Sum Of Two Arrays CodeStudio

You are given two arrays, a and b, which represent non-negative integers. You need to find the sum of the two numbers represented by the arrays and return the result as an array.

Example:

Input:

Array a: [1, 2, 4, 0, 1]

Array b: [9]

Output:

Sum: [1, 2, 4, 0, 1] + [9] = [1, 2, 4, 0, 1] + [0, 0, 0, 9] = [1, 2, 4, 1, 0]

Explanation:

The given arrays represent the numbers 12401 and 9. The sum of these numbers is 12401 + 9 = 12410. The result is represented by the array [1, 2, 4, 1, 0].

Approach 1: Calculates the sum of two arrays of digits

This approach iterates through the arrays from the least significant digit to the most significant digit. It performs digit-wise addition and keeps track of any carry during the addition. The sum is calculated by adding the corresponding digits of **a** and **b** along with the carry. The carry is updated as the floor division of the sum by 10, and the sum modulo 10 is added to the result array. Finally, the result array is reversed to obtain the correct order of digits.

Time Complexity: O(max(n, m))

- Here, n and m are the sizes of the input arrays **a** and **b**, respectively. The algorithm iterates through both arrays once to calculate the sum.

Space Complexity: O(max(n, m))

- The space complexity is determined by the size of the result array, which can be at most max(n, m)+1.

Approach 2: Calculates the sum of two arrays of digits (optimized version)

This approach is an optimized version of the first approach. It follows the same logic of digit-wise addition with carry tracking but avoids unnecessary steps. It performs the addition while considering the remaining digits in either **a** or **b** or any carry. It avoids the need for reversing the result array by adding digits to the result array from the most significant digit to the least significant digit.

Time Complexity: O(max(n, m))

- The algorithm iterates through both arrays once to calculate the sum.

Space Complexity: O(max(n, m))

- The space complexity is determined by the size of the result array, which can be at most max(n, m)+1.