节点的位置



7.



解法：用两个stack把node的val压进，再取出来（也就是从末尾的节点开始取出来）把他们的值加起来放到head的next的位置。也就是newNode.next=head.next + head.next=newNode 要注意的是进位

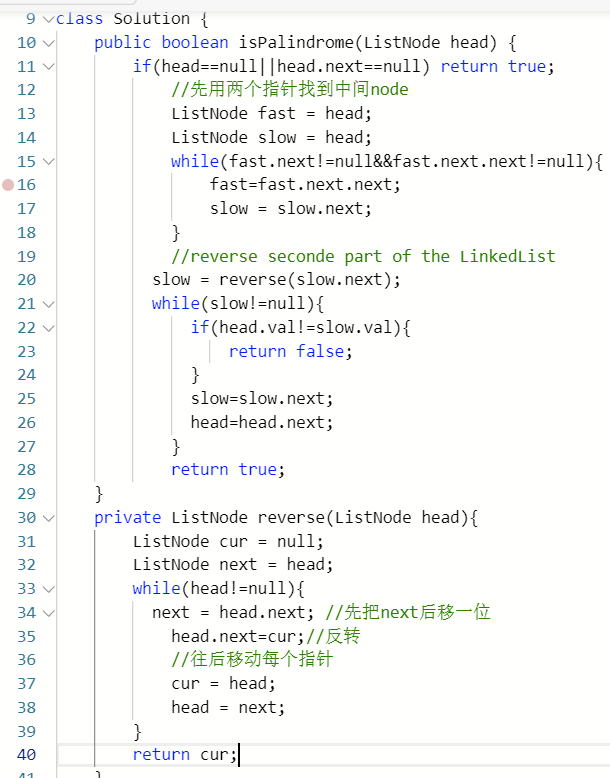


8.

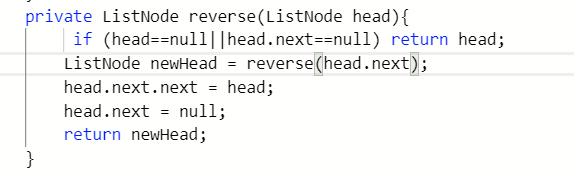


利用两个指针可以到达链表中间，因为链表没有index所以可以靠这样

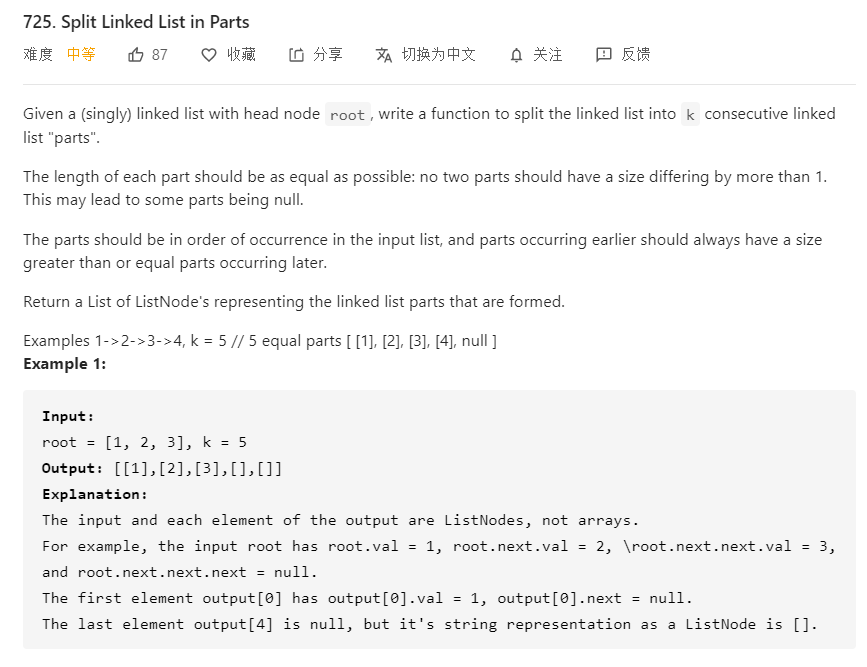
解法1：先用两个指针找到中间的变化的节点，再把第二部分的链表反转去看是不是和第一部分的链表一样 reverse可while可递归 ，while里用两个指针记录，一个记录第一个节点之前，第二个节点用head来 第三个指针next来记录下次head的位置，每个while交换head和第一个指针每次所有指针往后移动一位

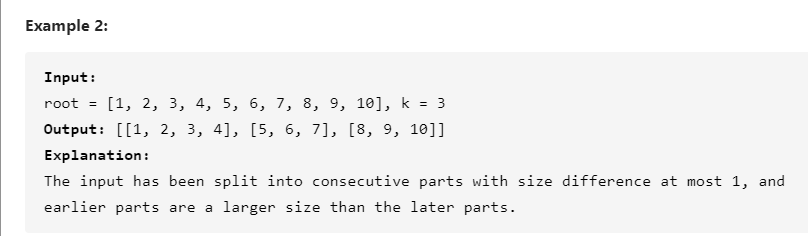


解法2：用递归去反转链表



9.





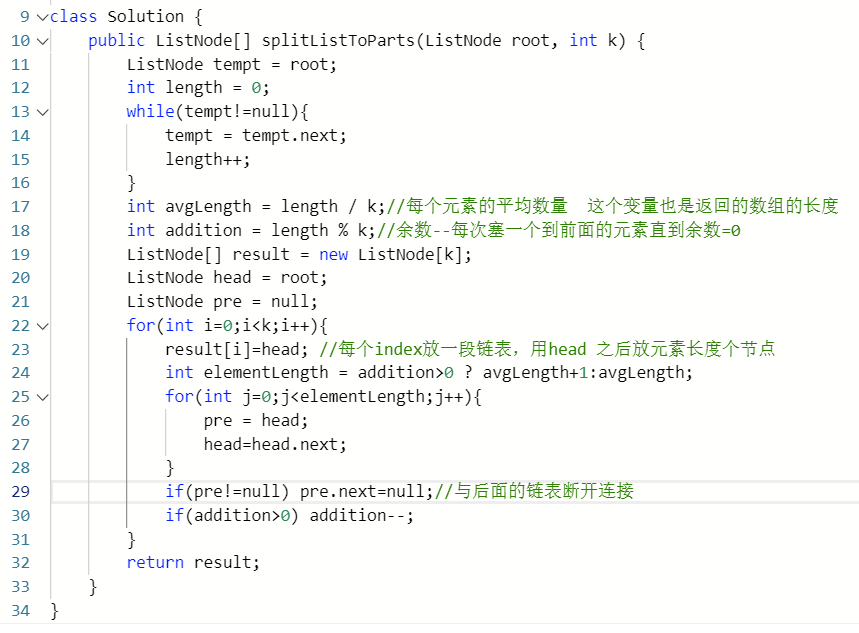
解法1：1，遍历链表获取长度 length（这个跑不掉 2，length 除以 k 得到每段链表的平均长度 aveLength 和 余数 remainder，remainder 的值就是有多少个长度为 (aveLength + 1) 的子链表排在前面。

2.1，举个例子帮助理解一下 11 / 3 = 3 余 2： 一共有3段，每段平均3个节点，但是剩下了2个节点，剩下的2个节点不能丢啊，得全部塞到子链表里面去，怎么塞呢？

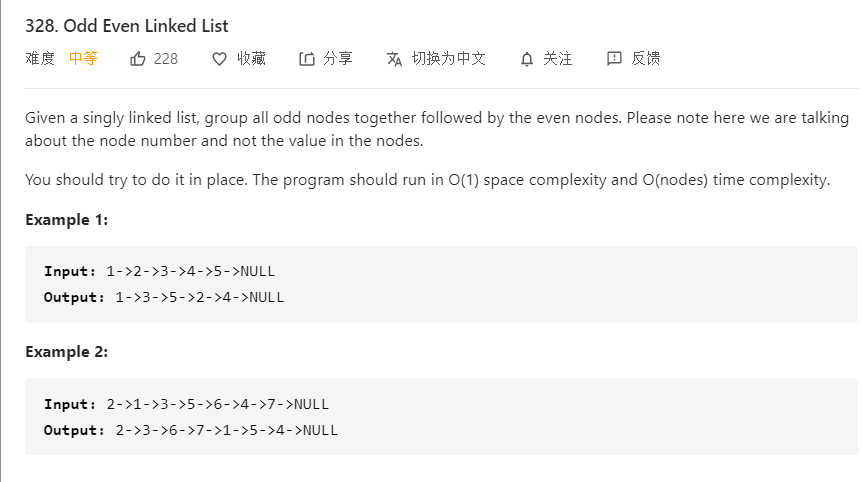
2.2，根据题意长的链表排前面，短的链表排后面，所以只有前面的两个子链表一人分担一个多余的节点，如此一来便形成了 4 4 3 的结构。

3，接下来的事儿就比较简单了，按照每个子链表应该的长度[4, 4, 3]去截断给定的链表。

放入特定长度的链表作为第一个元素可以通过把末尾节点的next设置为null来断开，但要确保head是下个元素的head



10.



解法：用三个指针，第一和第二个分别指向奇数和偶数个节点，每个while把他们的next设置成后两位，这样第1个的next是3，第2的next是4，这样就变成了把奇数个和偶数的元素分别组成两个链。第三个指针记录下第一个偶数个元素，这样就可以连接两个链表

