今天说个Easy题。但是，请用严肃的眼光看待它，因为方法真的很巧妙！题目叫[**Intersection of Two Linked Lists**](https://leetcode.com/problems/intersection-of-two-linked-lists/)**，**题目并不很难，大家可以先随意感受一下。10分钟写不出来就缴枪吧，因为即便你写出来了，方法估计也不是什么可取的好办法，让他随风去吧。

》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》

先介绍题目：**Write a program to find the node at which the intersection of two singly linked lists begins.**

 For example, the following two linked lists:

A: a1 → a2

↘

c1 → c2 → c3

↗

B: b1 → b2 → b3

begin to intersect at node c1.

意思就是：给出两个不同的list 返回第一个相同的节点（默认后面都相同，大大降低了难度，可以思考一下如果后面有可能不同咋办。）

》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》

EASY题这个帽子决定了我们几乎可以上来就写，毕竟考的不难嘛，仔细想想问题不大~

动笔！

第一件事！null check，easy。但是与此同时，我的习惯是，再思考一下，什么情况下会返回null，而不是仅仅想一下输入是null怎么处理。

顺着想下来，无外乎两种情况： 1，其中一个是null； 2，俩list完！全！不同。比如

1-> 2 -> 3-> null 和 4-> 5-> 6-> null 非要让你返回，那除了null，臣妾也想不到了。

再接下来，步入正轨，解题。题目本身的难度决定了即使我们用最直白的方法解题，时间复杂度也不会很高，长驱直入即可：

既然要我们找到第一个公共的节点，最直观的想法就是，倒着数嘛！第一个不一样的后一个就是了！（这句话有点奇怪，仔细想想其实就是你脑子里的想法）**然而**，singly linkedlist并不能让我们简单的倒着数……recursive貌似可以，却得不偿失，pass！

那退而求其次，倒着不能就顺着数。

在此之前！首先思考一下，为什么我们愿意倒着数而不是正着数。原因有： 1，我们可以保证从结尾开始数，两个list末尾长度是一致的，不会出现类似扣子错位的情况； 2，我们保证后面是一样的，所以找第一个不一样的就好了，方便。

照着这个思路去理解顺着数的问题，那就是：万一俩list长度不一样怎么办？【凉拌！】那就按短的那个算！长的那段截取！嘛！

说到这，大家都会了吧？两边遍历，第一遍找到两个长度，从公共长度开始比较，第一个相同的即是我们想要寻找的node，大功告成，提交开始下一道！撒花~鼓掌~啦啦啦~

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

**球头麻袋！【日语音译：等等！】**

你以为我为什么要用这道简单题来说事？

大神告诉你，其实还有一种神奇的算法。

---------------------------------------------------------------------------------------------------------------------

【以下为背景介绍】

作为预备，看过Cracking coding interview 150的同学应该对一道题还有印象。给你两个string，a, b. 请你判断a是不是b的anagram。意思就是比如a="abcdef" b="defabc". a是否由b从某一个char作为开始循环而来的。

题目不难，直观做法为把string排序，然后俩一样即可。

（别以为我不知道你在想可以把a按各个字母开头循环一次，和b比较，我一开始也是这么想的。然后被作者提都不提一句的鄙视了，真是醉了。）

同时作者又给出了 xyxy解题方案。意思为我们可以肯定如果ab是anagram的话，一定可以将a分割成xy两部分。比如上例中：a = "abc"+"def" = x + y，以保证b = y + x。 不用证明啦，想想就对~于是乎 我们把a重复两次，得到aa = xyxy = x + yx + x = x + b + x。

接下来判断b是不是a的一部分就好了。就问一句，吓人不吓人。被作者的智商活活碾压了十条街。

【以上为背景介绍】

--------------------------------------------------------------------------------------------------------------

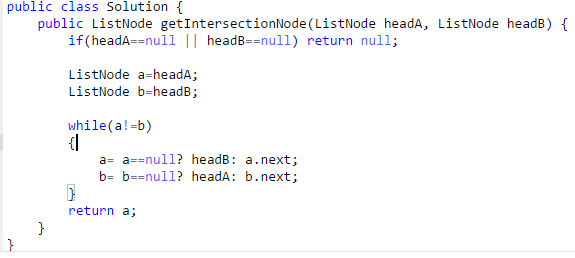
你想想这个背景，能用上么？

【以上为本世纪最大的hint】

我们把两个list， a 和 b 分别组成 ab 和 ba 【这明显和背景不一样啊……】【触类旁通啊！！】然后，奇迹就发生了。

贴上代码，神算法要自己去体会，看不明白写两个list自己顺着代码比划比划。一会儿你就服了~

（试试两个list完全不一样会怎么样，简直智能的快要飞起来了！）



=====================================================================

就是这么酷炫，选这道题的原因是：虽然题目本身并不难解决，但是仍然存在令人眼前一亮的算法方案。并且这是一个很值得推广的方法，把他记在脑子里，留有印象，一定会有很大的帮助。