160. Intersection of Two Linked Lists

Description | Hints | Submissions | Discuss | Solution

Pick One

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.

Notes:

- If the two linked lists have no intersection at all, return `null`.
- The linked lists must retain their original structure after the function returns.
- You may assume there are no cycles anywhere in the entire linked structure.
- Your code should preferably run in O(n) time and use only O(1) memory.

```
public class ListNode {
           int val;
           ListNode next;
           ListNode(int x) {
              val = x;
              next = null;
           }
       }
```

```
/*
 * 如果两个链长度相同的话，那么对应的一个个比下去就能找到，所以只需要把长链表变短即可。具体算法为：分别遍历两个链表，得到分别对应的长度。然后求长度的差值，把较长的那个链表向后移动这个差值的个数，然后一一比较即可
 */
public ListNode getIntersectionNode(ListNode headA, ListNode headB) {
    int len1 = 0;
    int len2 = 0;
    ListNode p1 = headA, p2 = headB;
    if(p1 == null || p2 == null)
        return null;

    while (p1 != null) {
             len1 ++;
             p1 = p1.next;
```

```
        }

    while (p2 != null) {
            len2 ++;
            p2 = p2.next;
        }

    int diff = 0;
    p1 = headA;
    p2 = headB;

    if(len1 > len2) {
        diff = len1 - len2;
        int i = 0;
        while (i < diff) {
                p1 = p1.next;
                i ++;
            }
    }else {
            diff = len2 - len1;
            int i = 0;
            while (i < diff) {
                p2 = p2.next;
                i ++;
            }
        }
    while (p1 != null && p2 != null) {
            if(p1.val == p2.val)
                return p1;
            else {
                p1 = p1.next;
                p2 = p2.next;
            }
        }
    return null;
}
```