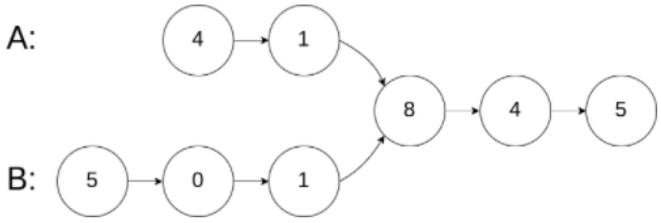
user: sakura

第53题——两个链表的第一个公共结点输入两个链表，找出它们的第一个公共节点。

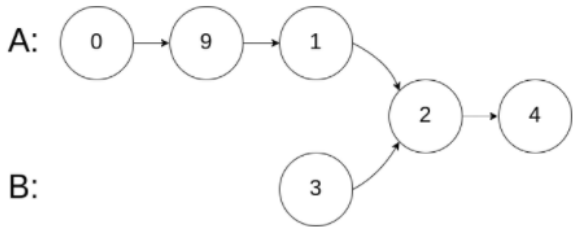
题目描述:输入两个链表，找出它们的第一个公共节点。

示例 1:



输入： intersectVal = 8, listA = [4,1,8,4,5], listB = [5,0,1,8,4,5], skipA = 2, skipB = 3
输出： Reference of the node with value = 8
输入解释： 相交节点的值为 8 （注意，如果两个列表相交则不能为 0）。从各自的表头开始算起，链表 A 为 [4,1,8,4,5]，链表 B 为 [5,0,1,8,4,5]。在 A 中，相交节点前有 2 个节点；在 B 中，相交节点前有 3 个节点。

示例 2:



输入： intersectVal = 2, listA = [0,9,1,2,4], listB = [3,2,4], skipA = 3, skipB = 1
输出： Reference of the node with value = 2
输入解释： 相交节点的值为 2 （注意，如果两个列表相交则不能为 0）。从各自的表头开始算起，链表 A 为 [0,9,1,2,4]，链表 B 为 [3,2,4]。在 A 中，相交节点前有 3 个节点；在 B 中，相交节点前有 1 个节点。

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode(int x) : val(x), next(NULL) {}
 * };
 */
class Solution {
public:
    ListNode *getIntersectionNode(ListNode *headA, ListNode *headB) {
        ListNode* head_a=headA;
        ListNode* head_b=headB;
        while(head_a != head_b)
        {
            if(head_a==nullptr){
                /*复位到B的第一个结点*/
```

```
                head_a=headB;
            }
            else{
                head_a=head_a->next;
            }

            if(head_b==nullptr){
                /*复位到A的第一个结点*/
                head_b=headA;
            }
            else{
                /*向后遍历*/
               head_b=head_b->next;
            }
        }
        return head_a;
    }
};
```

解题思路:

> 本题参考自LeetCode力扣用户——腐烂的橘子的题解, 思路就是, 两个指针同时遍历,每当遍历到末尾时, 就把它重新定位到另一条链表的首个结点即可, 循环条件为两个节点不相等, 当循环跳出时则两个指针相遇, 此时我们的结点就是第一个共节点。