第7章 用数组处理批量数据

§1 什么是数组

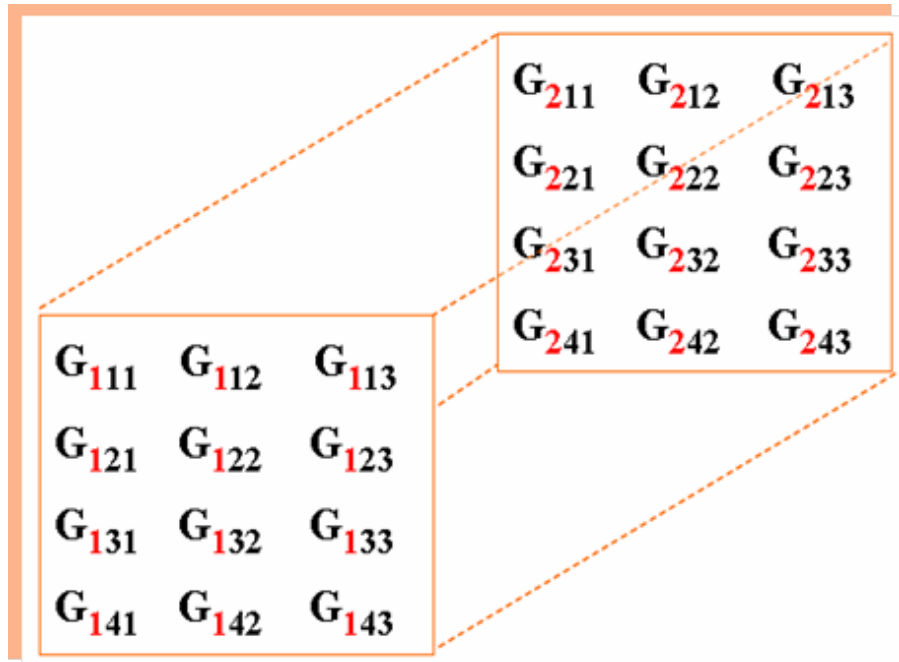
例1:有1000个工厂的月产值(product)，编程对产值加保存，并找出最高产值的工厂编号及其产值。

```
main( )
{    int i,num,p[1001];
     ……
     num=1;          /*num为最大产值厂的编号*/
     max=p[1];       /*假设第1个工厂产值最高*/
     for(i=2;i<=1000;i++)
       if(max<p[i]) { max=p[i]; num=i; }
     printf("NO:%d\tProduct:%d\n",num,max);
 }
```

§1 什么是数组

例2：某班4个同学，每人4门课程成绩(score)。设计合理的成绩表示方法。

学号	课程1	课程2	课程3	课程4
1	s11	s12	s13	s14
2	s21	s22	s23	s24
3	s31	s32	s33	s34
4	s41	s42	s43	s44



例3:某班4个同学,每人3门课成绩 (grade) 。表示2个班成的合理数据结构是什么 ？

§1 什么是数组

数组是同类型、相关数据的有序集合，它由若干个数组元素（也称为下标变量）组成。

数组名

数组名取名采用标识符规则，同一数组的下标变量同数组名，用不同下标加以区分

数组类型

数组的类型是指数组中下标变量的类型

数组逻辑结构

维数：数组元素下标的个数

各维上下界：指各维最后及第一个数组元素下标值

§2 一维数组

1.定义

类型符 数组名[常量表达式], … ;

— 常量表达式之值称作数组长度，即数组元素的个数

2.数组元素(下标变量）的引用

数组名[整型表达式]

— 整型表达式之值是该元素的下标之值

eg1: int digit[10];

eg2: float f1[5],f2[6];

eg5: int digit[10];

……

digit[6]=digit[5]+1;

eg3: #define MAX 1001

…

int p[MAX];

eg4: #define MAX 1000

…

int p[2*MAX];

§2 一维数组

3.说明

C中，数组的维下界例行从0开始

数组中的各元素占据连续的主存单元

C中，数组名代表数组的起始地址

C中，不允许对数组的长度作动态定义

```
#include <stdio.h>
main( )
{ int i,digit[10];
  for(i=1;i<=10;i++)
    digit[i]=i+1;
  for(i=1;i<=10;i++)
   printf("%6d",digit[i]);
}
```

下标越界

§2 一维数组

3.说明

C中，数组的维下界例行从0开始

数组中的各元素占据连续的主存单元

C中，数组名代表数组的起始地址

C中，不允许对数组的长度作动态定义

```
#include <stdio.h>
main( )
{ int i,digit[10];
  for(i=0;i<= 9 ;i++)
    digit[i]=i+1;
  for(i=0;i<=9 ;i++)
   printf("%6d",digit[i]);
 }
```

正确

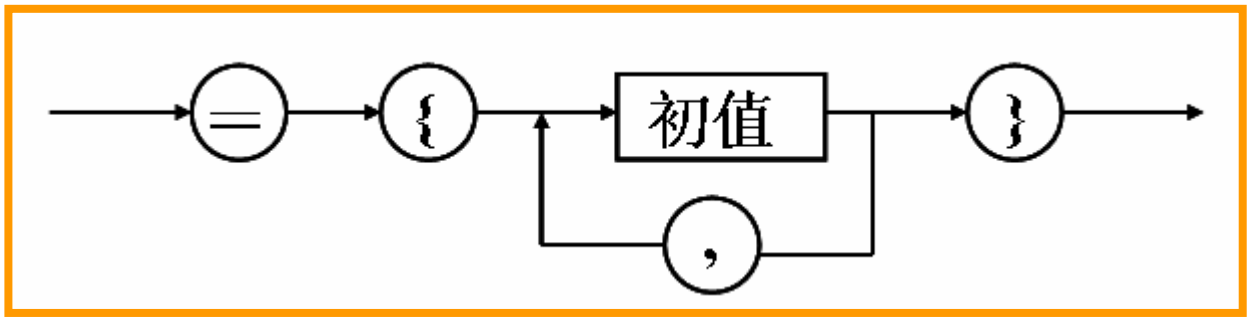
§2 一维数组

3.说明

- C中，数组的维下界例行从0开始
- 数组中的各元素占据连续的主存单元
- C中，数组名代表数组的起始地址
- C中，不允许对数组的长度作动态定义

§2 一维数组

4.数组初始化



eg1: int a[5];

不等价

eg2: int a[5]={0,0,0,0,0};

等价

eg3: int a[5]={0};

eg4: int a[5]={0,1,2};

eg5: int a[5]={0,1,2,3,4};

等价

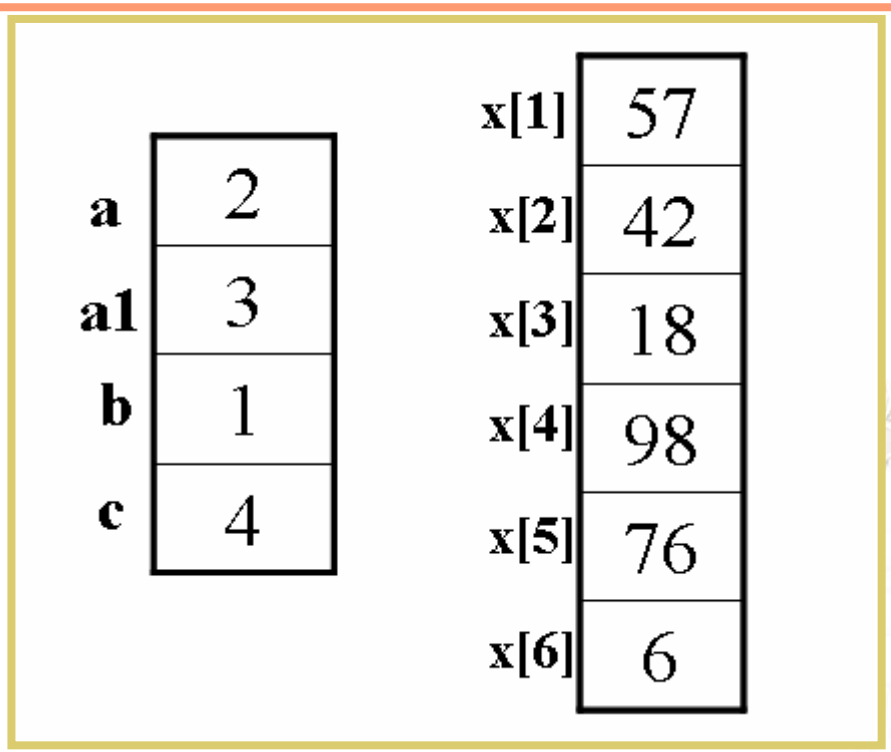
eg6: int a[ ]={0,1,2,3,4};

§2 一维数组

例1: 已知变量之值如图所示，写出以下变量之值。

x[a+a1]= 76 ; x[a*a1]= 6 ;

x[x[a+c]-a1]= 18 ; x[x[a1]-x[a+c]-a*c]= 98 。



§2 一维数组

例2:成绩分档统计： 0—9,10—19… 90—99以及100分的人数

```
main( )
{ int g,i,count[11]={0};
  for( ; ; )
    { scanf("%d",&g);
      if(g==-1) break;
      i=g/10;   count[i]++;  }     /*统计程序段*/
   for(i=0;i<=10;i++)          /*打印统计结果*/
    { if(i==10)  goto aa;
      printf("%2d---%-2d\t%d\n",10*i,10*i+9,count[i]);
      continue;
aa:  printf("-----%d\t%d\n",10*i,count[i]);
    }
}
```

§2 一维数组

例3:编程用“冒泡法”将10个整数由小到大排序

```
main( )
{ int i,j,t,a[11];
  printf(“Input 10 numbers:\n”);
  for(i=1;i<=10;i++)
    scanf(“%d”,&a[i]);
  printf(“\n”);
 for(i=1;i<=9;i++)
 for(j=1;j<=10-i;j++)
   if(a[j]>a[j+1])  {t=a[j]; a[j]=a[j+1]; a[j+1]=t;}
 printf(“The sorted numbers:\n”);
for(i=1;i<=10;i++)   printf(“%6d”,a[i]);
}
```

§2 一维数组

例4:编程用“选择法”将10个整数由小到大排序

```
main( )
{ int i,j,k,t,a[11];
  printf(“Input 10 numbers:\n”);
  for(i=1;i<=10;i++)  scanf(“%d”,&a[i]);
  for(i=1;i<=9;i++)
   { k=i;      /*假设第i轮中最小数是第i个数*/
     for(j=i+1;j<=10;j++)
       if(a[k]>a[j]) k=j;   /*选出第i轮中最小数所在位置 k*/
     if(i!=k) { t=a[k]; a[k]=a[i]; a[i]=t;}
   }
 printf(“\nThe sorted numbers:\n”);
 for(i=1;i<=10;i++)   printf(“%6d”,a[i]);  }
```

§2 一维数组

例5：10个数按序存放在数组中，今输入一个数num，使用”折半查找法”找出该数是数组中第几个数，否则输出“未找到”。

```
main( )
{ int bot,top,mid,num,flag,a[11]={0,1,3,5,6,8,23,35,47,59,68};
  scanf(“%d”,&num); flag=0;
  bot=1,top=10;
  while(bot<=top)
   { mid=(bot+top)/2;   /*折半*/
     if(a[mid]==num) { flag=1; break;}   /*已找到,位置是mid*/
     else if(a[mid]>num)  top=mid-1;   /*在前半区找*/
     else bot=mid+1; }  /*在后半区找*/
 if(flag) printf(“%d is found! Position is %d\n”,num,mid);
 else  printf(“%d is not found!\n”,num); }
```

§2 一维数组

例5:数组中有5个已排序整数,编程将数num插入数组适当位置

插入操作步骤

- 确定数num的插入位置loc
- 数据从loc开始依次后移
- 在loc位置处插入数num

§2 一维数组

例5:数组中有5个已排序整数,编程将数num插入数组适当位置

```
main( )
{ int loc,i,num,a[7]={0,1,3,7,9,12};
  printf("Input a number");
  scanf("%d",&num);
  for(i=1;i<=5;i++)  printf("%6d",a[i]);
  printf("\n");
  if(num>a[5])  a[6]=num;
  else  { loc=1;
          while(a[loc]<num) loc++;  /*确定插入位置loc*/
           for(i=5;i>=loc;i--)  a[i+1]=a[i];  /*挪出插入位置*/
           a[loc]=num;  /*插入数据*/
        }
  for(i=1;i<=6;i++)  printf("%6d",a[i]);
}
```

§2 一维数组

经典算法

- 数据排序(冒泡、选择)
- 数据检索(顺序查找、折半查找)
- 数据插入(定位、后移挪腾、插入)
- 数据删除(定位、前移)

§3 二维数组

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \end{pmatrix}$$

```
int  a[3][4];
```

§3 二维数组

一、定义

行数 列数

类型符 数组名[常量表达式1] [常量表达式2], … ;

类型符 数组名[页数] [行数][列数], … ;

类型符 数组名[size1] [size2] …[sizen], …;

二、引用

数组名 [行下标][列下标]

eg1: float b[5][10],b1[3][10];

eg2: float a[2][3][4];

eg3: 引用b[4][9]

正确

eg4:引用b1[2][3]

eg5: 引用b[5][10]

错误

eg6:引用b1[3][3]

§3 二维数组

三、说明

1.C中，二维数组以行为序占据连续主存单元

2.C中，数组名代表数组首址,即数组第0行起始地址

3.C中，a[m][n]定义相当于m个一维数组定义,一维数组名分别是:a[0]、a[1] …a[m-1]。长度均为n

4.a[m][n]中存在着m+1个地址表示法：

a —— 二维数组首址，即0行首地址；

a[0] —— 二维数组0行0列地址，a[0] ≡&a[0][0]；

a[1] —— 二维数组1行0列地址，a[1] ≡&a[1][0]；

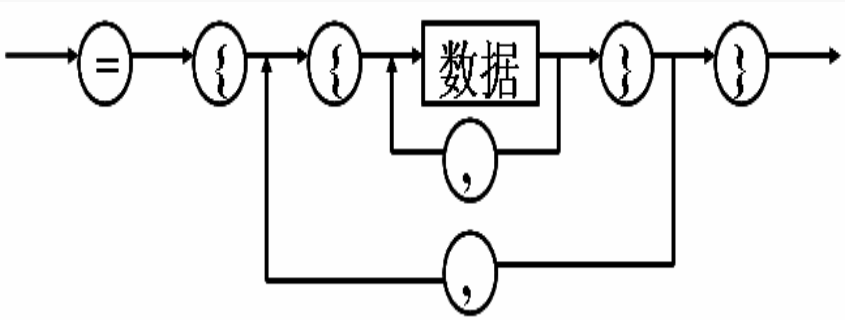
……

a[m-1] —— 二维数组m-1行0列地址，a[m-1] ≡&a[m-1][0]；

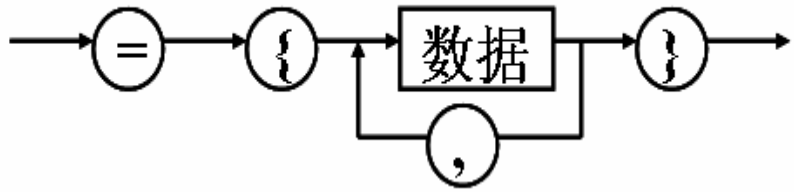
§3 二维数组

四、数组初始化

- 分行初始化



- 不分行初始化



eg1: int a[3][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};

eg1: int a[ ][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};

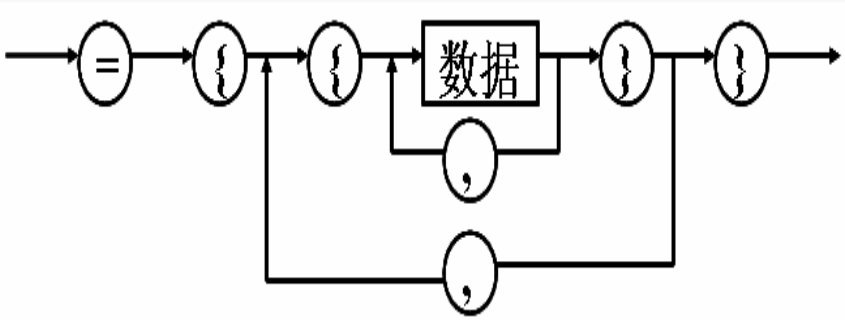
eg2: int a[4][4]={{1},{5},{9},{-11,11}};

eg2: int a[ ][4]={{1},{5},{9},{-11,11}};

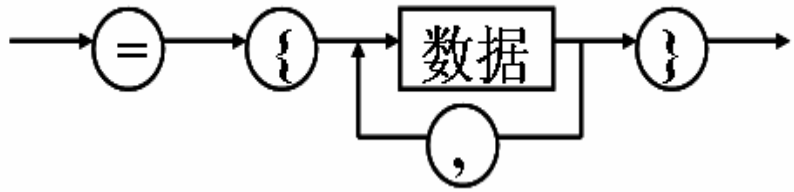
§3 二维数组

四、数组初始化

- 分行初始化



- 不分行初始化



eg3: int a[3][4]={{1},{ },{9}};

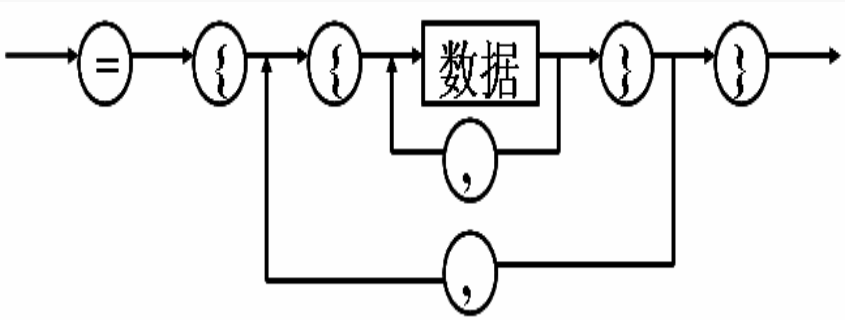
eg4: int a[3][4]={{1},{5,6},{9}};

eg5: int a[3][4]={{1},{5,6}};

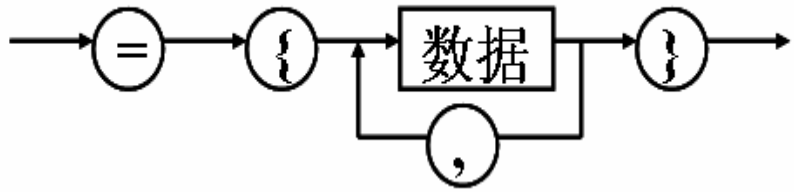
§3 二维数组

四、数组初始化

- 分行初始化



- 不分行初始化



eg6: int a[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};

eg6: int a[ ][4]={1,2,3,4,5,6,7,8,9,10,11,12};

eg7: int a[3][4]={1,2,3,4,5,6,7,8};

eg7: int a[3][4]={1,2,3,4,5,6,7,8,0,0,0,0};

§3 二维数组

例1:一个3×4矩阵如下,编程打印最大值元素及其所在位置

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 9 & 8 & 7 & 6 \\ -10 & 10 & -5 & 2 \end{pmatrix}$$

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

§3 二维数组

例2:以直角三角形形式打印杨辉三角形前10行

```
#include <stdio.h>
main( )
{  int y[11][11], i, j;
   for(i=1; i<=10; i++)  { y[i][1]=1; y[i][i]=1; }
   for(i=3; i<=10; i++)
   for(j=2; j<=i-1; j++)
     y[i][j]=y[i-1][j-1]+y[i-1][j];
   for(i=1; i<=10; i++)
     { for(j=1;j<=i; j++)  printf("%7d", y[i][j]); /*打印第i行*/
       printf("\n");
      }
 }
```

§3 二维数组

例3:编程将5阶方阵就地转置

a

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} & a_{04} \\ a_{10} & a_{11} & a_{12} & a_{13} & a_{14} \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} \\ a_{30} & a_{31} & a_{32} & a_{33} & a_{34} \\ a_{40} & a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

b

$$\begin{pmatrix} a_{00} & a_{10} & a_{20} & a_{30} & a_{40} \\ a_{01} & a_{11} & a_{21} & a_{31} & a_{41} \\ a_{02} & a_{12} & a_{22} & a_{32} & a_{42} \\ a_{03} & a_{13} & a_{23} & a_{33} & a_{43} \\ a_{04} & a_{14} & a_{24} & a_{34} & a_{44} \end{pmatrix}$$

- n阶方阵就地转置
  - i: 0 ~ n-2 , 1
  - j: i+1 ~ n-1 , 1
  - a[i][j]⇔a[j][i]

§3 二维数组

```
#include <stdio.h>
main( )
 {  int i, j;
    float a[5][5], t;
    for(i=0; i<5; i++)
    for(j=0; j<5; j++)
      scanf( "%f",&a[i][j] );
   printf("\n Matrix A\n");
   for( i=0; i<5; i++)
   { for(j=0; j<5; j++)
      printf("%16.2f",a[i][j]);
     printf("\n");
    }
```

```
for( i=0; i<=3; i++)
for(j=i+1; j<=4; j++)
 { t=a[i][j];
   a[i][j]=a[j][i];
   a[j][i]=t;
 } /*就地转置*/
printf("Transposed Matrix A\n");
for(i=0; i<5; i++)
{ for(j=0; j<5;j++)
     printf("%16.2f",a[i][j]);
   printf("\n");
 }
}
```

§3 二维数组

矩阵相关经典算法

- 矩阵相加、相减、*相乘
- 矩阵转置(异地、就地)
- 方阵主对角线、辅对角线的处理
  - 主对角线：所有满足i=j的a[i][j]
  - 辅对角线：所有满足i+j=n-1的a[i][j]
- 方阵上三角、下三角控制
  - 上三角元素：所有i<j的a[i][j]
  - 下三角元素：所有i>j的a[i][j]

§3 二维数组

思考题：编程生成如下方阵并将其输出出来

```
1  3  3  3  3
2  1  3  3  3
2  2  1  3  3
2  2  2  1  3
2  2  2  2  1
```

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

§4 字符数组

一、定义和作用

1.使用一维字符数组存放一个串

char 数组名[exp];

2.使用二维字符数组存放多个相关串

char 数组名[exp1][exp2];

eg1: char s[10];

s[0]='I',s[1]='␣',s[2]='a',s[3]='m',s[4]='␣';

s[5]='h',s[6]='a',c[7]='p',s[8]='p',s[9]='y';

存储状态

I		a	m		h	a	p	p	y

§4 字符数组

一、定义和作用

1.使用一维字符数组存放一个串

char 数组名[exp];

2.使用二维字符数组存放多个相关串

char 数组名[exp1][exp2];

eg2:

```
char s[4][9]={“BASIC”,”Computer”,”FORTRAN”,”Design”};
  ……
 printf(“%s”,s[1]);
```

s[0]	BASIC\0
s[1]	Computer\0
s[2]	FORTRAN\0
s[3]	Design\0

Computer

§4 字符数组

二、初始化

1. 用字符常量以数组元素为单位初始化

2. 用字符串对字符数组整体初始化

eg1: char s [10];

不等价

eg2:char s[10]={'T','u','r','b','o','␣','C'};

s[10]	T	u	r	b	o	␣	C	\0	\0	\0

eg3:char str[ ]={'C','h','i','n','a'};

不等价

eg4:char str[ ]={"China"};

§4 字符数组

二、初始化

1. 用字符常量以数组元素为单位初始化

2. 用字符串对字符数组整体初始化

1 eg5: char str[6]={“China”};

等价

2 eg5:char str[6]= “China” ;

等价

3 eg5:char str[ ]= “China” ;

§4 字符数组

二、初始化

1. 用字符常量以数组元素为单位初始化

2. 用字符串对字符数组整体初始化

eg6:阅读程序运行结果

```
main( )
{char s[4][9]={“BASIC”, “Computer”,
                “FORTRAN”,“Design”};
 int i;
 for(i=0;i<4;i++)
  printf(“\n%s”,s[i]);
}
```

s[0]	BASIC\0
s[1]	Computer\0
s[2]	FORTRAN\0
s[3]	Design\0

BASIC
Computer
FORTRAN
Design

§4 字符数组

三、串的输入/输出

1. 用getchar/putchar函数或%c格式说明一次I/O一个字符

eg1:
```
char i,s[5];
for(i=0;i<5;i++)
  s[i]=getchar( );
```

eg2:
```
char s[5]={'c','h','i','n','a'};
for(i=0;i<5;i++)
  putchar(s[i] );
```

eg1:
```
char i,s[5];
for(i=0;i<5;i++)
  scanf("%c",&s[i]);
```

eg2:
```
char s[5]={'c','h','i','n','a'};
for(i=0;i<5;i++)
  printf("%c",s[i]);
```

§4 字符数组

三、串的输入/输出

1. 用getchar/putchar函数或%c格式说明一次I/O一个字符
2. 使用%s格式说明一次性I/O整个字符串

- 输入/出串时,输入/出项都应书写数组名,表示串的起始地址
  输入/出数时,输入项书写变量的地址, 输出项书写变量名

```
eg3:
 char s[6];
 scanf("%s",s);
 printf("%s\n",s);
```

```
eg4: int i;
     scanf("%d",&i);
     printf("%d\n",i);
```

```
eg5: char c;
     scanf("%c",&c);
     printf("%c\n",c);
```

§4 字符数组

三、串的输入/输出

1. 用getchar/putchar函数或%c格式说明一次I/O一个字符

2. 使用%s格式说明一次性I/O整个字符串

- 输入/出串时,输入/出项都应书写数组名,表示串的起始地址
- 输入/出串时,输入项书写变量的地址,输出项书写变量的值

eg6: char s1[20]=”china\0Beijing”;
printf(“%s”,s1);

eg7:printf(“%s”,”china”);

eg8:c=‘a’;printf(“%c”,c);

eg9:i=3; printf(“%d”,i);

§4 字符数组

三、串的输入/输出

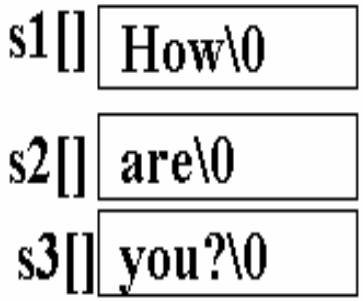
1. 用getchar/putchar函数或%c格式说明一次I/O一个字符

2. 使用%s格式说明一次性I/O整个字符串

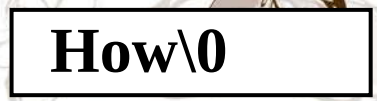
- 输入/出串时,输入/出项都应书写数组名,表示串的起始地址
- 输入/出数时,输入/出项书写变量的地址,输出项书写变量的值
- 用scanf输入串时,空白字符是串分隔符;空白字符也是数据分隔符

eg10:
```
char s1[5],s2[5],s3[5];
scanf("%s%s%s",s1,s2,s3);
```

How are you?<CR>



eg11:
```
char s[20];
scanf("%s",s);
```



§4 字符数组

四、字符串库函数

1. puts函数— #include <stdio.h>

- 首部描述： int puts(char *s);
- 调用格式： puts(字符指针s)
- 功能:输出s所指的串,并将\0转换为\n。输出成功返回换行符；失败返回EOF

eg1: puts(“china”);

eg1: printf(“%s\n”,”china”);

eg1: char str[ ]=“china”;
puts(str);

```
eg2:#include  <stdio.h>
    main( )
    { char s1[ ]=“abc”, s2[ ]=“def\nghi\n”, s3[ ]=“jkl”;
      puts(s1);  puts(s2);  puts(s3);}
```

§4 字符数组

四、字符串库函数

2. gets函数— #include <stdio.h>

- 首部描述： char *gets(char *s);
- 调用格式： gets(字符数组s)
- 功能:从标准输入终端接收以\n结束的串,将\n转换为\0后存入s,并返回指向串s的指针

```
eg1: char s[6];
     gets(s);
     puts(s);
```

```
eg1: char s[6],*p;
     p=gets(s);
     puts(p);
```

```
eg1: char s[6],*p;
     p=gets(s);
     printf("%s\n",s);
     printf("%s\n",p);
```

```
eg2: char s[13];
     gets(s); /*s串是How are you?*/
```

```
eg3: char s[13];
     scanf("%s",s);/*How */
```

输入: How are you?<回车>

§4 字符数组

四、字符串库函数

3.strcat函数— #include <string.h>

- 首部描述: char *strcat(char *str1,char *str2);
- 调用格式: strcat (字符数组,字符指针)
- 功能:将串1、2合并为串1,并返回串1地址

```
eg1: char  s1[15]="china␣",s2[ ]="beijing";
       printf("%s", strcat(s1,s2));
```

china␣ beijing

```
eg2: char  s1[15]="china␣";
       printf("%s", strcat(s1,"beijing"));
```

§4 字符数组

四、字符串库函数

4.strcpy函数— #include <string.h>

- 首部描述: char *strcpy(char *str1,char *str2);
- 调用格式: strcpy(字符数组, 字符指针)
- 功能:将串2拷贝至字符数组1中，并返回串1地址

```
eg1: char str1[10] =“Beijing”, str2[ ]=“china”;
     strcpy( str1,str2);
     puts(str1);  putchar(‘\40’); puts(str2);
```

china␣china

```
eg2: char str1[10] =“Beijing” ;
     strcpy( str1, “china”);
     puts(str1);  putchar(‘\40’); puts(“china”);
```

§4 字符数组

四、字符串库函数

5.strcmp函数— #include <string.h>

- 首部描述: int strcmp(char *str1,char *str2);
- 调用格式: strcmp(字符指针1,字符指针2)
- 功能:比较串1与串2的大小。若相等,返回0值；若不等,返回第一个不等字符的ASCII差值

```
eg1: char s1[6] , s2[ ]="china";
     strcpy(s1, s2);
     printf("%d",strcmp(s1, s2));
```

0

```
eg2: printf("%d",strcmp("china", "beijing"));
```

1

§4 字符数组

四、字符串库函数

5.strcmp函数— #include <string.h>

- 首部描述: int strcmp(char *str1,char *str2);
- 调用格式: strcmp(字符指针1,字符指针2)
- 功能:比较串1与串2的大小。若相等,返回0值；若不等,返回第一个不等字符的ASCII差值

eg3(考题)：判断两个字符串s1和s2是否相等，应当使用 D 。

A. if(s1==s2)　　B. if(s1=s2)

C. if(strcmp(s1, s2))　　D. if(!strcmp(s1, s2))

eg4(考题)：判断字符串s1是否大于字符串s2，应当使用 C 。

A. if(s1>s2)　　B. if(strcmp(s1,s2))

C. if(strcmp(s1, s2)>0)　　D. if(strcmp(s2, s1)>0)

§4 字符数组

四、字符串库函数

6.strlen函数— #include <string.h>

- 首部描述: int strlen(char *s);
- 调用格式: strlen(字符指针s)
- 功能:测试串s的实际长度并返回该值

```
eg1: char  str[10]="china";
     printf("%d", strlen(str));
```

5

```
eg2(考题):   main( )
              { printf("%d", strlen("ab\034\\\x89")); }
```

执行以上程序后,程序的输出是 5 。

```
eg3(考题):  main( )
             { printf("%d", strlen("abc\0defg")); }
```

执行以上程序后,程序的输出是 3 。

§4 字符数组

五、数值型和字符串型的异同

	数值型	字符串型
①数组初始化	只能逐个初始化	逐个或整体初始化
②格式I/O	地址/变量名	串地址/串地址
③引用值	变量名表示变量值	串名表示串地址
④赋值性	可以初始化 可以被赋值	可以初始化 不可被赋值 只能被strcpy
⑤数值比较	运用关系符 进行大小比较	不可运用关系符 只能用strcmp比较

§4 字符数组

例1: 编程模拟库函数strcat的功能。

- 首部描述: char *strcat(char *s1,char *s2);
- 功能:将串1、2合并为串1,并返回串1地址

```
char  *strcat(s1, s2)
char s1[ ],s2[ ];
{ int i=0, j=0;
 while( s1[ i ]!='\0')   i++;    /*将i移至串1的\0处*/
 while((s1[ i ]=s2[ j ])!='\0')
  { i++; j++;}       /* 将串2连至串1之后*/
   return(s1);    /* 返回串1的指针*/
 }
```

§4 字符数组

例2: 有3个长度均小于20的串,编程找出其中的最大串

```
#include  <stdio.h>
#include  <string.h>
main( )
{ char str[3][20], maxstr[20]; /* maxstr存放最大串*/
   int i;
   for(i=0; i<3; i++)  gets(str[i]);    /*输入3个串:str[0]~str[2]*/
   if( strcmp( str[0], str[1])>0)   strcpy(maxstr, str[0]);
   else  strcpy( maxstr, str[1]); /*maxstr串为串0,1中大者*/
   if(strcmp( str[2], maxstr)>0)  strcpy(maxstr, str[2]);
      /*maxstr串为串0,1,2中大者*/
   printf(" The largest string is:%s\n",maxstr);
}
```

§4 字符数组

例3: 输入一行字符,统计其中的单词数,单词之间用空格分隔

```
#include <stdio.h>
main( )
{  char c, str[81];
   int i, num=0, word=0;
   gets( str);
   for( i=0; (c=str[i])!='\0'; i++)
     if(c=='\40')  word=0;
     else if(word==0)  { word=1; num++; }
  printf(" There are %d words in the line!\n", num);
 }
```

§4 字符数组

例4:编写val(s),将数字字符串s转化为等价数值。允许串有前导数符+或-,遇到非数字字符结束处理,返回数串的对应数值

```
int val(s)
char s[ ];
{ int i=0, n, sign;
  sign=1;
  if( s[i]=='+'||s[i]=='-')  sign=(s[i++]=='+')?1:-1;/*量化数符*/
  for( n=0; s[i]>='0'&&s[i]<='9'; i++)
     n=10*n+s[i]-'0';   /*将数字串s转化为等价数值n*/
   return(sign*n);
}
```

§4 字符数组

例5: ①有大小不等10个整数，编程打印最大值。

②有大小不等的10个串（串长均小于20），编程打印最大串。

```
#include <stdio.h>
main( )
{  int a[11],max,i;
    for(i=1;i<=10;i++)   scanf("%d",&a[i]);
    max=a[1];
    for(i=2;i<=10;i++)
      if(a[i]>max)   max=a[i];
   printf("max=%d\n",max);
}
```

§4 字符数组

例5: ①有大小不等10个整数，编程打印最大值。

②有大小不等的10个串（串长均小于20），编程打印最大串。

```
#include <stdio.h>
#include <string.h>
main( )
{  int i;  char str[11][20],maxstr[20];
    for(i=1;i<=10;i++)   gets(str[i]);
    strcpy(maxstr,str[1]);
    for(i=2;i<=10;i++)
      if(strcmp(str[i],maxstr)>0)   strcpy(maxstr,str[i]);
   printf("The largest string:%s\n",maxstr);
}
```

§4 字符数组

例6: 有大小不等的10个串(串长<20),用“冒泡法”将它们排序

```
#include <stdio.h>
main( )
{ int i,j,t,a[11];
  printf(“Input 10 numbers:\n”);
  for(i=1;i<=10;i++)
   scanf(“%d”,&a[i]);
 for(i=1;i<=9;i++)
 for(j=1;j<=10-i;j++)
   if(a[j]>a[j+1])  {t=a[j]; a[j]=a[j+1]; a[j+1]=t;}
 printf(“The sorted numbers:\n”);
for(i=1;i<=10;i++)   printf(“%6d”,a[i]);
}
```

§4 字符数组

例6: 有大小不等的10个串(串长<20),用“冒泡法”将它们排序

```
#include <stdio.h>
#include <string.h>
main( )
{ int i,j;     char str[11][20],t[20];
  for(i=1;i<=10;i++)   gets(str[i]);
  printf(“\n Before sorted:\n”);
  for(i=1;i<=10;i++)   puts(s);
 for(i=1;i<=9;i++)
 for(j=1;j<=10-i;j++)
   if(strcmp(str[j],str[j+1])>0)
  {strcpy(t,str[j]);strcpy(str[j],str[j+1]);strcpy(str[j+1],t) ;}
 /*输出排序后的各个串*/ }
```

学习指导

数组技术

相关语法

- 一维数组定义、初始化及数组元素引用
- 二维数组定义、初始化及数组元素引用
- 字符数组定义、初始化、I/O方法及与数值型数据的差异
- 字符串库函数的正确运用

经典算法

- 最大/小、次大值求解（数值型、串）
- 排序算法（数值型、串）
- 检索算法（数值型、串）
- 插入与删除算法（数值型、串）
- 单词数统计、单词长度/行长计算
- 数串与数值的相互转换
- 矩阵常用运算（加、减、乘、转置等）
- 矩阵主、辅对角线控制,上三角、下三角控制
- 矩阵置值与输出形状问题
- 图案输出问题