Add Two Numbers Represented by Linked List LeetCode

You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list

Example:

List 1: 5 -> 6 -> 4 -> 9

List 2: 2 -> 4 -> 9

Result: 7 -> 0 -> 4 -> 0 -> 1

Approach 1: Adds two numbers represented by linked lists iteratively

In this approach, we simulate the process of addition as we do it manually, from the least significant digit to the most significant digit. We traverse both linked lists simultaneously, adding the corresponding digits and carry. If one of the linked lists has more digits, we treat the missing digits as zeroes. We keep track of the carry as we add digits and update it for the next iteration. The result is stored in a new linked list.

1. Initialize a carry variable to 0.
2. Create two pointers to traverse the linked lists, and a pointer to the result linked list.
3. Traverse both linked lists:
    - Add the values of the current nodes along with the carry.
    - Update the carry by dividing the sum by 10.
    - Append the digit (sum % 10) to the result linked list.
4. Return the result linked list.
5. **Time Complexity: O(max(N, M)), where N and M are the lengths of the two linked lists.**
6. **Space Complexity: O(max(N, M)), as we create a new linked list to store the result.**

Approach 2: Add two numbers represented by linked lists recursively

In this approach, we use recursion to add digits starting from the least significant digit and propagate the carry. At each step, we calculate the sum of the current digits and the carry, update the carry for the next iteration, and create a new node with the sum % 10. We continue the recursion until both linked lists are exhausted and the carry is zero.

1. Base case: If both linked lists are empty and there's no carry, return nullptr.

2. Calculate the sum of the current digits and the carry:
    - If linked list 1 has a node, add its value to the sum and move the pointer.
    - If linked list 2 has a node, add its value to the sum and move the pointer.
3. Calculate the carry for the next iteration by dividing the sum by 10.
4. Create a new node with the value (sum % 10).
5. Make a recursive call to the next digits, passing the carry.
6. Link the current node to the result of the recursive call.
7. Return the current node.
8. **Time Complexity: O(max(N, M)), where N and M are the lengths of the two linked lists.**
9. **Space Complexity: O(max(N, M)), as the recursive call stack can go up to the maximum length of the linked lists.**