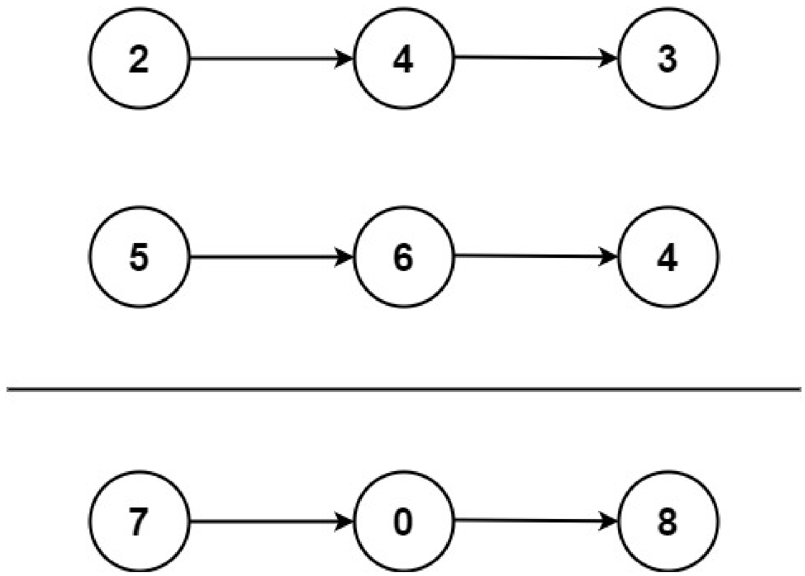
Problem 2. Add Two Numbers

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.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Example 1:




```
Input: l1 = [2,4,3], l2 = [5,6,4]
Output: [7,0,8]
Explanation: 342 + 465 = 807.
```

Example 2:

```
Input: l1 = [0], l2 = [0]
Output: [0]
```

Example 3:

```
Input: l1 = [9,9,9,9,9,9,9], l2 = [9,9,9,9]
Output: [8,9,9,9,0,0,0,1]
```

Constraints:

- The number of nodes in each linked list is in the range `[1, 100]`.
- `0 <= Node.val <= 9`
- It is guaranteed that the list represents a number that does not have leading zeros.

Solution(s)

Solution 1

Let $M, N \in \mathbb{N} \bigcup \{0\}$ be m- and n-digit numbers, $m, n \in \mathbb{N} \bigcup \{0\}$, and suppose that

$$\begin{aligned} M &= \sum_{i=0}^{m-1} m_i \times 10^i = m_{m-1} \cdots m_1 m_0, \\ N &= \sum_{i=0}^{n-1} n_i \times 10^i = n_{n-1} \cdots n_1 n_0, \end{aligned}$$

where juxtaposition represents concatenation and $m_0, \ldots, m_{m-1}, n_0 \ldots, n_{n-1} \in \{0, 1, \ldots, 9\}$ are the digits of M and N, resp. Suppose further that $K = k_{k-1} \cdots k_1 k_0$ is the k-digit sum of M and N,

$$K = M + N.$$

To simplify the notation, suppose that, without loss of generality, $n \leq m$, let $n_n, n_{n+1}, \ldots, n_{m-1} = 0$, and, for each $i \in \{0, 1, \ldots, m-1\}$, let c_i denote the i^{th} *carry*, where

$$\begin{aligned} c_0 &= \left\lfloor \frac{m_0 + n_0}{10} \right\rfloor, \\ c_1 &= \left\lfloor \frac{m_1 + n_1 + c_0}{10} \right\rfloor, \\ c_2 &= \left\lfloor \frac{m_2 + n_2 + c_1}{10} \right\rfloor, \\ &\vdots \\ c_{m-1} &= \left\lfloor \frac{m_{m-1} + n_{m-1} + c_{m-2}}{10} \right\rfloor. \end{aligned}$$

Owing to the fact that the m_i and n_i are always less than 10, c_i is guaranteed to be either 0 or 1. The k digits of K can then be written

$$\begin{aligned} k_0 &= \operatorname{mod}(m_0 + n_0, 10), \\ k_1 &= \operatorname{mod}(m_1 + n_1 + c_0, 10), \\ k_2 &= \operatorname{mod}(m_2 + n_2 + c_1, 10), \\ &\vdots \\ k_{m-1} &= \operatorname{mod}(m_{m-1} + n_{m-1} + c_{m-2}, 10), \\ k_m &= c_{m-1}, \end{aligned}$$

where, as is implicitly expressed, $k = m$ if $c_{m-1} = 1$, or $k = m - 1$ if $c_{m-1} = 0$.

The sum $K = M + N$ can then be computed using the following algorithm:

Data: The sequence of digits $(m_i)_{i\in\{0,1,\ldots,m-1\}}$ of a number M
Data: The sequence of digits $(n_i)_{i\in\{0,1,\ldots,n-1\}}$ of a number N, $n \leq m$
Result: The sequence of digits $(k_i)_{i\in\{0,1,\ldots,k-1\}}$ of the sum $K = M + N$, where k may be either m or $m+1$

```
k_0 ← mod(m_0 + n_0, 10);
c_0 ← ⌊(m_0 + n_0)/10⌋;
for i ∈ {1, . . . , m − 1} do
    if i > n − 1 then
        n_i ← 0;
    end
    k_i ← mod(m_i + n_i + c_{i−1}, 10);
    c_i ← ⌊(m_i + n_i + c_{i−1})/10⌋;
end
if c_{m−1} > 0 then
    k ← m + 1;
    k_{k−1} ← c_{m−1};
else
    k ← m;
end
return (k_i)_{i∈{0,1,...,k−1}}
```

This has $O(m)$ time complexity and $O(m)$ space complexity.

References

[1] https://leetcode.com/problems/add-two-numbers