**LeetCode 题库分门别类详细解析笔记（第5章）**

第5章 在链表中穿针引线（19题） 3

5-1 在链表中穿针引线（2题） 3

 （完成）例题1：LeetCode 第 206 题。 3

 （完成）练习1：LeetCode 第 92 题。 3

5-2 测试你的链表程序（5题） 4

 （完成）练习1：LeetCode 第 83 题。 4

 （完成）练习2：LeetCode 第 86 题。 4

 （完成）练习3：LeetCode 第 328 题。奇数（Odd）偶数（Even）链表。 5

 （完成）练习4：LeetCode 第 2 题。两个数相加。 5

 （完成）练习5：LeetCode 第 445 题。两个数相加。 5

5-3 设立链表的虚拟头结点（3题，这是处理链表问题常用的技巧） 5

 （完成）例题1：LeetCode 第 203 题。 5

 （完成）练习1：LeetCode 第 82 题。给定一个排序链表，删除所有含有重复数字的节点，只保留原始链表中 没有重复出现 的数字。 6

 （完成）练习2：LeetCode 第 21 题。合并两个有序的链表。 7

5-4 复杂的穿针引线（4题） 8

 （完成）例题1：LeetCode 第 24 题。两两交换链表中的节点。 8

 练习1：LeetCode 第 25 题。k个一组翻转链表。 8

 练习2：LeetCode 第 147 题。 8

 练习3：LeetCode 第 148 题。 8

5-5 不仅仅是穿针引线（1题） 9

 例题1：LeetCode 第 237 题。 9

5-6 链表与双指针（4题） 9

 （完成）例题1：LeetCode 第 19 题。 9

 练习1：LeetCode 第 61 题。 10

 练习2：LeetCode 第 143 题。 10

 练习3：LeetCode 第 234 题。 11

模板：

题目要求：

题目难度：

中文网址：

英文网址：

求解关键：

我的解答：

参考资料：

第5章 在链表中穿针引线（19题）

链表是一种特殊的线性结构，由于不能像数组一样进行随机的访问，所以和链表相关的问题有他自身的特点。我将之称为穿针引线。我们在这一章，就来看一看，如何在链表中穿针引线。

5-1 在链表中穿针引线（2题）

* （完成）例题1：LeetCode 第 206 题。

题目要求：反转链表，常规问题，要很熟练做出来。

题目难度：

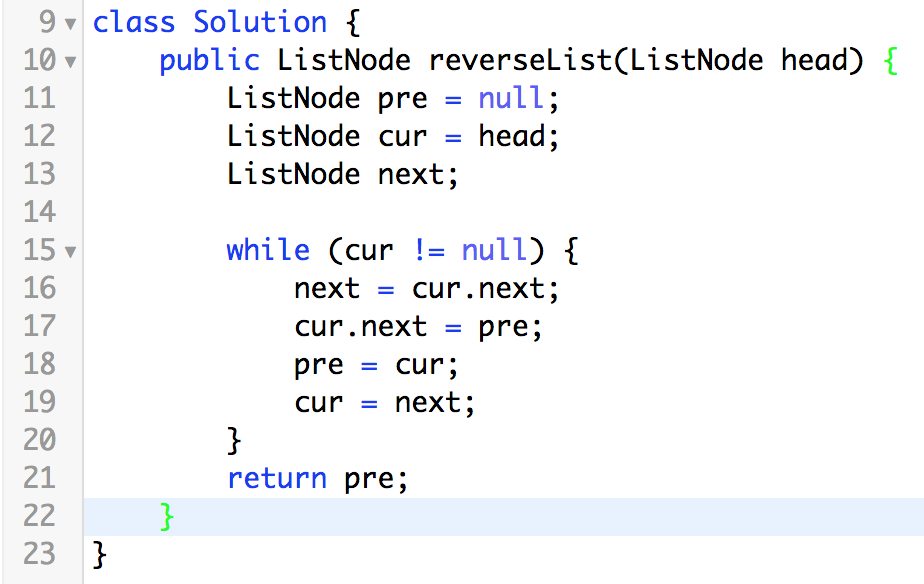
中文网址：https://leetcode-cn.com/problems/reverse-linked-list/description/

英文网址：

求解关键：（1）需要三个指针 pre、cur、next，其中 next 可以在 cur 确定以后初始化；

（2）画图分析 next 指针的指向，注意先后顺序，通常跟交换操作一样，程序写出来是“头尾相连”的。（3）最后一定不要忘记，返回的是 pre 节点（**画图就会很清晰了**）。

我的解答：



* （完成）练习1：LeetCode 第 92 题。

题目要求：

题目难度：

中文网址：

英文网址：

求解关键：

我的解答：

参考资料：

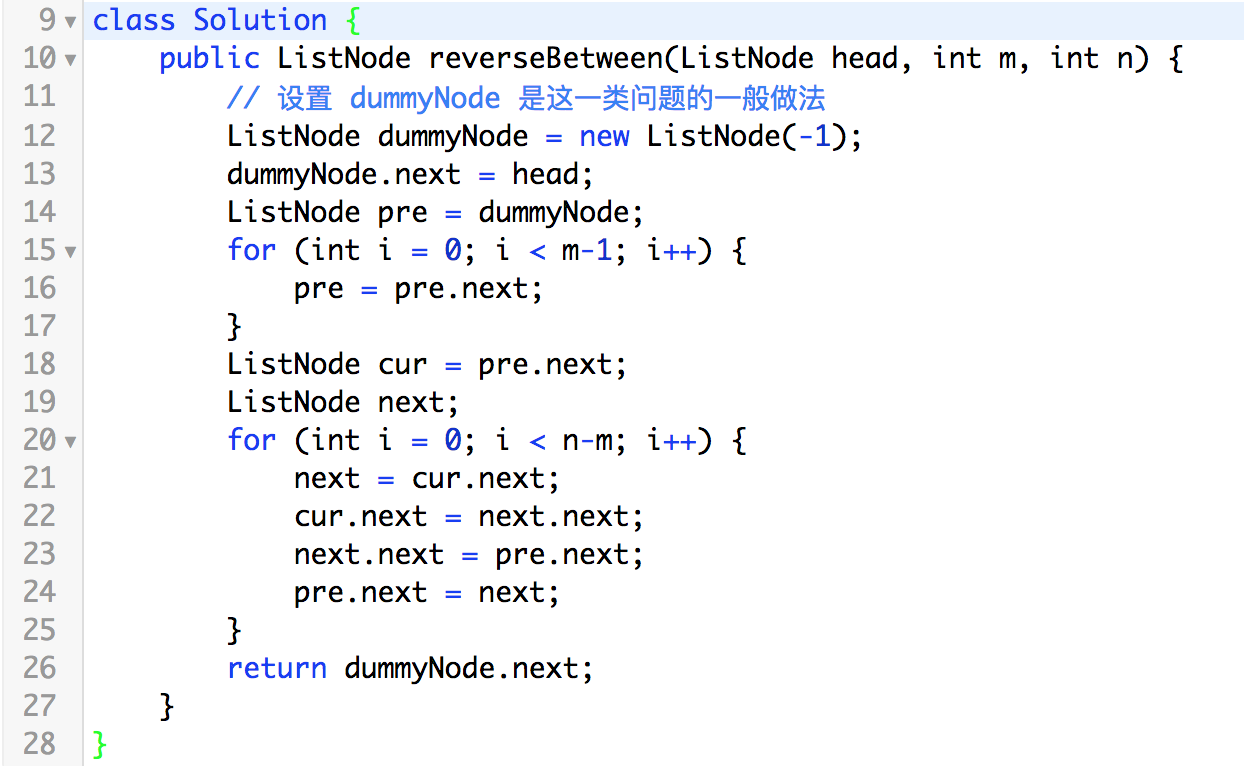
https://leetcode-cn.com/problems/reverse-linked-list-ii/description/

提示：（1）反转链表指定的部分，用到了设置虚拟头结点，只要设计到头节点的前一个节点的操作，一般都会用到虚拟头结点这个技巧，使得我们的代码更加简洁，一定要注意，返回的时候，要返回虚拟头结点的 next。

（2）关注临界值，pre 要循环几次，链表要“滚”几轮，都是这里要关注的点，因此代入一些具体值就能避免出错，正确的结果无非就是我们以为的那个数字 +1 或者 -1；

（3）自己在纸上画出图来，验证一下两轮以后代码是不是得到我们想要的一致结果；

（4）关于代码：每“滚”一次，其实 cur 的指针都不变，pre 也不变，next 变化，但是一直跟在 cur 的后面，所以它的位置在循环开始的时候确定。每次 cur 的 next 都会移到 pre 的 next，这就是循环体内第 2 行代码的含义。看这个循环体的结构，是不是还是“头尾相连”的呢？

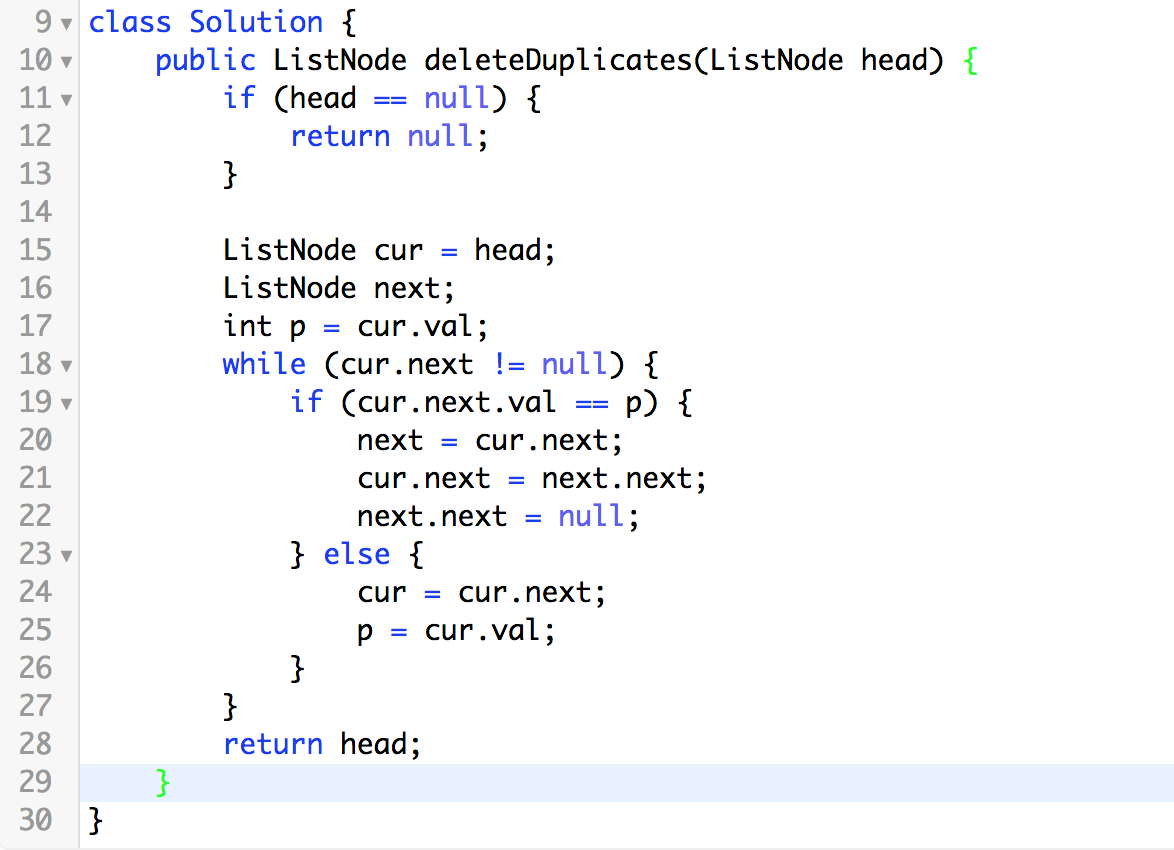


5-2 测试你的链表程序（5题）

* （完成）练习1：LeetCode 第 83 题。

https://leetcode-cn.com/problemset/all/?search=83

提示：给定一个排序链表，删除所有重复的元素，**使得每个元素只出现一次**。（简单：但是还是不熟练，做起来很慢。）思路：3 个指针。套路是：一开始就初始化 next 指针。



（1）空的情况不要忘记写在最开始，最简单的情况最容易忽略；

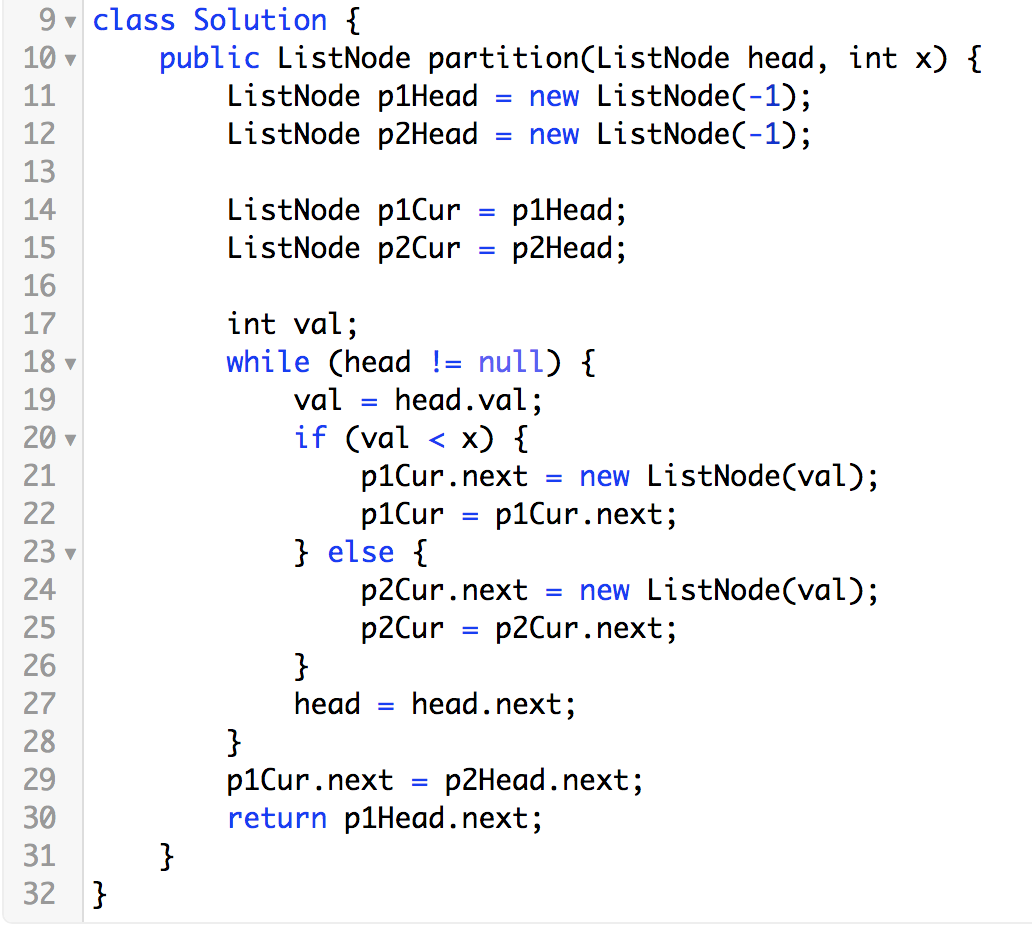
（2）这里不会涉及头结点的前一个节点的操作，因此不需要设立虚拟的头结点；

（3）删除链表中的节点的**固定的套路**是 while(cur.next!=null)，即**去判断当前节点的下一个节点是不是我们要删除的节点**，如果是，则当前节点的下一个节点的指向跳过它，所以这里只要 cur 和 next 就可以了，不用 pre（图示）。

* （完成）练习2：LeetCode 第 86 题。

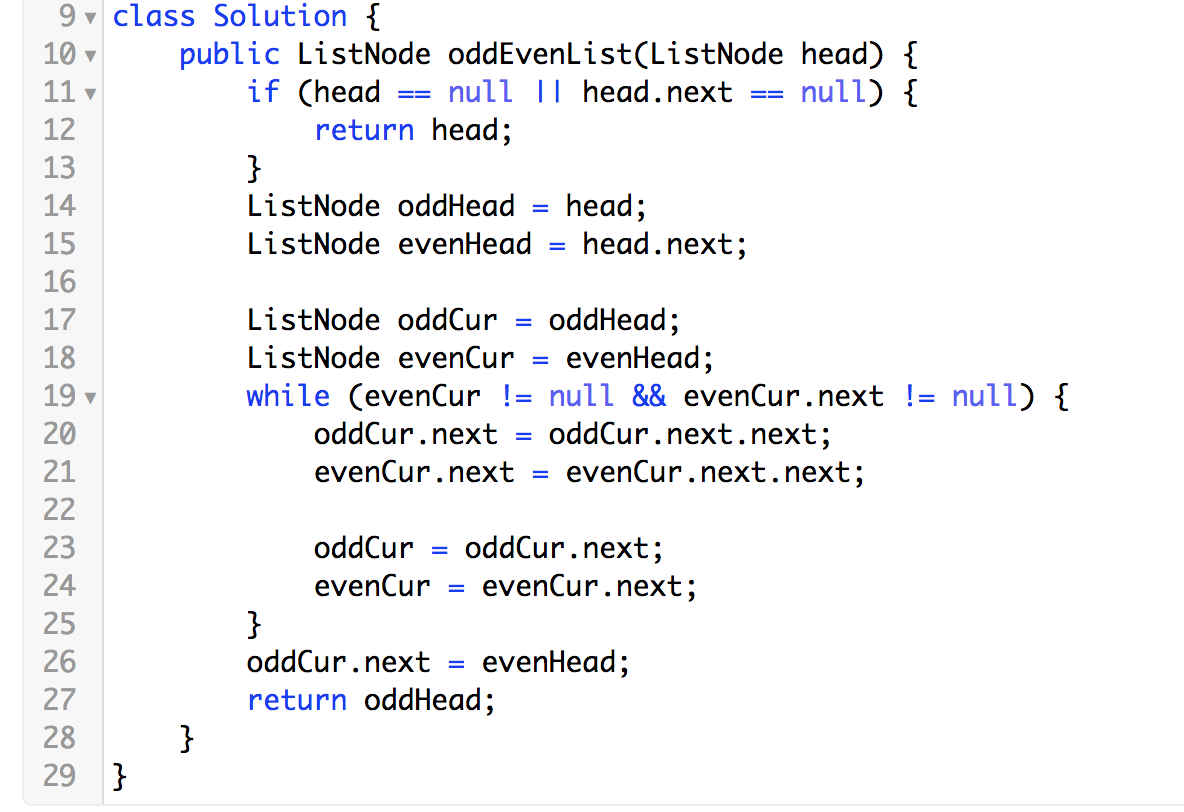
https://leetcode-cn.com/problems/partition-list/description/

提示：分隔链表。其实就是我们在数组中的 partition 这个过程，只不过要注意，在数组中，我们要通过不断地交换元素的位置来 partition ，在这里，我们利用链表的动态特性，可以很高效地解决问题，思路也非常自然，所以不要被在数组中的 partition 的固定思维带跑偏了。



* （完成）练习3：LeetCode 第 328 题。奇数（Odd）偶数（Even）链表。

提示：除了分别创建两个节点以外，然后拼在一起，我们还有下面这种写法：在遍历的过程中逐渐改变指针的指向，最后串在一起，此时的时间复杂度为 O(1)。



注意1： while 循环的执行条件一定要判断正确。

注意2：代码写到这里，有些地方已经很熟练了，不需要添加注释，所以量变产生质变，一定要多练习。

* （完成）练习4：LeetCode 第 2 题。两个数相加。

提示：这是一个相对简单的题目，但是它的姐妹问题第 445 号问题更能考查对相关知识点的掌握。要掌握的技巧是如何表示进位。

* （完成）练习5：LeetCode 第 445 题。两个数相加。

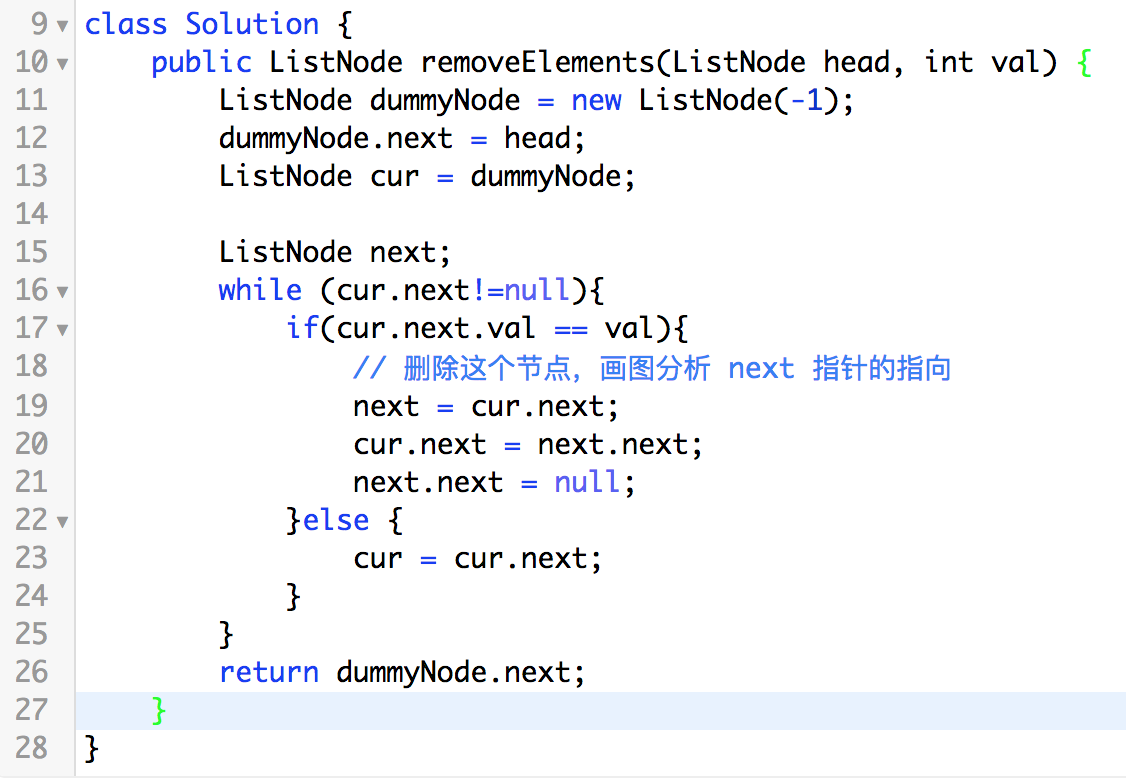
提示：这道题以第 2 题为基础。

5-3 设立链表的虚拟头结点（3题，这是处理链表问题常用的技巧）

* （完成）例题1：LeetCode 第 203 题。

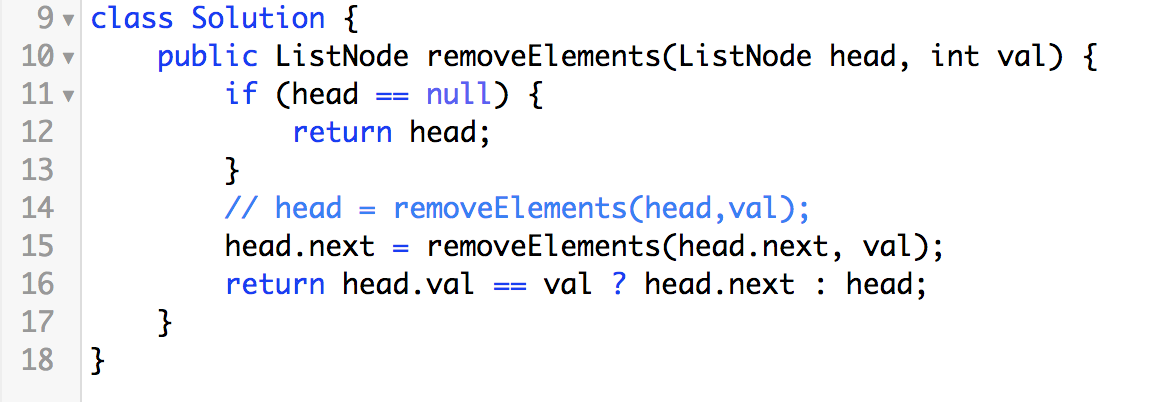
https://leetcode-cn.com/problems/remove-linked-list-elements/description/

解法1：删除节点这件事情很可能发生在链表的头结点，因此需要对头结点特殊处理。常用的处理头结点的技巧是设立虚拟头结点。这样头结点的处理逻辑和非头结点就可以统一起来。



解法2：使用递归删除。

思路：（1）处理最简单的情况。（2）假设规模小的情况解决了，处理规模大的情况。



* （完成）练习1：LeetCode 第 82 题。给定一个排序链表，删除所有含有重复数字的节点，只保留原始链表中 没有重复出现 的数字。

这里解题的关键还是画图，然后看图编码把自己要表达的逻辑表示出来。分类讨论是关键。

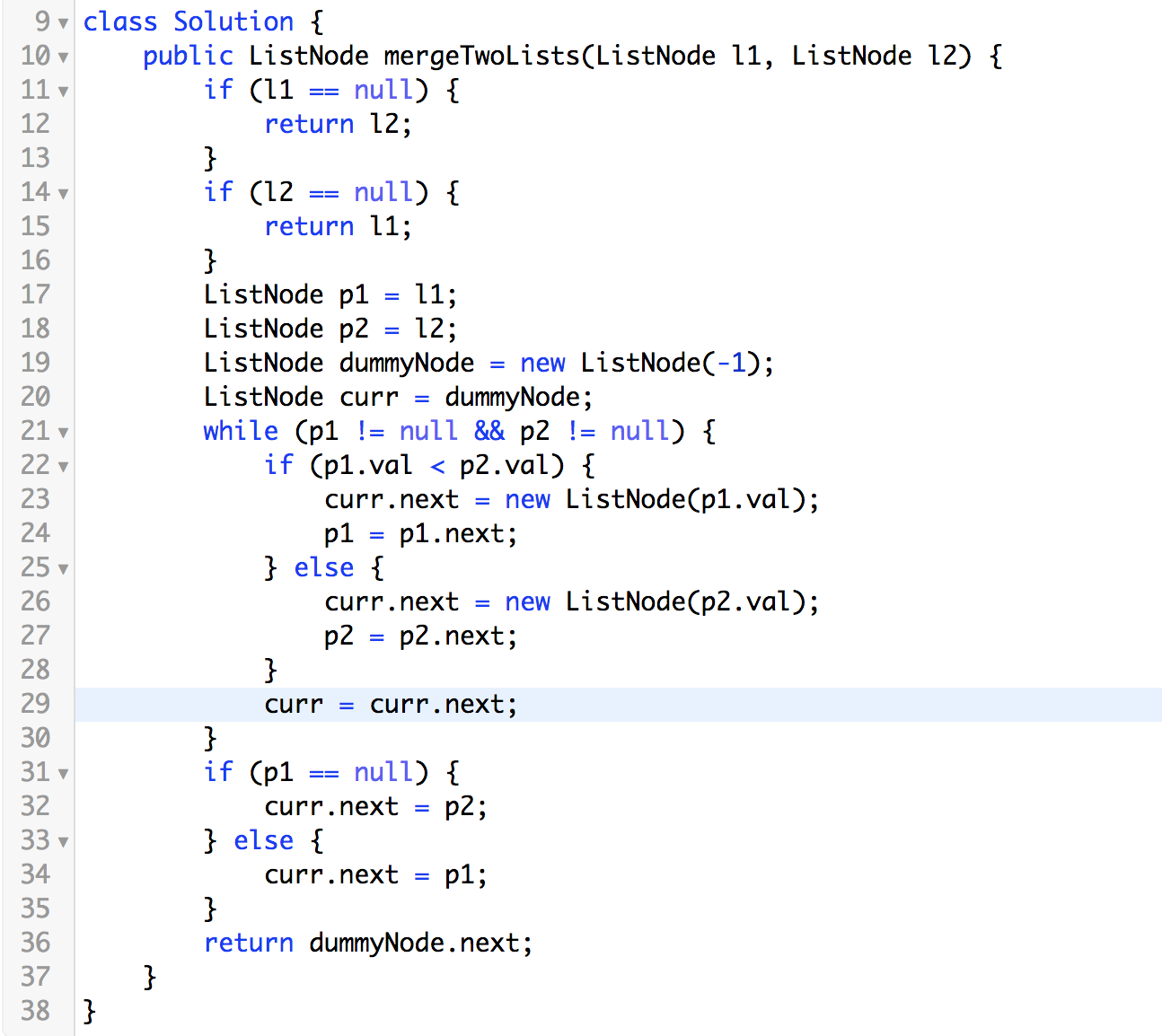


说明：我的写法我觉得不太好的一个地方就是最简单的情况没有写在最前面，虽然也通过了。

* （完成）练习2：LeetCode 第 21 题。合并两个有序的链表。

https://leetcode-cn.com/problems/merge-two-sorted-lists/description/

提示：非常常规的一个问题，只要是熟悉了归并排序，不用画图都可以直接写出正确的答案。

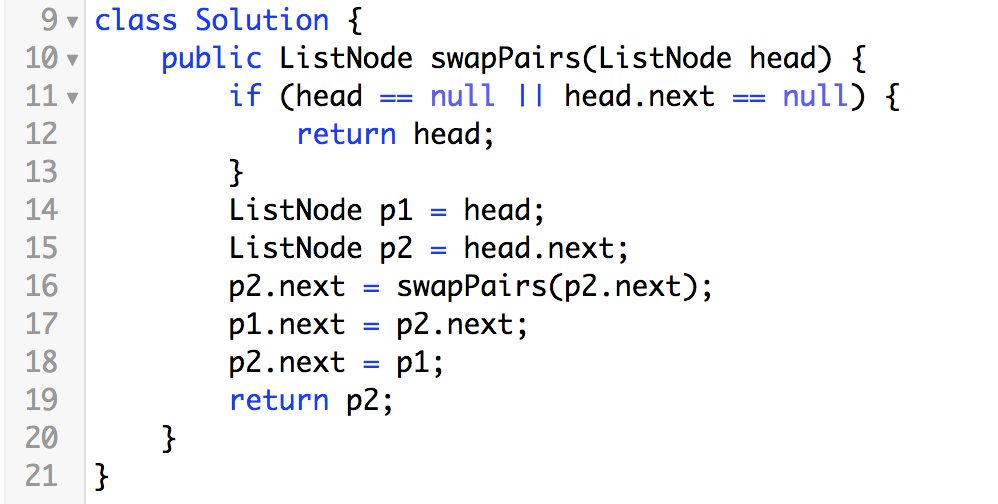


5-4 复杂的穿针引线（4题）

* （完成）例题1：LeetCode 第 24 题。两两交换链表中的节点。

解法1：指针的交换。

解法2：递归写法。



总结：递归的写法很简洁，不用去思考节点中指针的指向，可以达到简化问题的作用，毕竟我认为处理节点的 next 指向谁是一件比较容易出错的事情。

* 练习1：LeetCode 第 25 题。k个一组翻转链表。

有了上一题作为铺垫，自然也会想到使用递归来解决这个问题。

* 练习2：LeetCode 第 147 题。

题目要求：对链表进行插入排序。

题目难度：中等。

中文网址：https://leetcode-cn.com/problems/insertion-sort-list/description/

英文网址：

求解关键：

我的解答：

参考资料：

* 练习3：LeetCode 第 148 题。

题目要求：在 O(n log n) 时间复杂度和常数级空间复杂度下，对链表进行排序。

题目难度：中等。

中文网址：https://leetcode-cn.com/problems/sort-list/description/

英文网址：

求解关键：分治，归并排序，这道题基本就只有这一种思路。要使用归并排序（并且是**自底向上的归并排序**），就要找到链表中间的那个元素，可以设置两个从头结点开始的节点（一个小技巧）。

我的解答：

参考资料：

5-5 不仅仅是穿针引线（1题）

* 例题1：LeetCode 第 237 题。

题目要求：请编写一个函数，使其可以删除某个链表中给定的（非末尾的）节点，您将只被给予要求被删除的节点。

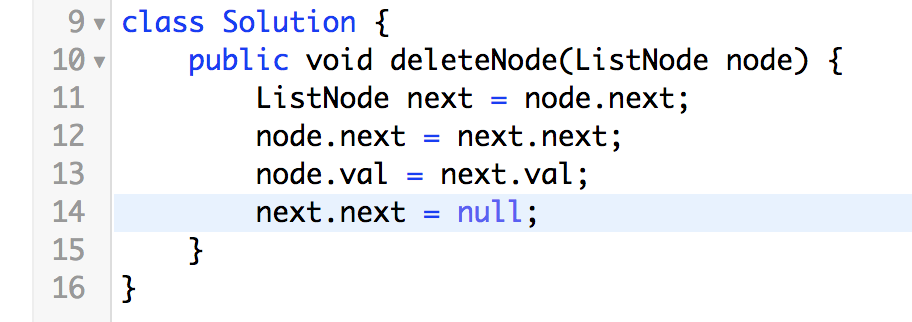
题目难度：**简单**。

中文网址：https://leetcode-cn.com/problems/delete-node-in-a-linked-list/description/

英文网址：

求解关键：弄清楚题意，题目只给你这个节点的引用，让你从链表中删除这个节点，所以不要和 dummyNode 那种套路混在一起了。

我的解答：



参考资料：

5-6 链表与双指针（4题）

* （完成）例题1：LeetCode 第 19 题。

题目要求：给定一个链表，删除链表的倒数第 n 个节点，并且返回链表的头结点。

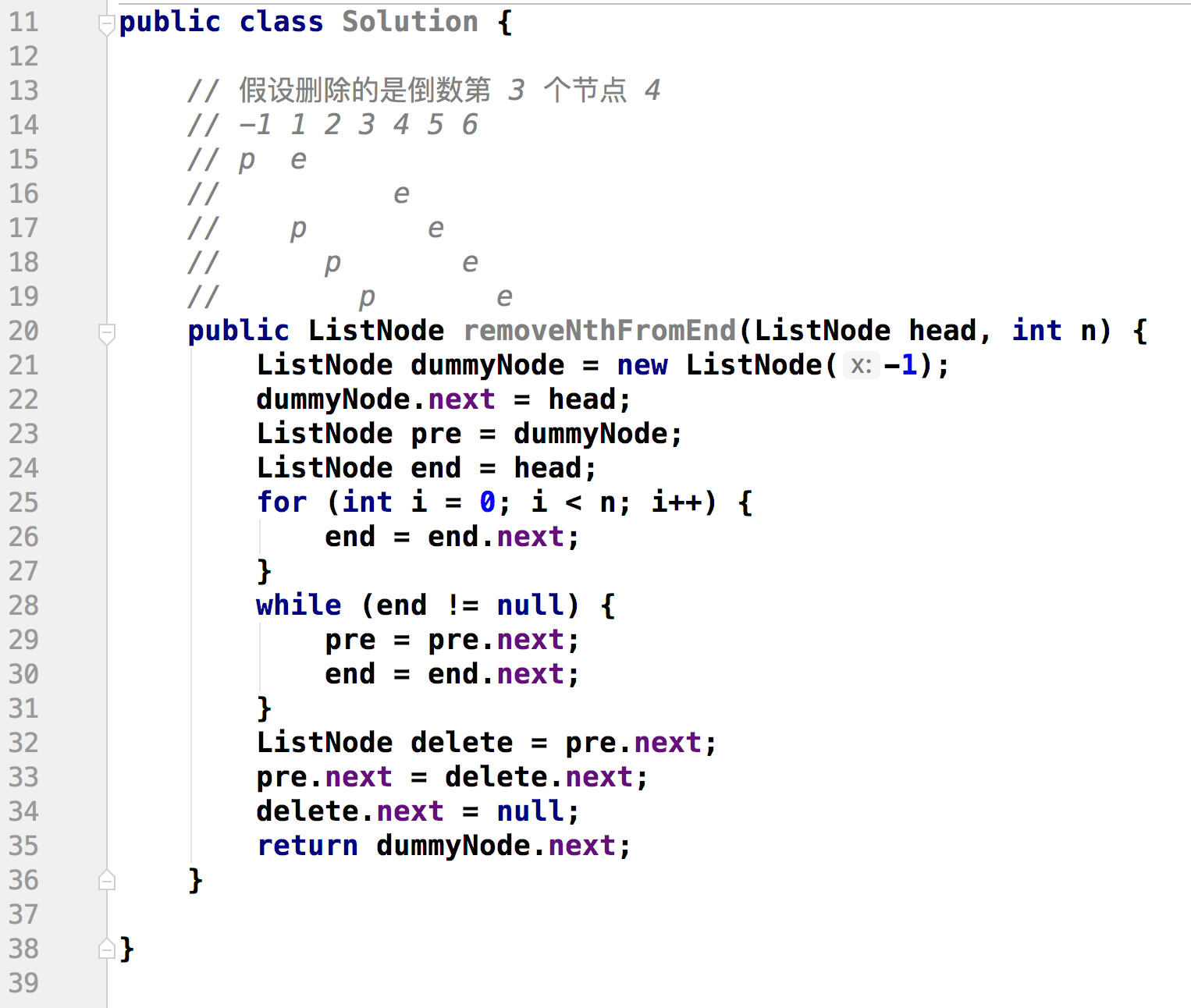
题目难度：中等。

中文网址：https://leetcode-cn.com/problems/remove-nth-node-from-end-of-list/description/

英文网址：

求解关键：其实只要掌握了如何找到距离末尾 n 个元素的位置，就很容易了。还要注意的就是边界值的选取，其实往往我们认为的值与正确值无非就是 +1 或者 -1 ，为了避免因为粗心出错，我们可以拿一个距离的例子，发现变量之间的关系。

我的解答：



参考资料：

* 练习1：LeetCode 第 61 题。

题目要求：给定一个链表，旋转链表，将链表每个节点向右移动 k 个位置，其中 k 是非负数。

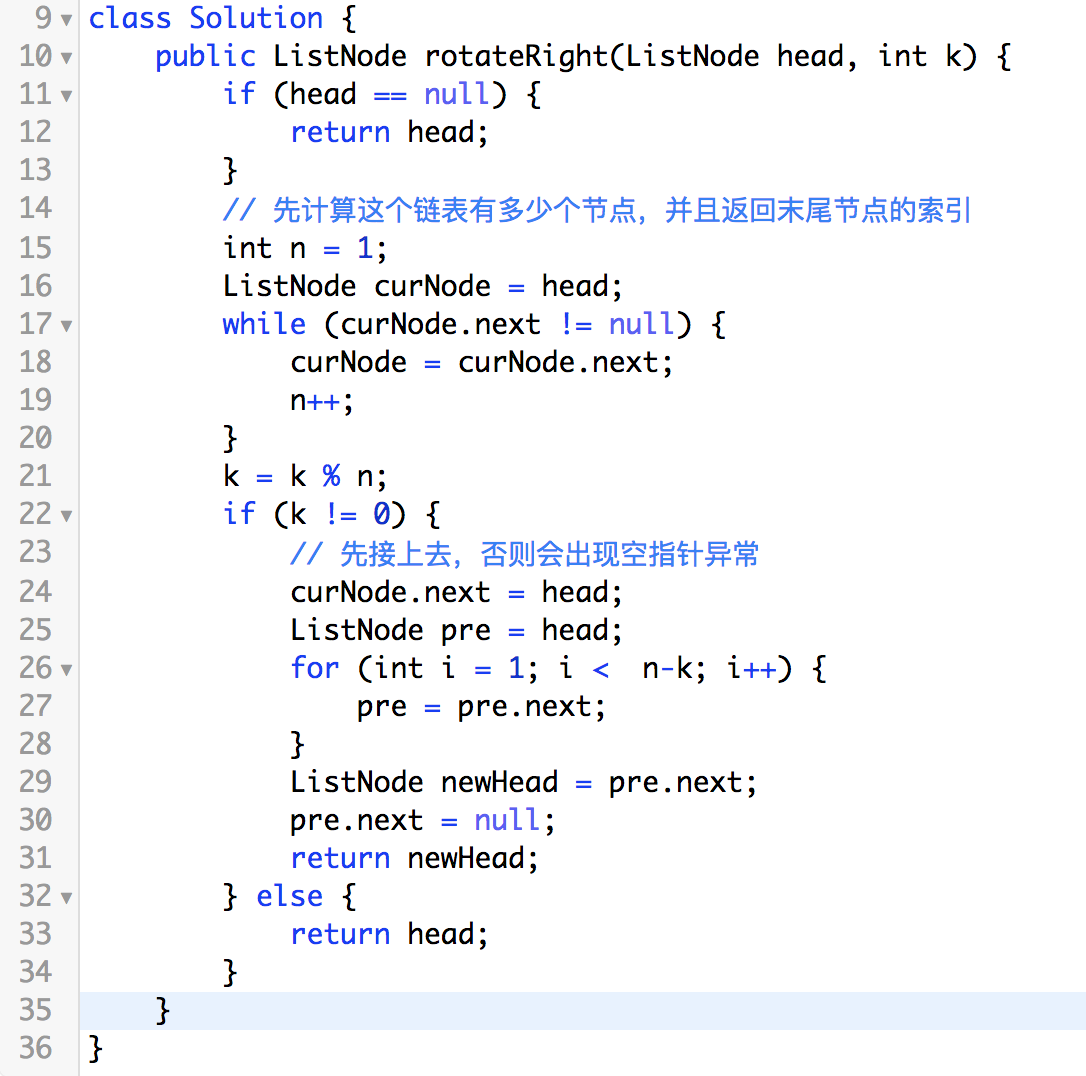
题目难度：

中文网址：https://leetcode-cn.com/problems/rotate-list/description/

英文网址：

求解关键：问题本身不难，但是要处理一些细节。（1）一定要先求出链表的总长度；（2）求得总长度的时候，顺便标记好末尾节点，并且把末尾节点的 next 指针指到头结点去，形成环，否则容易出现空指针异常；（3）到底多少 pre 指针还要走多少步，举 1-2 个具体的例子带进去就知道了。

我的解答：



参考资料：

* 练习2：LeetCode 第 143 题。

题目要求：重排链表。

题目难度：中等。

中文网址：https://leetcode-cn.com/problems/reorder-list/description/

英文网址：

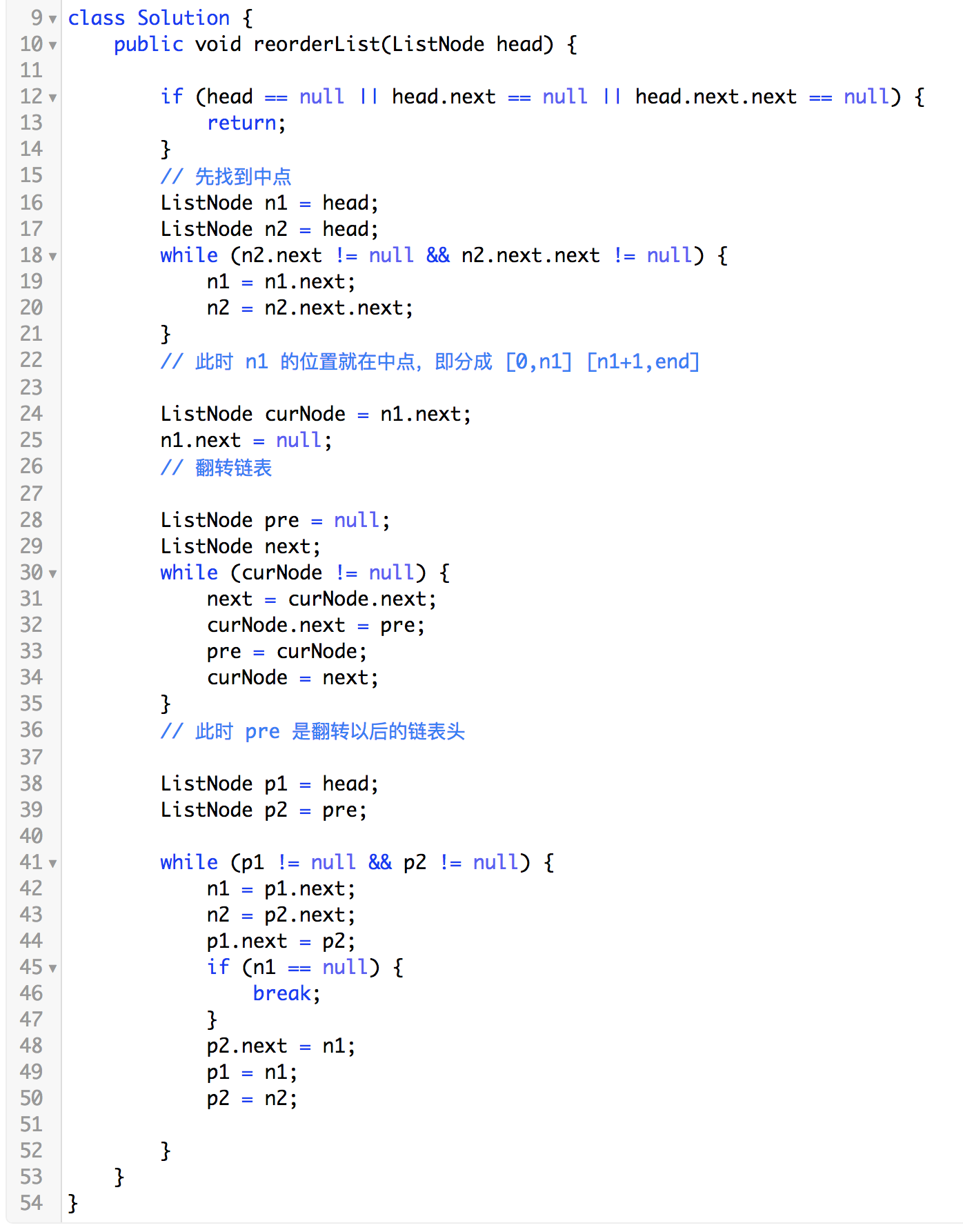
求解关键：找到中点，然后翻转。考查的都是常规的链表操作，把它们合在一起就是这道题的解答。

（1）找链表的中点；

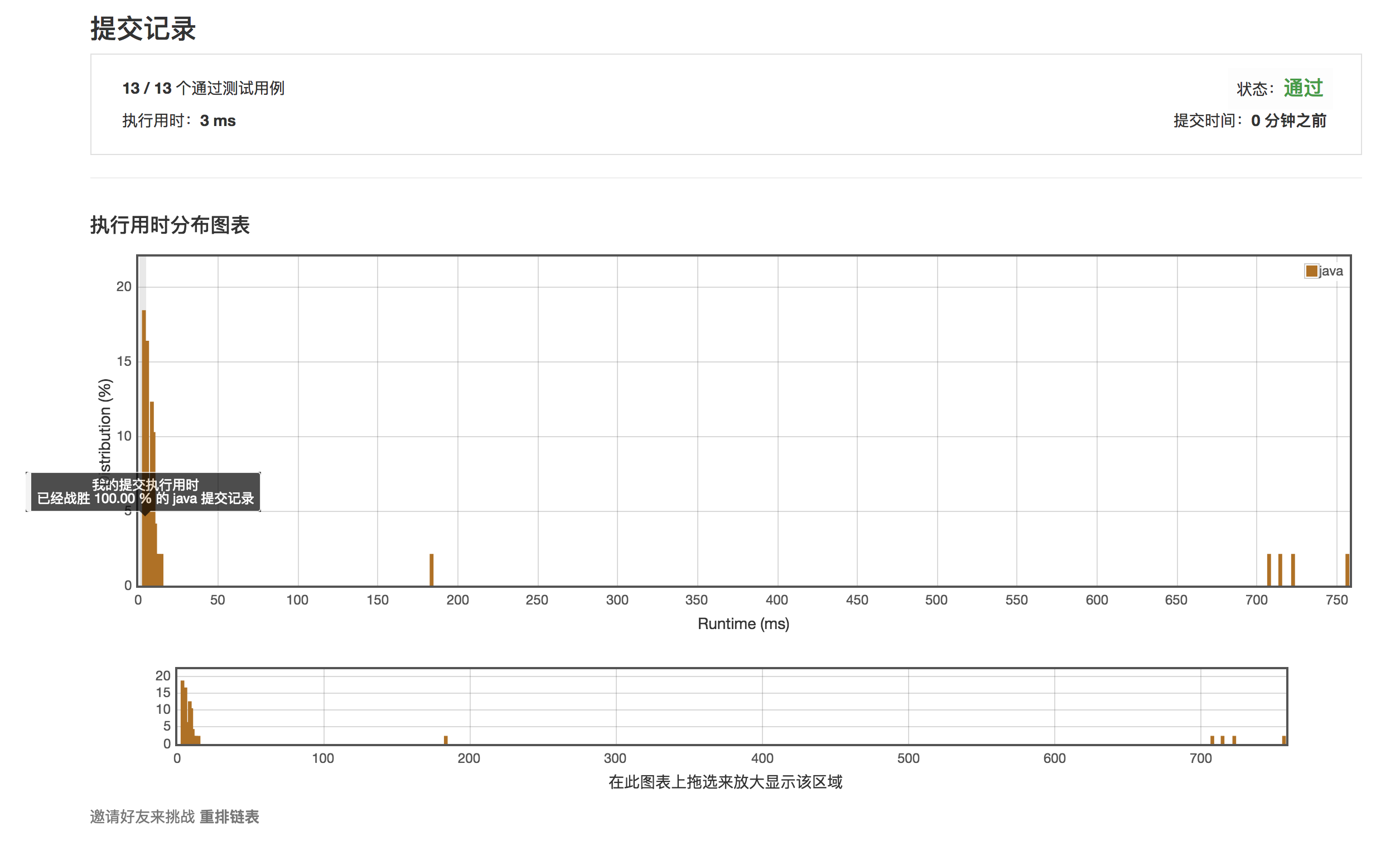
（2）翻转链表；

（3）用 S 形连接两个链表。

我的解答：



值得一提的是，这道题的解法，其实就是设置了很多指针，“穿针引线”，把 next 指针指来指去，就完事了。



参考资料：

* 练习3：LeetCode 第 234 题。

题目要求：回文链表。

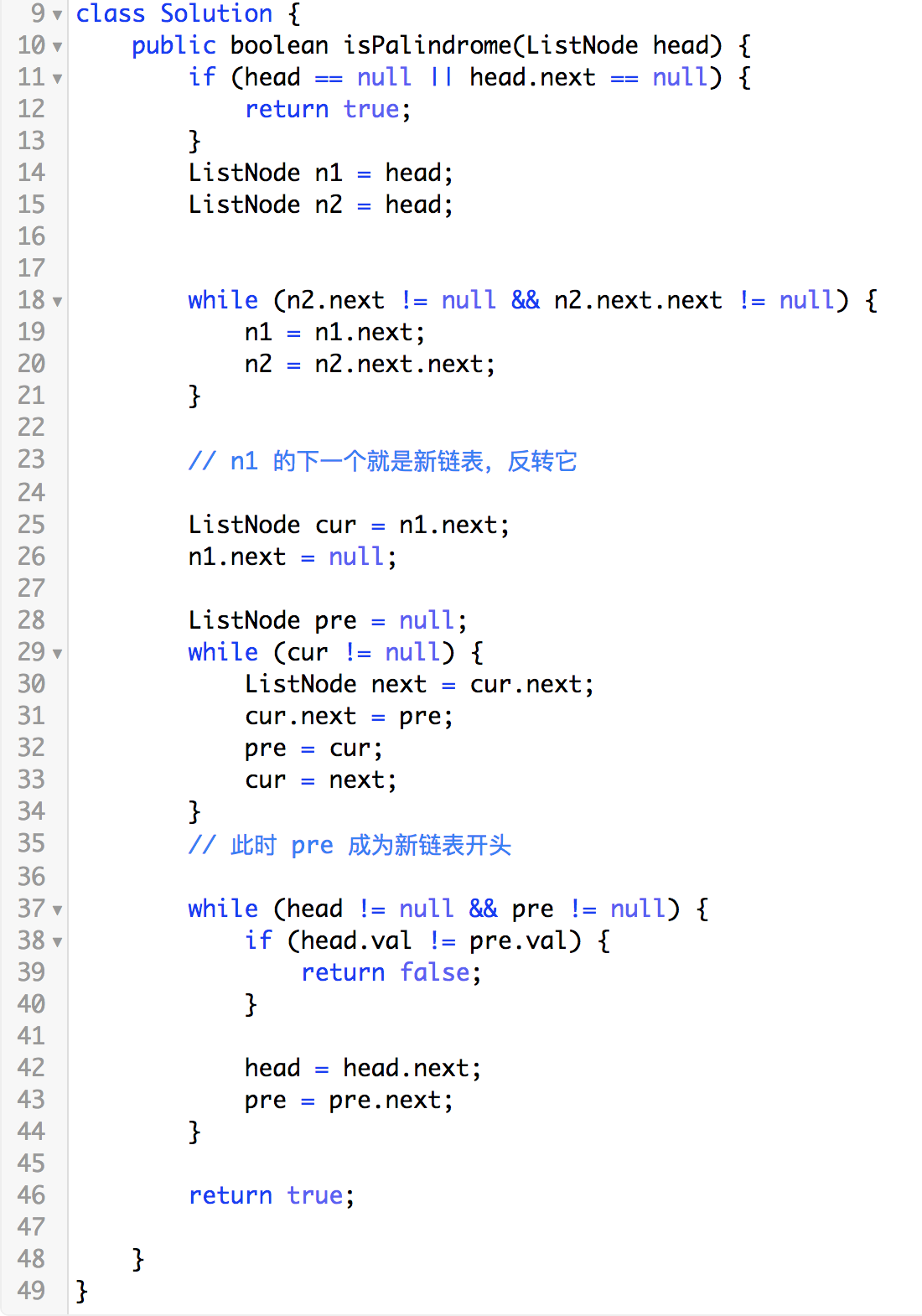
题目难度：简单

中文网址：https://leetcode-cn.com/problems/palindrome-linked-list/description/

英文网址：https://leetcode.com/problems/palindrome-linked-list/description/

求解关键：仍然是考查关于链表的基本操作：（1）找中点；（2）翻转链表。

我的解答：



参考资料：