4/29 Leetcode

题目：判断环形链表

给你一个链表的头节点 head ，判断链表中是否有环。

如果链表中有某个节点，可以通过连续跟踪 next 指针再次到达，则链表中存在环。 为了表示给定链表中的环，评测系统内部使用整数 pos 来表示链表尾连接到链表中的位置（索引从 0 开始）。注意：pos 不作为参数进行传递 。仅仅是为了标识链表的实际情况。

如果链表中存在环 ，则返回 true 。 否则，返回 false 。

示例 1：



输入：head = [3,2,0,-4], pos = 1

输出：true

解释：链表中有一个环，其尾部连接到第二个节点。

示例 2：



输入：head = [1,2], pos = 0

输出：true

解释：链表中有一个环，其尾部连接到第一个节点。

示例 3：



输入：head = [1], pos = -1

输出：false

解释：链表中没有环。

来源：力扣（LeetCode）

链接：<https://leetcode-cn.com/problems/linked-list-cycle>

思路：

这道题要求是判断链表，链表其实是一个很好的储存方法，可以方便的进行增加，删除链表，但是相比于数组也是有缺点的，比如不能直接通过下标来访问链表，需要通过头链表慢慢往后遍历，所在判断环形链表上会有点困难。以下是我的思路，

1. 定义两个指针
2. 让一个指针一次向前一步，一个向前两步
3. 如果能相遇，则代表是环形链表，就像跑步一样，在一个圆形跑到，前面的只要足够快，总会追上后面的人。
4. 若两个指针重叠，则代表是环形链表，经历一定次数后还没相遇，则不是环形链表

这样就可以解决这道题了，在期间遇到的问题也比较少，比如在判断循环结束时的条可能出现越界（不是环形链表），但是稍加思考也可以自己解决掉这个问题。

总结：

写完这个题目我认为收获就是对链表的其他算法进行练习，更好的理解了链表的优势和劣势，知道怎么处理链表的一些复杂结构，通过上网查询，我也学到了怎么处理查询链表倒数第N个数的方法，所以收获还是比较大的，更好的掌握了链表的有关算法。