高级语言程序设计

主讲：王老师

尚德机构

学习是一种信仰

第五章　数组

数组

数组是具有相同数据类型的一组有序数据的集合。

数组中的数据称为数组元素，数组元素通过数组名和下标引用，下标是数组元素在数组中的位置序号，表示该数组元素在数组中的相对位置。

一维数组

一维数组的定义：

数组同变量一样，必须先定义后使用。数组占用内存单元，在定义数组时必须指定数组元素的数据类型和数组元素的个数，这样才能为其分配相应的内存单元。

一维数组是只有一个下标的数组，其定义的一般形式：

数据类型符 数组名1[长度1]，数组名2[长度2]，...；

1、“数据类型符”是指数组元素的数据类型。数据类型可以是任何基本类型。

2、“数组名”与变量名一样，必须遵循标识符的命名规则。

一维数组

3、“长度”必须是整型的常量表达式，必须用一对方括号括起来，表示数组的元素个数（又称数组长度），可由整型常量或符号常量组成，但其中不能出现变量。需要注意的是，在数组定义时“长度”外面的一对方括号，以及数组元素引用时“下标表达式”外面的一对方括号，都是C语言语法规则要求的。

4、数组元素的下标是元素相对于数组首地址或起始地址的偏移量，所以从0开始顺序编号。

5、数组名表示整个数组所占用的内存空间的首地址。同一数组中的所有元素，按其下标的顺序占用若干连续的存储单元。

6、一个数组定义语句中可以只定义一个数组，也可以定义多个数组，还可以同时定义数组和变量。

一维数组元素的引用

引用一维数组中任意一个元素的方法：

数组名[下标]

1、“下标”可以是一个整型常量、整型变量或整型表达式，其取值范围是0~（长度-1）。需要注意的是，在C程序运行过程中，系统并不自动检查数组元素的下标是否越界，即下标可以不在0~（长度-1）的合法范围内。

2、任何一个数组元素，本质上就是一个变量，它具有和相同类型单个变量一样的属性，可以被赋值，可以接收键盘输入的数据，也可以组成表达式。如果数组元素参与表达式运算，则必须已被赋值。

一维数组的定义和引用

★一维数组的定义

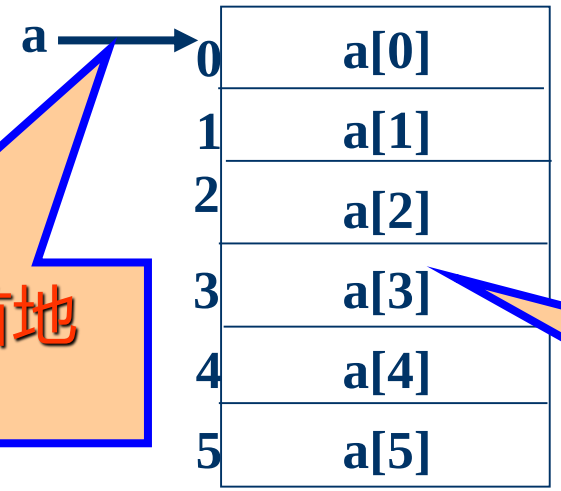
❖定义方式： 数据类型 数组名[常量表达式];

表示元素个数
下标从0开始

例 int a[6];

合法标识符

[ ]：数组运算符
单目运算符
优先级(1)，左结合
不能用( )



数组名表示内存首地址，是地址常量

编译时分配连续内存
内存字节数=数组大小×每元素的字节

例 int i=15;
int data[i]; (×不能用变量定义数组维数)

例 int data[5]; //C语言对数组不作越界检查，使用时要注意
data[5]=10;

★一维数组元素的引用

- ❖数组必须先定义，后使用
- ❖只能逐个引用数组元素，不能一次引用整个数组。
- ❖数组元素表示形式： 数组名[下标]

  其中：下标可以是常量或整型表达式

例 int a[10];

printf(“%d”,a); (×)

必须 for(j=0;j<10;j++)

printf(“%d\t”,a[j]); (✓)

```
#include <stdio.h>
void main()
{
   int i,a[10];
   for(i=0;i<=9;i++)
      a[i]=i;
   for(i=9;i>=0;i- -)
       printf(“%d”,a[i]);
}
```

运行结果：

9 8 7 6 5 4 3 2 1 0

一维数组的初始化

数组元素和变量一样，可以在定义时对数组元素赋初值，称为数组的初始化。

一维数组初始化的一般格式：数据类型符 数组名[长度]={初值表},...;

1、如果对数组的全部元素赋初值，定义时数组长度可以省略（系统根据初值个数自动确定）；如果被定义数组的长度，与初值个数不同，则数组长度不能省略。

2、“初值表”中初值个数、可以少于数组元素的个数，即允许只给部分元素赋初值。若只对数组的前若干个元素赋初值，则没有给出初值的元素均有默认的初值。对于数值型数组默认的初值为0；对于字符型数组，默认的初值为空字符“‘\0’”（ASCII码值为0）。

一维数组的初始化

具体地，一维数组的初始化可以分为以下几种情况：

1、给一维数组的全部元素赋初值。

例如：设int a[4]={1,2,3,4};

2、给一维数组的部分元素赋初值。

例如：设int a[4]={1,2};

3、初值的个数不能超过一维数组元素的个数。

例如：int a[4]={1,2,3,4,5};

一维数组a的长度是4，而初值的个数是5，初值的个数超出了数组长度，则编译时系统会报错。

一维数组的初始化

4、给一维数组的全部元素赋初值时允许省略数组长度的说明。

例如：int a[4]={1,2,3,4};

则可以写成：int a[]={1,2,3,4};

需要注意的是，只能给数组元素赋值，不能给数组名赋值，因为数组名代表数组的首地址，数组名是常量。

★一维数组的初始化

在定义数组时，为数组元素赋初值
(在编译阶段使之得到初值)

实现的方法：

❖在定义数组时对数组元素赋初值。

```
int a[5]={1,2,3,4,5};
等价于：a[0]=1; a[1]=2; a[2]=3; a[3]=4; a[4]=5;
```

❖只给一部分元素赋值。

```
如   int a[5]={6,2,3};
等价于：   a[0]=6; a[1]=2;a[2]=3; a[3]=0; a[4]=0;
如    int a[3]={6,2,3,5,1};    (×)
```

❖数组元素值全部为0。

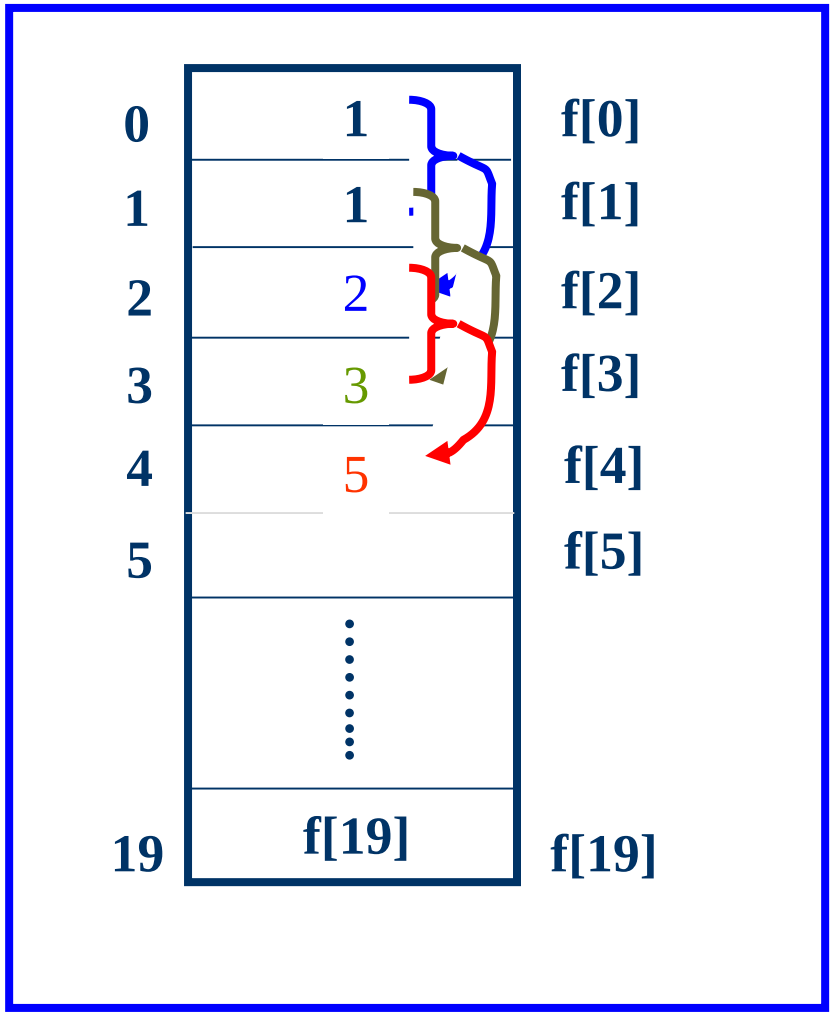
```
int a[5]={0,0,0,0,0};     或int a[5]={0};
```

❖对整个数组元素赋初值时，可以不指定长度。

```
int a[]={1,2,3,4,5,6};
编译系统根据初值个数确定数组大小
```

例 用数组来处理求Fibonacci数列问题

$$F1=1 \qquad (n=1)$$
$$F2=1 \qquad (n=2)$$
$$F_n=F_{n-1}+F_{n-2} \qquad (n\geq 3)$$




```
#include <stdio.h>
void main()
{   int i;
    int f[20]={1,1};
   for(i=2;i<20;i++)
      f[i]=f[i-2]+f[i-1];
   for(i=0;i<20;i++)
   {   if(i%5==0)  printf("\n");
       printf("%12d",f[i]);}
}
```

运行结果：

```
           1           1           2           3           5
           8          13          21          34          55
          89         144         233         377         610
         987        1597        2584        4181        6765
```

课堂练习

1、设int a［10］；则数组a的下标正确的使用范围是（　　）。

A. 0到9

B. 0到10

C. -1到9

D. 1到10

课堂练习

1、设int a［10］；则数组a的下标正确的使用范围是（　　）。

A. 0到9

B. 0到10

C. -1到9

D. 1到10

答案：A

解析：数组名[下标]其“下标”可以是一个整型常量、整型变量或整型表达式，其下标取值范围为0~（长度-1）。故a［10］；的下标是从0到9。

课堂练习

2、设int a[5]={1,2,3,4}；，元素a[‘E’ - ‘A’]的值是_____。

课堂练习

2、设int a[5]={1,2,3,4}；，元素a[‘E’ - ‘A’]的值是_____。

答案：0

解析：在ASCⅡ中，A的值为65，E的值为69，所以a[‘E’ - ‘A’]就等于a[4]。

在一维数组中，编号也是从0开始编号，因为int a[5]={1,2,3,4}；，所以a[4]=0。

二维数组

二维数组的定义

二维数组也需要先定义后使用，二维数组的定义形式：

数据类型符 数组名[行长度][列长度],...;

1、数据类型可以是任何基本类型。

2、数组名和变量名一样，必须遵循标识符的命名规则。

3、行长度说明二维数组有几行，列长度说明二维数组有几列。行长度和列长度都是一个“整型常量表达式”，表达式中不能出现变量。

4、二维数组的元素个数=行长度*列长度。

二维数组

5、二维数组元素在内存中的排列顺序是“按行存放”，即先顺序存放第一行的各个元素，再存放第二行的各个元素，依此类推。

6、可以把二维数组看作是一种特殊的一维数组，其每个元素都是一个一维数组。

7、一个数组定义语句中可以只定义一个二维数组，也可以定义多个二维数组；可以在一条定义语句中同时定义一维和二维数组，同时还可以定义变量。

❖二维数组理解

0	a[0][0]	a[0]
1	a[0][1]	
2	a[0][2]	
3	a[0][3]	
4	a[1][0]	a[1]
5	a[1][1]	
6	a[1][2]	
7	a[1][3]	
8	a[2][0]	a[2]
9	a[2][1]	
10	a[2][2]	
11	a[2][3]	

二维数组元素的引用

定义了二维数组后，就可以引用该数组的任意元素。二维数组元素的引用方法：

数组名[行下标][列下标]

1、“行下标”和“列下标”可以是整型常量、整型变量、整型表达式或符号常量。

2、“行下标”和“列下标”均从0开始，都应在已定义数组大小的范围内，不能越界。行下标的合法取值范围是0~（长度-1），列下标的合法取值范围是0~（长度-1）。

3、与一维数组元素的引用方法相同，任何一个二维数组元素均可以看作一个变量，它具有和同类型单个变量一样的属性，可以被赋值，可以接收键盘输入的数据，也可以组成表达式。如果二维数组元素参与表达式运算，则必须已被赋值。

二维数组的初始化

二维数组的初始化分为以下几种情况。

1、按行给二维数组的全部元素赋初值，其一般形式：

数据类型符　数组名[行长度][列长度]={{第0行初值表},{第1行初值表},...,{最后一行初值表}};

2、按二维数组在内存中的排列顺序给各元素赋初值，即不分行给二维数组的全部元素赋初值，其一般形式：

数据类型符　数组名[行长度][列长度]={初值表};

二维数组的初始化

3、只对每行的前若干个元素赋初值，此时所有未赋初值的二维数组二维数组元素均为0值（即对于整型二维数组是0，对于实型二维数组是0.0，对于字符型二维数组是 ‘\0’ ）。

4、只对前若干行的前若干个元素赋初值，此时所有未赋初值的的数组元素均为0值。

5、如果对全部元素都赋初值，则“行长度”可以省略。需要注意的是只能省略“行长度”，但第二维的长度即“列长度”不能省略。使用这种方式赋初值，允许给出的初值不是列长度的整数倍。此时，行长度=初值个数整除列长度后加1。

6、如果分行给所有行的前若干个元素赋初值，则行长度可以省略。

★二维数组的初始化

- ❖分行初始化
- ❖按元素排列顺序初始化

第一维长度省略初始化

二部初始化

例　int a[][3]={1,2,3,4,5};

1	2	3	4	5	0
a[0][0]	a[0][1]	a[0][2]	a[1][0]	a[1][1]	a[1][2]

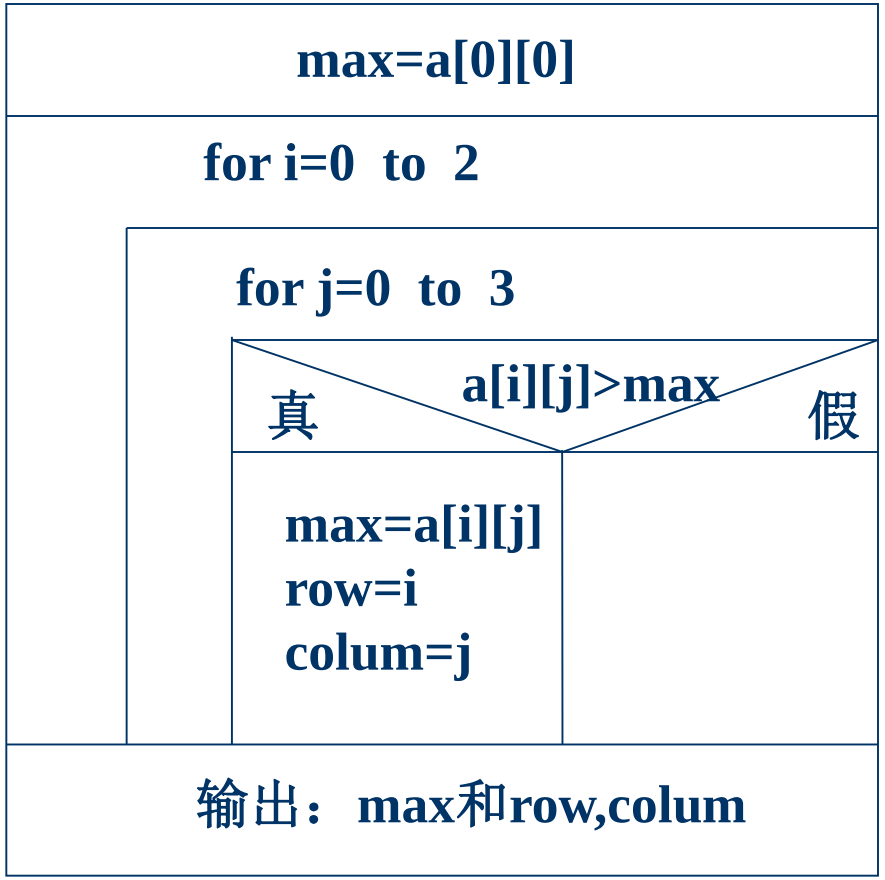
★二维数组程序举例

例 将二维数组行列元素互换，存到另一个数组中

$$a=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix} \qquad b=\begin{bmatrix}1 & 4\\ 2 & 5\\ 3 & 6\end{bmatrix}$$

```
#include <stdio.h>
void main()
{ int a[2][3]={{1,2,3},{4,5,6}};
  int b[3][2],i,j;
  printf("array a:\n");
  for(i=0;i<=1;i++)
   { for(j=0;j<=2;j++)
       { printf("%5d",a[i][j]);
         b[j][i]=a[i][j];}
       printf("\n");
   }
  printf("array b:\n");
  for(i=0;i<=2;i++)
   { for(j=0;j<=1;j++)
       printf("%5d",b[i][j]);
      printf("\n");}
}
```

例 求二维数组中最大元素值及其行列号




```
#include <stdio.h>
void main()
{ int a[3][4]={{1,2,3,4},
        {9,8,7,6},{-10,10,-5,2}};
  int i,j,row=0,colum=0,max;
  max=a[0][0];
  for(i=0;i<=2;i++)
    for(j=0;j<=3;j++)
      if(a[i][j]>max)
       { max=a[i][j];
         row=i;
         colum=j;
       }
  printf("max=%d,row=%d,
 colum=%d\n",max,row,colum);
}
```

课堂练习

3、设int a[2][3]={{1},{2,3}};,则元素a[1][0]的值是（　）。

A. 0

B. 1

C. 2

D. 3

课堂练习

3、设int a[2][3]={{1},{2,3}};,则元素a[1][0]的值是（　）。

A. 0

B. 1

C. 2

D. 3

答案：C

解析：二维数组int a[2][3]={{1},{2,3}};,是一个2行3列的二维数组，故元素a[1][0]的值是2。

课堂练习

4、设float a [4][5]；则数组a中第一维下标的最大值是_____。

课堂练习

4、设float a [4][5]；则数组a中第一维下标的最大值是_____。

答案：3

解析：在二维数组中，行长度是第一维，列长度是第二维。“行下标”和“列下标”均是从0开始编号，所以行长度的取值范围是0~3，则数组a中第一维下标的最大值是3。

字符数组和字符串

字符数组的定义和初始化：

一维字符数组，用于存储和处理一个字符串，其定义格式与一维数值型数组一样。

一维字符数组的定义形式：

char 数组名[数组长度]={初值表};

其功能是定义一维字符型数组，并为其赋初值。

例　char c[10], ch[3][4];

字符数组和字符串

二维字符数组用于同时存储和处理多个字符串，其定义格式与二维数值型数组一样。二维字符数组定义形式：

char 数组名[行长度][列长度]={{初值表},{初值表},...,{初值表}};

其功能是定义二维字符型数组，并为其赋初值。

字符数组

字符数组：存放字符数据的数组。

一维字符数组：存放一个字符串（每个数组元素存放一个字符）

二维字符数组：存放多个字符串（行数是字符串的个数）

★字符数组的定义

❖形式：

- char 数组名[常量表达式]
- char 数组名[常量表达式][常量表达式]
- 常量表达式：整数、字符、符号常量

例 char c[10], ch[3][4];

❖可以用整型数组存放字符型数据，但浪费存储空间。

一维字符数组的初始化

1、一维字符数组的初始化

（1）用字符初始化字符数组

例如：char str1[8]={ ‘p’ ，‘r’ ，‘o’ ，‘g’ ，‘r’ ，‘a’ ，‘m’ ，‘\0’ };

char str2[5]={ ‘C’ ，‘h’ ，‘i’ ，‘n’ ，‘a’ };

（2）用字符串初始化字符数组

例如：char str1[8]={ “program” }

字符串是用双引号括起来的一串字符。在字符串末尾系统会自动地添加一个字符串结束标志符 ‘\0’ ，表示字符串在此位置结束。字符 ‘\0’ 的ASCII码值为0，称为空字符，它不作为字符串的有效标志，只起到标志结束的作用，但需要占用1字节的内存空间。

一维字符数组的初始化

（3）用字符的ASCII码值初始化字符数组。

例如：char str1[8]={112,114,111,103,114,97,109,0};

（4）初始化时如果只提供了部分元素的值。未提供初值的元素自动赋值为 ‘\0’ 。

例如：char str1[10]={“program”}

二维字符数组的初始化

2、二维字符数组的初始化

（1）用字符初始化二维字符数组。

例如：char s[2][10]

={{ ‘c’ ，‘o’ ，‘m’ ，‘p’ ，‘u’ ，‘t’ ，‘e’ ，‘r’ },

{ ‘s’ ，‘c’ ，‘i’ ，‘e’ ，‘n’ ，‘c’ ，‘e’ }};

（2）用字符串初始化二维字符数组。

例如：char s[2][10]={“computer”，“science”};

二维字符数组在初始化时，未提供初值的元素自动赋值为 ‘\0’ 。

★字符数组的初始化

- ❖逐个字符赋值
- ❖用字符串常量

二维字符数组初始化

例 char diamond[][5]={{' ',' ','*'},{' ','*',' ','*'},
{'*',' ',' ',' ','*'},{' ','*',' ','*'},{' ',' ','*'}};

diamond[0]			*	\0	\0
diamond[1]		*		*	\0
diamond[2]	*				*
diamond[3]		*		*	\0
diamond[4]			*	\0	\0

字符数组的输入和输出

1、一维字符数组的输入和输出

（1）使用scanf()和printf()函数

逐个字符输入和输出。在scanf()函数或printf()函数中用“%c”格式说明符每次输入或输出一个字符，重复该过程完成整个字符串的输入和输出。

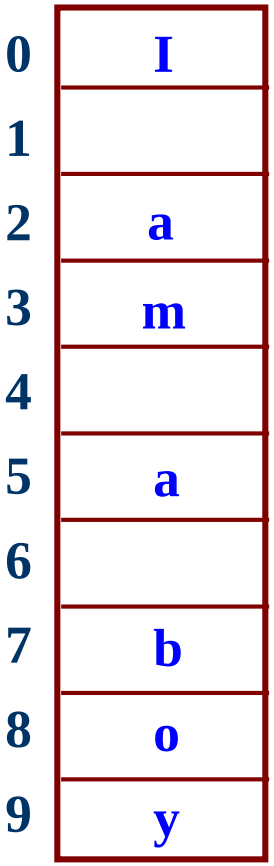
将字符串整个输入和输出。在scanf()函数或printf()函数中用“%s”格式说明符一次性输入或输出整个字符串。

（2）使用gets()和puts()函数。使用字符串输入函数gets()或字符串输出函数puts()将字符串整个输入或输出。

★字符数组的引用

输出一个字符串

```
#include <stdio.h>
void main()
{
    char c[10]={'I',' ','a','m',' ','a',' ','b','o','y'};
    int i;
    for(i=0;i<10;i++)
       printf("%c",c[i]);
    printf("\n");
}
```



★字符数组的输入输出

- ❖逐个字符I/O：　%c
- ❖整个字符串I/O：　%

用字符数组名,不要加&
长度<数组维数
或回车结束
动加 '\0'

```
例 用%c
void main()
{   char  str[5];
    int i;
    for(i=0;i<5;i++)
        scanf("%c", &str[i]);
    for(i=0;i<5;i++)
        printf("%c", str[i]);
}
```

```
例
vo
{

}
```

输入：China ↵
输出：China
输入：Program ↵
输出：Progr

用字符数组名,
遇 '\0'结束

其它注意事项：

```
#include <stdio.h>
void main()
{
    char a[ ]={'h','e','l','\0','l','o','\0'};
    printf("%s",a);
}
```

输出：hel

h	e	l	\0	l	o	\0

数组中有多个‘\0’时，
遇第一个结束

二维字符数组的输入和输出

（1）二维字符数组的输入。

对于二维字符数组，除了可以使用初始化的方法赋值外，还可以从键盘赋值。二维字符数组的每一行可以看作一个一维字符数组，可以将二维字符数组的每一行作为一个一维字符数组进行输入和输出。

(2) 二维字符数组的输出。

二维字符数组的输出一般有三种方式：

方法一：

```
int i,j;
for(i=0; i<3; i++)
{
        for(j=0; s[i][j]!='\0'; j++)
            printf("%c",s[i][j]);
        printf("\n");
}
```

方法二：

```
int i;
for(i=0; i<3; i++)
      printf("%s\n",s[i]);
```

方法三：

```
int i;
for(i=0; i<3; i++)
      puts(s[i]);
```

❖用字符串常量初始化字符数组

二维字符数组初始化

例 char fruit[][7]={”Apple”,”Orange”,
”Grape”,”Pear”,”Peach”};

fruit[0]	A	p	p	l	e	\0	\0
fruit[1]	O	r	a	n	g	e	\0
fruit[2]	G	r	a	p	e	\0	\0
fruit[3]	P	e	a	r	\0	\0	\0
fruit[4]	P	e	a	c	h	\0	\0

字符串的定义和初始化

字符串是指若干有效字符的序列，可以包括字母、数字、专用字符和转义字符等。

赋值方法有：

- 第一种方式是按单个字符的方式赋初值，其必须有一个是字符串的结束标志符 ‘\0’ ；
- 第二种方式是直接在初值表中给出一个字符串常量。

需要注意的是，由于系统在存储字符串常量时，会在字符串末尾自动加上一个结束标志符 ‘\0’ ，所以无需认为添加。另外，由于字符串结束标志符 ‘\0’ 也需要在字符数组中占用一个元素的存储空间，即1字节，因此在说明字符数组长度时，至少为字符串所需长度加1。

字符串的输入和输出

1、使用函数scanf( )和printf( )

（1）逐个字符的输入和输出。在scanf()或printf()函数中使用“%c”格式说明符每次输入或输出一个字符，重复执行多次完成整个字符串的输入或输出。

（2）字符串的整体输入和输出。在scanf()或printf()函数中使用“%s”格式说明符一次性输入或输出整个字符串。

字符串的输入和输出

说明：

（1）scanf()函数或printf()函数中，与“%s”格式说明符对应的输入项或输出项应是地址。

（2）使用“%s”格式说明符输入字符串时，系统会自动在字符串末尾加一个‘\0’，将字符串连同添加的‘\0’一起存入字符数组中。

（3）使用“%s”格式说明符输入字符串时，遇到空格或回车则认为字符串输入结束。

（4）使用“%s”格式说明符输出字符串时，一旦遇到‘\0’则结束输出（‘\0’不输出），其后字符不在输出。

字符串的输入和输出

2、使用函数gets( )和puts( )

使用字符串输出函数gets()或字符串输入函数puts()将字符串整体输入或输出。

字符串输入函数gets( )

字符串输入函数gets( )

【调用格式】gets(字符数组名)

【参数】字符数组名是已经定义的字符数组名。

【功能】从标准输入设备即键盘上输入一个字符串（可以包含空格，仅以回车作为结束标志），并将其存储到指定的字符数组中。

【返回值】字符数组的首地址。

字符串输入函数gets( )

【说明】

1、gets( )函数输入的字符串的长度没有限制，编程者应保证字符数组有足够大的空间，存放输入的字符串。

2、gets( )函数和使用“%s”格式说明符的scanf()函数都可以从键盘输入字符串，但两者是有区别的。对于scanf()函数，回车和空格符都看成是输入字符串的结束标志；对于gets()函数输入字符串中允许包含空格，只有回车才看作是输入字符串的结束标志。

❖字符串输入函数gets

- 格式：gets (字符数组)
- 功能：从键盘输入一个以回车结束的字符串放入字符数组中，并自动加 ‘\0’ 。
- 说明：输入串长度应小于字符数组维数

例：gets和scanf输入比较

```
#include <stdio.h>
void main( )
{
   char  a1[15], a2[15] ;
   gets(a1);
   scanf(“%s”,a2);
   printf (“a1=%s\ n”,a1);
   printf (“a2=%s\ n”,a2);
}
```

注意：puts和gets函数只能输入输出一个字符串。

错 puts(str1,str2)　gets(str1,str2)

输入：china beijing ↵
　　china beijing ↵
输出：a1=china beijing
　　a2=china

字符串输出函数puts( )

字符串输出函数puts( )

【调用格式】puts( )

【参数】字符数组名是已经存放字符串的字符数组名。

【功能】把字符数组名中所存放的字符串，输出到标准输出设备即显示器，并以‘\n’取代字符串的结束标志‘\0’。所以用puts()函数输出字符串时，不要求另外加换行符。

【返回值】无。

字符串输出函数puts( )

【说明】

1、字符串中允许包含转义字符，输出时产生一个控制操作。

2、puts()函数和使用“%s”格式说明符的printf()函数都可以输出字符串，但两者是有区别的。对于printf()函数，不输出字符串结束标志符；对于puts()函数，字符串结束标志符‘\0’转换为‘\n’输出。此外，puts()函数一次只能输出一个字符串，而printf()函数用来输出字符串时一次可以输出多个。

★字符串处理函数

包含在头文件 string.h 中

❖字符串输出函数 puts

- 格式： puts（字符数组）
- 功能：向显示器输出一个字符串（输出完，换行）
- 说明：字符数组必须以 ‘\0’结束。可以包含转义字符。
  输出时 ‘\0’转换成 ‘\n’，即输出字符后换行。

这里是将
‘\0’ → ‘\n’
因此光标移
到下行

运行结果：
china
beijing
china
WUHAN

例：

```
#include <stdio.h>
void main( )
{ char  a1[ ]="china\ nbeijing" ;
  char  a2[ ]="china\ 0beijing" ;
  puts(a1);  puts(a2);
  puts("WUHAN" );
}
```

字符串比较函数strcmp( )

字符串比较函数strcmp( )

【调用格式】strcmp(字符串1,字符串2)

【参数】“字符串1”和“字符串2”可以是字符串常量，也可以是已经存放字符串的字符数组名。

【功能】比较两个字符串的大小，即两个字符串从左到右逐个字符进行比较（按照ASCII码值的大小进行比较），直到出现不同字符或遇到‘\0’为止。

如果“字符串1”等于“字符串2”，则函数返回值为0。

如果“字符串1”大于“字符串2”，则函数返回值是正整数（小于0的整数）。

如果“字符串1”小于“字符串2”，则函数返回值是负整数（大于0的整数）。

字符串比较函数strcmp( )

【说明】

1、如果一个字符串是另一个字符串从头开始的子串，则母串为大。

2、不能使用关系运算符“==”比较两个字符串，只能调用strcmp()函数进行处理。

❖字符串比较函数strcmp

- 格式：strcmp(字符串1,字符串2)
- 功能：比较两个字符串
- 比较规则：对两串从左向右逐个字符比较（ASCII码），
  直到遇到不同字符或‘\0’为止。
- 返回值：返回int型整数。其值是ASCII码的差值
  a. 若字符串1< 字符串2， 返回负整数
  b. 若字符串1> 字符串2， 返回正整数
  c. 若字符串1= = 字符串2， 返回零
- 说明：字符串比较不能用“==”,必须用strcmp，
  虽然编译无错，但结果不对

错 `if(str1==str2) printf(“yes”);`

`if(strcmp(str1,str2)==0) printf(“yes”);` 对

例：字符比较

```
#include <stdio.h>
void main( )
{ int  i,j,k;
  char  a1[ ]="wuhan",  a2[ ]="beijing" ;
  i=strcmp(a1,a2);
  j=strcmp("china", "korea");
  k=strcmp(a2, "beijing" );
  printf("i=%d\ nj=%d\ nk=%d\ n",i,j,k);
}
```

运行结果：

i=21

j=-8

k=0

i=21　　i=w-b=119-98=21

j=-8　　j=c-k=99-107=-8

k=0　　k=b-b=98-98=0

字符串复制函数strcpy( )

【调用格式】strcpy(字符数组名,字符串[,整型表达式])

【参数】“字符数组名”是已经定义的字符数组名。

“字符串”可以是字符串常量，也可以是已经存放字符串的字符数组名。

“整型表达式”可以是任何整型表达式，这一参数可以省略。

【功能】将“字符串”的前“整型表达式”个字符组成新的字符串复制到“字符数组”中，若省略“整型表达式”，则将“字符串”完整地复制到“字符数组中”，字符数组的原有内容被覆盖。

【返回值】字符数组的首地址

字符串复制函数strcpy( )

【说明】

1、字符数组长度必须足够大，以便容纳复制过来的字符串。复制时，连同字符串结束标志 ‘\0’ 一起复制。

2、不能用赋值运算符 “=” 将一个字符串直接赋值给一个字符数组，只能调用strcpy( )函数处理。

❖字符串拷贝函数strcpy

- 格式：strcpy(字符数组1,字符串2)
- 功能：将字符串2，拷贝到字符数组1中去
- 返值：返回字符数组1的首地址
- 说明：①字符数组1必须足够大，>字符串2

  ②字符数组1必须是数组名形式（str1），字符串 2可以是字符数组名或字符串常量。

  ③拷贝时‘\0’一同拷贝

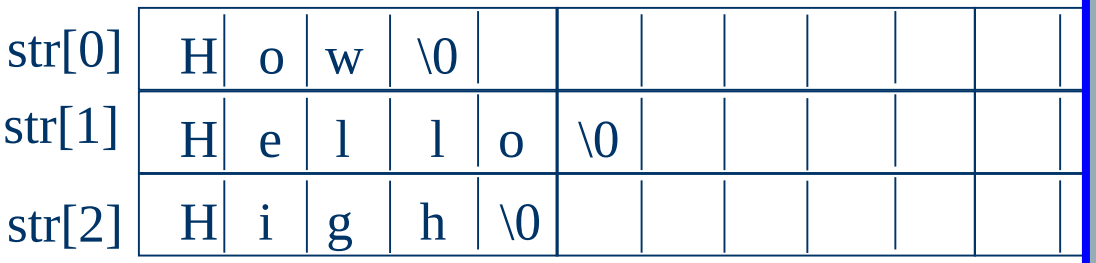
  ④不能使用赋值语句为一个字符数组赋值

例 char str1[20],str2[20];
str1={“Hello!”}; (×)
str2=str1; (×)

⑤可以只复制字符串2中的前几个字符，来取代字符数组1的前几个字符。

strcpy(str1,str2,2) —— 复制前2个。

例 有3个字符串，要求找出其中最大者。




```
#include <stdio.h>
#include <string.h>
void main()
{ char string[20],str[3][20];
   int i;
   for(i=0;i<3;i++)
      gets(str[i]);
   if(strcmp(str[0],str[1])>0)
            strcpy(string,str[0]);
   else  strcpy(string,str[1]);
   if(strcmp(str[2],string)>0)
            strcpy(string,str[2]);
   printf("\nThe largest string
                  is:\n%s\n",string);
}
```

字符串连接函数strcat( )

【调用格式】strcat(字符数组名,字符串)

【参数】“字符数组名”是已经定义的存放字符串的字符数组名。

“字符串”可以是字符串常量，也可以是已经存放字符串的字符数组名。

【功能】把“字符串”连接到“字符数组”中字符串的尾端（最后一个有效字符的后面），组成新的字符串并存储到“字符数组”。“字符数组”中原来的结束标志，被“字符串”的第一个字符覆盖，而“字符串”在操作中未被修改。

【返回值】字符数组的首地址。

字符串连接函数strcat( )

【说明】

1、由于没有边界检查，编程者应注意保证字符数组长度足够大，以便容纳连接后的新字符串；否则，会因长度不够出现问题。

2、连接前两个字符串都有结束标志 ‘\0’ ，连接后字符数组中存储的字符串结束标志 ‘\0’ 被舍弃，只在新字符串的最后保留一个 ‘\0’ 。

❖字符串连接函数strcat

- 格式：strcat (字符数组1,字符数组2)
- 功能：把字符数组2连到字符数组1后面
- 返值：返回字符数组1的首地址
- 说明：①字符数组1必须足够大
  ②连接前,两串均以‘\0’结束;连接后,串1的‘\0’取消,新串最后加‘\0’。

例:

```
#include <stdio.h>
void main( )
{
  char  str1[30]={“People’s Republic of “};
  char str2[]={China”};
  printf (“%s\n”,strcat(str1,str2));
}
```

str1: People’s Republic of \0
str2: china\0
str1: People’s Republic of china\0

测字符串长度函数strlen( )（其中len是length的缩写）

【调用格式】strlen(字符串)

【参数】“字符串”可以是字符串常量，也可以是已经存放字符串的字符数组名。

【功能】测字符串（字符串常量或字符数组）的实际长度（不包含字符串结束标志符 ‘\0’ ）。

【返回值】字符串的实际长度。

❖字符串长度函数strlen

- 格式：strlen(字符数组)
- 功能：计算字符串长度
- 返值：返回字符串实际长度，不包括‘\0’在内

例 对于以下字符串，strlen(s)的值为：

(1) char s[10]={‘A’,‘\0’,‘B’,‘C’,‘\0’,‘D’};

(2) char s[ ]=“\t\v\\\0will\n”;

(3) char s[ ]=“\x69\082\n”;

答案：1 3 1

例：测试字符串长度

```
#include <stdio.h>
void main( )
{
    char a1[10]=“ china” ;
    printf (“%d\ n”,strlen(a1));
    printf (“%d\ n”, strlen(“beijing\ 0wuhan”));
}
```

运行结果：5
7

例 把输入的字符串逆序排列，并显示。

逆序排列用交换算法，求出字符串最后一个字符的下标，然后将第一个和最后一个交换，第二个和倒数第二个交换，…。

```
#include <stdio.h>
void main()
{
    char str[80];
    int temp,i,j;
    printf("Enter a string:\n");
    scanf("%s",str);
    for(i=0,j=strlen(str)-1;i<j;i++,j--)
    {  temp=str[i];
       str[i]=str[j];                    /*交换i,j两个元素*/
       str[j]=temp;
    }
    printf("\nReversed string:\n%s\n",str);
}
```

字符串大写字母转换成小写函数strlwr( )

【调用格式】strlwr(字符串)

【参数】“字符串”可以是字符串常量，也可以是已经存放字符串的字符数组名。

【功能】将字符串中的大写字母转换成小写字母，其他字符（包括小写字母和非字母字符）不转换。

字符串小写字母转换成大写函数strupr( )

【调用格式】strupr(字符串)

【参数】“字符串”可以是字符串常量，也可以是已经存放字符串的字符数组名。

【功能】将字符串中的小写字母转换成大写字母，其他字符（包括大写字母和非字母字符）不转换。

- **大写字母转换成小写字母函数strlwr**
  - 格式：strlwr(字符串)
- **小写字母转换成大写字母函数strupr**
  - 格式：strupr(字符串)

例：字符转换

```
#include <stdio.h>
void main(   )
{ char  a1[6]="CHinA", a2[ ]="wuHAn" ;
   printf ("%s\ n",strlwr(a1)); printf ("%s\ n",strupr(a2));
}
```

运行结果：china
WUHAN

课堂练习

5、设char s[10]=〞abcde〞,t[ ]=〞12345〞；,则s和t在内存中分配的字节数分别是（）

A. 6和5

B. 6和6

C.10和5

D.10和6

课堂练习

5、设char s[10]=〞abcde〞,t[ ]=〞12345〞；,则s和t在内存中分配的字节数分别是（）

A:6和5　　B:6和6

C:10和5　　D:10和6

答案：D

解析：因为定义数组char s[10]=〞abcde〞中有10个元素，每个元素存放一个字节，所以是10个字节。数组t[ ]=〞12345〞没有定义有多少个元素，但是已经存在5个元素，一般在字符串末尾系统会自动添加一个字符串结束标志符'\0’表示字符串在此位置结束，所以字节数应为6。

课堂练习

2、设char s[20]=“China”；，执行语句printf（“%d\n”，strlen(s)）；后输出结果是_____。

课堂练习

2、设char s[20]=“China”；，执行语句printf（“%d\n”，strlen(s)）；后输出结果是_____。

答案：5

解析：strlen计算的是字符串的长度，其返回值为字符串的实际长度，因为char s[20]=“China”；，“China”的字符串的长度为5，所以printf（“%d\n”，strlen(s)）；最后的输出结果为5。

祝大家顺利通过考试！