计算最大乘积

题目描述：

给定一个元素类型为小写字符串的数组，请计算两个没有相同字符的元素 长度乘积的最大值，如果没有符合条件的两个元素，返回 0。

输入描述：

输入为一个半角逗号分隔的小写字符串的数组，2 <= 数组长度<=100，0 < 字符串长度 <= 50。

输出描述：

两个没有相同字符的元素 长度乘积的最大值。

示例 1

输入：

```
iwdvpbn,hk,iuop,iikd,kadgpf
```

输出：

```
14
```

说明：

数组中有 5 个元素。

iwdvpbn 与 hk 无相同的字符，满足条件，iwdvpbn 的长度为 7，hk 的长度为 2，乘积为 14（7*2）。

iwdvpbn 与 iuop、iikd、kadgpf 均有相同的字符，不满足条件。

iuop 与 iikd、kadgpf 均有相同的字符，不满足条件。

iikd 与 kadgpf 有相同的字符，不满足条件。

因此，输出为 14。

```
#include <iostream>

#include<vector>

#include<algorithm>
```

```
using namespace std;


int getMaxProduct(const vector<string>&arr) {
    int maxProduct=0;
    int n =arr.size();

    for(int i=0;i<n;i++){
        for(int j=i+1;j<n;j++){
            bool hasSameChar=false;

            for(char ch :arr[i]){
                if(arr[j].find(ch)!=string::npos){
                    hasSameChar=true;
                    break;
                }
            }
            if(!hasSameChar){
                int product=arr[i].length()*arr[j].length();
                maxProduct=max(maxProduct,product);
            }
        }
    }
```

```
    return maxProduct;
}
int main(){
    string input;
    getline(cin,input);
    vector<string>arr;
    size_t pos =0;
    string delimiter=",";
    while ((pos = input.find(delimiter))!=string::npos){
        string token=input.substr(0,pos);
        arr.push_back(token);
        input.erase(0,pos+delimiter.length());
    }
    arr.push_back(input);

    int result =getMaxProduct(arr);
    cout<<result<<endl;
    return 0;
}
// 64 位输出请用 printf("%lld")
```