0043. 字符串相乘

ITCharge 大约 2 分钟

- 标签：数学、字符串、模拟
- 难度：中等

题目链接

- 0043. 字符串相乘 - 力扣

题目大意

描述：给定两个以字符串形式表示的非负整数 `num1` 和 `num2` 。

要求：返回 `num1` 和 `num2` 的乘积，它们的乘积也表示为字符串形式。

说明：

- 不能使用任何标准库的大数类型（ | BigInteger）或直接将输入转换为整数来处理。
- $1 \leq num1.length, num2.length \leq 200$。
- `num1` 和 `num2` 只能由数字组成。
- `num1` 和 `num2` 都不包含任何前导零，除了数字0本身。

示例：

- 示例 1：

```py
输入：num1 = "2", num2 = "3"
输出："6"
```

- 示例 2：

```py
输入：num1 = "123", num2 = "456"
输出："56088"
```

解题思路

思路 1：模拟

我们可以使用数组来模拟大数乘法。长度为 `len(num1)` 的整数 `num1` 与长度为 `len(num2)` 的整数 `num2` 相乘的结果长度为 `len(num1) + len(num2) - 1` 或 `len(num1) + len(num2)`。所以我们可以使用长度为 `len(num1) + len(num2)` 的整数数组 `nums` 来存储两个整数相乘之后的结果。

整个计算流程的步骤如下：

1. 从个位数字由低位到高位开始遍历 `num1`，取得每一位数字 `digit1`。从个位数字由低位到高位开始遍历 `num2`，取得每一位数字 `digit2`。
2. 将 `digit1 * digit2` 的结果累积存储到 `nums` 对应位置 `i + j + 1` 上。
3. 计算完毕之后从 `len(num1) + len(num2) - 1` 的位置由低位到高位遍历数组 `nums`。将每个数位上大于等于 `10` 的数字进行进位操作，然后对该位置上的数字进行取余操作。
4. 最后判断首位是否有进位。如果首位为 `0`，则从第 `1` 个位置开始将答案数组拼接成字符串。如果首位不为 `0`，则从第 `^` 个位置开始将答案数组拼接成字符串。并返回答案字符串。

思路 1：代码

py

```py
class Solution:
    def multiply(self, num1: str, num2: str) -> str:
        if num1 == "0" or num2 == "0":
            return "0"

        len1, len2 = len(num1), len(num2)
        nums = [0 for _ in range(len1 + len2)]

        for i in range(len1 - 1, -1, -1):
            digit1 = int(num1[i])
            for j in range(len2 - 1, -1, -1):
                digit2 = int(num2[j])
                nums[i + j + 1] += digit1 * digit2

        for i in range(len1 + len2 - 1, 0, -1):
```

```
            nums[i - 1] += nums[i] // 10
            nums[i] %= 10

        if nums[0] == 0:
            ans = "".join(str(digit) for digit in nums[1:])
        else:
            ans = "".join(str(digit) for digit in nums[:])
        return ans
```

思路 1：复杂度分析

- **时间复杂度**：$O(m \times n)$，其中 m 和 n 分别为 `nums1` 和 `nums2` 的长度。
- **空间复杂度**：$O(m + n)$。