字符串相乘

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。

示例 1:

输入: num1 = "2", num2 = "3"

输出: "6"

示例 2:

输入: num1 = "123", num2 = "456"

输出: "56088"

```
          1   2   3
      乘  4   5   6
————————————————————————
          6   12  18
      5   10  15
  4   8   12
————————————————————————
  4   13  28  27  18
整理：  c[i + 1] += c[i] / 10, c[i] %= 10, 从低位开始。
step 0: 4   13  28  27  18
step 1: 4   13  28  28  8
step 2: 4   13  30  8   8
step 3: 4   16  0   8   8
step 4: 5   6   0   8   8
```

```
class Solution {
public:
    string multiply(string num1, string num2) {
        string ans;
        vector<int> a,b,c;
        //对存储空间 扩容
        c.resize(num1.size()+num2.size()-1);
        //逆序转换
        for(int i=num1.size()-1;i>=0;i--)
        {
            a.push_back(num1[i]-'0');
        }
        for(int j=num2.size()-1;j>=0;j--)
        {
            b.push_back(num2[j]-'0');
```

```
        }
        for(int i=0;i<num1.size();i++)
        {
            for(int j=0;j<num2.size();j++)
            {
                c[i+j]+=a[i]*b[j];
            }
        }
        int step=0;//进位
        for(int i=0;i<c.size();i++)
        {
            step+=c[i];
            char c=step%10+'0';//取余保留当前位
            ans=c+ans;
            step=step/10;
        }
        while(step)//如果 step 仍存在  需要最前面再进位
        {
            char c=step%10+'0';
            ans=c+ans;
            step=step/10;
        }
        while(ans.size()>1&&ans[0]=='0')
        {
            ans.erase(ans.begin());
        }
        return ans;
    }
};
```