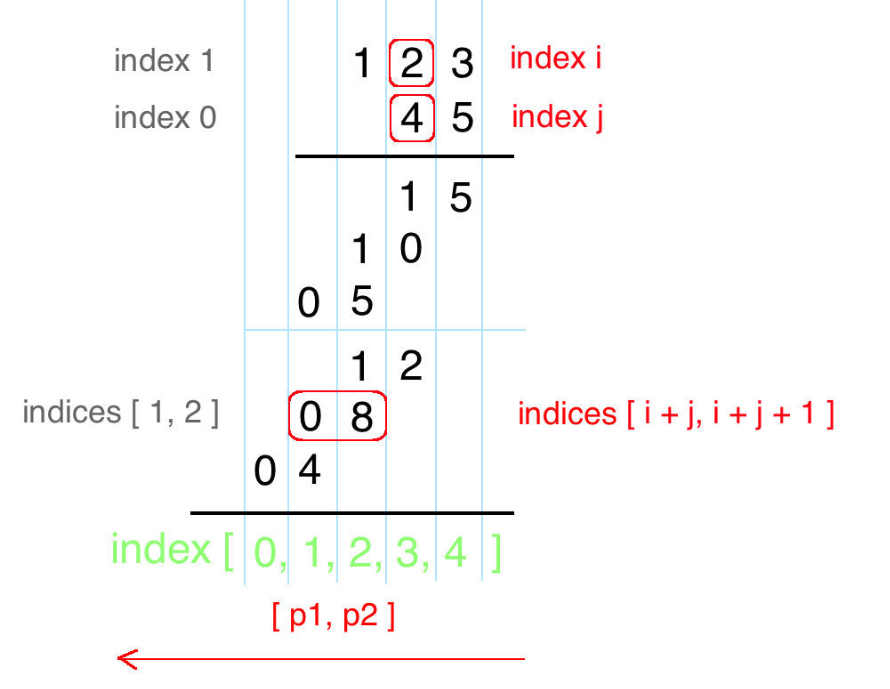
Given two non-negative integers num1 and num2 represented as strings, return the product of num1 and num2.

**Note:**

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

题目：就是两个长整数，做乘法。给两个整数以字符串的形式，乘出来的整数输出也是以字符串的形式输出的。

我们要了解乘法的过程是什么



我们从图中直到，结果的长度正好是第一个数字的长度+第二个数字的长度。所以我们保存结果用一个数组来保存，数组的大小为两个数字长度和，如图就是5.

做乘法的时候一定是从末尾开始的，先从一个数的末尾开始分别与另一个数字的每一位相乘，然后进位。等结束，然后爱是下一位数字与另一个数字的每一位相乘。这正好用嵌套循环就可以解决。

那么没位置的数字怎么算呢？

首先我们看，当两个数字都为末尾的时候，i=2，j=1，i位置的数字是3，j位置的数字是5，相乘得15，10进位，剩下数字五，所以最后一位数字就是5，那么最后一位和之前两个数字位置又有什么关系呢？其实，i+j+1=2+1+1=4，正好是数组的末尾。每个位置的i和j，在对应结果的当前位都是i+j+1，进位在i+j。也就是说，5的位置正好保存在i+j+1中，1进位加到了i+j的位置。

所以我们在每位计算的时候把进位都加到i+j中就行了，剩下的加在i+j+1中。

15，怎么算进位呢？15%10=5，15/10=1，这样把进位和进位后余下的数字就算出来了。

然后我们逐个加到对应位置就行了。

最后我们在输出的时候要注意，队首不能有0。

代码

public static String multiplyStrings(String num1,String num2){

int m = num1.length();

int n = num2.length();

int[] result = new int[m+n];

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

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

int temp = (num1.charAt(i)-'0')\*(num2.charAt(j)-'0');//对应位置相乘

result[i+j+1]=result[i+j+1]+temp%10;//余下数

result[i+j]=result[i+j]+temp/10;//进位

}

}

StringBuffer sb = new StringBuffer();

for(int i=0;i<m+n;i++){//数组转化成字符串

if(i==0&&result[i]==0)//处理第一个数字为0的情况，去掉队首的0

continue;

sb.append(result[i]);

}

return sb.toString();

}