# 第七章 数组

本章学习目标：

* 掌握数组声明符、数组声明的方法。
* 掌握一维数组、二维数组在内存中的存储。
* 掌握通过下标方式访问数组中各元素的方法。
* 掌握通过指向数组的指针访问数组中各元素的方法。
* 掌握字符数组与其他内置类型数组之间的细微差别。

## 7.1 实践题

**一、一维数组**

**实验目的**

1. 了解一维数组的概念
2. 掌握一维数组的定义和初始化
3. 一维数组的元素的使用
4. 一维数组常见应用

**实验步骤**

步骤一 用原有知识解决以下问题

1. 从键盘上接收3个学生成绩
2. 计算3个学生的成绩和计算
3. 计算3个学生的平均值
4. 显示学生的平均值

分析问题后，主要代码如下：

int main(int argc,char \*argv[]){

int s1;

int s2;

int s3;

int sum = 0;

int avg = 0;

scanf(“%d”,&s1);

scanf(“%d”,&s2);

scanf(“%d”,&s3);

sum = s1 + s2 +s3;

printf(“sum is %d\n”,sum);

avg = sum/3;

printf(“avg is %d\n”,avg)

}

步骤二 用数组的方式解决该问题

int main(int argc,char \*argv[]){

int a[3] = {0,0,0}

int i;

int sum = 0;

int avg = 0;

for(i=0;i<10;i++){

scanf(“%d”,&a[i]);

sum += a[i];

}

printf(“sum is %d\n”,sum);

avg = sum/3;

printf(“avg is %d\n”,avg)

}

**实验结果/结论**

1. **实验结果**

* 用数组的方式一样能够解决我们提出的问题，并且代码更清晰简短。

2. **实验结论**

* 当我们对大量类型相同的数据进行操作时，采用数组来解决问题，会减少很多重复的工作及冗余的代码。

**二、一维数组及指针**

**实验目的**

1. 了解一维数组的存储规则
2. 掌握使用指针操纵数组的方法
3. 了解指针算数运算、比较运算

**实验步骤**

步骤1：用数组的方法解决在上一实验中提出的问题，并准备好代码。

步骤2：用指针操作数组解决上面的问题。得到主要代码如下：

int main(int argc,char \*argv[]){

int a[3] = {0,0,0}

int \*p = a;

int i;

int sum = 0;

int avg = 0;

for(i=0;i<10;i++){

scanf(“%d”,p+i);

sum += \*(p+i);

}

printf(“sum is %d\n”,sum);

avg = sum/3;

printf(“avg is %d\n”,avg)

}

**实验结果/结论**

1. **实验结果**

* 可以用指针来控制变量，得到数组的操作。

2. **实验结论**

* 数组元素在内存中是连续存放的

**二、 二维数组及指针**

**实验目的**

1. 了解二维数组的存储规则
2. 掌握使用指针操纵二维数组的方法

**实验步骤**

步骤1：一重指针操纵二维数组

问题：用一重指针将数组a[2][3]的数据打印出来

经分析后，主要代码如下：

int main(int argc,char \*argv[]){

int a[2][3] = {{0,0,0}{0,0,0}};

int \*p = &a[0][0];

int i;

int j;

for(i=0;i<2;i++){

for(j=0;j<3;j++){

scanf(“%d”,&a[i][j]);

}

}

for(i=0;i<2\*3;i++){

printf(“%d\t”,\*(p+i));

}

}

步骤2：通过二重循环遍历二维数组

问题：用二重循环的方式，打印出数组a[2][3]中的数据,主要代码如下：

int main(int argc,char \*argv[]){

int a[2][3] = {{0,0,0}{0,0,0}};

int \*p = &a[0][0];

int i;

int j;

for(i=0;i<2;i++){

for(j=0;j<3;j++){

scanf(“%d”,&a[i][j]);

}

}

for(i=0;i<2;i++){

for(j=0;j<3;j++){

printf(“%d\t”,\*(p+i\*3+j));

}

printf(“\n”);

}

}

**实验结果/结论**

1. **实验结果**

* 用指针可以操纵二维数组
* 二维数组用二重循环来操纵更加方便

2. **实验结论**

* 二维数组用二重循环来操纵更加方便
* 二维数组中的数也是连续存放的

## 7.2 理论题

### A类

1. **填空题**
2. 构成数组的各个元素必须具有相同的 。
3. 下面的程序是输出数组中最大元素的下标(p表示最大元素的下标)。

void main()

{

int s[]={1,-3,0,-9,8,5,-20,3};

for(i=0,p=0;i<8;i++)

if(s[i]>s[p]) ;

}

1. 输入20个数，输出他们的平均值，输出与平均值之差的绝对值最小的数组元素。

#include <stdio.h>

void main()

{

float a[20],pjz=0,s,t;

int i,k;

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

{

scanf(“%f”,&a[i]);

pjz+= ；

}

s=fabs(a[0]-pjz);

t=a[0];

for(i=1;i<20;i++)

if( fabs(a[i]-pjz)<s )

{

t=a[i];

}

}

1. 输出行、列号之和为3的数组元素。

main()

{ char ss[4][3]={'A','a','f','c','B','d','e','b',

'C','g','f','D'};

int x,y,z;

for (x=0; (1) ;x++)

for (y=0; (2) ;y++)

{ z=x+y;

if ( (3) ) printf("%c\n",ss[x][y]);

}

}

1. 将一个数组中的元素按逆序重新存放。例如原来的顺序为：8,5,7,4,1, 要求改为：1,4,7,5,8 。

#define N 7

void main()

{ int a[N]={12,9,16,5,7,2,1},k,s;

printf("\n初始数组:\n");

for (k=0;k<N;k++)

printf("%4d",a[k]);

for (k=0;k< (1) ;k++)

{ s=a[k]; a[k]= (2) ; (3) =s; }

printf("\n交换后的数组:\n");

for (k=0; (4) ;k++)

printf("%4d",a[k]);

}

1. 有一行文字，要求删去某一个字符。此行文字和要删去的字符均由键盘输入，要删去的字符以字符形式输入（如输入 a表示要删去所有的 a字符）。

#include <stdio.h>

void main()

{ /\*str1表示原来的一行文字，str2表示删除指定字符后的文字\*/

char str1[100],str2[100];

char ch;

int i=0,k=0;

printf(“please input an sentence:\n”);

gets(str1);

scanf("%c",&ch);

for (i=0; (2) ;i++)

if (str1[i]!=ch)

{ str2[ (3) ]=str1[i]; k++; }

str2[ (4) ]='\0';

printf("\n%s\n",str2);

}

1. 找出10个字符串中的最大者。

#include <stdio.h>

#include <string.h>

#define N 10

void main()

{ char str[20],s[N][20];

int i;

for (i=0;i<N;i++)

gets( (1) );

strcpy(str,s[0]);

for(i=1;i<N;i++)

if ( (2) >0) strcpy(str,s[i]);

printf("The longest string is : \n%s\n",str);

}

1. 某人有四张 3分的邮票和三张 5分的邮票，用这些邮票中的一张或若干张可以得到多少种不同的邮资？

main()

{ static int a[27];

int i,j,k,s,n=0;

for (i=0;i<=4;i++)

for (j=0;j<=3;j++)

{ s= (1) ;

for (k=0;a[k];k++)

if (s==a[k]) (2) ;

if ( (3) )

{ a[k]=s; n++;}

}

printf("%d kind:",n);

for (k=0; (4) ;k++)

printf("%3d",a[k]);

}

1. 求矩阵的马鞍点。马鞍点即它的值在行中最大，在它所在的列中最小。

#define N 10

#define M 10

main()

{ int i,j,k,m,n,flag1,flag2;

int a[N][M],max;

printf("\n输入行数 n:");

scanf("%d",&n);

printf("\n输入列数 m:");

scanf("%d",&m);

for (i=0;i<n;i++)

for (j=0;j<m;j++)

scanf("%d", (1) );

for (i=0;i<n;i++)

{ for (j=0;j<m;j++)

printf("%5d",a[i][j]);

(2) ;

}

flag2=0;

for (i=0;i<n;i++)

{ max= (3) ;

for (j=1;j<m;j++)

if (a[i][j]>max) max=a[i][j];

for (j=0;j<m;j++)

{ flag1=0;

if (a[i][j]==max)

{ for (k=0,flag1=1;k<n&&flag1;k++)

if ( (4) ) flag1=0;

if (flag1)

{ printf("第%d行, 第%d列的 %d是鞍点\n", (5) );

flag2=1;

}

}

}

}

if (!flag2)

printf("\n矩阵中无鞍点!\n");

}

**二、选择题**

1. 在定义int a[10];之后，对a的引用正确的是 。

A. a[10] B. a[6.3] C. a(6) D. a[10-10]

1. 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 a[3][2]={{1,2},{3,4}}

1. 在执行int a[][3]={1,2,3,4,5,6};语句后，a[1][0]的值是 。

A. 4 B. 1 C. 2 D. 5

1. 以下不能正确赋值的是 。

A. char s1[10];s1="test"; B. char s2[]={’t’,’e’,’s’,’t’}

C. char s3[20]= "test"; D. char s4[4]={ ’t’,’e’,’s’,’t’}

1. 下面程序段运行时输出结果是 。

char s[12]= "A book";

printf("%d\n",strlen(s));

A. 12 B. 8 C. 7 D. 6

**三、综合题**

1. 有一个数组，内放10个整数。要求找出最小的数和它的下标，然后把它和数组中最前面的元素对换位置。
2. 求一个 3×3矩阵两条对角线上元素之和（每个元素只加一次）。
3. 打印如下形式的杨辉三角形

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

输出前10行，从 0行开始，分别用一维数组和二维数组实现。

1. 有一个二维数组整型数组中，每一行都有一个最大值，编程求出这些最大值以及它们的和。

### B类

**一、选择题**

1. 下面各语句中，能正确进行赋字符串操作的语句是（ ）

A.char s[5] ={“ABCDE”}；

B.char s[5] ={‘A’，‘B’，‘C’，‘D’，‘E’}；

C. char \*s；s＝“ABCDE”；

D. char \*s；scanf（“%s”, &s）；

3. 下述对C语言字符数组的描术中错误的是

A) 字符数组可以存放字符串

B) 字符数组中的字符串可以整体输入、输出

C) 可以在赋值语句中通过赋值运算符"="对字符数组整体赋值

D) 不可以用关系运算符对字符数组中的字符串进行比较、

C、2

4. 设有下面的程序段，则下列正确的是 。

char s[]="china";

char \*p; p=s;

A)s和p完全相同

B)数组s中的内容和指针变量p中的内容相等

C)s数组长度和p所指向的字符串长度相等

D)\*p与s[0]相等

5. 下面各语句中，能正确进行赋字符串操作的语句是（ ）

A.char s[5] ={“ABCDE”}；

B.char s[5] ={‘A’，‘B’，‘C’，‘D’，‘E’}；

C. char \*s；s＝“ABCDE”；

D. char \*s；scanf（“%s”, &s）；

6.下列语句中，正确的是（）

A.Char a[3][]={‘adc”,1}

B.Char a[][3]={‘abc’,’1’}

C.Char a[3][]={‘a’,”1”}

D.Char a[][3]={“a”,”1”}

E.Char a[]={0,1,2,3,4,5,6,7}

解释：

1. **综合题**
2. 打印魔方阵。所谓魔方阵是指这样的方阵，它的每一行、每一列和对角线之和均相等。例如：三阶魔方阵为

8 1 6

3 5 7

4 9 2

要求打印由 1到的自然数构成的魔方阵。

提示：魔方阵中各数的排列规律如下：

⑴ 将“1”放在第一行中间一列；

⑵ 从“2”开始直到n×n为止各数依次按下列规则存放：每一个数存放的行比前一个数的行数减1，列数加1；

⑶ 如果上一个数的行数为1，则下一个数的行数为 n（指最下一行）；

⑷ 当一个数的列数为 n，下一个数的列数应为1，行数减1；

⑸ 如果按上面规则确定的位置已有数，或上一个数是第 1行第 n列时， 则把下一个数放在上一个数的下面。

## 本章答案

### A类

**一、填空题**

* 1. 类型
  2. (1) int i,p

(2) p=i

(3) printf(“%d\n”,p);

* 1. (1) #include “math.h”

(2) a[i]/20

(3) s=fabs(a[i]-pjz);

(4) printf(“%f,%f\n”,pjz,t);

* 1. (1) x<4

(2) y<3

(3) z==3

* 1. (1) N/2

(2) a[N-1-k]

(3) a[N-1-k]

(4) k<N

* 1. (1) str[i]!=’\0’

(2) k

(3) k

* 1. (1) s[i]

(2) s[i],str

* 1. (1) i\*3+j\*5

(2) break

(3) s!=a[k]

(4) k<n

* 1. (1) &a[i][j]

(2) printf(“\n”)

(3) a[i][0]

(4) a[k][j]<max

(5) i,j,a[i][j]

**二、选择题**

1.A，2.D，3.A，4.A，5.D

**三、综合题**

### B类

**一、填空题**

**二、选择题**

**三、综合题**