第七章 数组

郎大鹏

第七章 数组

第七章 数组

百度笔试

- 有一个监控系统，会监控访问的url和ip还有时间。需求是可以存储1000亿条信息。

  1. 可以按照url查询某个时间段内的访问量。

  2. 可以按照ip查询某个时间段内的访问量。

  请设计一个系统？

7.1　成组数据处理问题实例及解决方法

日常中经常遇到大量的成组数据问题。如：需要对某班某门课的成绩进行排序，找出不及格的学生，统计各分数段的人数，找出最高分、最低分及其它分析计算工作。看下面的例题：

例7.1　20100611班有10人，“大学计算机基础”课程的期末考试成绩为：90，80，95，56，65，47，93，82，75，61。求最高分。

7.1　成组数据处理问题实例及解决方法

求解方案：

（1）先设第1个人的成绩为最高，将其值存入到变量max中。

（2）用第2个人的成绩与max比较，若大于max的值，则用该值更新max。

（3）再用下一人的成绩与max比较，若大于max的值，则用该值更新max。

（4）以此方法类推，直到所有人的成绩都处理完毕。

（5）此时max的值即为最高分。

分析上述方案，第（1）至第（4）步有重复的部分，算法实现时可考虑使用循环来完成。比较的部分可以用分支来实现。

7.1　成组数据处理问题实例及解决方法

例7.2　在例7.1的基础上，求完最高分后再求20100611班“大学计算机基础”不及格的人数。

求解方案：

（1）用变量count表示不及格人数，并将count清0。

（2）用第1个人的成绩与60作比较。若小于60，则执行count=count+1。

（3）再用下一个人的成绩与60作比较。若小于60，则执行count=count+1。

（4）以此方法类推，直到所有人的成绩都处理完毕。

（5）此时count的值即为不及格人数。

分析上述方案，第（2）至第（4）步有重复的部分，算法实现时可考虑使用循环来完成。比较的部分可以用分支来实现。

第七章 数组

7.2 一维数组的定义与引用

7.2.1 一维数组的定义

定义一维数组的语法格式：类型说明符 数组名[常量表达式];

说明：

① 类型说明符指定数组中每个元素的数据类型。

② 数组名的命名规则与变量名的命名规则相同。

③ 常量表达式指定数组包含元素的个数，即数组长度。

④ 数组元素的下标从0开始排序。

⑤ 数组名代表数组首元素的存储地址。

例如：

```
int a[5];
/*定义整形数组a，共有5个元素：a[0]，a[1]，a[2]，a[3]，a[4]*/
```

7.2　一维数组的定义与引用

7.2.2　一维数组元素的引用

数组元素的引用格式：数组名[下标]

说明：

1、对数组只能引用数组元素；

2、下标可以是整型常量或整型表达式；

3、数组元素可当作变量编程。

7.2　一维数组的定义与引用

7.2.2　一维数组元素的引用

例7.3　用有10个元素的数组代表前10个自然数并输出。输出时每个数的宽度为4。

```
# include <stdio.h>
void main()
{int a[10],i;                    /*定义有10个元素的整型数组a用来代表自
                                   然数*/
                                 /*变量i作数组的下标*/
 for(i=0;i<10;i++)               /*为数组元素赋值*/
   a[i]=i+1;                     /*注意元素下标与元素值的对照关系：*/
 for(i=0;i<10;i++)               /*输出每个元素的值*/
   printf("%4d",a[i]);           /*用%4d指定输出的宽度为4*/
}
```

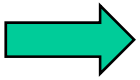
7.2　一维数组的定义与引用

7.2.3　一维数组的初始化

定义数组时可以对数组元素进行初始化。

(1) 对全部元素赋初值。如：

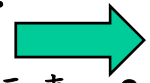
```
int a[3]={1,2,3};
/*初值个数与数组元素的个数相等*/
```



```
int a[3];
a[0]=1;a[1]=2;a[2]=3;
```

(2) 给数组前面的部分元素赋初值。例如：

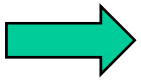
```
int a[5]={1,2,3};
/*初值个数与数组元素的个数不等，5个元素，3个初值*/
```



```
int a[5];
a[0]=1;a[1]=2;a[2]=3;
/*前3个元素被赋指定初值*/
a[3]=0;a[4]=0;
/*后2个元素初值为0*/
```

(3) 在定义的同时给全部元素赋初值这种情况下，可以省略数组长度。例如：

```
int a[3]={1,2,3};
/*初值个数与数组元素的个数相等*/
```



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

7.2　一维数组的定义与引用

7.2.3　一维数组的初始化

例7.4　求斐波那契数列的前20项，存入数组中并输出，每行5个数，每个数占8列。

分析：

从题目的要求上粗略看来，与前面章节中用循环的方式求斐波那契数列没有太大的差别。编程时定义一个存放数列的数组。编写2个循环，用第1个循环计算数列每一项的值病存入数组，用第2个循环输出。

7.2 一维数组的定义与引用

7.2.3 一维数组的初始化

```
# include <stdio.h>
void main()
{int a[20]={1,1},i;                    /*定义整型数组存储斐波那契数列*/
                                /*前2个数不用计算，在数组定义时直接给出*/
 clrscr();                             /*此函数是库函数，功能是清屏*/
 for(i=2;i<20;i++)                     /*从第3个数开始计算每一个数*/
   a[i]=a[i-1]+a[i-2];
 for(i=0;i<20;i++)                     /*输出每个元素的值*/
   {if(i%5==0)                  /*控制5个数一行*/
      printf("\n");
    printf("%8d",a[i]);
   }
}
```

第七章 数组

7.3　二维数组的定义与引用

7.3.1　二维数组的定义

定义二维数组的语法格式：

类型说明符 数组名[常量表达式1][常量表达式2]；

说明：

① 常量表达式1指定第一维的长度，常量表达式2指定第二维的长度。

② 二维数组元素在内存中占用连续的存储空间，按行存放，即先存第1行的各元素再存第2行的各元素。

③ 可以从两个角度去看二维数组：一是二维数组由元素组成，另一是二维数组由行组成。按第二种观点二维数组是一种特殊的一维数组，其每个元素又是一个一维数组。

7.3　二维数组的定义与引用

7.3.2　二维数组元素的引用

二维数组元素的引用格式：

数组名[下标1][下标2]

例：7.5　定义一个2行3列的二维数组，第1行的值为1、2、3，第2行的值为4、5、6，并按矩阵的方式输出。

程序由两个部分组成。第一部分构造数组。由于各元素的值由规律，可由行列号计算出来：元素值=行号*3+列号+1。用一个双重循环遍历数组元素，外层循环变量i代表行号，内层循环变量j代表列号。第二部分输出数组。同样用双重循环完成。外层循环每执行一次输出一个回车，就会实现矩阵的效果。

7.3 二维数组的定义与引用

7.3.2 二维数组元素的引用

```
# include <stdio.h>
void main()
{int a[2][3],i,j;              /*定义2×3的数组，i代表行号，j代表列号*/
 clrscr();
 for(i=0;i<2;i++)                  /*构造数组元素的值*/
   for(j=0;j<3;j++)
     a[i][j]=i*3+j+1;
 for(i=0;i<2;i++)                  /*输出数组*/
   {for(j=0;j<3;j++)
      printf("%4d",a[i][j]);
      printf("\n");                /*每行元素输出完成后输出一个回车*/
   }
}
```

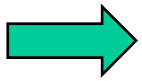
7.3　二维数组的定义与引用

7.3.3　二维数组的初始化

定义二维数组的同时可进行初始化。

(1) 可分行对二维数组赋初值。例如：

```
int a[2][3]={{1,2,3},{4,5,6}};
/*外层花括号包含2套子花括号，*/
/*第1套子花括号代表第1行的值，*/
/*第2套花括号代表第2行的值*/
```



```
int a[2][3];
a[0][0]=1;a[0][1]=2;a[0][2]=3;
a[1][0]=4;a[1][1]=5;a[1][2]=6;
```

(2) 可将所有数据写在一套花括号中，按顺序给各元素赋值。例如：

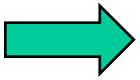
```
int a[2][3]={1,2,3,4,5,6};
/*与前一种方法等价*/
```

7.3　二维数组的定义与引用

7.3.3　二维数组的初始化

(3) 可对部分元素赋初值。例如：

```
int a[3][3]={{1}, {}, {4,5}};
/*中间的空的1套子花括号，/*
/*表示对应行无值*/
```

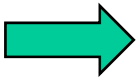


```
int a[3][3];
a[0][0]=1;a[2][0]=4;a[2][1]=5;
```

未被赋值的元素初值为0。

(4) 在定义的同时给全部元素赋初值这种情况下，可以省略数组第一维的长度。例如：

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



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

7.3　二维数组的定义与引用

7.3.3　二维数组的初始化

例7.6　数组a的值为 $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$，将其转置后存于数组b中，使b的值为 $\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$。

转置时a中第i行第j列的元素在b中变为第j行第i列，即进行了行列互换。写程序时注意行列的对应关系。

7.3 二维数组的定义与引用

7.3.3 二维数组的初始化

```
# include <stdio.h>
void main()
{int a[2][3]={1,2,3,4,5,6},b[3][2],i,j;
                                /*定义a，b两个数组*/
 clrscr();
 for(i=0;i<2;i++)               /*i代表a中的行号*/
   for(j=0;j<3;j++)             /*j代表a中的列号*/
     b[j][i]=a[i][j];           /*a中的i行j列变为b中的j行i列*/
 for(i=0;i<3;i++)               /*输出转换后的结果*/
   {for(j=0;j<2;j++)
      printf("%4d",b[i][j]);
      printf("\n");             /*每行后输出回车换行*/
   }
}
```

7.3　二维数组的定义与引用

7.3.3　二维数组的初始化

例7.7　将矩阵a进行自身转置，转换前a的值为 $\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$，转换后a的值为 $\begin{bmatrix}1&4&7\\2&5&8\\3&6&9\end{bmatrix}$。

自身转置时，是以主对角线为轴，互换对称元素。写程序时，以下三角元素为主进行遍历。

7.3 二维数组的定义与引用

7.3.3 二维数组的初始化

```
# include <stdio.h>
void main()
{int a[3][3]={1,2,3,4,5,6,7,8,9},i,j,x;
/*x用作互换的中间变量*/
 clrscr();
 for(i=0;i<3;i++)                    /*以下三角为主进行遍历*/
   for(j=0;j<i;j++)
     {x=a[i][j];                     /*互换对称元素*/
      a[i][j]=a[j][i];
      a[j][i]=x;
     }
 for(i=0;i<3;i++)                     /*输出结果*/
   {for(j=0;j<3;j++)
      printf("%4d",a[i][j]);
      printf("\n");                  /*每行后换一新行*/
   }
}
```

第七章 数组

7.4　字符数组的定义与引用

7.4.1　字符数组的初始化

(1) 对全部元素赋初值。例如：

```
char c[5]={'a','b','c','d','e'};
/*初值个数与数组元素的个数相等*/
```

```
char c[5];
c[0]='a';
c[1]='b';
c[2]='c';
c[3]='d';
c[4]='e';
```

7.4　字符数组的定义与引用

7.4.1　字符数组的初始化

(2) 给数组前面的部分元素赋初值。

例如：

```
char c[5]={'a','b','c'};
/*初值个数与数组元素的个数不等，
5个元素，3个初值*/
```

```
int a[5];
c[0]='a';
/*前3个元素被赋指定初值*/
c[1]='b';
c[2]='c';
a[3]='\0';a[4]='\0';
/*后2个元素初值为'\0'*/
```

7.4　字符数组的定义与引用

7.4.1　字符数组的初始化

(3) 在定义的同时给全部元素赋初值
这种情况下，可以省略数组长度。
例如：

```
char c[5]={'a','b','c','d','e'};
/*初值个数与数组元素的个数相等*/
```

```
char c[]={'a','b','c','d','e'};
```

7.4　字符数组的定义与引用

7.4.2　字符串和字符串结束标志

在C语言中没有专门的字符串变量，通常用一个字符数组来存放一个字符串。前面介绍字符串常量时，已说明字符串总是以’\0’作为串的结束符。因此当把一个字符串存入一个数组时，也把结束符’\0’存入数组，并以此作为该字符串是否结束的标志。有了’\0’标志后，就不必再用字符数组的长度来判断字符串的长度了。

7.4　字符数组的定义与引用

7.4.2　字符串和字符串结束标志

C语言允许用字符串的方式对数组作初始化赋值。例如：

```
char c[10]={'c',' ','p','r','o','g','r','a','m'};          /*9 个常量*/
```

```
char c[10]={'c',' ','p','r','o','g','r','a','m','\0'};     /*最后一个是空值*/
```

```
char c[]={'c',' ','p','r','o','g','r','a','m','\0'};       /*可省略数组长度*/
```

```
char c[]={"C program"};                                    /*写成字符串的方式*/
```

```
char c[]="C program";                                      /*可去掉{}*/
```

7.4 字符数组的定义与引用

7.4.3 字符数组的输入输出

在采用字符串方式后，字符数组的输入输出将变得简单方便。

可用两种方式进行输入输出。

① %c格式，每次输入输出一个字符。

② %s格式，每次输入输出一个串。

说明：

① 用%s格式输出时，输出项用数组名。例如：

```
char c[]="C program";
printf(" s" ,c);
```

② 用%s格式输出时，输出字符不包含结束符’\0’。

③ 用%s格式输出时，输出时从第一个字符开始，到’\0’为止。

④ 用%s格式输入时，可输入一个字符串。输入项用数组名。例如：

```
char c[10];
scanf(" %s" ,c);
```

7.4　字符数组的定义与引用

7.4.4　字符串处理函数

C语言提供了丰富的字符串处理函数，大致可分为字符串的输入、输出、合并、修改、比较、转换、复制、搜索几类。使用这些函数可大大减轻编程的负担。用于输入输出的字符串函数，在使用前应包含头文件"stdio.h"，使用其它字符串函数则应包含头文件"string.h"。

1）字符串输出函数 puts

语法格式：puts(字符数组名)

功能：把字符数组中的字符串输出到显示器。即在屏幕上显示该字符串。

例7.8　定义一个字符数组并输出。

```
# include <stdio.h>
void main()
{
 char c[]="Windows\nUnix";
 puts(c);
}
```

7.4　字符数组的定义与引用

2）字符串输入函数gets

语法格式：gets(字符数组名)

功能：从标准输入设备键盘上输入一个字符串。本函数得到一个函数值，即为该字符数组的首地址。

用该函数可以接受包含空格的串。输入时按回车结束。

例7.9　先输入一个串，再输出它。

```
# include <stdio.h>
void main()
{
 char st[15];
 clrscr();
 printf("Please input a string:\n");
 gets(st);
 printf("The resualt string is:\n");
puts(st);
}
```

7.4　字符数组的定义与引用

3）字符串连接函数strcat

语法格式：strcat(字符数组名1，字符数组名2)

功能：把字符数组2中的字符串连接到字符数组1中字符串的后面，并删去字符串1后的串标志”\0”。本函数返回值是字符数组1的首地址。

例7.10　输入一个人名，连接上前缀后输出。

```
# include <stdio.h>
void main()
{
 char st1[30]="My name is ";
int st2[10];
 clrscr();
 printf("Please input your name:\n");
 gets(st2);
 strcat(st1,st2);
printf("The resualt string is:\n");
 puts(st1);
}
```

7.4　字符数组的定义与引用

4）字符串拷贝函数strcpy

语法格式：strcpy(字符数组名1，字符数组名2)

功能：把字符数组2中的字符串拷贝到字符数组1中。串结束标志”\0”也一同拷贝。字符数名2，也可以是一个字符串常量。这时相当于把一个字符串赋予一个字符数组。

例7.11　拷贝一个串后输出。

```
# include <stdio.h>
void main()
{
 char st1[15],st2[]="C Language";
 clrscr();
 strcpy(st1,st2);
 puts(st1);
}
```

7.4　字符数组的定义与引用

7.4.4　字符串处理函数

5）字符串比较函数strcmp

语法格式：strcmp(字符数组名1，字符数组名2)

功能：按照ASCII码顺序比较两个数组中的字符串，并由函数返回值返回比较结果。

字符串1＝字符串2，返回值＝0；

字符串2>字符串2，返回值>0；

字符串1<字符串2，返回值<0。

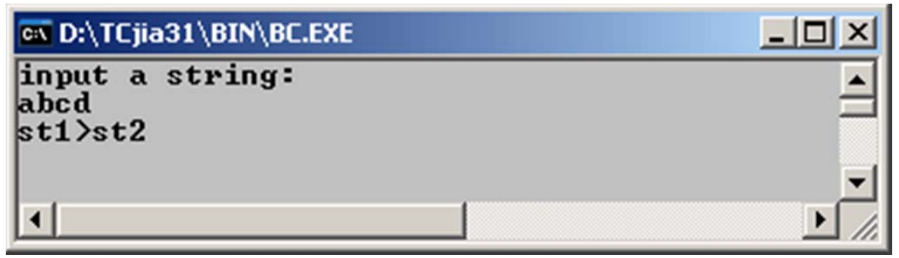
本函数也可用于比较*两个字符串常量*，或比较*数组和字符串常量*。

7.4　字符数组的定义与引用

7.4.4　字符串处理函数

例7.12　输入一个串，并与串"C Language"进行比较。

```
# include <stdio.h>
# include <string.h>
void main()
{int k;
 char st1[15],st2[]="C Language";
 clrscr();
 printf("input a string:\n");
 gets(st1);
 k=strcmp(st1,st2);
 if(k==0) printf("st1=st2\n");
 if(k>0) printf("st1>st2\n");
 if(k<0) printf("st1<st2\n");
}
```



7.4　字符数组的定义与引用

6）测字符串长度函数strlen

语法格式：strlen(字符数组名)

功能：测字符串的实际长度(不含字符串结束标志'\0')，并作为函数返回值。

例7.13　求一个串的长度

```
# include <stdio.h>
# include <string.h>
void main()
{int k;
char st[]="C language";
 clrscr();
 k=strlen(st);
 printf("The lenth of the string is %d\n",k);
}
```

第七章 数组

7.5　数组及数组元素做函数的参数

7.5.1　数组元素做函数的参数

数组元素可以充当函数的实参，用法与作用与普通变量相同。函数发生调用时，将实参数组元素的值传递给形参。

例7.14　利用函数求两数组元素的最大值。

7.5 数组及数组元素做函数的参数

```
# include <stdio.h>
int fmax(int x,int y)        /*函数的定义与前述方法相同*/
{int z;
 if(x>y)
    z=x;
 else
    z=y;
 return z;
}
void main()
{int a[2]={3,5};              /*定义一个有2个元素的数组*/
 int max;
 max=fmax(a[0],a[1]);        /*调用时，用数组元素做实参*/
 printf("The max = %d",max);
}
```

7.5　数组及数组元素做函数的参数

7.5.2　数组做函数的参数

数组做函数参数即可以充当*实参*，也可以充当*形参*。函数发生调用时，传递给形参数组的是实参数组*首元素的地址*，使得形参数组与实参数组占据相同的*内存单元*。可以认为，形参数组与实参数组为同一数组，形参数组名为实参数组的别名。*在子函数中对形参数组的改变相当于改变了实参数组。*

数组充当实参的语法格式：数组名

数组充当形参的语法格式：类型说明符 数组名[ ]

7.5　数组及数组元素做函数的参数

例7.15　利用函数对数组元素做乘2计算。

```
# include <stdio.h>
void farr(int x[])
 /*定义形参数组*/
{int i;
 for(i=0;i<3;i++)
 /*在子函数中改变形参数组的值*/
    x[i]*=2;
}
```

```
void main()
{int a[3]={1,3,5};
 /*定义一个有3个元素的数组*/
 int i;
 printf("Diao yong han shu
qian:\n");
 for(i=0;i<3;i++)
 /*调用函数前输出一遍a数组*/
   printf("%4d",a[i]);
farr(a);
 /*用数组做实参*/
printf("\n Diao yong han shu
hou:\n");
for(i=0;i<3;i++)
/*调用函数后再输出一遍a数组*/
   printf("%4d",a[i]);
}
```

第七章 数组

7.6　编程举例

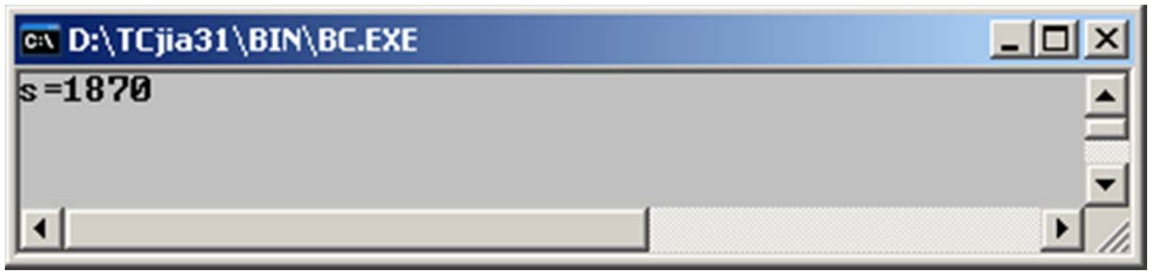
例7.16 求斐波那契数列前20项中第1，5，9，13，17项的和。

分析：

若仅仅计算斐波那契数列前若干项的值，那么用数组或用循环都可以。若想对数列中的数进行进一步的计算，则2种方法的效率是完全不同的。若用循环解本题需要记住当前计算到了第几项，还要判断是否是被求和项，非常麻烦。若用数组解本题可考虑用两个循环。第1个循环计算数列的每一项，第2个循环求和。

7.6　编程举例

```
# include <stdio.h>
void main()
{int a[20]={1,1},i,s=0;
        /*变量s表示和，定义时直接赋初值清零*/
 clrscr();
 for(i=2;i<20;i++)
        /*计算数列各项*/
   a[i]=a[i-1]+a[i-2];
 for(i=0;i<20;i++)
   if(i+1==1||i+1==5||i+1==9||i+1==13||i+1==17
        /*判断当前项是否为求和项*/
     s+=a[i];
 printf("s=%d",s);
}
```



7.6　编程举例

例7.17 从键盘接收任意10个整数存入数组，并最小数及其下标位置。

分析：

设计程序时，可以考虑采用循环完成对数组的遍历。设计2个变量：用value存放最小数的值，pos存放下标位置。

value的初值为数组首元素，pos的初值为0。从下标为1的元素开始遍历数组时，若某元素的值比value还小，则更新value和pos的值。

程序分成3个部分：第1部分输入10个数据；第2部分计算；第3部分输出。

7.6 编程举例

```
# include <stdio.h>
void main()
{int a[10];                                  /*定义有10个元素的数组*/
 int i,value,pos;                            /*i做循环变量*/
 for(i=0;i<10;i++)                           /*输入10个初始数据*/
   scanf("%d",&a[i]);
 value=a[0];                                 /*为value和pos赋初值*/
 pos=0;
for(i=1;i<10;i++)                            /*从下标为1的元素开始遍历数组*/
   if(a[i]<value)
                  /*若找到新的最小值，则更新value和pos的值*/
     {
      value=a[i];
      pos=i;
}
 printf("value=%d,pos=%d",value,pos);        /*输出结果*/
}
```

7.6　编程举例

请思考：变量value和pos有什么关系？编程时是否可以省略变量value。若可以，请改写程序。

7.6　编程举例

例**7.18**　将有**10**个元素的数组的最小值与首元素互换位置。

```
# include <stdio.h>
void main()
{int a[10];                                  /*定义有10个元素的数组*/
 int i,value,pos;                            /*i做循环变量*/
int x;                                       /*x做互换中间变量*/
 for(i=0;i<10;i++)                           /*输入10个初始数据*/
   scanf("%d",&a[i]);
 printf("\n yuan shi shu zhu:\n");           /*输出原始数据*/
 for(i=0;i<10;i++)
   printf("%4d",a[i]);
 value=a[0];                                 /*为value和pos赋初值*/
 pos=0;
for(i=1;i<10;i++)                            /*从下标为1的元素开始遍历数组*/
   if(a[i]<value)                            /*若找到新的最小值，则更新value和pos的值*/
     {
      value=a[i];
      pos=i;
}
     x=a[0];                                 /*互换两数的位置*/
     a[0]=a[pos];
     a[pos]=x;
printf("\n zui hou jie guo:\n");             /*输出最后结果*/
 for(i=0;i<10;i++)
   printf("%4d",a[i]);
}
```

请思考：语句**a[0]=a[pos];**可否改为**a[0]=value;**？为什么？如果不要**x**做互换的中间变量，是否可以完成互换？若可以请改写程序。

关于排序

Google的成功得益于其强大的功能和独到的特点：

Google检索网页数量达24亿，搜索引擎中排名第一；

Google支持多达132种语言，包括简体中文和繁体中文；

Google网站只提供搜索引擎功能，没有花里胡哨的累赘；

Google速度极快，年初时据说有15000多台服务器，200多条T3级宽带；

Google的专利网页级别技术PageRank能够提供准确率极高的搜索结果；

Google智能化的“手气不错”功能，提供可能最符合要求的网站；

Google的“网页快照”功能，能从Google服务器里直接取出缓存的网页。

Google具有独到的图片搜索功能；

Google具有强大的新闻组搜索功能；

Google具有二进制文件搜索功能（PDF，DOC，SWF等）；

Google排名的因素

- 排名 得分 分类 详细说明

1 4.9 关键词 关键词在网站TITLE上的使用
2 4.4 外部链接 外部链接的锚文字
3 4.4 网站品质 网站的外部链接流行度、广泛度
4 4.1 网站品质 域名年龄（从被搜索引擎索引开始计算）
5 4 页面质量 网站内部链接结构
6 3.9 网站品质 网站的外部链接页面内容与关键词的相关性
7 3.9 网站品质 网站在主题相关的网站群中的链接流行度
8 3.7 关键词 关键词在网页内容上的应用
9 3.6 外部链接 外部链接页面本身的链接流行度
10 3.5 网站品质 网站新外部链接产生的速率
11 3.5 页面质量 导出链接的质量和相关性
12 3.5 外部链接 外部链接页面的主题性
13 3.5 外部链接 外部链接页面在相关主题的网站社区中的链接流行度
14 3.4 关键词 页面内容和关键词的相关性（语义分析）
15 3.4 页面质量 页面的年龄
16 3.3 关键词 关键词在H1标签中的使用
17 3.2 网站品质 网站收录数量
18 3.2 外部链接 链接的年龄
19 3.1 网站品质 用户查询的关键词与网站主题的相关性（防止Google bombing）
20 3.1 外部链接 链接的周围文字

Google排名的因素

- 排名 得分 分类 详细说明
  21 3 关键词 关键词在网站域名中的使用
  22 3 页面质量 页面内容的质量
  23 2.8 关键词 关键词在页面URL中的使用
  24 2.8 关键词 关键词在H2、H3等Headline标签中的使用
  25 2.8 页面质量 网站的结构层次
  26 2.8 网站品质 用户行为
  27 2.8 外部链接 同域名下外部链接页面的链接流行度
  28 2.6 关键词 图片的关键词优化
  29 2.6 网站品质 Google的人工授予权重
  30 2.6 网站品质 域名的特殊性（.edu .gov等）
  31 2.5 网站品质 新页面产生的速率
  32 2.5 外部链接 外部链接的创建和更新时间
  33 2.5 外部链接 外部链接网站域名的特殊性
  34 2.4 外部链接 外部链接网站的PR值
  35 2 关键词 关键词在Meta Description中的使用
  36 2 网站品质 用户搜索网站的次数
  37 1.9 页面质量 URL中“/”符号的出现次数
  38 1.8 页面质量 拼写和语法的正确性
  39 1.4 页面质量 HTML代码是否通过W3C认证
  40 1.3 网站品质 网站是否通过Google Webmaster Central的确认
  41 1.2 关键词 关键词在Meta Keywords中的使用

Google的PageRank值排序算法

- 基本思想

如果网页T存在一个指向网页 A的链接，则表明的所有者认为 A比较重要，从而把的一部分重要性得分赋予A。

这个 重要性得分的值由T的PageRank值PR(T)和T的出链(从链出的链接)数 C(T)决定。具体公式为：PR(T) / C(T)。而对于页面 A，其PageRank值PR(A)的计算如下 ：

PR(A)=PR(T1) / C(T1)+… +PR(Tn) / C(Tn)

其中，T1，T2 ，… ，Tn为含有指向A链接的页面。

7.6 编程举例

例7.19 从键盘接收任意10个整数，用选择排序法升序排列后输出。

选择排序法的基本思想是：将一组数中最小的数调整到最前面（升序）。

设有n个数，算法具体实现过程如下：

① 设第1个数最小，用变量k记录其位置；

② 用k所在位置的数与第2个数比，若第2个数小，则k记录新的最小数的位置，否则不变；

③ 以此类推；

④ 用k所在位置的数与第n个数比，若第n个数小，则k记录新的最小数的位置，否则不变；互换k所在位置的数与第1个数。经过上述步骤后完成1遍扫描，最小的数被调整到了最前面；

⑤ 对剩余的n-1个数进行①至④步的第2遍扫描。第2小数被调整到了第2的位置；

⑥ 以此类推；

⑦ 进行n-1遍扫描后，只剩1个数即最大的数，不用再比，排序完成。

将程序分成3个部分：第1部分输入10个任意整数；第2部分进行排序；第3部分输出排序后的结果。

7.6　编程举例

```
# include <stdio.h>
void main()
{int a[10],i,j,x,k;                    /*变量k表示当前最小数的位置*/
 clrscr();
 for(i=0;i<10;i++)                     /*读入10个数*/
   scanf("%d",&a[i]);
 for(j=0;j<=8;j++)                     /*变量j控制比较的遍数，*/
                                       /*且表示每1遍最小数的最终位置*/
   {k=j;                               /*设第1个数最小*/
    for(i=j+1;i<=9;i++)                /*从下一个数开始找新的最小数*/
      if(a[i]<a[k])k=i;
    x=a[k];                            /*调整最小数的位置*/
    a[k]=a[j];
    a[j]=x;
   }
 for(i=0;i<10;i++)                     /*输出*/
   printf("%5d",a[i]);
}
```

7.6 编程举例

例：7.20 从键盘接收任意10个整数，用插入排序法升序排列后输出。

插入排序法的基本思想是：将一个数插入到升序数列的所有比它大的数前面后，数列仍然有序（升序）。

设有n个数，算法具体实现过程如下：

① 只有1个数时，数列是升序的；

② 2个数与第1个数比，若第2个数不小于第1个数，排列顺序不变，否则互换两数，完成1遍排序；

③ 设有n-1个数已升序排好；

④ 第n个数与第n-1个数比，若不小于第n-1个数，排列顺序不变，本遍排序完成，否则互换两数；

⑤ 再与其前面的数（即第n-2个数）比，若不小于，则排列顺序不变，本遍排序完成，否则互换两数；

⑥ 以此类推，直至第1个数处理完为止，排序完成。

将程序分成3个部分：第1部分输入10个任意整数；第2部分进行排序；第3部分输出排序后的结果。

7.6 编程举例

```
# include <stdio.h>
void main()
{int a[10],i,j,x;                          /*变量x用于存放被插入的数*/
 clrscr();
 for(i=0;i<10;i++)                         /*输入*/
   scanf("%d",&a[i]);
 for(j=1;j<=9;j++)                         /*j控制排序遍数和被插入的数*/
   {i=j-1;                                 /*从当前数的前一个数开始比*/
    while(i>=0)                            /*当还有没比完的数时*/
      {if(a[i+1]>=a[i]) break;             /*不小于前一数时，结束本遍排序*/
x=a[i];                                    /*与前一数互换*/
a[i]=a[i+1];
a[i+1]=x;
       i--;
      }
   }
 for(i=0;i<10;i++)                         /*输出*/
   printf("%5d",a[i]);
}
```

7.6 编程举例

例：7.21 输入一个数，并在有序数列中用二分查找法进行查找。若有相同数，则输出其位置，否则给出提示信息“Not found.”。

在一数组中查找是否存在某数完全可以用顺序的方法从头到尾一个一个地查。若有n个数，最好的情况是1次比较就有结论，最坏的情况是n次比较才有结论，总体效率并不高。对于无序的数列必须用此方法。对于有序的数列可用二分查找法提高效率。这种方法类似于查英文字典。单词是按字母顺序排列的。若想查单词“people”可随手打开词典，若当前页的单词都是以字母“k”开头的，可想而知目标单词“people”应在当前页的后面，当前页前面的部分都不用查了。设有n个数，算法具体实现过程如下：

① 置l=1，表示左边界位置，r=n，表示右边界位置；

② 若l>r，表示没找到，查找结束；

③ 计算中间位置m=(l+r)/2；

④ m位置的数与目标数相比；

⑤ 若相等表示找到该数，查找结束；

⑥ 若大于目标数，修改右边界位置，r=m-1，转第②步继续；

⑦ 若小于目标数，修改左边界位置，l=m+1，转第②步继续。

7.6 编程举例

```
# include <stdio.h>
void main()
{int a[10]={1,3,5,7,12,14,16,27,45,99},l,r,m,x;          /*变量x用于存放目标数*/
 clrscr();
 printf("Please input a number\n");                      /*输入前做一个提示*/
 scanf("%d",&x);
 l=0;                                                    /*置左右边界的初值*/
 r=9;
 while(l<=r)                /*满足l<=r时意味着范围非空，需继续查找*/
{m=(l+r)/2;                                              /*计算中间位置*/
  if(a[m]==x)                                            /*等于目标数，表示找到*/
{printf(“Posication is %d”,m);
 break;
}
if(a[m]>x)   /*大于目标数，修改右边界*/
r=m-1;
```

7.6 编程举例

例：7.22 将数组中元素的值按逆序重新存放。

逆序存放时，原序排第1的数现在排最后，原序排最后的数现在排第1，即将数组中的数与按中间轴对称的数进行互换。

进行互换时要注意元素的个数。可以以中间轴左侧的元素作基准进行遍历，将其与对称元素进行互换。编程时注意中间轴的位置。若数组a有10的数，下标从0开始，中间轴在a[4]，a[5]之间，遍历时左侧元素下标从0遍历到4。若数组a有9的数，下标从0开始，中间轴是a[4]，遍历时左侧元素下标从0遍历到3。写循环时循环变量的终值小于表达式“数组长度/2”即可。还要注意对称元素的下标之间的关系：对称元素下标之和=数组长度-1。即下标为i的元素，其对称的元素下标为数组长度-1-i。

7.6 编程举例

```
# include <stdio.h>
void main()
{int a[10],i,x;                    /*变量x用作互换时的中间变量*/
 clrscr();
 for(i=0;i<10;i++)                 /*输入数组元素的值*/
   scanf("%d",&a[i]);
 for(i=0;i<10/2;i++) /*以中间轴左侧的元素作基准进行遍历*/
   {x=a[i];                        /*互换对称元素*/
    a[i]=a[10-1-i];
    a[10-1-i]=x;
   }
 for(i=0;i<10;i++)                 /*输出结果*/
    printf("%5d",a[i]);
}
```

7.6　编程举例

例7.23 从键盘输入一3×4的矩阵的值，并求出最大元素及其位置。写程序时，预设下标为[0][0]的元素为最大值。遍历数组元素时每一数都与当前最大值相比，比最大值大时，更新所存的数据。

7.6　编程举例

```
# include <stdio.h>
void main()
{int a[3][4],i,j,x,l,r;                /*变量x存放最大值，l存放行号，r存放列号*/
 clrscr();
 for(i=0;i<3;i++)
   for(j=0;j<4;j++)
     scanf("%d",&a[i][j]);                          /*读入数组*/
 x=a[0][0];                          /*设a[0][0]为最大值*/
 l=r=0;                                             /*记录下标*/
 for(i=0;i<3;i++)                                   /*遍历数组*/
   for(j=0;j<4;j++)
     if(a[i][j]>x)                                  /*找到新的最大值时，更新原值*/
       {x=a[i][j];
        l=i;
        r=j;
       }
 printf("Max number is %d,posation is %d,%d\n",x,l,r);
}                                                   /*输出结果*/
```

7.6 编程举例

例7.24 用函数完成例7.1和例7.2。

分析：

在主函数中实现数据的输入和输出。用fmax函数求最高分。用fcount函数求不及格人数。为fmax函数设计另两个形参：arr[]和n。arr[]用来接收从主函数传来的学生成绩。n用来存数组元素的个数，也即学生数。在主函数中函数调用作为表达式直接输出。

主函数只有3条语句，统计计算分别由子函数来完成，使得程序清晰，更好地体现了模块化程序设计的思想。

7.6 编程举例

```
# include <stdio.h>
int fmax(int arr[],int n)      /*定义形参数组*/
{
int max,i;
 max=arr[0];                   /*设首元素为最大值的初值*/
 for(i=1;i<n;i++)                          /*遍历数组求最大值*/
   if(arr[i]>max)
      max=arr[i];
 return max;                   /*返回计算结果*/
}
int fcount(int arr[],int n)
{
int count,i;
count=0;                                   /*设count的初值为0*/
for(i=0;i<n;i++)                           /*统计不及格人数*/
   if(arr[i]<60)
      count++;
 return count;                             /*返回计算结果*/
}
void main()
{
 int a[10]={ 90,80,95,56,65,47,93,82,75,61};
                                           /*预置数组初值*/
 printf("\n zui gao fen wei:%d",fmax(a,10));
                                           /*将函数调用做表达式，直接输出*/
printf("\n bu ji ge ren shu wei :%d",fcount(a,10));
}
```

7.6 编程举例

例7.25 输入五个国家的名称按字母顺序排列输出。

国家名是字符串，5个国家名应用一个二维字符数组来表示。此题相当于对5个字符进行升序排序。排序时用strcmp函数比较各串的大小。用选择排序法排序

7.6 编程举例

```
# include <stdio.h>
# include <string.h>                    /*使用strcmp函数须加上头文件string.h*/
void main()
{char st[20],cs[5][20];                 /*定义二维数组存放5个国家名，*/
                                        /*一维数组作临时数组*/
 int i,j,p;
 clrscr();
 printf("input country's name:\n");
 for(i=0;i<5;i++)                       /*输入5个名字*/
   gets(cs[i]);
 for(i=0;i<4;i++)                       /*用选择法排序*/
   {p=i;strcpy(st,cs[i]);
    for(j=i+1;j<5;j++)
      if(strcmp(cs[j],st)<0)
        {p=j;
         strcpy(st,cs[j]);
        }
    if(p!=i)
      {strcpy(st,cs[i]);
       strcpy(cs[i],cs[p]);
       strcpy(cs[p],st);
      }
   }
 printf("The resualt is:\n");
 for(i=0;i<5;i++)                       /*输出结果*/
   puts(cs[i]);
}
```