Intersection Point In Y Shaped Linked Lists

solved by	Senan
Platform	GeeksForGeeks
difficulty	Medium
tags	Linked List
language	C++
solved on	@13/11/2024
link	https://www.geeksforgeeks.org/problems/intersection-point-in-y-shapped-linked-lists/1
Completion	✓

Intuition

In Y-shaped linked lists, the two lists eventually converge at a common node after which they share all the following nodes. To find the intersection point, we can use two pointers and traverse each list. By switching to the other list when reaching the end, both pointers eventually align at the intersection node after equalizing the path length.

Approach

1. Initialize two pointers, `temp1` pointing to `head1` and `temp2` to `head2`.
2. Traverse both lists simultaneously.
   - If a pointer reaches the end of one list, reset it to the head of the other list.
   - Continue moving the pointers until they meet at the intersection node or both reach `NULL`.
3. When `temp1` and `temp2` meet, that node is the intersection point.
4. If they both reach `NULL` without meeting, it means there is no intersection.

Complexity

Time Complexity:

The time complexity is **O(m + n)**, where **m** and **n** are the lengths of the two linked lists. Each pointer traverses each list at most once.

Space Complexity:

The space complexity is **O(1)** since we only use a constant amount of additional space.

Code

```
class Solution {
  public:
    // Function to find intersection point in Y-shaped Linked Lists.
    int intersectPoint(Node* head1, Node* head2) {
        if (head1 == NULL || head2 == NULL) return -1;

        Node* temp1 = head1;
        Node* temp2 = head2;

        while (temp1 != temp2) {
            temp1 = temp1->next;
            temp2 = temp2->next;

            if (temp1 == NULL) temp1 = head2;
            if (temp2 == NULL) temp2 = head1;
        }

        return (temp1 == NULL) ? -1 : temp1->data;
    }
};
```