# 题目

给定两个以字符串形式表示的非负整数 num1 和 num2，返回 num1 和 num2 的乘积，它们的乘积也表示为字符串形式。

注意：不能使用任何内置的 BigInteger 库或直接将输入转换为整数。

示例 1:

输入: num1 = "2", num2 = "3"

输出: "6"

示例 2:

输入: num1 = "123", num2 = "456"

输出: "56088"

提示：

1 <= num1.length, num2.length <= 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。

# 分析

## 方法一：模拟

思路：

可以使用竖式乘法的思路来实现。具体步骤如下：

1、初始化一个长度为 num1.size() + num2.size() 的数组 res，用于存放乘积的各位数字。

2、从 num1 的个位开始，与 num2 的个位相乘，结果存入 res 的对应位置。进位部分放入高位。

3、依次处理 num1 的十位、百位...与 num2 的个位、十位...相乘，将结果累加到 res 的对应位置。

4、将 res 转换为字符串，去除前导零后返回结果。

代码：

class Solution {

public:

string multiply(string num1, string num2) {

int m = num1.size(), n = num2.size();

vector<int> res(m + n, 0);

for (int i = m - 1; i >= 0; --i) {

for (int j = n - 1; j >= 0; --j) {

int mul = (num1[i] - '0') \* (num2[j] - '0');

int sum = mul + res[i + j + 1];

res[i + j] += sum / 10;

res[i + j + 1] = sum % 10;

}

}

string result = "";

for (int num : res) {

if (!(result.empty() && num == 0)) {

result += to\_string(num);

}

}

return result.empty() ? "0" : result;

}

};