



第六章

数 组

实例

例：输入10个学生的成绩，计算平均成绩，并统计成绩高于平均成绩的人数。

如何计算平均成绩？在循环中已解决。

如何统计高于平均成绩的人数？

用每个人的成绩与平均成绩比较

如何存储每个人的成绩？

批量数据的存储。

用变量存储：变量之间无联系，不方便操作。

解决方法：使用数组。

```
#include <stdio.h>
void main()
{int k=0;
float
j1, j2, j3, j4, j5, j6, j7, j8, j9, j10,
s=0, a;
scanf("%f",&j1); scanf("%f", &j2);
scanf("%f", &j3);
scanf("%f", &j4); scanf("%f", &j5);
scanf("%f", &j6);
scanf("%f", &j7); scanf("%f", &j8);
scanf("%f", &j9);
scanf("%f", &j10);
s=j1+j2+j3+j4+j5+j6+j7+j8+j9+j10;
a=s/10;
if(j1>=a) k++; if(j2>=a) k++;
if(j3>=a) k++;
if(j4>=a) k++; if(j5>=a) k++;
if(j6>=a) k++;
```

使用数组，
简化程序。

```
#include <stdio.h>
void main()
{int i, n, k=0;
float
a[10], s=0, ave;
n=10;
for(i=0;i<n;i++)
{
scanf ("%f", &a[i]);
s=s+a[i];
ave=s/n;
for(i=0;i<n;i++)
```

实例

例：输入三个整数a、b、c，按降序排序后输出。

思路：a存放最大数；b存放中间数；c存放最小数；

输出a、b、c。

实现：若 $a < b$ ：交换a、b的值；

若 $a < c$ ：交换a、c的值；

若 $b < c$ ：交换b、c的值；

输出a、b、c。

扩充：四个数a、b、c、d排序？

十个数如何排序？

实例

10个数排序：

给10个数统一命名：如 a;

如何表示某个数：加序号，如0、1、2.....10

a[0]、a[1]、a[2].....a[9]

用以上方法表示的一组数称为数组。

排序过程：

1) 在a[0].....a[9]之间找最大的，与a[0]交

换；

2) 在a[1].....a[9]之间找最大的，与a[1]交

换；

.....

9) 在a[8].....a[9]之间找最大的，与a[8]交

换

实例

```
#include <stdio.h>
#define n 5
void main()
{int i,j,k,a[10],m;
 for(i=0;i<n;i++)  scanf("%d",&a[i]);
 for(i=0;i<n-1;i++)
 { m=i;
  for(j=i+1;j<n;j++)
   if(a[j]>=a[m])  m=j;
  k=a[i];  a[i]=a[m];  a[m]=k; }
 for(i=0;i<n;i++)
 printf("%4d",a[i]);
}
```

数组的概念

批量数据

一个班学生的学习成绩

一行文字

一个矩阵

这些数据的特点是：

- 1、具有相同的数据类型
 - 2、使用过程中需要保留原始数据
- C语言为这些数据，提供了一种构造数据类型：数组。

数组

是一组具有**相同数据类型**的数据的有序集合。

数组的概念

1. 数组：一组具有相同数据类型的数据的有序集合。

例如：一个班级某课程的成绩。

特点：有共同的类型、共同的名字。

2. 数组元素：数组中的成员。

3. 种类：一维数组、二维数组、字符数组

一维数组：元素按线性排列。

二维数组：元素按行列排列。

4. 下标：数组元素在数组中的位置或序号。

一维数组：1个下标

数组的定义

■ 一维数组的定义

(1) 形式：类型说明符 数组名[常量表达式]

例如：int g1[45] float b[20]

(2) 说明：

① 类型说明符：指明数组中各数据的类型。

int float char

② 数组名：指明数组的标识符。命名规则同变量名。

③ 常量表达式：指明数组的大小。即数组元素个数

数组的定义

■ 二维数组的定义

(1) 形式：

类型说明符 数组名[常量表达式1] [常量表达式
2]

例如：int g2[45][3] float c[20][5]

(2) 说明：

- ① 常量表达式1：指明数组的行数。
- ② 常量表达式2：指明数组的列数。。
- ③ 数组元素个数： $m * n$
- ④ 下标的取值范围 行 0 ~ m - 1 列 0 ~ n - 1

数组元素的表示

1. 一维数组

形式： 数组名[下标]

例如： g1[23] g1[10]

2. 二维数组

形式： 数组名[行下标][列下标]

例如： a1[5][0] a2[35][2]

3. 说明

下标应为整型表达式，且取值范围为0~n-1。

n：一维数组的元素或二维数组的行、列数。

数组的存储

一个数组的各个元素占用一片连续的内存单元。

1. 一维数组

各元素按下标顺序占用存储单元。

例如： int a[5]

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

2. 二维数组

按行序存放，行内按列序存放。

例如： int a[2][3]

	a[0][0]	a[0][1]	a[0][2]	a[1][0]	a[1][1]	a[1][2]	
--	---------	---------	---------	---------	---------	---------	--

数组的使用

例：

一个一维数组,初始值为：87、65、78、92、84。将每个元素的值除10后输出。

分析：

初始值的设置

对元素的处理过程

元素值的输出

初始化
赋值

```
printf("%d",a[0]);  
printf("%d",a[1]);  
printf("%d",a[2]);  
printf("%d",a[3]);  
printf("%d",a[4]);
```

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

数组初始化

■ 一维数组：

① 全部赋值： int a[5]={1,2,3,4,5};

② 前n个赋值： int a[10]={1,2,3,4,5};

只给前面5个元素赋初值，后5个元素值为0。

③ 全部赋0值： int a[10]={0};

不能写成： int a[10]={0*10};

④ 如果对全部数组元素赋初值，可以不指定数组长度。

int a[5]={1,2,3,4,5}; int a[]={1,2,3,4,5};

数组初始化

■ 二维数组

可以用下面**4**种方法对二维数组初始化

(1) 分行给二维数组赋初值。如：

```
int a[3][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};
```

(2) 可以将所有数据写在一个花括弧内，按数组排列的顺序对各元素赋初值。如：

```
int a[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```

数组初始化

(3) 可以对部分元素赋初值。如

```
int a[3][4]={{1}, {5}, {9}};
```

也可以对各行中的某一元素赋初值，如

```
int a[3][4]={{1}, {0,6}, {0,0,11}};
```



1	0	0	0
5	6	0	0
0	0	0	0



1	0	0	0
0	6	0	0
0	0	11	0



1	0	0	0
5	0	0	0
9	0	0	0

也可以只对某几行元素赋初值。如：

```
int a[3][4]={{1}, {5, 6}};
```

数组初始化

(4) 如果对全部元素都赋初值，则定义数组时对第一维的长度可以不指定，但第二维的长度不能省。

如：

```
int a[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```

它等价于：

```
int a[ ][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```

在定义时也可以只对部分元素赋初值而省略第一维的长度，但应分行赋初值。如：

```
int a[ ][4]={ {0,0,3}, {}, {0,10} }; →
```

0	0	3	0
0	0	0	0
0	10	0	0

数组的使用

数组元素的作用

与一个同类型的变量相同。

例如：

被赋值 `aa[2]=4;`

输入、输出 `scanf("%d",&aa[3]);`

`printf("%d",aa[2]+aa[3]);`

参加运算 `aa[2]*2+aa[3]/3`

条件判断 `if(aa[2]>0)`

`while(aa[3]<2)`

数组的使用

数组的赋值

在程序的执行部分，只能通过给数组元素赋值实现对数组整体的赋值。

例如：将一个有20个元素的一维数组赋值为

```
int a[20];
```

```
a[0]=1;
```

```
a[1]=2;
```

```
a[2]=3;
```

```
.....
```

```
a[19]=20;
```

a[i]=i+1

```
void main( )
```

```
{int a[20],i;
```

```
for(i=0;i<20;i++)
```

```
    a[i]=i+1;
```

```
for(i=0;i<20;i++)
```

```
    printf("%d",a[i]);}
```

数组的使用

数组的输入、输出

使用循环结构控制输入、输出数组的全部或部分元素。

一维数组：使用一层循环

```
void main( )
{int a[ ]={1,2,3,4,5,6},i;
for(i=0;i<6;i=i+2)
printf("%d",a[i]);}
```

```
void main( )
{int a[ ][3]={1,2,3,4,5,6},i,j;
for(i=0;i<2;i++)
{for(j=0;j<3;j++)
printf("%d",a[i][j]);
printf("\n");}}
```

字符数组

字符数组：

用来存放字符数据的数组，一个元素存放一个字符。

字符数组的定义

char 数组名[数组的大小]

char 数组名[行数][列数]

char c[80],c1[20][20]

字符串数组—初始化

使用字符常量初始化

```
char c[5]={'t','h','e','r','e'};
```

用字符串常量初始化。

例如 char c[]={"I am happy"};

 char c[]="I am happy";

与下面的数组初始化等价

```
char c[ ]={ 'I','','a','m','','h','a','p','p','y','\0'}
```

 char c[10]={"China"};

字符数组—输入、输出

1、逐个字符输入输出：

用格式符“%c”输入或输出一个字符。

使用循环结构输出所有字符。

2、将整个字符串一次输入或输出：

用“%s”格式符

*输出字符串：

printf函数中的输出项是字符数组名。

printf("%s", c);

输出字符不包括结束符'\\0'。

字符数组—输入、输出

※ 输入字符串：

例如：char c[6]; scanf("%s",c);

输入项使用字符数组名；

输入的字符串应短于已定义的字符数组的长度；

系统自动在字符串后加'\\0'结束符；

输入的字符串中不能含空格；

可同进输入多个字符串，输入时用空格分隔字符串，例如：

char str1[5],str2[5],str3[5];

scanf("%s%s%s",str1,str2,str3);

输入数据一：1234567890

2

字符串函数

1. puts函数

形式: puts (字符数组)

作用: 将一个字符串(以'\\0'结束的字符序列)输出到终端。

如: char st1[]="china";

 puts(st1);

 char str[]={"China\\nBeijing"};

 puts(str);

输出结果:

China

Beijing

字符串数组—字符串函数

2. gets函数

形式： gets(字符数组)

作用： 从终端输入一个字符串到字符数组。例

例： gets(str)

从键盘输入：

Computer ↵

注意：

用puts和gets函数只能输入或输出一个字符串。

不能写成： puts(str1, str2)

或 gets(str1, str2)

注意： 送给
数组的共有
9个字符。

字符串函数

3. strcat函数

形式： `strcat(字符数组1, 字符数组2)`

作用： 连接两个字符数组中的字符串，把字符串2接到字符串1的后面，结果放在字符数组1中。

例如：

```
char str1[30]={" People' s Republic  
of " };
```

```
char str2[ ]={" China" };
```

```
printf ("%s" , strcat(str1, str2));
```

输出：

People's Republic of China

字符串数组—字符串函数

4. strcpy函数

形式: **strcpy(字符数组1, 字符串2)**

作用: 将字符串**2**复制到字符数组**1**中去。

例如:

```
char str1[10], str2[] = {"China"};
```

strcpy(str1, str2);

说明:

1. 字符数组1必须定义得足够大。
2. 复制时连同字符串后面的' \0' 一起复制到字符数组1中。

字符串数组—字符串函数

5. strcmp函数

形式：strcmp(字符串1， 字符串2)

作用：比较字符串1和字符串2。

例如：strcmp(str1, str2)；

strcmp(" China" , " Korea");

strcmp(str1, " Beijing");

字符串比较的规则：

从第一对字符开始；

以第一对不等字符比较结果为准

字符数组—字符串函数

比较的结果由函数值带回

- (1) 如果字符串1=字符串2， 函数值为0。
- (2) 如果字符串1>字符串2， 函数值为一正整数。
- (3) 如果字符串1<字符串2， 函数值为一负整数。

字符数组—字符串函数

6. strlen函数

形式 : `strlen (字符数组)`

作用：测试字符串长度的函数。函数的值为字符串中的实际长度(不包括' \0' 在内)。

如：`char str[10]={" China" };`
`printf(" %d" , strlen(str));`

直接测试字符串常量的长度，如

`strlen(" China");`

字符数组—字符串函数

7. strlwr函数

形式： strlwr (字符串)

作用：将字符串中大写字母换成小写字母。

8. strupr函数

形式： strupr (字符串)

作用：将字符串中小写字母换成大写字母。

程序举例

例1：在一组数中找最小元素，输出其下标和它的值。

分析：

数据结构：数组a，变量m；

算法：

从前向后找：设 $a[0]$ 最小，与其它数比较；

从后向前找：设 $a[n-1]$ 最小，与其它数比较。

程序举例

例2：在一组按升序排列的数中插入一个数，使其仍然有序。

分析：

数据结构：数组a，变量k、m

算法：

找插入位置：

从前向后找：先找到位置，然后移动；

从后向前找：边找边移动。

插入

程序举例

例3：将10个数按从小到大排序并输出。

排序问题：

排序方式：

升序、降序。

排序方法：

选择、冒泡、插入

数组存储方式：

使用数组

■ 查找

顺序查找

折半查找

程序举例

例4：矩阵转置。

分析：

数据结构：二维数组

算法：

$$a[i][j] \leftrightarrow a[j][i]$$

$$1 \leq i \leq n-1$$

$$0 \leq j \leq i$$

程序举例

例5：将十进制数转换成十六进制数，然后输出

分析：

数制转换方法：除16取余

余数的存储：使用字符数组

余数的处理：10~15的转换

转换后数据的输出：从后向前输出。

程序举例

- 更多的算法

- “折半” 查找

- “冒泡法” 排序

- 一维数组的循环移动(左移、右移)

- 在数组中删除一个元素

- 打印 “杨辉三角形”

- 找 “鞍点”

- 字符排序

- 字符串函数的实现