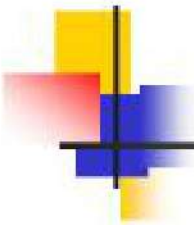


第6章 数组

学习目的和要求：

- 掌握一维数组的定义和使用；
- 掌握二维数组的定义和使用；
- 掌握字符数组的定义和使用；
- 能够熟练地将循环与数组结合起来进行批量数据处理。

一、一维数组

数组的引入

在学生成绩信息管理系统的设计中有许多批量数据处理的问题：向系统依次输入并存储100个学生的C语言成绩；存储完成后，输入学生的学号，显示该生的成绩。

有了前一章的基础，我们可以很快判断这里需要一个循环结构。

问题：该处提出了一个新的要求--存储问题

一、一维数组

问题1：要存储100个学生的成绩

```
int score1,score2,score3,……score100;
printf("请第1个学生的信息：");
scanf("%d",&score1);
……
printf("请第100个学生的信息：");
scanf("%d ",&score100);
```

定义100个变量显然不现实。

问题2：如何实现输入学号，显示该生的成绩

```
int num;
printf("请输入学生的学号： \n");
scanf("%d",&num);
switch(num)
{ case 1:printf("学号为1的成绩%d",score1);break;
  case 2:printf("学号为2的成绩%d",score2);break;
 ……
 case 100:printf("学号为100的成绩%d",score1);break;
default: printf("输入错误！ \n");
}
```

一、一维数组

1、数组的概念

- 数组：是有序数据的集合，数组中的每一个元素都属于同一个数据类型。
- 数组类型：构造类型
- 数组元素：基本类型
- 存放形式：同一数组的所有元素在存储器中占用一片连续的存储单元。

一、一维数组

2、数组的定义

1）一维数组的定义

格式

类型说明符　　数组名 [数组大小]

例：int　score [100]

数组名为 score，有100个整型数据元素：

score[0], score[1], score[2] …… score[99]

0～99：数组元素的下标

```
#include <stdio.h>
int main()
{ char a[5];
  int i;
  for(i=0;i<=4;i++)
     printf("%d       ",&a[i]);
  printf("\n%d",a);
}
```

一、一维数组

对一维数组定义的说明

- 数组名定名规则和变量名相同，但数组名不能与其他变量名相同。
- 数组大小必须是整型常量表达式，它表示元素的个数，即数组长度，但数组元素的下标是从0开始。

例：

```
main ( )
{       int  x;
        scanf("%d",&x);
        int  a [x];
        ……
}
```

×

```
#define  N  100
main ( )
{       int  a [ N ];
        int  b [ 20 ];
        ……
}
```

√

一、一维数组

3、数组的使用

1）一维数组元素的使用

格式

数组名[下标]

例： a[2]=a[0]+a[1]

a[5]=a[2*3]+a[7]

a[i]=a[i+1]+a[i+2]

一、一维数组

一维数组元素的使用说明

- 数组必须先定义，再引用。
- 数组元素只能逐个引用，不能一次引用整个数组。
- 下标引用范围从0开始，且不能超过数组的长度。
- 下标必须是整型（常量、表达式）。

一、一维数组

例1、向系统依次输入并存储100个学生的C语言成绩；存储完成后，输入学生的学号，显示该生的成绩。

解题思路：

1、存储设计—用数组实现批量数据的存储；学号和数组的下标对应；

2、循环实现批量数据的输入。

```
#include <stdio.h>
int main()
{ int score[100],i,num;
  printf("请依次输入100个学生的C语言课程成绩： \n");
  for(i=0;i<100;i++)
  {printf("请输入第%d个学生成绩： \n",i);
   scanf("%d",&score[i]); }
  printf("请输入查询的学号（0~99）:\n");
  scanf("%d",&num);
  if(num<0 || num>99)
     printf("输入有误!\n");
  else
     printf("学号为%d的学生成绩为%d",num,score[num]);
  return 0;
}
```

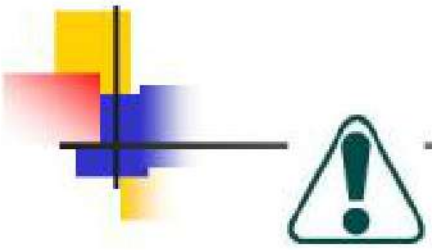
一、一维数组

例2 由键盘输入50个学生的C语言程序设计考试成绩，求出并输出平均成绩。

```
#include<stdio.h>
int main()
{    int i;
     float sum=0,ave,score[50];
     for(i=0;i<=49;i++)
     { scanf("%f",&score[i]);
       sum=sum+score[i];}
     ave=sum/50;
     printf("平均成绩是：%6.2f\n",ave);
}
```

例3:从键盘输入10个学生的C语言成绩到数组中，求平均分、最高分和最低分。

```
#include <stdio.h>
int main()
{  int i,sum=0,max=0,min=100,c[10];
   float ave;
   for(i=0;i<10;i++)
        scanf("%d",&c[i]);
   for(i=0;i<10;i++)
   {    sum=sum+c[i];
        if (c[i]>max)  max=c[i];
        if(c[i]<min)    min=c[i]; }
   ave=(float)sum/10;
   printf("平均分：%f，最高分：%d，最低分：%d\n",ave,max,min);
}
```



只能逐个引用数组元素，不能将把整个数组的所有元素作为一个整体一次调用。例如，以下调用方式是错误的：**scanf("%d",c);**

一、一维数组

4、数组的初始化

数组的初始化是指在定义数组的同时对数组元素赋予初值。各数组元素的初值放在花括号{}内，初值之间用逗号分隔。

①在定义数组时给数组元素赋初值：

```
int a[10]={ 3, 1, 5, 7, 4, 20, 31, 0, 2, 56 };
```

结果：a[0]=3, a[1]=1, a[2]=5.....a[9]=56

②在定义数组时可以只给一部分数组元素赋初值：

```
int a[10]={3， 1， 5， 7 };
```

结果：a[0]=3, a[1]=1, a[2]=5, a[3]=7, 其余元素为 0

一、一维数组

③使一个数组中全部元素值为0：

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

结果：a[0]=0, a[1]=0,……, a[9]=0

注意：不能给数组整体赋值。 a[ ]=0 ; ×

④在对全部数组元素赋初值时，可以不指定数组长度。

例：int a[5]={1,2,3,4,5};

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

0:00:00

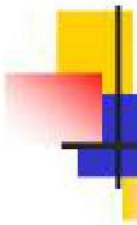
1. 对赋值语句“int a[10]={6,7,8,9,10};”的正确理解是( )

A) 将5个初值依次赋给a[1]至a[5]

B) 将5个初值依次赋给a[0]至a[4]

C) 将5个初值依次赋给a[6]至a[10]

D) 因为数组长度与初值的个数不相同，所以此语句不正确



以下不正确的定义语句是( )

A) double x[5]={4.0,4.0,4.0,4.0,4.0};

B) int y[5]={0,1,3,5,7,9};

C) char c1[]={'1','2','3','4','0'};

D) char c2[]={'\x18','\x3a','\x18'};



例**4**：编写一个仿骰子程序，将六面体掷**10000**次，统计各面出现的次数。

解题分析：对于骰子的六个点出现的次数纪录可以采用一维数组**dot[6]**分别纪录；每次投掷骰子到底出现哪一面，由随机数发生器**rand()**提供的随机数经过处理而决定。

```
#include <stdlib.h>
#include <stdio.h>
int main()
{ int dig,i,dot[6]={0};
 for(i=1;i<=10000;i++)
 { dig=rand()%6; /*rand()为库函数，其功能是随
  机产生0到32767间的整数*/
  dot[dig]++; }
 printf("\n digit  frequency\n");
for(i=0;i<=5;i++)
  if(i!=0) printf("%d       %d\n",i,dot[i]);
       else printf("6       %d\n",dot[i]);
return 0;}
```

一、一维数组

例5 求Fibonacci数列：1，1，2，3，5，8，...的前40个数，即

$F_1=1$ （n=1）

$F_2=1$ （n=2）

$F_n=F_{n-1}+F_{n-2}$ （n≥3）

算法：

定义一个一维数组存放Fibonacci数列，数组的长度为40，初始化前两个元素为1，1，后面的元素根据$F_n=F_{n-1}+F_{n-2}$计算。

一、一维数组

例5 程序

```
#include<stdio.h>
void main()
{     int f[40]={1,1},i,n=0;
      for(i=2;i<40;i++)
      {  f[i]=f[i-2]+f[i-1];      }
      for(i=0;i<40;i++)
      {  printf("%10d",f[i]);
         n++;
         if(n%5==0)
                  printf("\n");
      }
}
```

一、一维数组

例6：从键盘输入10个数，求这10个数的最大值。

算法：

定义一个长度为10的一维数组，利用循环结构为数组的各个元素输入值，再求这个数组中元素的最大值。

一、一维数组

例6 程序

```
#include<stdio.h>
void main()
{     int i;
      float a[10],max;
      for(i=0;i<=9;i++)
      {  scanf("%f",&a[i]);  }
      max=a[0];
      for(i=1;i<=9;i++)
      {  if(max<a[i])  max=a[i];     }
      printf("max=%f\n",max);
}
```

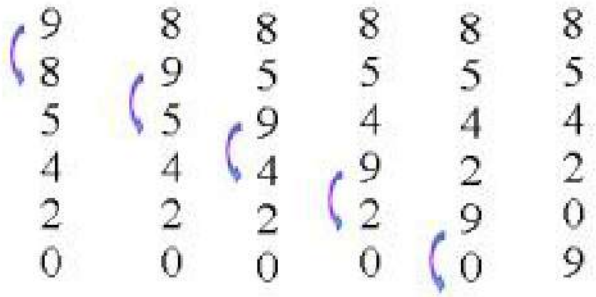
例7 用冒泡法对10个数从小到大排序。

排序是一种重要的算法，排序的方法很多，冒泡排序法是其中一种；基本思想如下：依次比较相邻的两个数，将小数放在前面，大数放在后面，第一趟结束时，最大的数被排到了最后；第二趟，采用相同的方法，一直比较到倒数第二个数（倒数第一的位置上已经是最大的），第二趟结束时，次大的数被排到了倒数第二的位置上；继续重复以上过程，直至完成排序。该方法之所以称为“冒泡排序”，是因为排序过程中总是大的数像石头一样“沉底”，小的数像水中气泡一样“上升”，每一趟都有一块当前最大的石头沉底。

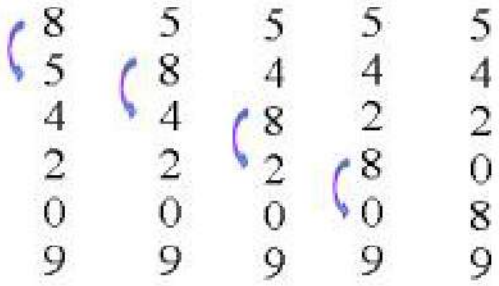
一、一维数组

例7 用起泡法对 10 个数从小到大排序。

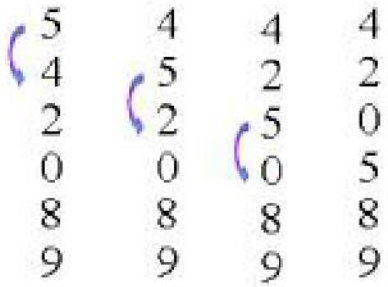
第一趟：



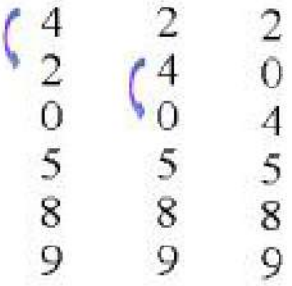
第二趟：



第三趟：



第四趟：



第五趟：

归纳法：

通过以上分析可以看出，如果有**n**个数，最多经过**n-1**趟完成排序；第**i**趟排序中进行两两比较的次数为**n-i**。这样，可以采用二重循环来实现冒泡排序。

外循环：控制排序的趟数，循环控制变量**i**的取值范围设为是**0<= i<n-1**；

内循环：控制每一趟两两比较的次数，由于数组的下标从**0**开始，为方便处理，将循环控制变量**j**的取值范围设为：**0<= j<n-i**。

一、一维数组

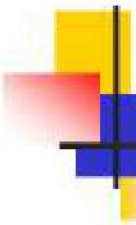
例7 程序

```
#include<stdio.h>
int main()
{    int a[10]={2,6,5,3,7,9,10,4,1,12};
     int i,j,t;
     for(i=1;i<=9;i++)
     {  for(j=1;j<=10-i;j++)
        {       if(a[j-1]>a[j])
                {       t=a[j-1];a[j-1]=a[j];a[j]=t;  }
        }
     }
     for(i=0;i<=9;i++)
     {  printf("%d ",a[i]);  }
}
```

第五章习题

已知**abc+cba=1333;**其中**a**、**b**、**c**均为一位数，编程求出满足条件的**a,b,c**的所有组合。

【解析】可以用三重循环穷举所有的解。



```
#include <stdio.h>
int main()
{  int a,b,c;
   for(a=0;a<=9;a++)
   {
   for(b=0;b<=9;b++)
        {
     for(c=0;c<=9;c++)
                {
        if((a*100+b*10+c)+(c*100+b*10+a)==1333)
                  printf("a=%d,b=%d,c=%d\n",a,b,c);
                }
        }
   }
}
```

输入一行字符串(以回车结束)，分别统计出其中中英文字母，空格，数字的个数。

```
……
{  char c;   int letter=0,space=0,digit=0,other=0;
do
   { scanf("%c",&c);
       if(c>='a'&&c<='z'||c>='A'&&c<='Z')   letter++;
              else if(c==' ')     space++;
                    else if(c>='0'&&c<='9')   digit++;
                          else    other++;
        }while(c!='\n');
printf("字母数：%d\n空格数：%d\n数字数:%d\n其他字符
   数%d\n",letter,space,digit,other-1);}
```

```
#include <stdio.h>
void main()
{  char c;
   int letters=0,space=0,digit=0,others=0;
   printf("please input some characters\n");
   while((c=getchar())!='\n')
   {
        if(c>='a'&&c<='z'||c>='A'&&c<='Z')  letters++;
                else if(c==' ')  space++;
                        else if(c>='0'&&c<='9') digit++;
                                else others++;
   }
printf("char=%d space=%d digit=%d others=%d\n",
   letters, space, digit, others);
}
```

用循环语句打印以下图形

AAAAAA
  BBBBBB
    CCCCCC
      DDDDDD
        EEEEEE

解题思路：1、外循环控制行数

2、内循环一打印空格；内循环二打印字

3、打印回车（属于外循环 or内循环？）

```
#include <stdio.h>
int main()
{int i,j;
 for(j=1;j<=5;j++)
 {
   for(i=1;i<j;i++)
       printf(" ");
   for(i=1;i<=6;i++)
       printf("%c",i+'A'-1);
   printf("\n");
 }
}
```

```
#include <stdio.h>
int main()
{char c='A';
 int i;
 for(;c<='E';c++)
 {
    for(i=1;i<=c-'A';i++)
         printf(" ");
    for(i=1;i<=6;i++)
         printf("%c",c);
    printf("\n");
 }
}
```

某场比赛评委会共有**10**人，评委会对每一个参赛人员打分，参赛人员最终得分的计算规则是:去掉一个最高分和一个最低分，然后计算平均分。试编写一个程序，从键盘输入每位评委的评分，计算出参赛人员的得分

解题思路：

1、需要定义一个浮点型数组用来记录每一个评委的评分；

2、通过循环程序将每个评委的评分输入；

3、通过循环程序比较获得最高分和最低分

4、通过循环程序将按规则计算出参赛人员的得分。

```
#include <stdio.h>
int main()
{
   float score[10],sum=0,min,max,ave;
   int i;
   printf("请输入评分:");
   for(i=0;i<10;i++)
        scanf("%f",&score[i]);
   min=max=score[0];
   for(i=0;i<10;i++)
   {     if(score[i]<min) min=score[i];
        if(score[i]>max) max=score[i];  }
   for(i=0;i<10;i++)
                sum=sum+score[i];
   ave=(sum-min-max)/8;
   printf("该选手的得分是：%.2lf\n",ave);  }
```

一、一维数组

例8 有一个已经排好序的数组，现在从键盘上输入一个数，将其按原来的排序规律插入到该数组中。

算法：

假设有一个数组a[6]={9,7,5,3,1};如果想把4从插入到数组a，需要把a[3]后面的数整体往后移动一位；在移动的时候要倒着移，假如有n个数，a[n+1]=a[n];一直移到a[3]。

一、一维数组

例8 有一个已经排好序的数组，现在从键盘上输入一个数，将其按原来的排序规律插入到该数组中。

9	7	5	3	1	0
a[0]	a[1]	a[2]	a[3]	a[4]	a[5]

n 

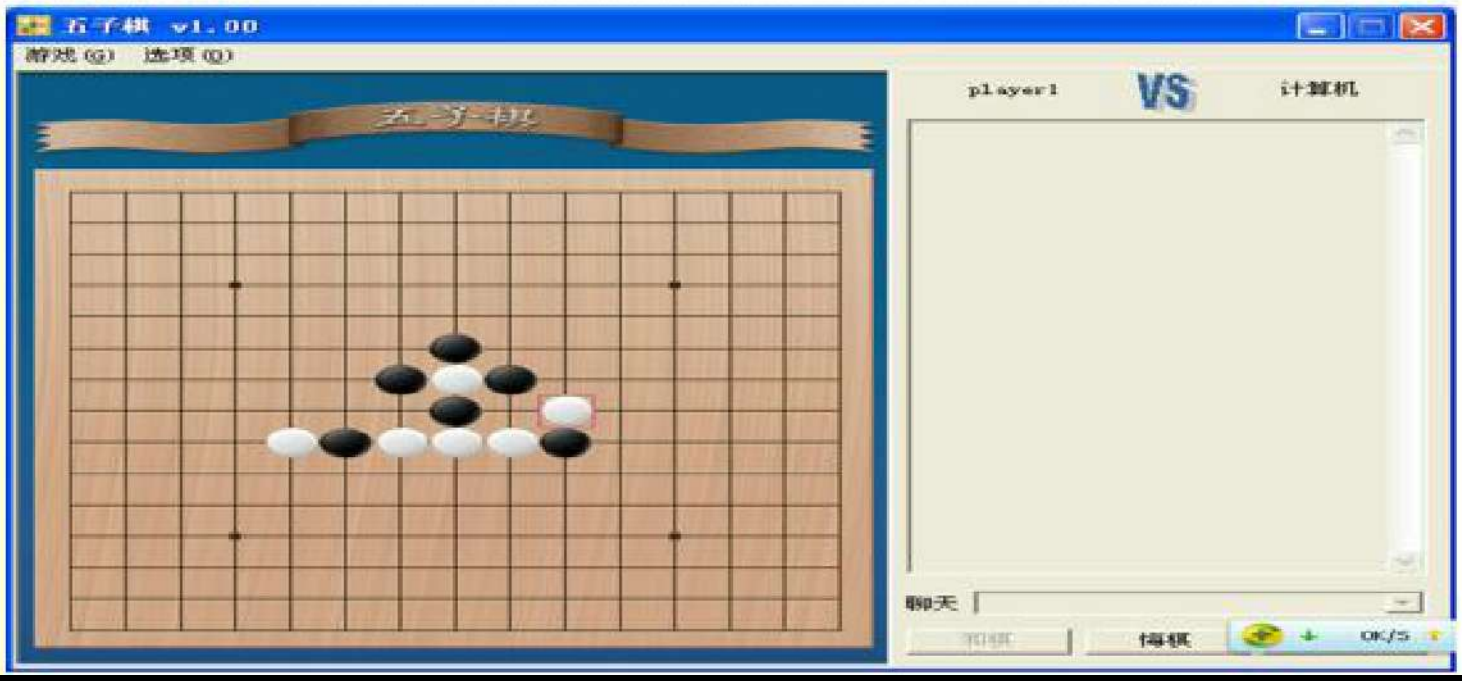
一、一维数组

例8 程序

```
#include<stdio.h>
void main()
{     int a[6]={6,5,4,3,1} ,n,position,i;
      scanf("%d",&n);
      for(i=0;i<=4;i++)
      {  if(n>a[i])       break;          }
      position=i;
      for(i=4;i>=position;i--)
      {  a[i+1]=a[i];   }
      a[position]=n;
      for(i=0;i<=5;i++)
      {  printf("%d ",a[i]);   }
}
```

二、二维数组

前面介绍了对于一维线性关系的批量数据存储的方式，但有的问题可能存在二维平面关系：方阵、矩阵等；下图所示的五子棋棋盘状态的存储等；这时就需要用到二维数组。



二、二维数组

1、二维数组的定义

二维数组顾名思义既是在一位数组的基础上增加了一维数组，其定义方法和一位数组相似；二维数组定义的一般形式为：

类型符 数组名[常量表达式1][常量表达式2]；

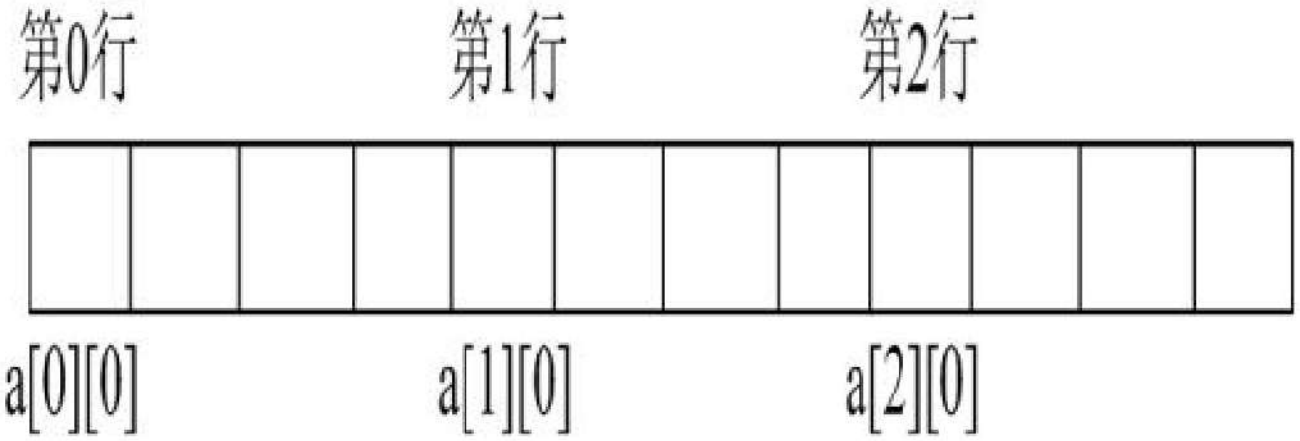
其中常量表达式1表示第一维下标的长度，通常也称为行标；常量表达式2也称为列标，表示第二维下标的长度。

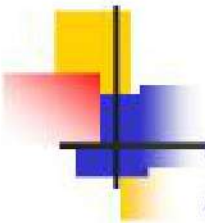
int a[3][4];

上例定义了一个三行四列的整型数组，数组名称为**a**,该数组共有**3*4**个元素构成，二维数组的行下标和列下标都是从**0**开始，因此，数组**a**的各元素是：

- **a[0][0],a[0][1],a[0][2],a[0][3]**
- **a[1][0],a[1][1],a[1][2],a[1][3]**
- **a[2][0],a[2][1],a[2][2],a[2][3]**

二维数组在内存中也是占有连续的存储空间，并且是按行排列的。

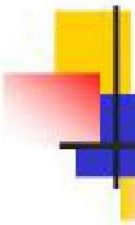




2、二维数组的引用

引用形式为：数组名[行下标] [列下标]

```
int a[3][4];
a[0][1]=3;    //直接给数组元素a[0][1]赋值
scanf("%d",&a[0][1]);  //从键盘输入数据给数组元素a[0][1]
printf("%d",a[0][1]);  //输出数组元素a[0][1]的值
```



例9：从键盘输入一个3×4矩阵，保存到一个二维数组a中，将数组的每个元素加1后，按行输出矩阵a。

```
#include <stdio.h>
int main(void)
{  int a[3][4],i,j;
   printf("输入数组各元素:");
   for(i=0;i<3;i++)
       for(j=0;j<4;j++)
       {       scanf("%d",&a[i][j]);
               a[i][j]=a[i][j]+1;}
   for(i=0;i<3;i++)
   {
       for(j=0;j<4;j++)
               printf("%6d",a[i][j]);
       printf("\n");
   }
}
```

3、二维数组的初始化

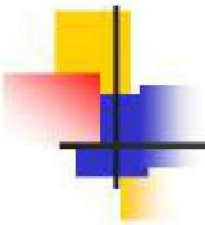
1） 分行赋初值

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

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

```
int b[3][3]={{1,2,3},{},{7,8}};
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \\ 7 & 8 & 0 \end{bmatrix}$$



2） 顺序赋初值

顺序赋初值将所有数据写在一个花括号内，按数组元素在内存中的排列顺序依次对各元素赋初值。

- **int a[3][3]={1,2,3,4,5,6,7,8,9};**等价于
- **int a[3][3]={{1,2,3},{4,5,6},{7,8,9}};**

顺序方式不直观，容易出错。

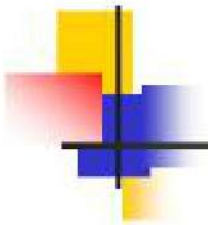
和一维数组一样，如果对全部二维数组元素都赋了初值，或分行初始化时在初值表中列出了全部行，则可以省略二维数组的行标。例如：

int a[3][3]={1,2,3,4,5,6,7,8,9};等价于

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

int b[3][3]={{1,2,3},{},{7,8}};等价于

int b[][3]={{1,2,3},{},{7,8}};



定义int a[][3]={ 4,5,6,7,8,91,2,3};则a数组第一维的大小是( )

A) 2　　B) 3　　C) 4　　D) 无确定值

0:00:00

int a[][2]={0,0,0,0,0};则下面不正确的叙述是( )

- **A)** 数组**a**的每个元素都可得到初值**0**
- **B)** 二维数组**a**的第一维大小为**3**
- **C)** 因为二维数组**a**中第二维大小的值除经初值个数的商为**2.5,**故数组**a**的行数为**3**
- **D)** 只有元素**a[0][0]**和**a[0][1]**可得到初值**0**，其余元素均得不到初值**0**

以下能正确定义二维数组的是( )

A) int a[][3];

B) int a[][3]= {2*3};

C) int a[][3]={};

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

以下不能正确定义二维数组的选项是( )

A) int a[2][2]={{1},{2}};

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

C) int a[2][2]={{1},{2,3}};

D) int a[2][]={{1,2},{3,4}}

以下能正确定义数组并正确赋初值的语句是( )

A) int N=5,b[N][N];

B) int a[1][2]={{1},{3}};

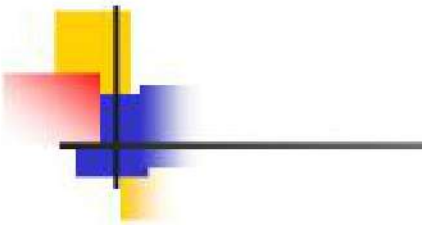
C) int c[2][]={{1,2},{3,4}};

D) int d[3][2]={{1,2},{34}};

例10：从键盘输入一个3×4矩阵，保存到一个二维数组a中，求该矩阵的转置矩阵b。

$$a=\begin{bmatrix}1 & 2 & 3 & 11\\ 4 & 5 & 6 & 12\\ 7 & 8 & 9 & 13\end{bmatrix}$$

$$b=\begin{bmatrix}1 & 4 & 7\\ 2 & 5 & 8\\ 3 & 6 & 9\\ 11 & 12 & 13\end{bmatrix}$$



- 这样3×4矩阵a的转置矩阵b为4×3矩阵。矩阵a中的元素a[1][2](即6)对应矩阵b中的元素b[2][1]，同理，矩阵a中的元素a[i][j]对应矩阵b中的元素b[j][i]。

```
#include <stdio.h>
int main()
{   int a[3][4],b[4][3],i,j;
    printf("输入数组各元素:");
    for(i=0;i<3;i++) //输入矩阵a
        for(j=0;j<4;j++)
            scanf("%d",&a[i][j]);
    for(i=0;i<3;i++)  //矩阵转置
        for(j=0;j<4;j++)
            b[j][i]=a[i][j];
    for(i=0;i<4;i++)  //输出矩阵b
    {   for(j=0;j<3;j++)
          printf("%6d",b[i][j]);
       printf("\n");
    }}
```

例11有一个3×4的矩阵，求该矩阵中最大的元素值以及最大元素所在的行号和列号。

```
int main()
{  int i,j,max,row,col;
   int a[3][4]={{1,2,3,4},{9,8,7,6},{2,100,-
   5,200}};
   max=a[0][0]; // max变量初始化
   row=col=0;
   for(i=0;i<3;i++)
        for(j=0;j<4;j++)
             if(max<a[i][j])
             {      max=a[i][j];
                    row=i;
                    col=j;}
printf("最大元素：%d，行号：%d，列号：%d\n",max,row,col);
}
```

例12：设某班级每位同学有三门课程的考试成绩，试编写程序实现对每个同学进行成绩的录入并计算平均成绩。

解题思路：

1、对于学生成绩的存储可以采用二维数组grade[N][4],其中grade[i][0]、grade[i][1]、grade[i][2]依次存放第i位同学的三门课程单科成绩，grade[i][3]存放第i位同学的平均成绩；

2、对于同学单科成绩的录入可以用二重循环来实现。

```
int main()
{ int grade[100][4]={0},i,j,n;
do
{printf("please input the number of ");
   scanf("%d",&n);
   }while(n>100);   //本循环控制学生人数
   printf("please input the individual grade:\n");
   for(i=0;i<n;i++)
   { printf("student number %d",i);
      for(j=0;j<3;j++)        //逐一输入第i个学生成绩
        {printf("\n class %d",j);
              scanf("%d",&grade[i][j]);}
   grade[i][3]=(grade[i][0]+grade[i][1]+grade[i][2])/3;
        }
   for(i=0;i<n;i++)
printf("\n the %d student average is %d",i,grade[i][3]);
}
```

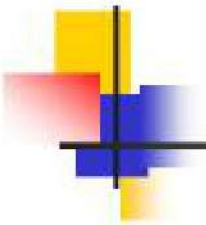
课后习题：

1、输入**10**个整数，求其中正数的个数及平均值，并输出结果。（平均值精确到小数点后两位）。

2、定义一个**10**×**10**的矩阵，要求该矩阵主对角线上的元素均**1**，其余均为**0**，然后输出该矩阵。

3、设某班级每位同学有十门课程的考试成绩，试编写程序实现对每个同学进行成绩的录入并计算平均成绩。

4、键盘上输入10×10个实数放到10行10列的二维数组中,找出该二维数组的最大值和最小值，输出二值并输出其行、列标 。



5、从键盘上给**5**×**6**数组赋值，求它的转置矩阵，并在屏幕上显示出来。

三、字符数组

1、字符数组的定义

C语言中没有字符串类型的变量，学生信息信息管理系统中的学号、姓名等都是字符串数据；如何对字符串变量来保存、处理？

字符数组：存放字符型数据的数组，字符数组中的每个元素都是一个字符型变量，一个元素存放一个字符。

char 数组名[数组长度];

三、字符数组

2、字符数组的引用

引用字符数组元素也是用下标，且下标也是从**0**开始。例如：

```
char a[5];
a[0]= 'C'; //直接给数组元素a[0]赋值
scanf("%c",&a[0]);  //输入数据给数组元素a[0]
printf("%c",a[0]);  //输出数组元素a[0]的值
```

三、字符数组

3、字符数组的初始化

1） 用字符常量初始化

```
char a[8]={ 'C', 'h' , 'i' , 'n' , 'a'};
a[0]= 'C',a[1]= 'h',a[2]= 'i',
a[3]= 'n',a[4]= 'a';
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
C	h	i	n	a	\0	\0	\0

三、字符数组

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

等价于

char a[5]={‘C’, ‘h’ ,’ i’ ,’n’ ,’a’};

2）用字符串常量初始化

char a[8]= {"China"};

或者省略花括号，直接写成

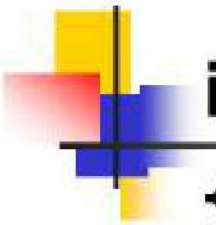
char a[8]= "China";

数组的下标也可以省略。例如，

char a[]= "China";

由于字符串常量的末尾由系统自动加上字符串结束标志'**\0**',因此，数组**a**的长度为**6**，上面的语句等价于：

char a[]={ 'C', 'h' , 'i' , 'n' , 'a' , '\0'};



```
int main()
{
   char s[]={"happy"};
    char ss[]={'h','a','p','p','y'};
   printf("%d",sizeof(s));
   printf("%d",sizeof(ss));
}
```

同数值型数组一样，也可以定义和初始化二维字符数组。例如，

char str[3][5]={ "C","math","VC"}

二维数组**str**有**3**行**5**列，每一行相当于一个一维字符数组，因此，二维数组**str**由三个一维字符数组组成，其数组名为**str[0]**、**str[1]**和**str[2]**，每个一维数组的长度为**5**。

以下选项中，不能正确赋值的是( )

A) char s1[10];s1="Ctest";

B) char s2[]={'C', 't', 'e', 's', 't'};

C) char s3[20]="Ctest";

D) char s4[]="Ctest\n";

以下不能正确进行字符串赋初值的语句是( )

A) char str[5]="good!";

B) char str[]="good!";

C) char str[10]="good!";

D) char str[5]={'g','o','o','d'};

有以下程序，程序运行结果是( )

```
#include <stdio.h>
main()
{
char a[10]=”abcd”;
printf(“%d”, sizeof(a));
}
```

三、字符数组

4、字符串输入输出函数

方法一：

逐个字符的输入输出：用‘ %c ’输入或输出一个字符

例：逐字符输入/输出字符数组c[10]

```
for (i=0;i<10;i++)
        scanf("%c",  &c[i]);
for (i=0;i<10;i++)
        printf("%c", c[i]);
```

三、字符数组

方法二：

将整个字符串一次输入或输出：用‘%s’格式符输出字符串。

例：
```
char c[ ]={“china”};
char  a[10];
printf(“%s”, c);
scanf(“%s”, a);
printf(“%s”, a);
```

printf("%s", c);

整串字符的输出说明

① 输出字符不包括结束符'\0';

② '%s'格式输出字符串时，printf函数中的输出项是字符数组名，而不是数组元素名;

③ 若数组长度大于字符串实际长度，也只输出到遇'\0'结束;

④ 若一个字符数组中包含一个以上'\0'，则遇第一个'\0'时结束输出

scanf(“%s”,c);

整串字符的输入：

① **scanf**函数中的输入项是字符数组名，由于数组名代表数组的起始地址，所以不要再加**&**。

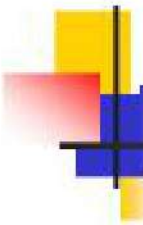
② 从键盘输入一个字符串，以回车结束，系统会自动在后面加一个**‘\0’**结束符。

③ 若利用一个**scanf**输入多个字符串，则以空格分隔。

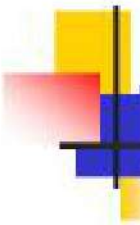
例：
```
char str1[5],str2[5],str3[5];
scanf(“%s%s%s”,str1,str2,str3);
```
输入：**How are you?**

例13：从键盘输入一个字符串，输出并求该字符串的长度。

```
#include <stdio.h>
int main()
{ char a[20];
  int i=0;
  scanf("%s",a);
  while(a[i]!='\0')
      i++;
  printf("%s的长度为：%d\n",a,i);
}
```



```
#include <stdio.h>
int main()
{ char a[20];
  int i=0;
  scanf("%s",a);
  while(a[i])
      i++;
  printf("%s的长度为： %d\n",a,i);
}
```



```
#include <stdio.h>
int main()
{ char a[5]={"abcde"};
  int i=0;
  while(a[i]!='\0')
      i++;
  printf("%s的长度为：%d\n",a,i);
}
```

三、字符数组

方法三：

使用**scanf**函数输入多个字符串时有一个问题，就是说以空格为一个字符串的结束，这样，字符串中不能包含空格字符。

gets(一维字符数组名);

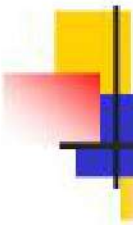
puts(一维字符数组名或字符串常量);

例13：从键盘输入一个字符串，输出并求该字符串的长度

```
#include <stdio.h>
int main()
{ char a[20];
  int i=0;
  gets(a);
  while(a[i]!='\0')
      i++;
  printf("%s的长度为：%d\n",a,i);
}
```

例14：编写一个数据加密程序，从键盘输入一个由字母、数字、空格组成的字符串，输出其所对应的密文。加密规则是：将字符串中的每个字符的ASCII码值加100。

编程思路：先从键盘输入一个字符串保存到字符数组中，然后，从数组的第1个元素开始，利用循环语句，在循环体中将每个数组元素的值加100。



```
#include <stdio.h>
int main()
{ char a[20];
  int i=0;
  gets(a);
  while(a[i])
  {  a[i]=a[i]+100;
     i++;}
  printf("密文为：%s\n",a);
}
```

例15：从键盘输入一个字符串，将其中的大写字符变成小写字母，其它字符不变。

```
#include <stdio.h>
int main()
{ char a[20];
  int i=0;
  gets(a);
  while(a[i])
  {  if(a[i]>='A' && a[i]<='Z')
          a[i]=a[i]+32;
     i++;}
  printf("转换结果为：%s\n",a);
}
```

5、字符串处理函数

C语言具有丰富的字符串处理函数；当使用这些库函数时需要包含头文件 **string.h**。

使用格式	函数功能
strlen(str)	求字符串 str 的长度，返回不包括串结束符'\0'在内的字符个数。
strcat(str1, str2)	将字符串 str2 连接到字符数组 str1 的后面，结果放在字符数组 str1 中。
strcmp(str1, str2)	比较两个字符串的大小。 如果 str1>str2，则返回正数； 如果 str1==str2，则返回 0； 如果 str11< str12，则返回负数。
strcpy(str1, str2)	将字符串 str12 复制到字符数组 str1 中。

例15：将键盘输入的字符串按逆序输出。

编程思路：先从键盘输入一个字符串保存、

到字符数组中，然后，利用循环语句，从字符串末尾向字符串首部方向依次输出每个字符。

```
#include <stdio.h>
#include <string.h>
int main()
{
  char a[20];
  int i;
  gets(a);
  for(i=strlen(a)-1;i>=0;i--)
      printf("%c",a[i]);
}
```

分析下面程序的执行结果。

```
#include <stdio.h>
#include <string.h>
int main()
{
  char str1[30]="abc";
  char str2[]="123";
  printf("%s\n",strcat(str1,str2));
}
```

分析下面程序的执行结果。

```
#include <stdio.h>
#include <string.h>
int main()
{ char str1[30];
  char str2[]="123";
  strcpy(str1,str2);
  printf("%s\n",str1);
  printf("%d\n",strlen(str1));
  printf("%d\n",strcmp(str1,str2));}
```

作业：

1、从键盘输入一个不超过20个字符的字符串，存储到一个字符数组中。编程查找在这一字符数组中有没有字符'a'，若有，则输出第一个'a'在该串字符中的位置，若没有，则输出"not found!"

2、编写程序实现对于字符数组str中除了下标为奇数、同时ASCII值也为奇数的字符外，其余的全部删除；str中剩余的字符所形成的一个新串放在字符数组s中并输出s。

3、编写程序实现对于字符数组ss输入字符串；然后求出ss字符串中指定字符的个数,并输出此值。

4、从键盘输入一个字符串，找出其中ASCII码值最大的字符并输出该字符及其ASCII码值。

5、编写程序实现对于字符数组**str**输入字符串；然后删除符数组**str**中一个指定下标的字符并将新字符转存至**result**数组中。例如输入一个字符串”**hello world**”，然后输入**4**；则**result**数组的字符串为”**hell world**”。