



## §. 基础知识题 – 字符数组的输入与输出

要求:

- 1、完成本文档中所有的题目并写出分析、运行结果，体会字符数组输入输出时不同用法的差异
- 2、题目明确指定编译器外，缺省使用VS2022即可
  - ★ 如果要换成其他编译器，可能需要自行修改头文件适配
  - ★ 部分代码编译时有warning，不影响概念理解，可以忽略
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
  - ★ 贴图要有效部分即可，不需要全部内容
  - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
  - ★ **不允许**手写在纸上，再拍照贴图
  - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
  - ★ 如果某题要求VS+Dev的，则如果两个编译器运行结果一致，贴VS的一张图即可，如果不一致，则两个图都要贴
- 4、转换为pdf后提交
- 5、**11月17日前**网上提交本次作业（在“文档作业”中提交）



## §. 基础知识题 - 字符数组的输入与输出

贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

例：无效贴图

```
Microsoft Visual Studio 调试控制台  
Hello, world!  
D:\Workspace\VS2019-Demo\Debug\cpp-demo.exe (进程 7484)已退出, 代码为 0。  
按任意键关闭此窗口. . .
```

例：有效贴图

```
Microsoft Visual Studio 调试控制台  
Hello, world!
```



# §. 基础知识题 - 字符数组的输入与输出

- 注意：
- 1、部分内容的填写，如果能确定是“不确定值/随机值”的，可直接填写“\*\*/随机”

demo-CPP

(全局范围)

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int i, a[5];
6      for (i = 0; i < 5; i++)
7          cout << a[i] << endl;
8      return 0;
9  }
10
```

Microsoft Visual Studio 调试控制台

```
-858993460
-858993460
-858993460
-858993460
-858993460
D:\Workspace\VS2019-demo\Debug\
按任意键关闭此窗口...
```

输出的5行内容是:

\*\*
\*\*
\*\*
\*\*
\*\*

输出的5行内容是:

随机
随机
随机
随机
随机

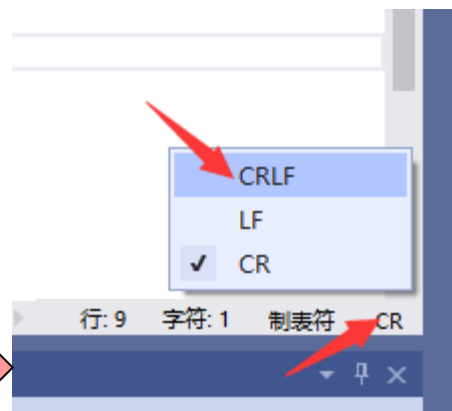
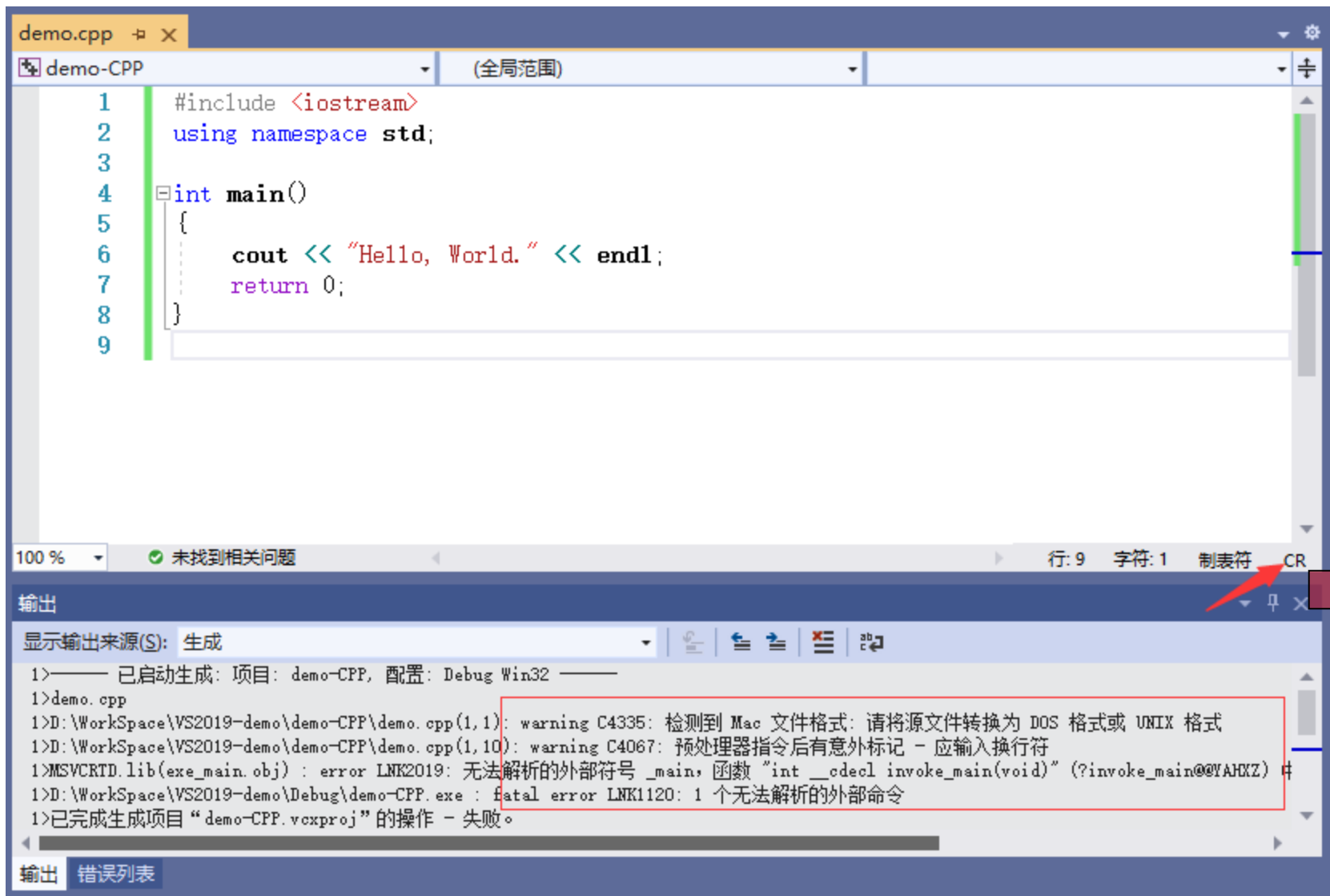


## §. 基础知识题 - 字符数组的输入与输出

注意:

2、附: 用WPS等其他第三方软件打开PPT, 将代码复制到VS2022中后, 如果出现类似下面的**编译报错**, 则观察源程序编辑窗

的右下角是否为CR, 如果是, 单击CR, 在弹出中选择CRLF, 再次CTRL+F5运行即可





## §. 基础知识题 - 字符数组的输入与输出

### 1. 输入

逐个输入: scanf("%c",&数组元素)    C方式

cin >> 数组元素    C++方式

#### 例1: C方式输入单个字符

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;

int main()
{
    char a[10];
    int i;

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    scanf("%c%c", &a[3], &a[7]);

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    return 0;
}
```

数组下标表示前有  
取地址符号&  
因为scanf规定后面  
必须是变量的地址

scanf前首先输出10行, 内容是:

```
**
**
**
**
**
**
**
**
**
**
scanf时, 输入AB并回车, 输出是:
**
**
**
65
**
**
**
66
**
**
**
//用不同颜色标注出有变化的内容
```

本页需填写答案





## §. 基础知识题 – 字符数组的输入与输出

### 1. 输入

逐个输入: scanf("%c",&数组元素)    C方式  
          cin >> 数组元素            C++方式

### 例3: C方式多次逐个输入时回车的处理

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;

int main()
{
    char a[10];
    int i;

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    scanf("%c%c", &a[3], &a[7]);
    scanf("%c", &a[0]);

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    return 0;
}
```

scanf前首先输出10行, 内容是

```
**
**
**
**
**
**
**
**
**
**
```

scanf时, 输入AB并回车, 输出是:

```
10
**
**
65
**
**
**
66
**
**
```

//用不同颜色标注出有变化的内容

本页需填写答案



## §. 基础知识题 - 字符数组的输入与输出

### 1. 输入

逐个输入: scanf("%c",&数组元素)    **C方式**

cin >> 数组元素    **C++方式**

例4: C++方式多次逐个输入时回车的处理

```
#include <iostream>
using namespace std;

int main()
{
    char a[10];
    int i;

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    cin >> a[3] >> a[7];
    cin >> a[0];

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    return 0;
}
```

cin前首先输出10行, 内容是

\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*

cin时, 输入AB并回车, 表现如何?  
多按几次回车, 表现如何?  
最后再输入C并回车, 则输出是:

67  
\*\*  
\*\*  
65  
\*\*  
\*\*  
\*\*  
66  
\*\*  
\*\*

//用不同颜色标注出有变化的内容

综合例3/4得到结论: 当多次逐个输入时,

**C方式**处理回车的方式是\_将回车读取变量中\_,

**C++方式**处理回车的方式是\_忽略回车\_

本页需填写答案





字符串形式: scanf("%s", 数组名)      C方式  
cin >> 数组名      C++方式

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;
```

直接数组名，无下标，  
也不加&  
因为C/C++规定，数组名  
代表数组的起始地址

2、Hello后面的一个字符是什么？尾零 \0

本页需填写答案



## §. 基础知识题 - 字符数组的输入与输出

### 1. 输入

字符串形式: scanf("%s", 数组名)      C方式

cin >> 数组名      C++方式

#### 例6: C方式输入字符串(错误)

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;
```

```
int main()
{
```

```
    char a[10];
    int i;
```

```
    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;
```

```
    scanf("%s", a);
```

```
    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;
```

```
    return 0;
```

```
}
```

直接数组名, 无下标,  
也不加&  
因为C/C++规定, 数组名  
代表数组的起始地址

scanf前首先输出10行, 内容是

```
**
**
**
**
**
**
**
**
**
**
**
```

等待键盘输入:

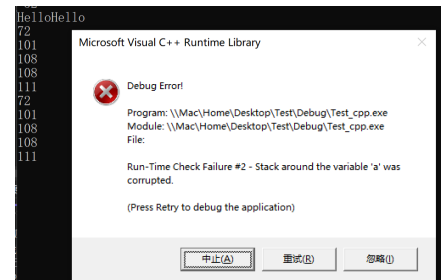
测试1: 输入9个及以下字符并回车, 输出?

测试2: 输入10个及以上字符并回车, 输出?

测试1: 比如还是输入Hello, 结果和前一题相同

测试2: 比如输入HelloHello, 会出现弹窗  
编译报错

问: 如果要想保证输入正确, 输入的字符个数  
要\_+ 1后小于\_\_定义的字符数组的长度



本页需填写答案



## §. 基础知识题 - 字符数组的输入与输出

### 1. 输入

字符串形式: scanf("%s", 数组名)

C方式

cin >> 数组名

C++方式

#### 例7: C++方式输入字符串(正确)

```
#include <iostream>
using namespace std;

int main()
{
    char a[10];
    int i;

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    cin >> a;

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    return 0;
}
```

直接数组名, 无下标,  
也不加&

cin前首先输出10行, 内容是

\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*

等待键盘输入, 输入Hello并回车, 输出为

72  
101  
108  
108  
111  
0

\*\*  
\*\*  
\*\*  
\*\*

//用不同颜色标注出有变化的内容

问: 1、回车是否在数组中? 不在

2、Hello后面的一个字符是什么? 尾零 \0

本页需填写答案



# §. 基础知识题 - 字符数组的输入与输出

## 1. 输入

字符串形式: scanf("%s", 数组名)

C方式

cin >> 数组名

C++方式

### 例8: C++方式输入字符串(错误)

```
#include <iostream>
using namespace std;

int main()
{
    char a[10];
    int i;

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    cin >> a;

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    return 0;
}
```

直接数组名, 无下标,  
也不加&

cin前首先输出10行, 内容是

\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*

等待键盘输入:

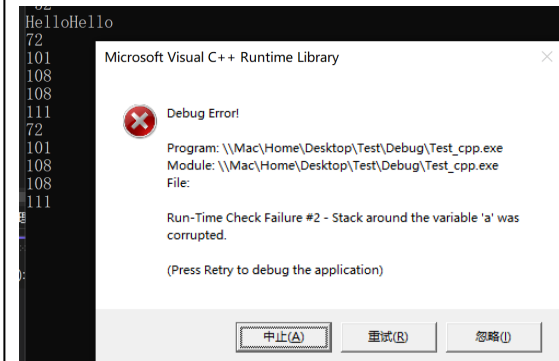
测试1: 输入9个及以下字符并回车, 输出?

测试2: 输入10个及以上字符并回车, 输出?

测试1: 比如还是输入Hello, 结果和前一题相同

测试2: 比如输入HelloHello, 会出现弹窗  
编译报错

问: 如果要保证输入正确, 输入的字符个数  
要\_+ 1后小于\_\_定义的字符数组的长度



本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 2. 输出

逐个: printf("%c", 数组元素)

C方式

cout << 数组元素

C++方式

例9: C/C++方式输出单个字符

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
    char a[]="Student"; //长度缺省为8

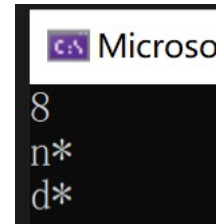
    cout << sizeof(a) << endl;

    printf("%c*\n", a[5]);

    cout << a[3] << '*' << endl;

    return 0;
}
//输出加*是为了确认只输出了一个字符
```

输出为:



本页需填写答案



## §. 基础知识题 - 字符数组的输入与输出

### 2. 输出

逐个: printf("%c", 数组元素)

C方式

cout << 数组元素

C++方式

例10: C/C++方式以单个字符+循环形式输出整个数组

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
    int i;
    char a[]="Student";

    for(i=0; i<7; i++)
        printf("%c", a[i]);
    cout << endl; //换行

    for(i=0; i<7; i++)
        cout << a[i];
    cout << endl; //换行

    return 0;
}
```

数组 a 缺省长度为8  
输出[0]-[6]，尾零不输出

输出为:

```
选择 Micro:
Student
Student
```

本页需填写答案



## §. 基础知识题 - 字符数组的输入与输出

### 2. 输出

逐个: printf("%c", 数组元素)

C方式

cout << 数组元素

C++方式

例11: C/C++方式以单个字符+循环形式输出整个数组

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
    int i;
    char a[]="Student";

    for(i=0; i<7; i++)
        printf("%c", a[i]);
    cout << endl; //换行

    for(i=0; i<7; i++)
        cout << a[i] << '*';
    cout << endl; //换行

    return 0;
}
```

%c后面多一个,  
cout方式每个字符  
后面多一个\*

输出为:

Microsoft Visual Studio

```
S, t, u, d, e, n, t,
S*t*u*d*e*n*t*
```

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 2. 输出

字符串形式: printf("%s", 数组名) **C方式**

cout << 数组名 **C++方式**

例12: C/C++以字符串方式输出字符数组

```
#include <iostream>
using namespace std;
```

```
int main()
{
```

```
    char a[]="Student";
```

```
    printf("%s\n", a);
```

```
    cout << a << endl;
```

```
    return 0;
```

```
}
```

跟数组名  
不是数组元素名

输出为:

```
Microsoft Visual Studio
Student
Student
```

问: 尾零输出了吗? 如何证明?

尾零没有输出, 可在字符串后  
紧接着输出一个\*

```
#include <iostream>
using namespace std;

int main()
{
    char a[] = "Student";

    printf("%s\n", a);
    cout << a << "*" << endl;

    return 0;
}
```

```
Microsoft Visual Studio
Student
Student*
```

本页需填写答案





## §. 基础知识题 – 字符数组的输入与输出

### 2. 输出

字符串形式: `printf("%s", 数组名)` C方式

`cout << 数组名` C++方式

例13: C/C++以字符串方式输出字符数组

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
    char a[]="Student\0china";

    cout << sizeof(a) << endl;

    printf("%s*\n", a);
    cout << a << '*' << endl;

    cout << a[12] << endl;

    return 0;
}
```

输出为:

```
14
Student*
Student*
a
```

问1: 从本例的结果可知,  
数组a的长度是\_14\_\_\_\_,  
最后是否还有隐含的\0? 是  
a中的字符串的长度是\_7\_\_\_\_

问2: 字符串形式输出字符数组,  
如果数组中包含显式'\0',  
则输出到\_第一个显式'\0' 前\_\_为止

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 2. 输出

字符串形式: printf("%s", 数组名) **C方式**

cout << 数组名 **C++方式**

例14: C/C++以字符串方式输出字符数组(不含尾零)

```
#include <iostream>
using namespace std;

int main()
{
    //注意: 不能以字符串方式初始化
    char a[5]={'C','h','i','n','a'};

    printf("%s\n", a);
    cout << a << endl;

    return 0;
}
```

输出为:



问1: 为什么会有乱字符?

char数组不包含尾零, 直接以数组名方式输出时, 直到遇到尾零才停止, 所以会访问超出数组范围的内存, 这些内存的值为被初始化, 内容未知

问2: 如果%s方式换成下面形式

```
int i;
for (i=0; i<5; i++)
    printf("%c", a[i]);
```

还会看到乱字符吗? 为什么?

不会。不管字符数组中有没有尾零, 打印5个字符就停止了这5个字符都已经被赋值过

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 2. 输出

字符串形式: printf("%s", 数组名) **C方式**

cout << 数组名 **C++方式**

例15: C/C++以字符串方式输出字符数组(不含尾零)

```
#include <iostream>
using namespace std;

int main()
{
    char a[5]; //不初始化

    printf("%s\n", a);
    cout << a << endl;

    return 0;
}
```

输出为:



问1: 为什么会有乱字符?

char数组未初始化, 直接以数组名方式输出时, 直到遇到尾零才停止, 所以会访问超出数组范围的内存, 由于这些内存的值未被初始化, 内容未知

问2: 乱字符出现几行是正常的?  
一行? 多行? 或者都正常?  
都正常

结论: 不能字符串形式输出不含  
\_尾零\_的字符数组, 否则  
可能会得到不正确的结果

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 3. 从任一元素开始以字符串形式输入/输出

例16: 从任一元素开始以字符串形式输出

```
#include <iostream>
using namespace std;

int main()
{
    char a[]="Student";
    printf("%s\n", &a[3]);
    cout << &a[3] << endl;

    return 0;
}
```

%s形式

&数组元素名形式

输出为:

dent  
dent



## §. 基础知识题 – 字符数组的输入与输出

### 3. 从任一元素开始以字符串形式输入/输出

例17: C方式从任一元素开始以字符串形式输入

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;

int main()
{
    int i;
    char a[10];

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    scanf("%s", &a[3]);

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    return 0;
}
```

&数组元素名形式

scanf先输出10行, 内容是

```
**
**
**
**
**
**
**
**
**
**
**
72
101
108
108
111
0
**
```

等待键盘输入, 输入Hello并回车, 输出为

```
**
**
**
**
**
**
**
**
**
**
**
72
101
108
108
111
0
**
```

//用不同颜色标注出有变化的内容

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 3. 从任一元素开始以字符串形式输入/输出

例18: C++方式从任一元素开始以字符串形式输入

```
#include <iostream>
using namespace std;

int main()
{
    int i;
    char a[10];

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    cin >> &a[3];

    for(i=0; i<10; i++)
        cout << int(a[i]) << endl;

    return 0;
}
```

&数组元素名形式

cin先输出10行，内容是

\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*  
\*\*

等待键盘输入，输入Hello并回车，输出为

\*\*  
\*\*  
\*\*  
72  
101  
108  
108  
111  
0  
\*\*

//用不同颜色标注出有变化的内容

综合例16-18的结果，得出的结论是：

C/C++方式从任一元素开始以字符串形式

输入输出时，表示形式都是\_数组名[下标]\_的形式

本页需填写答案



# § . 基础知识题 – 字符数组的输入与输出

1-3. 总结

完成下表 (给出了第一行的答案供参考):

	C方式	C++方式
输入单个字符	<code>scanf("%c", &amp;元素名);</code>	<code>cin &gt;&gt; 元素名;</code>
输入字符串	<code>scanf("%s", 数组名);</code>	<code>cin &gt;&gt; 数组名;</code>
输出单个字符	<code>printf("%c", 数组名[下标]);</code>	<code>cout &lt;&lt; 数组名[下标];</code>
输出字符串	<code>printf("%s", 数组名);</code>	<code>cout &lt;&lt; 数组名;</code>
任一元素开始输入串	<code>scanf("%s", &amp;数组名[下标]);</code>	<code>cin &gt;&gt; &amp;数组名[下标];</code>
任一元素开始输出串	<code>printf("%s", &amp;数组名[下标]);</code>	<code>cout &lt;&lt; &amp;数组名[下标];</code>

本页需填写答案



## §. 基础知识题 - 字符数组的输入与输出

### 4. 多个字符串的输入

#### 例19: C方式多个字符串的输入

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
#include <cstdio>
using namespace std;

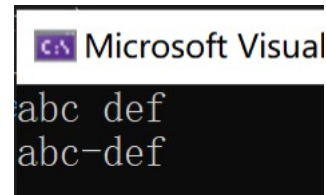
int main()
{
    char a[10], b[20];

    scanf("%s%s", a, b);

    printf("%s-%s\n", a, b);

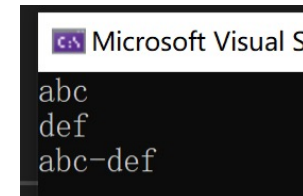
    return 0;
}
```

- 1、假设输入为`abc`空格`def`并回车  
则输出为:



```
Microsoft Visual
abc def
abc-def
```

- 2、假设输入为`abc`回车  
`def`回车  
则输出为:



```
Microsoft Visual S
abc
def
abc-def
```

- 结论: 空格是\_\_B\_\_  
A. 输入串中的合法字符  
B. 输入分隔符





## §. 基础知识题 – 字符数组的输入与输出

### 4. 多个字符串的输入

#### 例20: C++方式多个字符串的输入

```
#include <iostream>
using namespace std;

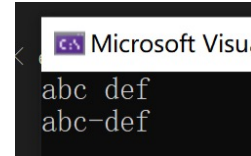
int main()
{
    char a[10], b[20];

    cin >> a >> b;

    cout << a << '-' << b << endl;

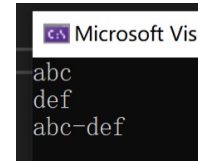
    return 0;
}
```

- 1、假设输入为`abc空格def`并回车  
则输出为:



```
Microsoft Visual Studio
abc def
abc-def
```

- 2、假设输入为`abc`回车  
`def`回车  
则输出为:



```
Microsoft Visual Studio
abc
def
abc-def
```

- 结论: 空格是\_\_B\_\_  
A. 输入串中的合法字符  
B. 输入分隔符

综合例19-20可知:  
scanf/cin从键盘上输入的字符串  
不能包含\_\_空格\_\_



## §. 基础知识题 – 字符数组的输入与输出

### 4. 多个字符串的输入

★ 从键盘输入含空格字符串的方法(不同编译器不同)

- VS2022 : 有gets\_s, 无gets, 有fgets
- Dev C++ : 有gets, 无gets\_s, 有fgets
- fgets函数的原型定义为:

fgets(字符数组名, 最大长度, stdin);

但与gets/gets\_s的表现有不同, 请自行观察

★ scanf/cin通过某些高级设置方式还是可以输入含空格的字符串的, 本课程不再讨论



## §. 基础知识题 – 字符数组的输入与输出

### 4. 多个字符串的输入

★ 从键盘输入含空格字符串的方法(不同编译器不同)

例21: VS下用gets\_s输入含空格的字符串

```
#include <iostream>
using namespace std;

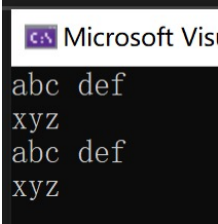
int main()
{
    char a[10], b[20];

    gets_s(a);
    gets_s(b);

    cout << a << endl;
    cout << b << endl;

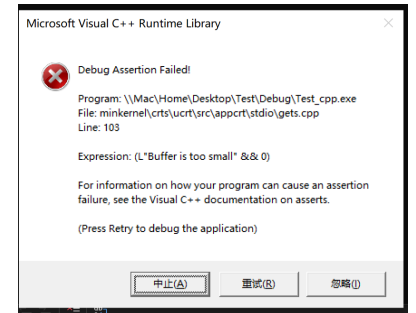
    return 0;
}
```

- 1、键盘输入abc空格def并回车，  
会继续等待输入，  
再输入xyz并回车  
则输出为：



```
Microsoft Visual Studio
abc def
xyz
abc def
xyz
```

- 2、键盘输入超过9个字符，观察
- 3、键盘先输入Hello并回车，  
再输入超过19个字符，观察



问：为什么a最长输入只能是9？  
a数组大小为10，以字符串输入时隐含尾0，  
所以最多输入9个字符  
为什么b最长输入只能是19？  
b数组大小为20，以字符串输入时隐含尾0，  
所以最多输入19个字符

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 4. 多个字符串的输入

★ 从键盘输入含空格字符串的方法(不同编译器不同)

例22: DevC++下用gets输入含空格的字符串

```
#include <iostream>
#include <cstdio>
using namespace std;

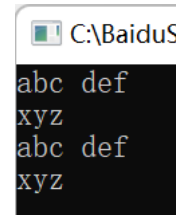
int main()
{
    char a[10], b[20];

    gets(a);
    gets(b);

    cout << a << endl;
    cout << b << endl;

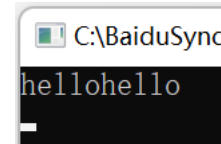
    return 0;
}
```

- 1、键盘输入`abc空格def`并回车，  
会继续等待输入，  
再输入`xyz并回车`  
则输出为：



```
C:\BaiduS
abc def
xyz
abc def
xyz
```

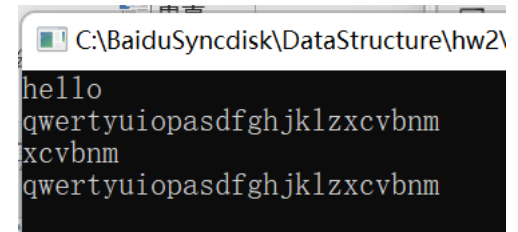
- 2、键盘输入超过9个字符，观察



```
C:\BaiduSync
hellohello
```

光标闪烁，  
等待输入

- 3、键盘先输入`Hello`并回车，  
再输入超过19个字符，观察



```
C:\BaiduSyncdisk\DataStructure\hw2\
hello
qwertyuiopasdfghjklzxcvbnm
xcvbnm
qwertyuiopasdfghjklzxcvbnm
```

问：为什么a最长输入只能是9？ 数组a的长度为10，除去尾零有9位  
为什么b最长输入只能是19？ 数组b的长度为20，除去尾零有19位

本页需填写答案



# § . 基础知识题 – 字符数组的输入与输出

## 4. 多个字符串的输入

★ 不同编译器从键盘输入含空格字符串的方法不同

例23: VS和Dev C++均可用fgets输入含空格的字符串

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
    char a[10], b[20];

    fgets(a, 10, stdin);
    fgets(b, 20, stdin);

    cout << a << endl;
    cout << b << endl;

    int i;
    for(i=0; a[i]!='\0'; i++)
        cout << int(a[i]) << ' ';
    cout << endl;

    for(i=0; b[i]!='\0'; i++)
        cout << int(b[i]) << ' ';
    cout << endl;

    return 0;
}
```

1、键盘输入abc空格def并回车，  
会继续等待输入，  
再输入xyz并回车  
则输出为：

Vs和dev结果相同





问1: 和例21-22的输出区别在哪里？

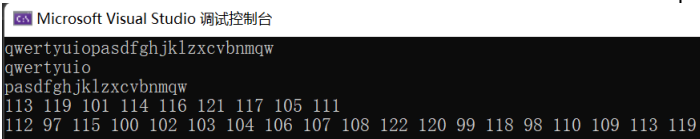
例23多了一个换行

问2: 后面两段红色代码的目的是什么？  
查看数组中是否存在换行符

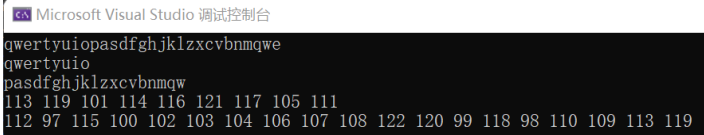
2、键盘输入9个字符并回车，则输出为：



3、如果输入28个字符并回车，则输出为：



4、如果输入超过28个字符并回车，  
则输出为：



本页需填写答案



## §. 基础知识题 - 字符数组的输入与输出

### 5. 二维字符数组的输入/输出

★ 数组名加双下标表示元素，单下标表示一维数组

例24：二维字符数组以双下标形式输出单个字符/单下标形式输出字符串

```
#include <iostream>
using namespace std;

int main()
{
    char a[3][30]={"ABCDEFGHJKLMNOPQRSTUVWXYZ",
                  "abcdefghijklmnopqrstuvwxyz",
                  "0123456789" };
    // 单个字符输出(数组名+双下标)
    printf("a[0][2]=%c\n", a[0][2]);
    cout << "a[1][20]=" << a[1][20] << endl;

    // 字符串输出(数组名+单下标)
    printf("a[0]=%s\n", a[0]);
    cout << "a[2]=" << a[2] << endl;

    return 0;
}
```

输出为：



```
Microsoft Visual Studio 调试控制台
a[0][2]=C
a[1][20]=u
a[0]=ABCDEFGHIJKLMNOPQRSTUVWXYZ
a[2]=0123456789
```

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 5. 二维字符数组的输入/输出

★ 数组名加双下标表示元素，单下标表示一维数组

例25：二维字符数组以双下标形式输入单个字符

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;

int main()
{
    char a[3][30]={"ABCDEFGHJKLMNOPQRSTUVWXYZ",
                  "abcdefghijklmnopqrstvwxyz",
                  "0123456789" };

    // 单字符输入(数组名+双下标)
    scanf("%c\n", &a[0][2]); //格式符为%c
    cin >> a[1][20];         //无&

    // 字符串输出(数组名+单下标)
    printf("a[0]=%s\n", a[0]);
    cout << "a[1]=" << a[1] << endl;

    return 0;
}
```

1、键盘输入#@并回车，输出为：

```
Microsoft Visual Studio 调试控制台
#@
a[0]=AB#DEFGHIJKLMNOPQRSTUVWXYZ
a[1]=abcdefghijklmnopqrst@vwxyz
```

2、键盘输入#并回车，  
输入@并回车  
输出为：

```
Microsoft Visual Studio 调试控制台
#
@a[0]=AB#DEFGHIJKLMNOPQRSTUVWXYZ
a[1]=abcdefghijklmnopqrst@vwxyz
```

本页需填写答案





## §. 基础知识题 - 字符数组的输入与输出

### 5. 二维字符数组的输入/输出

★ 数组名加双下标表示元素，单下标表示一维数组

例26：二维字符数组以单下标形式输入字符串

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;

int main()
{
    char a[3][30]={"ABCDEFGHIJKLMNOPQRSTUVWXYZ",
                  "abcdefghijklmnopqrstuvwxyz",
                  "0123456789" };

    scanf("%s", a[1]); //a[1]是一维数组名, 无&

    cout << "a[0]=" << a[0] << endl;
    cout << "a[1]=" << a[1] << endl;
    cout << "a[2]=" << a[2] << endl;

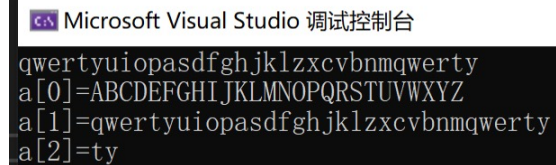
    return 0;
}
```

1、输入≤29个字符，输出为：



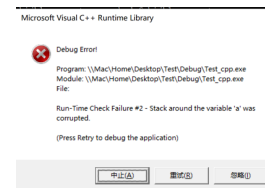
```
Microsoft Visual Studio 调试控制台
qwe
a[0]=ABCDEFGHIJKLMNOPQRSTUVWXYZ
a[1]=qwe
a[2]=0123456789
```

2、输入30–59个字符，输出为：



```
Microsoft Visual Studio 调试控制台
qwertyuiopasdfghjklzxcvbnmqwerty
a[0]=ABCDEFGHIJKLMNOPQRSTUVWXYZ
a[1]=qwertyuiopasdfghjklzxcvbnmqwerty
a[2]=ty
```

3、输入60个以上字符，输出为：



将scanf换为 cin >> a[1];  
再重复1、2、3，观察结果

结果相同

本页需填写答案





问1：输入30~59个字符为什么不出现错误？a[2]中是什么？

二维数组在内存中时连续存放的，输入30~59个字符时，不仅修改了a[1]，还修改了a[2]，但由于没有访问未初始化的内存空间，所以不会报错

a[2]中是输入的字符串后从第31个字符起后面的字符

问2：简述你是怎么理解二维数组越界的？

通过双下标输入时，行列均不能越界；

通过单下标输入时，可以把二维数组拉直，在某个行最开始输入，输入的总长度不能超过最后一行。



## §. 基础知识题 – 字符数组的输入与输出

### 5. 二维字符数组的输入/输出

★ 数组名加双下标表示元素，单下标表示一维数组

例27：二维字符数组从任一位置开始输出字符串

```
#include <iostream>
using namespace std;

int main()
{
    char a[3][30]={"ABCDEFGHJKLMNOPQRSTUVWXYZ",
                  "abcdefghijklmnopqrstuvwxyz",
                  "0123456789" };

    //（第1组）单字符输出(数组名+双下标)
    printf("a[0][2]=%c\n", a[0][2]);
    cout << "a[1][20]=" << a[1][20] << endl;

    //（第2组）字符串输出(&+数组名+双下标)
    printf("a[0][2]=%s\n", &a[0][2]);
    cout << "a[1][20]=" << &a[1][20] << endl;

    //（第3组）字符串输出(数组名+单下标)
    printf("a[0]=%s\n", a[0]);
    cout << "a[2]=" << a[2] << endl;

    return 0;
}
```

输出为：

```
Microsoft Visual Studio 调试控制台
a[0][2]=C
a[1][20]=u
a[0][2]=CDEFGHIJKLMNOPQRSTUVWXYZ
a[1][20]=vwxyz
a[0]=ABCDEFGHIJKLMNOPQRSTUVWXYZ
a[2]=0123456789
```

问1：同样双下标形式(第1/2组)，  
怎样输出单个字符？  
怎样输出字符串？

给数组名[][]本身，输出单个字符  
给数组名[][]的地址，输出字符串

问2：如何修改第2组的输出  
(必须保持双下标形式不变)，  
使输出结果与第3组一致？

```
//（第2组）字符串输出(&+数组名+双下标)
printf("a[0][0]=%s\n", &a[0][0]);
cout << "a[2][0]=" << &a[2][0] << endl;
```

本页需填写答案



## §. 基础知识题 – 字符数组的输入与输出

### 5. 二维字符数组的输入/输出

★ 数组名加双下标表示元素，单下标表示一维数组

例28：二维字符数组从任一位置开始输入字符串

```
#define _CRT_SECURE_NO_WARNINGS //VS需要
#include <iostream>
using namespace std;

int main()
{
    char a[3][30]={"ABCDEFGHJKLMNOPQRSTUVWXYZ",
                  "abcdefghijklmnopqrstuvwxyz",
                  "0123456789" };

    scanf("%s", &a[1][3]); //&+数组名+双下标

    cout << "a[0]=" << a[0] << endl;
    cout << "a[1]=" << a[1] << endl;
    cout << "a[2]=" << a[2] << endl;

    return 0;
}
```

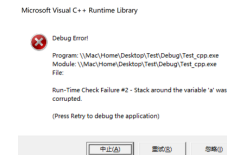
1、输入≤26个字符，输出为：

```
Microsoft Visual Studio 调试控制台
qwe
a[0]=ABCDEFGHIJKLMNOPQRSTUVWXYZ
a[1]=abcqwe
a[2]=0123456789
```

2、输入27-56个字符，输出为：

```
Microsoft Visual Studio 调试控制台
qwertyuiopasdfghjklzxcvbnmqwe
a[0]=ABCDEFGHIJKLMNOPQRSTUVWXYZ
a[1]=abcqwertyuiopasdfghjklzxcvbnmqwe
a[2]=we
```

3、输入56个以上字符，输出为：



将scanf换为 cin >> &a[1][3];  
再重复1、2、3，观察结果  
结果相同

本页需填写答案



问1: 输入27~56个字符为什么不出现错误? a[2]中是什么?

二维数组在内存中时连续存放的, 输入27~56个字符时, 不仅修改了a[1][3]到a[1][29], 还修改了a[2]中字符, 但由于没有访问未初始化的内存空间, 所以不会报错  
a[2]中是输入的字符串后从第28个字符起后面的字符

问2: 如果想不影响a[2],  
例26中是 $\leq 29$ 个字符,  
本例中是 $\leq 26$ 个字符,  
差别在哪?

例26是从a[1][0]开始修改, 修改到a[1][28], 最多29个字符, 要保证不能修改a[1][29], a[1][29]是尾零, 若修改, 会影响a[1]和a[2]  
而本例是从a[1][3]开始修改, 修改到[1][28], 最多26个字符, 要保证不能修改a[1][29], a[1][29]是尾零, 若修改, 会影响a[1]和a[2]

### 例29：在不同的控制台及字体设置下尾零输出的差异

## 1、新版控制台+新宋体28点阵

## 2、旧版控制台+新宋体28点阵

### 3、旧版控制台+新宋体16点阵

## 结论:

- 本页需填写答案

### 例30：在不同的控制台及字体设置下其它非图形字符输出的差异

```
#include <iostream>
using namespace std;

int main()
{
    int i;
    char a[4] = { 3, 4, 5, 6 };

    cout << "0          1          2          " << endl; //标尺
    cout << "012345678901234567890123456789" << endl; //标尺

    for (i = 0; i < 4; i++)
        cout << a[i] << '$'; //确认a[i]是否输出

    cout << '#' << endl; //加行尾识别符

    return 0;
}
```

```
C:\Users\lei\Desktop\Project\Debug>Test.exe
0          1          2
012345678901234567890123456789
♥$♦$*$$♠$#
C:\Users\lei\Desktop\Project\Debug>
```

```
C:\Users\lei\Desktop\Project\Debug>Test.exe
0      1      2
012345678901234567890123456789
1 $  $  $  $ #
C:\Users\lei\Desktop\Project\Debug>
```

上页的结论1也\_\_适用\_\_(适用/不适用)  
于其它非图形字符

本页需填写答案