

add two numbers leetcode solution

Add two numbers LeetCode solution is a popular problem that often appears in coding interviews and algorithm practice. It presents a unique challenge that requires a good understanding of linked lists and basic arithmetic operations. In this article, we will explore the problem statement, discuss various approaches to solve it, and provide a detailed solution using Python. By the end of this article, you will have a solid grasp of the problem and how to tackle it effectively.

Understanding the Problem Statement

The "Add Two Numbers" problem can be summarized as follows:

You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order, which means the 1's digit is at the head of the list. You need to add the two numbers and return it as a linked list.

For example, if the input linked lists are:

- List1: 2 -> 4 -> 3 (represents the number 342)
- List2: 5 -> 6 -> 4 (represents the number 465)

The output linked list should be:

- Result: 7 -> 0 -> 8 (represents the number 807)

This problem tests your ability to manipulate linked lists and perform basic arithmetic operations.

Key Concepts to Understand

Before diving into the solution, let's review some key concepts that are essential for understanding the problem:

1. Linked Lists

A linked list is a linear data structure where elements are stored in nodes. Each node contains a value and a reference (or pointer) to the next node in the sequence. In this problem, we use linked lists to represent numbers, with the least significant digit at the head of the list.

2. Carry in Addition

When adding two digits, you may encounter a situation where the sum exceeds 9. In such cases, you need to carry over the extra value to the next significant digit. For example:

- Adding 7 and 5 gives you 12. You write down 2 and carry over 1 to the next digit.

3. Traversing Linked Lists

To solve the problem, you must traverse the linked lists node by node, extracting the values and performing the addition while taking care of the carry.

Approaches to Solve the Problem

There are several approaches to solving the "Add Two Numbers" problem. Here, we will discuss two primary methods:

1. Iterative Approach

This approach involves using a loop to traverse both linked lists simultaneously. You will maintain a pointer to the current node of the result linked list and update it as you compute the sum.

2. Recursive Approach

In the recursive approach, you can break down the problem into smaller subproblems. You can add the digits from the head of both lists and handle the carry recursively. This method is elegant but may lead to stack overflow for very long lists due to deep recursion.

In this article, we will focus on the iterative approach, as it is generally more efficient and easier to implement.

Iterative Solution

Let's go through the steps required to implement the iterative solution for adding two numbers represented by linked lists.

Step-by-Step Implementation

1. Initialize Pointers and Variables:

- Create a dummy node to simplify the result linked list construction.
- Use a pointer to keep track of the current position in the result list.
- Initialize a variable to hold the carry value (starting at 0).

2. Traverse the Linked Lists:

- Use a loop to iterate through both linked lists until you reach the end of both.
- At each step, retrieve the current values from both linked lists (0 if the list has been fully traversed).
- Calculate the sum of the two values and the carry.
- Update the carry for the next iteration.
- Create a new node for the result linked list with the digit obtained from the sum.

3. Handle Remaining Carry:

- If there is a carry left after the loop, create a new node with the carry value.

4. Return the Result:

- Return the next node of the dummy node, which points to the head of the result list.

Sample Code in Python

Here's how you can implement the above steps in Python:

```
```python
class ListNode:
 def __init__(self, val=0, next=None):
 self.val = val
 self.next = next
```

```

def addTwoNumbers(l1, l2):
 dummy = ListNode(0) Create a dummy node
 current = dummy Pointer to construct the result list
 carry = 0 Initialize carry

 while l1 or l2:
 Get the current values from the linked lists
 val1 = l1.val if l1 else 0
 val2 = l2.val if l2 else 0

 Calculate the sum and the carry
 total = val1 + val2 + carry
 carry = total // 10 Update carry for the next position
 current.next = ListNode(total % 10) Create a new node for the result
 current = current.next Move to the next node

 Move to the next nodes in the input lists
 l1 = l1.next if l1 else None
 l2 = l2.next if l2 else None

 If there's a carry left, add a new node
 if carry:
 current.next = ListNode(carry)

 return dummy.next Return the next node of the dummy, which is the head of the result list
 ...

```

## Complexity Analysis

When analyzing the complexity of the iterative solution, we can break it down as follows:

- Time Complexity:  $O(\max(n, m))$ , where  $n$  and  $m$  are the lengths of the two linked lists. In the worst case, we will traverse both lists completely.
- Space Complexity:  $O(\max(n, m))$ , as we are creating a new linked list to store the result.

## Conclusion

The **add two numbers LeetCode** solution is a fundamental problem that helps to solidify your understanding of linked lists and basic arithmetic. The iterative approach we discussed is efficient and straightforward, making it a popular choice among developers. By mastering this problem, you'll be better prepared for similar challenges in coding interviews and software development tasks. Keep practicing, and soon you'll be able to tackle even more complex problems with confidence!

## Frequently Asked Questions

### What is the problem statement for 'Add Two Numbers' on LeetCode?

The problem requires you to add two numbers represented by linked lists, where each node contains a single digit, and the digits are stored in reverse order.

### What are the constraints for the 'Add Two Numbers' problem?

The linked lists have a length of at least 1 and at most 100, and each node's value is a digit from 0 to 9.

### What is the expected time complexity for an optimal solution to this problem?

The expected time complexity is  $O(\max(N, M))$ , where  $N$  and  $M$  are the lengths of the two linked lists.

**What data structure is primarily used to represent the numbers in the 'Add Two Numbers' problem?**

The numbers are represented using singly linked lists.

**How do you handle carry in the addition process for the linked lists?**

You keep track of a carry variable that is updated as you add the digits from both linked lists, and if the sum exceeds 9, you carry over to the next significant digit.

**Can you describe a basic algorithm to solve the 'Add Two Numbers' problem?**

Iterate through both linked lists, add the corresponding digits along with the carry, create a new node for the result, and update the carry. Continue until both lists are fully traversed.

**What edge cases should be considered in the 'Add Two Numbers' problem?**

You should consider cases where one linked list is longer than the other, as well as cases where there is a carry left after processing both lists.

**What is a common mistake to avoid when solving the 'Add Two Numbers' problem?**

A common mistake is forgetting to check for a remaining carry after finishing the addition of the linked lists, which could lead to missing an additional node in the result.

**[Add Two Numbers Leetcode Solution](#)**

Add Two Numbers Leetcode Solution

## Related Articles

- [algebra 2 imaginary numbers](#)
- [algebra 1 workbook with answer key](#)
- [adaptive behavior scale assessment](#)

[Back to Home](#)