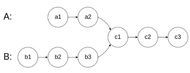
160. Intersection of Two Linked Lists

March 9, 2016 | 217.5K views

Average Rating: 3.97 (218 votes)

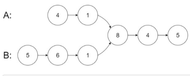
Write a program to find the node at which the intersection of two singly linked lists begins.

For example, the following two linked lists:



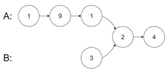
begin to intersect at node c1.

Example 1:




```
Input: intersectVal = 8, listA = [4,1,8,4,5], listB = [5,6,1,8,4,5], skipA = 2, skipB
Output: Reference of the node with value = 8
Input Explanation: The intersected node's value is 8 (note that this must not be 0 if
```

Example 2:




```
Input: intersectVal = 2, listA = [1,9,1,2,4], listB = [3,2,4], skipA = 3, skipB = 1
Output: Reference of the node with value = 2
Input Explanation: The intersected node's value is 2 (note that this must not be 0 if
```

Example 3:


```
Input: intersectVal = 0, listA = [2,6,4], listB = [1,5], skipA = 3, skipB = 2
Output: null
Input Explanation: From the head of A, it reads as [2,6,4]. From the head of B, it rea
Explanation: The two lists do not intersect, so return null.
```

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.
- Each value on each linked list is in the range `[1, 10^9]`.
- Your code should preferably run in O(n) time and use only O(1) memory.

Solution

Approach 1: Brute Force

For each node a_i in list A, traverse the entire list B and check if any node in list B coincides with a_i.

Complexity Analysis

- Time complexity : $O(mn)$.
- Space complexity : $O(1)$.

Approach 2: Hash Table

Traverse list A and store the address / reference to each node in a hash set. Then check every node b_i in list B: if b_i appears in the hash set, then b_i is the intersection node.

Complexity Analysis

- Time complexity : $O(m + n)$.
- Space complexity : $O(m)$ or $O(n)$.

Approach 3: Two Pointers

- Maintain two pointers pA and pB initialized at the head of A and B, respectively. Then let them both traverse through the lists, one node at a time.
- When pA reaches the end of a list, then redirect it to the head of B (yes, B, that's right.); similarly when pB reaches the end of a list, redirect it the head of A.
- If at any point pA meets pB, then pA/pB is the intersection node.
- To see why the above trick would work, consider the following two lists: A = {1,3,5,7,9,11} and B = {2,4,9,11}, which are intersected at node '9'. Since B.length (=4) < A.length (=6), pB would reach the end of the merged list first, because pB traverses exactly 2 nodes less than pA does. By redirecting pB to head A, and pA to head B, we now ask pB to travel exactly 2 more nodes than pA would. So in the second iteration, they are guaranteed to reach the intersection node at the same time.
- If two lists have intersection, then their last nodes must be the same one. So when pA/pB reaches the end of a list, record the last element of A/B respectively. If the two last elements are not the same one, then the two lists have no intersections.

Complexity Analysis

- Time complexity : $O(m + n)$.
- Space complexity : $O(1)$.

Rate this article:

Previous Next

Comments: 120 Sort By

Type comment here... (Markdown is supported)

Preview Post

TrafalgarZZZ 165 August 28, 2018 3:07 PM

Here's my solution which is like approach 3 but a little different. I store the size of ListA and ListB as len1 and len2. Then I reset the pointers to headA and headB and find the difference between len1 and len2, and then let the pointer of the longer list proceed by the difference between len1 and len2. Finally, traverse through the lists again, the intersection node can be easily found.

Read More

165 Share Reply

SHOW 22 REPLIES

jianchao-li 14342 July 30, 2018 3:35 PM

My accepted codes for the three approaches. Maybe someone could optimize them further and add them to the Solution? Thank you.

Approach 1: Brute Force (527ms)

Read More

108 Share Reply

SHOW 14 REPLIES

lenchen1112 976 September 13, 2018 8:21 PM

Solution 3 in Python:

```
# Definition for singly-linked list.
# class ListNode(object):
#     def __init__(self, x):
```

Read More

60 Share Reply

SHOW 8 REPLIES

Marcus_X 125 March 14, 2019 5:48 AM

You can do this by simply counting the lengths of both lists. Pass one: count lengths while searching to see if they have the same tail. Diff tails? Return NULL. Pass two: In an ideal world the lists would be the same length and simply walking them forward until they had the same pointer would be enough. To deal with uneven lists skip the longer list ahead X nodes, where X = abs(len1-len0). Now if you walk both lists forward one at a time they meet at the intersection point. This solution is still O(N) time and

Read More

58 Share Reply

SHOW 7 REPLIES

Anow 26 March 19, 2019 11:57 AM

Is there any problem with example 1? Why not 1 but 8 ?

26 Share Reply

SHOW 6 REPLIES

terrible_whiteboard 627 May 19, 2020 6:20 PM

I made a video if anyone is having trouble understanding the solution (clickable link)
https://youtu.be/c7dOI-hDa2Q

[redacted]

Read More

18 Share Reply

jkvavadiya 17 October 19, 2018 1:16 AM

```
public ListNode getIntersectionNode(ListNode headA, ListNode headB) {
ListNode pA = headA, pB = headB;
while (pA != pB) {
    pA = pA != null ? pA.next : headB;
    pB = pB != null ? pB.next : headA;
```

Read More

17 Share Reply

SHOW 1 REPLY

qiuyu8290 45 October 27, 2018 11:54 PM

similar idea with counting A, B length:

```
public class Solution {
    public ListNode getIntersectionNode(ListNode headA, ListNode headB) {
        ((ListNode curr = headA;
```

Read More

14 Share Reply

SHOW 2 REPLIES

ludi 9 December 19, 2017 1:02 PM

this question does not cover the following case:
if headA and headB has no intersection, the solution will go to dead loop.

10 Share Reply

SHOW 3 REPLIES

RubbishThird 9 August 29, 2018 10:21 AM

reverse either list, if two list have the intersection, we can reach the other list from the other list.

```
class Solution(object):
    def getIntersectionNode(self, headA, headB):
        """
```

Read More

9 Share Reply

1 2 3 4 5 6 ... 11 12