43. Multiply Strings

Description | Hints | Submissions | Discuss | Solution

Pick One

Given two non-negative integers `num1` and `num2` represented as strings, return the product of `num1` and `num2`, also represented as a string.

Example 1:

```
Input: num1 = "2", num2 = "3"
Output: "6"
```

Example 2:

```
Input: num1 = "123", num2 = "456"
Output: "56088"
```

Note:

1. The length of both `num1` and `num2` is < 110.
2. Both `num1` and `num2` contain only digits `0-9`.
3. Both `num1` and `num2` do not contain any leading zero, except the number 0 itself.
4. You **must not use any built-in BigInteger library** or **convert the inputs to integer** directly.

```java
public class L43 {


    /*
     * 这个题目是两个大数相乘，为了避免溢出，显然应该考虑对应位相乘，求结果的每一位的值最后串在一起。
     * 分别存进数组里面，在考虑低位向高位的进位，转换为字符串之后考虑首位为0的情况。
     */
    public String multiply(String num1, String num2) {
        //对字符进行反转
        String n1 = new StringBuilder(num1).reverse().toString();
        String n2 = new StringBuilder(num2).reverse().toString();
        //两数相乘，最大位数为两者位数之和
        int [] d = new int [num1.length() + num2.length()];

        for (int i = 0; i < n1.length(); i++) {
            for(int j = 0; j < n2.length(); j++) {
                //先考虑对应位相乘，不考虑进位
                d[i+j] += (n1.charAt(i) - '0') * (n2.charAt(j) - '0');
            }
        }
        StringBuilder sb = new StringBuilder();

        for(int i = 0; i < d.length; i++) {
            //对进位进行处理
            int mod = d[i] % 10;
            int carry = d[i] / 10;
            if(i+1 < d.length) {
                d[i+1] += carry;
            }
            sb.insert(0, mod);
        }

        //除去首部的0
        while (sb.charAt(0) == '0' && sb.length() > 1) {
            sb.deleteCharAt(0);
        }

        return sb.toString();
    }
}
```