1. 题目
2. 题干

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。

1. 示例

示例 1:

输入: num1 = "2", num2 = "3"

输出: "6"

示例 2:

输入: num1 = "123", num2 = "456"

输出: "56088"

说明：

num1 和 num2 的长度小于110。

num1 和 num2 只包含数字 0-9。

num1 和 num2 均不以零开头，除非是数字 0 本身。

不能使用任何标准库的大数类型（比如 BigInteger）或直接将输入转换为整数来处理。

1. 题解
2. 思路

模拟数字相乘的过程即可，数字相乘时，是用下面的数字的每一位数字乘以上面的数字，且下面的每一位数字做完乘积后，都会向前进一位，这里可以用在末尾添加0实现。最后，将每一次的乘积和都加起来即可。这里需要用到字符串中数字的相加，以及StringBuilder，注意，在StringBuilder中通过append添加数字，但是这个数字的顺序和相乘的顺序是相反的，因为每次相乘都是从数字的末尾开始的，而相乘完的结果会首先append到StringBuilder中，因此，StringBuilder的结果都需要进行reverse才是正确的结果。

字符串中数字的相加大体的思路也差不多，都是从每个数字的末尾开始操作，相加进位。注意，最后的进位不要忽略了，需要添加到字符串中。

1. 代码实现

Java：

class Solution {

    public String multiply(String num1, String num2) {

        if(num1.equals("0") || num2.equals("0")){

            return "0";

        }

        int n1 = num1.length();

        int n2 = num2.length();

        String sum = "0";

        for(int i = n2 -1; i >= 0; i--){

            StringBuilder curr = new StringBuilder();

            int zeroNum = n2-1-i;

            for(int j = 1; j <= zeroNum; ++j){

                curr.append(0);

            }

            int step = 0;

            for(int j = n1-1; j>=0; j--){

                int x = step;

                int x2 = num2.charAt(i) - '0';

                int x1 = num1.charAt(j) - '0';

                x += x1 \* x2;

                curr.append(x % 10);

                step = x / 10;

            }

            if(step != 0){

                curr.append(step%10);

            }

            sum = addString(sum, curr.reverse().toString());

        }

        return sum;

    }

    public String addString(String s1, String s2){

        int n1 = s1.length();

        int n2 = s2.length();

        int step = 0;

        StringBuilder addSum = new StringBuilder();

        for(int i= n1-1, j=n2-1; i >=0 || j >= 0; i--, j--){

             int sum = step;

             sum += i >= 0 ? s1.charAt(i)-'0' : 0;

             sum += j >= 0 ? s2.charAt(j)-'0' : 0;

            addSum.append(sum %10);

            step = sum / 10;

        }

        if(step != 0){

            addSum.append(step);

        }

        return addSum.reverse().toString();

    }

}