第5章 用数组处理批量数据

数组是一组具有相同数据类型的变量的集合，是一种简单的构造数据类型。在一个数组中，构成该数组的成员称为数组元素，数组中的每个元素都属于同一个数据类型。数组有两个基本要素：数据的类型和数据的位置。其中，数据的类型可以是前面介绍的基本数据类型（整型、实型、字符型），也可以是构造数据类型，数据的位置就是数据在数组中的相对位置，称之为下标。用一个统一的数组名和下标唯一地确定数组中的元素，通过数组的下标实现对数组元素的访问。由于数组元素可以看作是单个变量，所以对多个同类型变量的操作都可以推广到用数组操作。根据数组元素的数据类型，数组又可分为数值型数组（整型数组和实型数组）、字符数组、指针数组、结构型数组等。

在程序设计中，数组是一种十分有用的数据结构，可以将大量同类型的数据放到一起批量处理，它能够处理更复杂的数据，使数据的管理更加方便，许多问题如果不用数组，几乎难以解决。

为方便读者了解、学习本章内容，绘制本章思维导图，如图 5-1 所示。

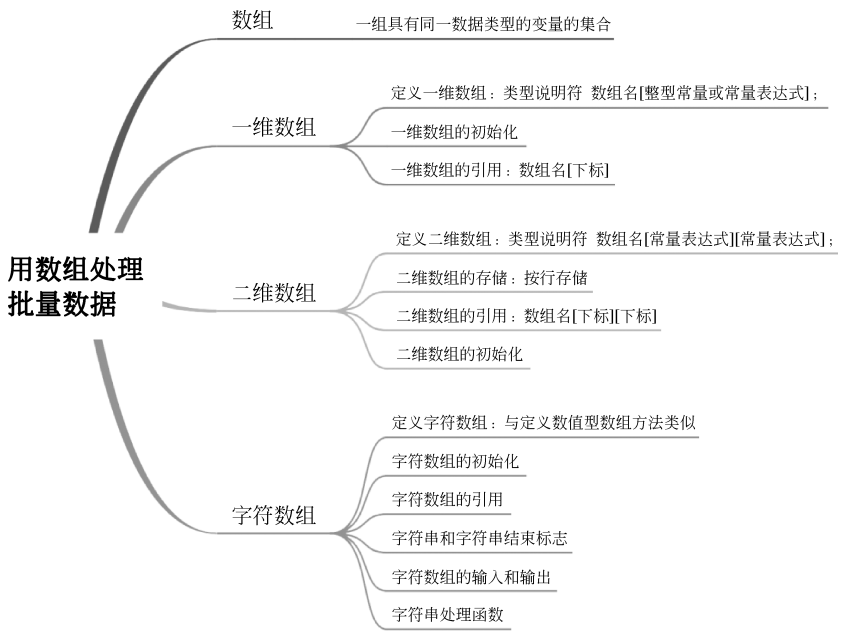

图 5-1 “用数组处理批量数据”思维导图

5.1　一维数组

一维数组是数组中最简单的，它的元素只需要用数组名加一个下标，就能唯一地确定。而有的数组，其元素要指定两个下标甚至多个下标，才能确定，它们是二维数组和多维数组。熟练掌握一维数组后，对二维或多维数组，很容易举一反三，迎刃而解。

5.1.1　知识点介绍

1. 一维数组的定义

当数组中每个元素都只带一个下标时，称这样的数组为一维数组。在 C 语言中，数组必须“先定义，后使用”。定义好一个一维数组后，C 语言系统就会在内存中为该数组开辟一片连续的存储空间。一维数组的定义形式见表 5-1。

表 5-1　一维数组的定义形式及说明

类型说明符 数组名[整型常量或整型常量表达式]；		
说　　明		示　　例
		int　a[10];
类型说明符	数组中所有元素的类型，如 int、float、char 等	int 为类型说明符
数组名	遵循 C 语言标识符命名规则	a 为数组名
方括号中的整型常量或表达式	数组中所含元素的个数，不能是变量，只能是整型常量或表达式	10 为数组的长度，a 数组中共含有 10 个元素，即 a[0]，a[1]，a[2]，…，a[9]，下标从 0 开始，到 9 结束

2. 一维数组的引用

每一个数组元素就是一个变量，所以，在程序中可以把数组中的每个元素当普通的变量来使用，即所谓的数组元素的引用。C 语言规定不能一次引用整个数组，只能使用循环逐个引用数组元素。引用形式为：**数组名[下标]**。

下标表示对应数组元素在数组中的顺序号，必须是整型常量、整型变量或整型表达式。C 语言规定，每个数组的第一个元素的**下标从 0 开始**，称为下标下界，最后一个元素的下标为数组元素的个数减 1，称为下标的上界，对数组元素引用时不能超过数组的界限，C 语言编译系统不对越界进行检查，因此在编写程序中，保证数组下标不越界是十分重要的。

如：

```
int i,j,a[100];
```

则：a[3]，a[i]，a[i+j*2]均为合法的数组元素引用，而 a[3.5]，a[100]都是不合法的数组元素引用。

在程序设计中，由于数组元素排列的规律性，通常可以通过改变其下标值，用循环的方

法对数组元素进行操作。

3. 一维数组初始化

当系统为所定义的数组在内存中开辟一片连续的存储单元时，这些存储单元中并没有确定的值。可以采用以下形式，在定义数组时为所定义数组的各元素赋初值，即初始化。一维数组初始化的形式见表 5-2。

表 5-2 一维数组的初始化

形　式	示　例	说　明
对数组元素全部赋初值	int a[5] = {1,2,3,4,5};	经过初始化后，数组各元素值：a[0]＝1，a[1]＝2，a[2]＝3，a[3]＝4，a[4]＝5
部分元素初始化	int a[5] = {1,2,3};	未赋值的部分元素值为 0，即：a[0]＝1，a[1]＝2，a[2]＝3，a[3]＝0，a[4]＝0
使数组中全部元素值为 0	int a[5] = {0};	初始化后，a 数组中 5 个元素值均为 0，即：a[0]＝0，a[1]＝0，a[2]＝0，a[3]＝0，a[4]＝0
在对全部元素赋初值时，可以不指定数组长度	int a[5] = {1,2,3,4,5};　等价于 int a[ ] = {1,2,3,4,5};	系统会根据所赋初值的个数确定数组的长度

一维数组除了上述初始化给数组赋值外，还可以通过循环语句实现动态输入。例如：

```
int a[10],i;
for(i = 0; i < 10; i++)
    scanf("%d",&a[i]);
```

运行时，用户可以从键盘输入 10 个数。数之间用空格或回车键隔开，如：1 2 3 4 5 6 7 8 9 10。

不能用一个语句一次给整个数组赋值，下面的写法都是错误的：

```
scanf("%d",&a);
scanf("%d",&a[10]);
```

5.1.2 实验部分

1. 实验目的

(1) 熟练掌握一维数组的定义和数组元素的引用方法。
(2) 熟练掌握一维数组的赋值以及输入输出的方法。
(3) 熟练掌握一维数组的相关算法(特别是排序算法和查找算法)。

2. 实验内容

(1) 程序填空一

请补充 main 函数，函数的功能是把一维数组中的元素逆置。结果仍保持在原数组中。

```
#include <stdio.h>
#define  N 10
int main()
{
    int a[N],i,j,t;
    printf("Input array:\n");
    /**************** FILL ****************/
    ________;
    while(i<=9)
    {
       scanf("%d",&a[i]);
       /**************** FILL ****************/
       ________;
    }
    printf("\nThe original array:\n");
    for(i=0;i<N;i++)
        printf("%4d",a[i]);
    /**************** FILL ****************/
    for(j=0, ________ ;j<=i;j++,i--)
    {
       t=a[j];
       /**************** FILL ****************/
        ________;
       a[i]=t;
    }
    printf("\nThe result:\n");
    for(i=0;i<N;i++)
     printf("%4d",a[i]);
    return 0;
}
```

【实验提示】

① 程序第9～10行的while循环语句是动态给数组输入值，所以循环条件中i初值值得关注；第15～16行循环输出数组，第18行for循环语句完成数组元素的逆置操作，即头尾元素互换，注意，此时j和i是调换元素的下标，j=0时，i就是最后一个元素的下标，以此类推。

② 思考：将18行的循环语句用下面的程序段替换，是否可以完成数组中的元素逆置？

```
for(i=0;i<10/2;i++)
{
    j=10-i-1;
    t=a[j];
    a[j]=a[i];
    a[i]=t;
}
```

（2）程序填空二

有10个数放在一个数组中，任意输入一个数，要求用顺序查找法查找该数。若该数存

在,则显示元素对应的位置值,然后继续查找直到最后一个元素为止;若该数不存在,则输出此数不存在。

请在程序的下画线处填入正确的内容,并把下画线删除,使程序得出正确的结果。

注意:不要增行或删行,也不得更改程序结构。

```
#include <stdio.h>
#define N 10
int main()
{
    int a[N],m,i,flag = 0;
    printf("\n Input m :");
    scanf("%d",&m);
    printf("Input array: \n");
    for(i = 0;i < N;i++)
        /**************** FILL ****************/
        scanf("%d", ________);
    /**************** FILL ****************/
    ________;
    while (i < N)
    {
      /**************** FILL ****************/
        if(________)
            i++;
        else
        {
            printf("\n Found!,position is: %d\n",i + 1);
            flag = 1;
            /**************** FILL ****************/
            ________;
        }
    }
    if(flag == 0)
        printf("%d Not found!\n",m);
    return 0;
}
```

【实验提示】

① 顺序查找法的思路是从数组第一个元素开始,从前向后依次与关键字进行比较。若找到该数,则显示元素对应的位置值,直到查找到数组尾部;若未找到该数,则输出不存在。程序从第 13 行起开始顺序查找。

② 注意程序中各变量的含义,变量 flag 是一个标志,找到为 1,没找到为 0。

③ 程序考虑了若在数组中存在多个关键字 m,并都要找到输出的情况。

④ 请用 for 循环语句编程处理上题。

(3) 程序改错一

下列给定的程序中,fun 函数的功能是求出 x 数组中 n 个同学成绩的平均分,并统计高于平均分(含平均分)的个数,将其作为函数的返回值,从主函数中输出。请改正程序中的

错误。

注意：不要改动 main 函数，不得增行或删行，不得更改程序的结构。

```
#include <stdio.h>
int fun(float x[],int n)     /*求平均分并统计高于平均分的个数*/
{
    int j,c=0;
    float ave=x[0];
    /**********ERROR**********/
    for(j=0;j<n;j++)
        ave+=x[j];
    ave=ave/n;
    printf("平均分是: %.2f\n",ave);
    /**********ERROR**********/
    for(j=0;j<=n;j++)
        /**********ERROR**********/
        if(x[j]>ave)
            c++;
    return c;
}
int main()
{
   float x[100]={70.5,82,64.5,95.5,78.5,65.45};
   printf("高于平均分的人数: %d\n",fun(x,6));
   return 0;
}
```

【实验提示】

本题可以拆分成两道简单的 C 语言题目。

① 求出 n 个数的平均值。可通过循环语句对 x 数组元素进行累加，求出 n 个同学的总分，再除以 n 即可求出平均分。

② 将大于平均值的数字个数统计出来。可利用循环和条件选择语句对 x 数组进行遍历，若数组中元素值大于平均值，则计数器累计。

③ 程序中 fun 函数功能是对数组中 n 个分数求平均值并输出，并且统计出高于平均分的人数放入变量 c 中，并将 c 返回给 main 函数。main 函数中 fun(x,5)是调用 fun 函数，此时，实参数组 x 把首地址传递给形参数组 x，实参 6 将值 6 传递给形参 n，从而对 x 数组中的 6 个数求平均。

（4）程序改错二

N 个已排好序的整数数列已放在一个数组中，给定下列程序中，fun 函数的功能是利用折半查找的算法查找整数 m 在数组中的位置。若找到该数，返回其下标值，反之，则返回 −1。请改正程序中的错误。

注意：不要改动 main 函数和其他函数中的任何内容，不得增行或删行，不得更改程序的结构。

```
#include <stdio.h>
#define N 10
int fun(int a[], int m)
{
    int low = 0,high = N - 1,mid;
    while(low <= high)
     {
        mid = (low + high)/2;
        if(m < a[mid])
            /***************** ERROR *****************/
            high = mid + 1;
        else
           /***************** ERROR *****************/
            if(m >= a[mid])
                /***************** ERROR *****************/
                low = mid - 1;
            else
                return(mid);
    }
    return( - 1);
}
int main()
 {
   int i,a[N] = { - 3,4,7,9,13,24,67,89,100,180},k,m;
   printf("Array a :");
   for(i = 0;i < N;i++)
      printf(" %d,",a[i]);
   printf("Input  m:");
   scanf(" %d",&m);
   k = fun(a,m);
   if(k >= 0)
       printf("m = %d,index = %d\n",m,k);
   else
       printf("Not be found!\n");
   return 0;
}
```

【实验提示】

① 折半查找算法，将数列按有序化（递增或递减）排列，当数列有序排列时，折半查找比顺序查找的平均查找速度要快得多，折半查找也称为对分查找。其基本思想是首先选取位于数组中间的元素，将其与查找键进行比较，如果它们的值相等，则查找键被找到，返回数组中间元素的下标；否则，将查找的区间缩小为原来区间的一半，即在一半的数组元素中继续查找。

② 本题中，每次查找前先确定数组中查找范围的上、下界：low（头位置下标）和 high（尾位置下标）（low < high），然后把查找键 m 与中间位置下标（mid）中元素的值进行比较，程序中第 8 行即在求查找范围的中间位置。如果 m 的值大于中间位置元素中的值，则下一次的查找范围放在中间位置之后的元素中，则头位置下标 low＝mid＋1；反之，下次查找范

围放在中间位置之前的元素中，则查找范围的尾位置下标 high＝mid－1。直到 low > high，查找结束。

③ fun 函数是在 a 数组中用折半查找法查找值 m 是否存在。若 m 存在，则返回其下标 mid；若 m 不存在，返回－1。main 函数通过调用 fun 函数，根据得到的返回值来确定是否找到。

④ 比较顺序查找和折半查找两种查找算法，分析各有什么特点。

（5）程序设计一

请写一 fun 函数，该函数的功能是把数组 a 中的数按从大到小的顺序排列（用冒泡排序法）。数组的值从 main 函数中输入，排序结果也在 main 函数中输出。

部分源程序已经给出，请勿改动 main 函数和其他函数中的任何内容，仅在 fun 函数中的空白处填入编写的语句。

```
#include <stdio.h>
#define N 10
void fun(int a[], int n)            //此函数用于冒泡法排序
{
    int i,j,t;
    for(i=1;i<=n-1;i++)
        for(j=0;j<n-i;j++)
    /*****************BEGIN*****************/


   /*****************END*****************/
}
int main()
{
   int i=0,a[N];
   printf("Input array a:\n");
   for(i=0;i<=N-1;i++)
      scanf("%d",&a[i]);
   fun(a,N);
   printf("\n The result is:\n");
   for(i=0;i<=N-1;i++)
   printf("%d,",a[i]);
   return 0;
}
```

【实验提示】

① 本题用冒泡排序法进行排序。冒泡排序法的思路是如果对 n 个数进行排序，则要进行 $n-1$ 轮比较，第一轮要进行 $n-1$ 次两两比较，第 i 轮要进行 $n-i$ 次两两比较，所谓两两比较就是每次将相邻两个数进行比较，并将较大的数调到前面。

② fun 函数完成对数组 a 的排序，main 函数中的函数调用语句：fun(a);是将实参数组 a 的首地址传递给形参数组 a，使形参数组和实参数组共用同一段内存单元，从而完成 fun 函数中对形参数组的排序即对实参数组的排序。

③ 如果将数组 a 中下标值为偶数的元素由大到小排序，其他元素不变，那么该如何修

改程序？

（6）程序设计二

请写一个 fun 函数，功能是任意输入一个数，将其插入一个已排好序的数组中，使原来排好序的数组仍然有序。请勿改动 main 函数和其他函数中的任何内容，仅在 fun 函数的花括号中填入编写的语句。

```
#include <stdio.h>
void fun(int a[11], int y)
{
    int i=0,j;
    if(y>a[9])
       a[10]=y;
    else
    /****************** BEGIN ******************/
    {

    }
    /****************** END ******************/
}
int main()
{
    int a[11]={3,4,6,9,13,16,19,24,35,88};
    int i,m;
    printf("\nInput a number: ");
    scanf("%d", &m);
    fun(a,m);
    for(i=0;i<=10;i++)
        printf("%d,", a[i]);
    return 0;
}
```

【实验提示】

① 本题关键点是要找出待插新数在有序数组中要插入的位置。首先确定该数是否应插在有序数组的最后。若是，就将该数插入到数组末尾；否则，将该数与数组每个元素依次比较，确定要插入的位置，再想办法空出该位置（从此位置开始依次向后移一个位置），继而将新数插入此处。简言之，找出位置，空出位置，插入进去。

② 思考：为什么程序中定义数组的长度为 11 而不是 10？

5.1.3 练习与思考

1. 单选题

（1）若要定义一个具有 5 个元素的整型数组，以下错误的定义语句是（　　）。

A. `int a[5]={0};`　　B. `int b[]={0,0,0,0,0};`

C. `int c[2+3];`　　D. `int i=5, d[i];`

(2) 若有说明：int a[10];则对 a 数组元素的正确引用是(　　)。

A. a[10]　　B. a[3.5]　　C. a[5]　　D. a[5.0]

(3) 以下程序运行后的输出结果是(　　)。

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

A. 6　　B. 10　　C. 11　　D. 15

(4) 若有int a[10] = {6,7,8,9,10};则对该说明语句正确的理解是(　　)。

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[0]至 a[4]

D. 因为数组长度与初值的个数不相同,所以此语句不正确

(5) 以下能对一维数组 a 进行正确初始化的语句是(　　)。

A. int a[10] = (0,0,0,0,0);　　B. int a[10] = {};

C. int a[ ] = {0};　　D. int a[10] = {10 * 1};

(6) 以下程序段运行的结果是(　　)。

```
int  a[10] = {1,2,3,4,5,6,7,8,9,10}, i, t;
for(i = 0;i < 5/2;i++)
{
    t = a[i];
    a[i] = a[5 - i - 1];
    a[5 - i - 1] = t;
}
for(i = 2;i < 8;i++)
    printf("%d",a[i]);
```

A. 345678　　B. 876543　　C. 1098765　　D. 321678

(7) 下列选项中,能正确定义数组的语句是(　　)。

A. int a[0..2008];　　B. int a[ ];

C. int N = 2008,int a[N];　　D. #define N 2008
int a[N];

(8) 若有int a[10];能给数组 a 的所有元素分别赋值为 1,2,3,…,10 的语句是(　　)。

A. for(i = 1;i < 11;i++)a[i] = i;　　B. for(i = 1;i < 11;i++)a[i - 1] = i;

C. for(i = 1;i < 11;i++)a[i + 1] = i;　　D. for(i = 1;i < 11;i++)a[0] = 1;

(9) 下面程序段的输出结果是(　　)。

```
int a[ ] = {2,3,5,4},i;
for(i = 0;i < 4;i++)
switch(i % 2)
```

```
{    case 0:switch(a[i]%2)
          {   case 0:a[i]++;break;
              case 1:a[i]--;
          }
          break;
     case 1:a[i]=0;
}
for(i=0;i<4;i++)
     printf("%d",a[i]);
```

A. 3344　　B. 2050　　C. 3040　　D. 0304

(10) 以下对一维数组赋初值,正确的是(　　)。

A. int arr[5]; arr={1,2,3,4,5};　　B. int arr[5]={1,2,3,4,5,6};

C. int arr[5]={0,0,2};　　D. int arr[5]=0,1,2,3,4,5;

2. 填空题

(1) 数组是一批具有_____数据类型的变量的集合。

(2) C 语言程序在执行过程中,不检查数组下标是否_____。

(3) 若有下列数组说明 int a[12],则数组元素最大和最小的下标分别是_____和_____。

(4) 假设 int 类型变量占用两字节,若定义 int a[15],则数组 a 占用的内存字节数是_____。

(5) 设int b[ ]={1,2,3,4},y,i=0;则执行语句y=a[i]++;之后,变量 y 的值为_____。

(6) C 语言的数组名是一个_____常量,不可以对它进行加、减和赋值等运算。

(7) 数组在内存中占据一片连续的存储空间,由_____代表它的首地址。

(8) 下面程序段的输出结果是_____。

```
int a[]={0,0,0,0,0,0},i;
for(i=1;i<=4;i++)
{
    a[i]=a[i-1]*2+1;
    printf("%d ",a[i]);
}
```

(9) 以下程序段可以求出所有水仙花数。(所谓水仙花数是指一个 3 位数正整数,其各位数字的立方和等于该正整数。)

```
int x,y,z,a[8],m,i=0;
for(_____;m++)
{
    x=m/100;
    y=_____;
    z=m%10;
    if(x*100+y*10+z==x*x*x+y*y*y+z*z*z)
    {     _____;        i++;        }
```

```
}
for(x = 0;x < i;x++)  printf(" % 6d", a[x]);
```

(10) 设数组 a 中的元素均为正整数,以下程序段是求 a 中偶数的个数和偶数的平均值,请填空。

```
int a[10] = {1,2,3,4,5,6,7,8,9,10},i,k,s;
float ave;
for(k = s = i = 0;i < 10;i++)
 {
    if(a[i] % 2!= 0)________;
        s += ________;
    k++;
 }
if(k!= 0)
{
    ave = (float)s/k;
    printf(" % d, % f\n",k,ave);
}
```

5.1.4　综合应用

1. 程序填空

从键盘输入 10 个整数,程序的功能是计算并输出其最大值、最小值及其所在的下标位置。

```
1   # include < stdio. h >
2   int main()
3   {
4       int a[10],i,max,min,maxpos,minpos;
5       for(i = 0;i < 10;i++)
6           scanf(" % d",&a[i]);
7       max = min = a[0];
8       / ********** FILL ********** /
9       maxpos = minpos = ________;
10      for(i = 1;i < 10;i++)
11      {
12          / ********** FILL ********** /
13          if(________)
14          {
15              max = a[i];
16              / ********** FILL ********** /
17              maxpos = ________;
18          }
19          else
20              / ********** FILL ********** /
21              if(________)
```

```
22              {
23                  min = a[i];
24                  /**************** FILL ****************/
25                  minpos = ________;
26              }
27      }
28      printf("max = %d,pos = %d\n",max,maxpos);
29      printf("min = %d,pos = %d\n",min,minpos);
30      return 0;
31  }
```

【实验提示】

(1) 一组数中找出最大值或最小值,即求最值,常采用“打擂台”的方法,程序中的变量max代表存放最大值的“擂台”,min表示存放最小值的“擂台”。程序中第7行“擂台”上先存放一个数,然后通过循环,数组中的每一个数依次与“擂台”上的数比较,如果比它大(小),就替代它成为“擂主”,同时,记下它所在的下标。比较完毕,“擂台”上的“擂主”就是所求的最大(小)数。

(2) 程序中,maxpos、minpos存放最大值、最小值元素的下标。

2. 程序改错一

下面给定的程序中,fun函数的功能是将十进制正整数m转换成k进制数(2≤k≤9),并按位输出。例如,若输入8和2,则应输出1000(即十进制8转换成二进制是1000)。请改正fun函数中的错误。

注意:不要改动main函数和其他函数中的任何内容,不得增行或删行,不得更改程序的结构。

```
1   #include <stdio.h>
2   void fun(int m,int k)
3   {
4       int aa[20],i;
5       for(i = 0;m;i++)
6       {
7           /**************** ERROR ****************/
8           aa[i] = m/k;
9           /**************** ERROR ****************/
10          m% = k;
11      }
12      for(;i;i--)
13          /**************** ERROR ****************/
14          printf("%d",aa[i]);
15  }
16  int main()
17  {
18      int b,n;
19      printf("\n please enter a number and a base:\n");
```

```
    scanf("%d,%d",&n,&b);
    fun(n,b);
    printf("\n");
    return 0;
}
```

【实验提示】

(1) 将十进制正整数转换成其他进制的数与十进制正整数转换成二进制数的方法是一样的。即把十进制整数 m 转换为 k 进制,采用 m 除以 k 取余数的方法来完成。

(2) 程序第 5～11 行,采用循环的方式用 m 除以 k 取余数,将所得余数放入数组,再用所得的商作为新的 m,再除以 k 取余数,如此重复,直到 m 变成 0 为止。

(3) 将存入数组中的余数,从后向前输出,即为转换成的 k 进制数,程序第 11～15 行即是。

(4) 注意,取余运算和整除运算的区别。

3. 程序改错二

功能:某个公司采用加密形式传递数据,数据是 4 位的整数,加密规则如下:每位数字都加上 5,然后用除以 10 的余数代替该位数字。再将新生成数据的第一位和第四位交换,第二位和第三位交换。

例如,输入一个 4 位整数 1234,则结果为 9876。

```
#include <stdio.h>
int main()
{
    int a,i,aa[4],t;
    printf("Input a 4-digit integer: ");
    /**************** ERROR ****************/
    scanf("%d", a);
    aa[0] = a%10;
    /**************** ERROR ****************/
    aa[1] = a%100%10;
    aa[2] = a%1000/100;
    aa[3] = a/1000;
    /**************** ERROR ****************/
    for(i=0;i<3;i++)
    {
        aa[i] += 5;
        aa[i] %= 10;
    }
    for(i=0;i<=3/2;i++)
    {
       t = aa[i];
       aa[i] = aa[3-i];
       aa[3-i] = t;
    }
```

```
    for(i=3;i>=0;i--)
        printf("%d",aa[i]);
    return 0;
}
```

【实验提示】

(1) 注意输入 scanf 函数的使用格式。

(2) 设计一个数组,存放该 4 位整数分离出的 4 个数位(个、十、百、千位)。

(3) 分离一个长整型数的各个数位的方法是这个题的关键所在,比如:整数 a 的个位分离方法是 a%10;将 a 缩小为原来的 1/10,即 a=a/10 后,a 的十位即变为个位,由此,一个长整型数的各个数位都可分离出来。

4. 程序设计一

输入某班级某门课程的成绩(最多不超过 40 人,具体人数由用户从键盘输入),编程统计不及格的人数。

【实验提示】

(1) 设计一个一维数组 score[40]用于存放学生的成绩。具体存放多少个学生成绩由用户自定(不超过 40)。代码如下:

```
int score[40],n,i,x=0;
scanf("%d",&n);
```

(2) 通过循环输入 n 个学生成绩。

(3) 循环遍历该数组,数组中每个学生成绩与及格分 60 进行比较,小于 60 则计数器 x 累计。

5. 程序设计二

兔子繁殖问题。假设一对兔子的成熟期是一个月,即一个月可长成成兔,如果每对成兔每个月都可以生一对小兔,一对新生的小兔从第二个月起就开始生兔子。试问从一对兔子开始繁殖,假如兔子都不死,一年以后可以有多少对兔子?试编程求解。

【实验提示】

(1) 依题意,从兔子的繁殖可以发现规律:①每月小兔的对数=上个月成兔的对数;②每月成兔的对数=上个月成兔的对数+上个月小兔的对数;综合①、②可以得出:每月成兔对数=前两个月成兔对数之和。即每个月兔子的总对数为:1,2,3,5,8,13,21,34,…,这就是著名的 Fibonacci 数列。

(2) 定义数组:int f[12]={1,2};存放每个月的兔子数。

(3) 由(1)知:f[2]=f[0]+f[1],f[3]=f[1]+f[2],以此类推,下面循环即可求出每个月的兔子数量:

```
for(i=2; i<12; i++)
    f[i]=f[i-1]+f[i-2];
```

5.2　二维数组

如果说一维数组在逻辑上可以想象成一行长表，那么二维数组在逻辑上可以想象成由若干行、若干列组成的表格或矩阵，二维数组常称为矩阵。把二维数组写成行和列的排列形式，可以有助于形象化的理解二维数组的逻辑结构。

5.2.1　知识点介绍

1. 二维数组的定义

当数组中的每个元素带有两个下标时，称这样的数组为二维数组。二维数组也必须“先定义，后使用”。在C语言中，二维数组的定义形式见表5-3。

表5-3　二维数组的定义形式及说明

类型说明符　数组名[整型常量表达式1][整型常量表达式2]；		示例：int　a[3][4]；
类型说明符	数组中所有元素的类型，如int、float、char等	int为类型说明符
数组名	遵循C语言标识符命名规则	a为数组名
常量表达式1	二维数组的行数，只能是整型常量或整型表达式	表示二维数组a有3行
常量表达式2	二维数组的列数，只能是整型常量或整型表达式	表示二维数组a有4列
数组元素个数	常量表达式1×常量表达式2，表示二维数组元素个数	数组a共有3×4即12个元素

如果有二维数组a[M][N]，则数组a含有M×N个元素，a[i][j]表示第i行第j列的元素，其中i的取值范围是0～M－1，列下标j的取值范围是0～N－1。

2. 二维数组的引用

二维数组的引用和一维数组的引用一样，也可以把二维数组中的每个元素当成普通的变量来使用，引用二维数组时必须带有两个下标。引用形式如下：**数组名[行下标][列下标]；**

行下标和列下标也可以是整型表达式，如a[2－1][2＊2]、a[i][j]、a[w＋v][i＋k]等都是合法的数组元素引用。需要注意的是，在引用二维数组元素时，每个下标的值都必须是**整型数据**，且应在已定义数组大小的取值范围内，不得超越数组定义中的上、下界；两个下标应该分别在两个方括号内。例如，a[1,2]和a[i,i]都是不合法的。

3. 二维数组元素初始化

对二维数组的初始化，即在定义二维数组的同时，给数组各元素赋值。可以有以下4种方式，见表5-4。

表 5-4 二维数组的初始化

形 式	示 例	说 明
分行对二维数组初始化	int a[3][3] = {{1,2,3},{4,5,6},{7,8,9}};	分行将值依次赋给每行各元素，如第一行：a[0][0] = 1 ,a[0][1] = 2,a[0][2] = 3
按二维数组在内存中的排列顺序给各元素赋值	int a[3][3] = {1,2,3,4,5,6,7,8,9};	二维数组在内存中是按行存储的，按顺序依次将值赋给各元素
所赋初值行数少于数组行数且各行部分赋值	int a[3][3] = {{1,2},{4,5}};	系统将自动给后面各行的元素赋初值 0
所有元素赋初值时，可以省略行下标，但不能省略列下标	int a[ ][3] = {{1,2,3},{4,5,6},{7,8,9}};	此时，只能省略第一维下标，第二维下标不能省，系统会根据所赋值的个数及二维下标数算出二维数组的行数

除了定义二维数组时对二维数组初始化赋初值外，程序中也可通过循环嵌套语句对数组元素逐个动态赋值。如下面程序段就可以给数组 a 所定义的 15 个元素输入值。

```
float a[3][5];
int i,j;
for(i = 0;i < 3;i++)
    for(j = 0;j < 5;j++)
        scanf(" % d",&a[i][j]);
```

运行程序时，用户从键盘输入数据，数据之间可以空格隔开，按行输入，如图 5-2 所示，也可在一行中输入，但不提倡在一行中输入数据。

```
12  34  56  70  54
65  14  59  90  87
67  85  44  98  36
```

图 5-2 二维数组数据输入示例

4. 二维数组的存储

在逻辑上可以把二维数组看成是有行有列的二维表格，在概念上是二维的，其下标在两个方向上变化，下标变量在数组中的位置也处于一个平面之中。但实际的硬件存储器却是连续编址的，存储器单元是一维的。那么，如何在一维存储器中存放二维数组呢？可以用把二维数组看作两个一维数组的叠加的方法来实现。

上例中数组 a 可看作由 3 个一维数组{a[0],a[1],a[2]}组成，其中每个元素又可看作是一个一维数组 a[i]={a[i][0],a[i][1],a[i][3],a[i][4]}(i=0,1,2)；然后按一维数组的存放方式依次对数组元素进行存放，即先存放数组 a[0]的元素，再存放 a[1]的元素，以此类推。由此可见，在 C 语言中，二维数组元素在内存排列的顺序是**按行存放**，即放完第一行之后，按顺序放入第二行，以此类推。

5. 多维数组

C 语言允许使用多维数组，有了二维数组的基础，再由二维数组推广到多维数组就相对容易了。

5.2.2　实验部分

1. 实验目的

(1) 熟练掌握二维数组的定义、二维数组元素的引用、赋值以及输入输出的方法。
(2) 比较二维数组与一维数组在操作上的异同。
(3) 熟练掌握用二维数组解决矩阵、行列式、二维表及平面图等问题的算法。
(4) 掌握二维数组以及多维数组的使用。

2. 实验内容

(1) 程序填空一

给定程序中，fac 函数的功能是计算 $N \times N$ 矩阵的主对角线元素与反向对角线元素的和，并作为函数值返回，请填空。

```
1  #include <stdio.h>
2  #define  N  4
3   int fac(int t[ ][N],int n)
4  {
5      int i,sum;
6      /**************** FILL ****************/
7      ________;
8      for(i=0;i<n;i++)
9          /**************** FILL ****************/
10         sum += ________ ;
11     for(i=0;i<n;i++)
12         /**************** FILL ****************/
13         sum += t[i][n-i- ________ ];
14     return sum;
15 }
16 int main()
17 {
18     int i,j,t[ ][N]={1,2,3,4,5,6,7,8,9,1,2,3,4,5,6,7};
19     printf("\nThe original data:\n");
20     for(i=0;i<N;i++)
21     {
22        for(j=0;j<N;j++)
23        printf(" %4d",t[i][j]);
24        printf("\n");
25     }
26     printf("The result is %d",fac(t,N));
27     return 0;
28 }
```

【实验提示】

① 该题关键在于找出主对角线元素行下标和列下标的关系以及反向对角线元素的行下标和列下标的关系。程序中第 10 行循环累加主对角元素，第 13 行循环累加反向对角

元素。

② main 函数给数组输入值，并调用 fac 子函数。程序第 25 行输出调用子函数后的结果，fac(t,N)实参 t 是数组名，意为将数组 t 的首地址传递给形参数组 t，如此，子函数中对形参数组的操作本质是对实参数组 t 的操作，实参 N 是方阵的行数(列数)。

③ 牢固掌握 $N\times N$ 矩阵主对角线和反向对角线元素的下标特征。

(2) 程序填空二

下列给定程序中，fun 函数的功能是求出一个 3×4 矩阵中值最大的那个元素的值，及其所在的行号和列号。main 函数中输入数组的值，请填空。

```
#include <stdio.h>
void fun(int a[3][4])
{
    int i,j,row=0,colum=0,max;
    /**************** FILL ****************/
    ________ ;
    /**************** FILL ****************/
    for(i=0; ________ ;i++)
        for(j=0;j<=3;j++)
            if(a[i][j]>max)
            {
                /**************** FILL ****************/
                ________ ;
                row=i;
                colum=j;
            }
   printf("max=%d,row=%d,colum=%d\n",max,row,colum);
}
 int main()
 {
    int a[3][4],i,j;
    printf("Please input array:\n");
     for(i=0;i<=2;i++)
         for(j=0;j<=3;j++)
            scanf("%d",&a[i][j]);
     fun(a);
    return 0;
}
```

【实验提示】

① 找出一组数中最大值或最小值，即求最值，常采用“打擂台”的方法。

② 第 6 行，“擂台”上先存放一个数，第 8～16 行通过循环依次打擂求最值，即数组中每一个数都和“擂台”上的数比，如果比它大(小)，就替代它，最后，“擂主”就是所求的最大(小)数。

③ 如果想求每行的最大值及其所在的列号，该如何修改程序？

(3) 程序改错一

下列给定程序中，yahui 函数的功能是按以下形式输出杨辉三角形(要求输出 10 行)，

请修改程序中的错误。

注意：不要改动 main 函数，不得增行或删行，不得更改程序的结构。

```
1
1  1
1  2  1
1  3  3  1
1  4  6  4  1
```

```
#include <stdio.h>
#define N 11
void yahui(int a[][N])
{
    int i,j;
    for(i=1;i<N;i++)
    {
       a[i][1]=1;
       a[i][i]=1;
    }
    /**************** ERROR ****************/
    for(i=1;i<N;i++)
         /**************** ERROR ****************/
         for(j=2;j<=i;j++)
            a[i][j]=a[i-1][j-1]+a[i-1][j];
    for(i=1;i<N;i++)
    {
        /**************** ERROR ****************/
        for(j=1;j<N;j++)
            printf("%6d",a[i][j]);
        printf("\n");
     }
}
int main()
{
     int a[N][N];
     yahui(a);
     return 0;
}
```

【实验提示】

① 题目中定义的二维数组是 11×11 的，该二维数组存放 10 行杨辉三角形数据是从第 1 行第 1 列开始的，所以定义的二维数组中第 0 行第 0 列元素没用。

② 杨辉三角形各行是$(a+b)^n$展开后各项的系数($n=0,1,2,\cdots$)。

③ 规律是各行第一个数都是 1，各行最后一个数都是 1；除 1 外的其余各数是上一行同一列和上一行的前一列两个数之和。程序第 12～15 行是在求杨辉三角形中除 1 外的其余各数，而杨辉三角形中除 1 外的数是从第三行开始才出现，所以控制外循环的 i 应从 3 开始；同理，除 1 外的数从第 2 列开始出现，但是每行有几列这样的数，是程序第 14 行改错的

关键。

④ yahui 函数求出并输出杨辉三角形，由 main 函数调用。

⑤ 输出杨辉三角形实际上是输出 10×10 方阵以主对角线为界的左下三角元素，注意内循环列标的变化。

⑥ 掌握 $N \times N$ 矩阵或行列式的下三角形的输出控制方法。

（4）程序改错二

在二维数组中存放一个 m 行 m 列的表格($2 \leqslant m \leqslant 9$)，例如：若输入 2，则输出：

```
1  2
2  4
```

如果输入 4，则应输出：

```
1  2  3   4
2  4  6   8
3  6  9   12
4  8  12  16
```

注意：不要改动 main 函数，不得增行或删行，不得更改程序的结构。

```
1  #include <stdio.h>
2  #define  M  10
3  int  a[M][M] = {0};
4  int main()
5  {
6      int i,j,m;
7      printf("ENTER m: ");
8      scanf("%d",&m);
9      for(i = 0;i < M;i++)
10       for(j = 0;j < M;j++)
11         /**************** ERROR ****************/
12         a[i][j] = i * j;
13     /**************** ERROR ****************/
14     for(i = 0;i < M;i++)
15     {
16         /**************** ERROR ****************/
17         for (j = 0;j < M;j++)
18             printf("%4d",a[i][j]);
19         printf("\n");
20     }
21     return 0;
22 }
```

【实验提示】

① int a[M][M]={0}；作用将数组 a 的所有元素赋初值为 0。

② 由题意知，数组元素的值为下一行与下一列的乘积，所以程序第 12 行中的错显而易见。

③ 最后输出的二维表格行、列数，依赖于主调函数输入的 m，所以程序第 14 行、第 17 行存在错误。

（5）程序设计一

有 3 名学生，每名学生有 4 门课成绩如表 5-5 所示，编程计算出每名学生的平均分。

表 5-5　学生成绩表

语　文	数　学	英　语	理　综
76	89	87	80
80	88	67	77
66	78	80	90

【实验提示】

① 每名同学的平均分等于每名学生的各科总分除以科目数，注意二维数组中行下标和列下标所代表的含义。

② 为了方便起见，可以定义一个 3×5 的二维数组，每行的最后一个元素专门用于存放每个学生的平均分。采用双循环把 3 名学生 4 门课成绩输入数组。

③ 使用双重循环，外循环控制 3 名学生，内循环控制学生的 4 门课累加，求出每名学生的平均分放入每行的最后一个元素中。

④ 输出 3×5 的二维数组元素，注意换行。

⑤ 如果求各科的平均分，该如何修改程序？

（6）程序设计二

下列程序定义了一个 $N \times N$ 的数组，并在 main 函数中赋值。试编写 fun 函数，功能是将数组左下三角中元素的值全部置 0。

例如：数组 a 中的值为：
1　9　7　3
2　4　6　8
5　6　7　4

则返回 main 函数后数组 a 中的值变为：
0　9　7　3
0　0　6　8
0　0　0　0

部分源程序已给出，如下所示。请勿改动 main 函数和其他函数中的任何内容，仅在 fun 函数的花括号中填入编写的语句。

```
#include <stdio.h>
 #define N 4
 void fun(int a[][N])
{
    int i,j;
    /****************** BEGIN ********************/


    /******************* END *********************/
}
int main()
{
```

```
    int a[][N] = {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16};
    int i,j;
    fun(a);
    for(i = 0;i < = N - 1;i++)
    {
        for(j = 0;j < = N - 1;j++)
            printf(" % 4d",a[i][j]);
        printf("\n");
    }
 return 0;
}
```

【实验提示】

① 控制 $N \times N$ 数组中左下三角元素的算法，实际上是下面千篇一律的循环语句，注意内循环的变化。

```
for(i = 0;i < N;i++)
   for(j = 0;j < = i;j++)
     …
```

② 一个 $N \times N$ 方阵，如果以主对角线为界线，可以将该方阵分为左下三角和右上三角；如果以反向对角线为界线，可将该方阵分为左上三角和右下三角。

③ 如何将一个 4×4 的方阵变为一个单位方阵(主对角线元素为 1，其他元素为 0)。

5.2.3 练习与思考

1. 单选题

(1) 以下定义数组的语句中错误的是(　　)。

A. int num[ ] = {1,2,3,4,5,6};

B. int num[][3] = {{1,2},3,4,5,6};

C. int num[2][4] = {{1,2},{3,4},{5,6}};

D. int num[][4] = {1,2,3,4,5,6};

(2) 以下程序段的输出结果是(　　)。

```
int b[3][3] = {0,1,2,0,1,2,0,1,2}, i, j, t = 1;
for(i = 0;i < 3;i++)
    for(j = 1;j < = 1;j++)
        t += b[i][b[j][i]];
printf(" % d\n",t);
```

A. 1　　B. 4　　C. 3　　D. 9

(3) 设有定义：int a[4][5];，按在内存中的存放顺序，数组 a 的第 11 个元素是(　　)。

A. a[2][1]　　B. a[1][2]　　C. a[2][0]　　D. a[2][2]

(4) 已知int c[3][4];则对数组元素正确引用的是(　　)。

A. c[1][4]　　B. c[1.5][0]

C. c[1 + 0][0]　　D. 以上表达都错误

(5) 以下语句定义正确的是(　　)。

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

B. float a[3][ ] = {{1},{2},{3}};

C. long a[2][3] = {{1},{1,2},{1,2,2},{0,0}};

D. double a[ ][3] = {0};

(6) 若二维数组 a 有 m 列,则在 a[i][j]前的元素个数为(　　)。

A. j×m+i　　B. i×m+j　　C. i×m+j−1　　D. i×m+j+1

(7) 以下定义数组的语句中错误的是(　　)。

A. int x[2][3] = {1,2,3,4,5,6};

B. int x[ ][3] = {0};

C. int x[ ][3] = {{1,2,3},{4,5,6}};

D. int x[2][3] = {{1,2},{3,4},{5,6}};

(8) 二维数组在内存中存放的顺序是(　　)。

A. 按行顺序　　B. 按列顺序

C. 按元素的大小　　D. 按元素被赋值的先后顺序

(9) 以下程序段的输出结果是(　　)。

```
int i, t[ ][3] = {9,8,7,6,5,4,3,2,1};
for(i = 0;i < 3;i++)
    printf(" % d ",t[2 - i][i]);
```

A. 3 5 7　　B. 7 5 3　　C. 3 6 9　　D. 7 5 1

(10) 以下程序段的输出结果是(　　)。

```
int a[3][3] = {1,2,3,4,5,6,7,8,9};
int sum = 0, i, j;
for(i = 0;i < 3;i++)
    for(j = 0;j < 3;j++)
    {
        a[i][j] = i + j;
        if(i == j)sum = sum + a[i][j];
    }
printf(" % d",sum);
```

A. 5　　B. 6　　C. 7　　D. 8

2. 填空题

(1) 若有定义int a[3][4] = {{1,2},{0},{4,6,8,10}};,则 a[1][2]得到的初值是_____,a[2][1]得到的初值是_____。

(2) 若有定义double a[2][2];,则数组 a 的 4 个元素在内存中的存放顺序是_____。

(3) 若有定义int a[3][5];,则数组 a 行下标的下界是_____,列下标的上界是_____。

(4) 以下程序段的输出结果是_____。

```
int a[4][5] = {1,2,4, - 4,5, - 9,3,6, - 3,2,7,8,4};
int i,j,n = 6;
i = n/5;
j = n - i % 5 - 2;
printf(" % d\n",a[i][j]);
```

(5) 在初始化多维数组时,可以省略_____。

(6) 对二维数组元素赋初值int a[3][4] = {5,12,7,4,8,3,9,24,11,2,6,4},则其中数组元素 a[2][2]的值为_____。

(7) 若有定义和语句 int a[3][3] = {{3,5},{8,9},{12,35}},i,sum = 0;for(i = 0;i < 3;i++) sum += a[i][2 - i];则 sum=_____。

(8) 若定义 int a[3][4]={{1},{5},{9}};它的作用是将数组各行第一列的元素赋初值,其余元素值为_____。

5.2.4 综合应用

1. 程序填空一

下面给定的程序中,fun 函数的功能是将 $N \times N$ 矩阵中元素的值按列向右移动 1 个位置,右边被移出矩阵的元素绕回左边第 1 列。

例如,若 $N=3$,有下列矩阵:

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

,则计算后结果为:

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

```
#include <stdio.h>
#define N 4
void fun(int t[N][N])
{
    int i,j,x;
    /**************** FILL ****************/
    for(i = 0;_____;i++)
    {
       /**************** FILL ****************/
        x = t[i][_____];
        for(j = N - 1;j > 0;j -- )
           t[i][j] = t[i][j - 1];
        /**************** FILL ****************/
        t[i][_____] = x;
    }
}
int main()
{
    int i,j,t[][N] = {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16};
    printf("\nThe original array:\n");
```

```
    for(i = 0;i < N;i++)
    {
       for(j = 0;j < N;j++)
            printf(" % 4d",t[i][j]);
        /****************FILL****************/
         _________________;
    }
    fun(t);
    printf("\nThe result is:\n");
    for(i = 0;i < N;i++)
    {
       for(j = 0;j < N;j++)
           printf(" % 4d", t[i][j]);
       printf("\n");
    }
    return 0;
}
```

请在程序的下画线处填入正确的内容，并把下画线删除，使程序得出正确的结果。

注意：不要增行或删行，也不得更改程序结构。

【实验提示】

(1) 每一行的第 1 列右移 1 列的方法是一样的，因此，需要寻求第 1 行第 1 列右移 1 列的方法，继而每行第 1 列的右移方法都相同，共有 N 行循环 N 次即可，程序中第 7 行即是。

(2) 若要右移 1 列，右边势必有被移出矩阵的元素，程序第 10 行先将其暂时存放在变量 x 中，然后通过循环从倒数第二列开始依次后移 1 列，最后将移除矩阵暂存在变量 x 中的元素放入第 1 列。

2. 程序填空二

功能：分别求出 $N \times N$ 矩阵左下和左上两个三角元素的和(含对角线元素)。分别以主对角线和反向对角线为界可以划分出 4 个三角形：左下三角和右上三角，左上三角和右下三角。

```
#include < stdio.h >
#define N 4
int fun1(int a[N][N])                    //求左下三角元素的和
{
    int i,j,s = 0;
    for(i = 0;i < N;i++)
        /****************FILL****************/
        for(j = 0;________;j++)
              /****************FILL****************/
              s = s + ________;
    return s;
}
```

```
int fun2 (int a[N][N])  //求左上三角元素的和
{
    int i,j,s = 0;
    for(i = 0;i < N;i++)
        /*****************FILL*****************/
        for(j = 0;________;j++)
           /*****************FILL*****************/
            s = s + ________;
    return s;
}
int main()
{
    int a[N][N],i,j;
    printf("Input array  a:\n");
    for(i = 0;i < N;i++)
        for(j = 0;j < N;j++)
            /*****************FILL*****************/
            scanf(" % d",________);
    printf("The sum of left - lower triangles is  % d\n",fun1(a));
    printf("The sum of upper left triangles is  % d\n",fun2(a));
    return 0;
}
```

【实验提示】

(1) $N \times N$ 方阵分别以主对角线和反向对角线为界可分出 4 个三角形,即左下和右上、左上和右下共 4 个三角形。

(2) 4 个三角形每个三角形都是 N 行,关键在于列数的变化,读者可以分别写出各三角形元素的行下标和列下标,以此找出每个三角形元素列标的变化特征,从而找出控制列的内循环的循环规律。

(3) 请自行写出右上和右下三角元素的和。

3. 程序改错

功能:实现 3 行 3 列矩阵的转置,即行列互换。

```
#include < stdio.h >
void fun(int a[3][3], int n)
{
    int i,j,t;
    for(i = 0;i < n;i++)
      for(j = 0;j < n;j++)
        /*****************ERROR*****************/
        scanf(" % d", a[i][j]);
    printf("\nThe original array is:\n");
    for(i = 0;i < n;i++)
    {
        for(j = 0;j < n;j++)
```

```
13          printf("%4d",a[i][j]);
14          printf("\n");
15      }
16      for(i=0;i<n;i++)
17         /**************** ERROR ****************/
18         for(j=0;j<n;j++)
19         {
20            /**************** ERROR ****************/
21            a[i][j]=t;
22            a[i][j]=a[j][i];
23            /**************** ERROR ****************/
24            t=a[j][i];
25         }
26       printf("\nThe result is:\n");
27       for(i=0;i<n;i++)
28       {
29          for(j=0;j<n;j++)
30          printf("%4d",a[i][j]);
31          printf("\n");
32        }
33  }
34   int main()
35   {
36     int b[3][3];
37     fun(b,3);
38     return 0;
39   }
```

【实验提示】

(1) 程序 5～8 行,采用双重循环实现二维数组的输入,第 8 行的 scanf 函数输入,错误显而易见。

(2) 将 3×3 矩阵转置,本质上就是以主对角线为对称轴,将左下三角与右上三角对应元素互换。

(3) 借助于变量 t 完成 a[i][j]和 a[j][i]的互换。

4. 程序填空三

给定程序中,fun 函数的功能是在 3×4 的矩阵中找出鞍点,即该位置上的元素在该行上最大,在该列上最小;若没有鞍点,则输出相应信息。

有下列矩阵
```
1   2   13   4
7   8   10   6
3   5    9   7
```

程序执行结果为:find: a[2][2]=9

注意:不得增行或删行,不得更改程序的结构。

```
#include <stdio.h>
#define M 3
#define N 4
void fun(int a[M][N])
{
    int i=0,j,find=0,rmax,c,k;
    while((i<M)&&(!find))
    {
        rmax=a[i][0];
        c=0;
        for(j=1;j<N;j++)
           if(a[i][j]>rmax)
           {
              rmax=a[i][j];
              /**************** FILL ****************/
             c=________;
            }
        find=1;
        k=0;
        while(k<M&&find)
        {
           if(k!=i&&a[k][c]<=rmax)
           /**************** FILL ****************/
           find=________;
           k++;
        }
        if((find))
           printf("find:a[%d][%d]=%d\n",i,c,a[i][c]);
        /**************** FILL ****************/
        ________;
    }
    if(!find)
       printf("not found\n");
}
int main()
{
    int x[M][N],i,j;
    printf("Input array x:\n");
    for(i=0;i<M;i++)
        for(j=0;j<N;j++)
           /**************** FILL ****************/
           scanf("%d",________);
    fun(x);
    return 0;
}
```

【实验提示】

(1) 题目要求找出矩阵的鞍点(也可能没有鞍点),即找出在该行上最大,在该列上最小的元素及其所在的行号和列号。该功能由 fun 子函数完成。

(2) 程序第 7～17 行循环语句,用"打擂法"找出行上最大的元素,记下其列号,第 18～26 行判断其是否是该列上最小的元素。变量 find 作为鞍点是否存在的标志。

(3) main 函数用双重循环输入矩阵的值,调用 fun 子函数。

5. 程序设计一

请编写 fun 函数,其功能是将 M 行 N 列的二维数组中的数据,按列的顺序依次放到一维数组中,一维数组中数据的个数作为函数的值返回。

若二维数组中的数据为：
$$\begin{matrix} 33 & 33 & 33 & 33 \\ 44 & 44 & 44 & 44 \\ 55 & 55 & 55 & 55 \end{matrix}$$

则一维数组中的内容应是：33 44 55 33 44 55 33 44 55 33 44 55。

注意：部分源程序如下所示,请勿改动 main 函数和其他函数中的任何内容,仅在 fun 函数的花括号中填入编写的语句。

```
#include <stdio.h>
int fun(int s[][10], int b[], int mm, int nn)
{
  int n=0,i,j;
  /****************** BEGIN ********************/


  /******************* END *********************/
  return n;
}
int main()
{
  int w[10][10]={{33,33,33,33},{44,44,44,44},{55,55,55,55}};
  int a[100]={0}, i, j, n;
  n=fun(w,a,3,4);
  printf("The A array:\n");
  for(i=0;i<n;i++)
     printf("%3d",a[i]);
  printf("\n");
  return 0;
}
```

【实验提示】

(1) 给出的 fun 函数首部中共有 4 个形参,其中形参 mm 代表二维数组的行数,形参 nn 表示列数。

(2) 题目要求实现将二维数组元素存入一维数组,需使用循环语句来控制二维数组元素的下标。

用双重循环嵌套处理，题目要求按二维数组列的顺序取出，所以外循环控制列下标，内循环控制行下标，每次循环将数组 s 中的一个元素取出放入数组 b 中，程序第 4 行定义的变量 n 可充当一维数组 b 的下标，恰好也是作为函数返回值的数组 b 中元素的个数。

6. 程序设计二

请编写 fun 函数，该函数的功能是求出二维数组周边元素的和，作为函数的值返回。二维数组中的值由 main 函数输入。

注意：部分源程序已给出，请勿改动 main 函数和其他函数中的任何内容，仅在 fun 函数的花括号中填入编写的语句。

```
#include <stdio.h>
#define M 4
#define N 5
int fun(int a[M][N])
{
  int i,j,s = 0;
  /****************** BEGIN ********************/


  /******************* END *********************/
  return s;
}
int main()
{
  int aa[M][N],i,j,y;
  for(i = 0;i < M;i++)
      for(j = 0;j < N;j++)
          scanf("%d", &aa[i][j]);
   y = fun(aa);
  printf("The sum is: %d\n", y);
  return 0;
}
```

【实验提示】

(1) 该题难度不大。要想求出 $M \times N$ 二维数组周边元素的和，可以先在草稿纸上写出周边元素的行列下标，观察规律，然后循环累加即可。

(2) 思考：如果求周边元素的平均值，该如何修改程序？

5.3 字符数组

用来存放字符数据的数组是字符数组。C 语言没有提供字符串数据类型，也没有字符串变量，因此，字符串的存取、处理等操作要用字符型数组来实现。在字符数组中，一个元素存放一个字符。

5.3.1　知识点介绍

因为C语言中没有提供字符串变量，字符串的存储完全依赖于字符数组，但字符数组又不等于字符串变量。字符数组是用来存放字符的数组，字符数组的一个元素就是一个字符。字符数组的定义与前面介绍的一维数组和二维数组的定义形式相同。不同的是，定义一个字符数组时，数组名前的类型说明符应该是char。

1. 字符串和字符串结束标志

在C语言中，没有字符串数据类型，但却允许使用字符串常量。字符串常量是由双引号括起来的一串字符。字符串是借助于字符数组存放的，并规定以字符'\0'作为字符串结束标识。在表示字符串常量时，不需要人为地在其末尾加入'\0'，C语言编译程序将自动地完成这一项工作，在末尾加'\0'。如：字符串“How are you!”等价于字符数组{'H','o','w',' ','a','r','e',' ','y','o','u','!','\0'}，但字符数组{'H','o','w',' ','a','r','e',' ','y','o','u','!'}不能等价地看作字符串"How are you! "，因为字符数组中缺少字符串结束标识'\0'。

字符数组在包含字符串结束标识的情况下，可以使用字符串处理函数，对其进行操作。

2. 字符数组的初始化

对字符数组的初始化有两种方法实现。

(1) 以单个字符的形式给每个数组元素赋值。例如，char c[5]={'c', 'h', 'i', 'n', 'a'};或者也可以是 char c[ ]={'c', 'h', 'i', 'n', 'a'};。

(2) 以字符串常量的形式给数组赋值。例如，char c[6]="china";或者：char c[ ]="china";。

注意：在程序执行过程中，不可以用赋值语句给字符数组整体赋一串字符。

例如：char m[10];

```
m="C program!";//赋值不合法
```

或char s1[10]="computer",s2[10];

```
s2=s1;  //赋值不合法
```

3. 字符数组的输入输出

C语言提供了两种格式符%c和%s进行字符数组的输入和输出。其中，格式符%c用来逐个字符地进行输入和输出，而格式符%s用来进行整串地输入和输出。在实际应用中，对字符串的输出往往不是根据数组定义的长度来判断字符串是否结束，而是检测到第一个'\0'的时候结束。字符串中实际含有的有效字符的个数可以通过循环语句检测。如：执行了下列程序段后，变量i中存放的是字符串string中所含的有效字符的个数。

```
char string[ ] = "China is a beautiful country."
int  i;
for(i = 0; string[i]!= '\0'; i++);
```

4. 字符串处理函数

C语言提供了一些用于字符串处理的库函数，常见的字符串处理函数见表5-6。

表5-6 常见的字符串函数

函数名	函数原型	功能	返回值	包含文件
strcpy	char *strcpy(char *str1,char *str2);	把str2所指的字符串复制到str1中去	返回str1	string.h
strcat	char *strcat(char *str1,char *str2);	把字符串str2连接到str1的后面，str1后面的'\0'被取消	返回str1	string.h
strlen	unsigned int strlen(char *str);	统计字符串str中字符的个数(不包括终止符'\0')	返回字符个数	string.h
strcmp	int strcmp(char *str1,char *str2);	比较两个字符串str1和str2	str1<str2，返回负数 str1=str2，返回0 str1>str2，返回正数	string.h
puts	int puts(char *str);	把str指向的字符串输出到标准的输出设备	返回换行符，若失败，返回EOF	stdio.h
gets	char*gets(char *str);	从标准输入设备读入字符串放入str指向的字符数组中	成功，返回str指针，否则返回NULL指针	stdio.h

5. 二维字符数组

二维字符数组用于同时存储和处理多个字符串，其定义格式与二维数值数组类似，可以看作若干个一维字符数组的叠加。如果以字符串的形式给字符数组赋值，同样每一个一维数组也是以'\0'结束。

5.3.2 实验部分

1. 实验目的

(1) 熟练掌握字符数组的定义，赋值和输入输出的方法。
(2) 熟练掌握用字符数组存储和处理字符串常量的方法。
(3) 熟记常用字符串处理函数，并掌握它们的使用方法。
(4) 掌握字符型数组和数值型数组在处理上的区别和联系。

2. 实验内容

(1) 程序填空一

下列给定的程序中，fun函数的功能是将两个字符串连接起来(不得调用字符串连接strcat函数)形成一个新串，请填空。字符串的输入与输出在main函数中进行。

```
1  #include <stdio.h>
2  #include <string.h>
3  void fun(char s1[],char s2[])
4  {
5      int i=0,j=0;
6      /**************** FILL ****************/
7      while(________)
8          i++;
9      while(s2[j]!='\0')
10     {
11         s1[i++]=s2[j];
12         /**************** FILL ****************/
13         ________;
14     }
15     /**************** FILL ****************/
16     ________;
17 }
18 int main()
19 {
20     char s1[50],s2[20];
21     printf("Input string s1:\n");
22     gets(s1);
23     printf("Input string s2:\n");
24     /**************** FILL ****************/
25     scanf("%s",________);
26     fun(s1,s2);
27     printf("The result is:");
28     puts(s1);
29     return 0;
30 }
```

【实验提示】

① 把第二个字符串连接到第一个字符串的后面，找出第一个字符串的连接位置是关键，程序第 7 行通过 while 循环语句遍历字符串，直到字符串结束，即结束标志'\0'的位置即为第二个串的连接点，程序第 9～14 行将第二个字符串的每个字符依次地循环接入到第一个字符串的后面。

② 程序第 16 行，注意连接后形成新字符串的结束标志'\0'。

③ 字符数组的输入有多种方法，读者应该清楚不同方法的区别和使用方法。

④ 如何将字符数组 s2 中的全部字符复制到字符数组 s1 中，而不调用 strcpy 函数。

（2）程序填空二

请补充 fun 函数，该函数的功能是删除字符数组中小于指定字符的字符，只保留大于指定字符的字符。指定字符从键盘输入，结果仍保存在原数组中。例如：输入 abcdefghij，指定字符为 d，则输出结果为：defghij。

注意：不得增行或删行，不得更改程序的结构。

```
#include <stdio.h>
#include <string.h>
#define N 80
void fun(char s[],char ch)
{
    int i = 0,j = 0;
    while(s[i]!= '\0')
    {
        if(s[i]< ch)
            /**************** FILL ****************/
            ________;
        else
        {
           /**************** FILL ****************/
           ________ = s[i];
           i++;
        }
    }
    /**************** FILL ****************/
    ________;
}
int main()
{
   char str[N],ch;
   printf("\nInput a string:\n");
   gets(str);
   printf("\nInput a character:\n");
   scanf("%c",&ch);
   fun(str,ch);
   printf("\nThe result is:\n");
   puts(str);
   return 0;
}
```

【实验提示】

① 程序第 7～18 行，对存放字符串的数组 s 进行循环遍历，每个字符都与指定字符进行比较，如果比指定字符小，访问下一个字符；如果大于指定字符，则进行保留，将它重新保存在数组 s 中，注意重新存放在数组 s 中时的下标。

② 通过保留比指定字符大的或相等的字符，实现删除比指定字符小的字符的目的。处理结束后，第 20 行，注意在新字符串的后面加上结束标志'\0'。

③ C 语言中没有直接删除字符的算法，做题时需注意方法。

④ 思考并编程：将一个字符串中 ASCII 码值为偶数的字符删除，只保留 ASCII 码值为奇数的字符。

（3）程序改错一

下面给定程序中，fun 函数的功能是判断用户输入的任意一个字符串是否为回文串。回文串是指从开头读和从末尾读（正读和反读）均为相同字符，例如："HELLEH"。请修改

程序中的错误。

注意：不得增行或删行，不得更改程序的结构。

```
#include <stdio.h>
#define N 50
int fun(char a[])
{
    int i = 0,num = 0,flag = 1;
    do
     {
        num++;
     /**************** ERROR ****************/
     }while(a[i]!= '\0');
     do
     {
        /**************** ERROR ****************/
        if(a[i]!= a[num - i])
        {
           flag = 0;
           break;
        }
        i++;
    /**************** ERROR ****************/
    }while(i < num);
    return(flag);
}
int main()
{
   char a[N];
   int m;
   printf("Input a string:\n");
   /**************** ERROR ****************/
   scanf("%c",a);
   m = fun(a);
   if(m == 1)
       printf("YES\n");
   else
       printf("NO\n");
   return 0;
}
```

【实验提示】

① 首先需清楚题目中各变量所代表的含义：变量 flag 作为判断字符串是否是回文串的标志，当 flag 为 1 时此字符串为回文串，当 flag 为 0 时此字符串不为回文串；变量 num 用来存放字符串的有效长度。

② 程序第 6～10 行 while 循环语句的目的是求出字符串的有效长度，对字符数组的循环处理往往用检测'\0'作为循环的结束条件，但是 i 作为循环条件的元素下标显然不合适。

③ 程序第 11～21 行，判断该字符串是否为回文串，本质上就是循环判断数组元素 a[i]

(i从0开始)与a[num－i－1](num为字符串有效长度)是否相等,i的变化范围只限于字符串的前半部分。

④ 变量flag作为字符串是否是回文串的标记,是flag为1,不是则为0。main函数通过调用fun函数得到返回的flag的值,并通过该值确定输出结果。

(4) 程序改错二

下列给定的程序中,fun函数的功能是判断字符ch是否与字符数组str中字符串的某一个字符相同;若相同,什么都不做,若不同,则将其插在该字符串的最后。

注意:不要改动main函数,不得增行或删行,不得更改程序的结构。

```
1  #include <stdio.h>
2  #include <string.h>
3  void fun(char str[],char ch)
4  {
5      int i=0;
6     /**************** ERROR ****************/
7      while(str[i]!='\0'||str[i]!=ch)
8          i++;
9      /**************** ERROR ****************/
10     if(str[i]==ch)
11     {
12        str[i]=ch;
13        /**************** ERROR ****************/
14        str[i++]='\0';
15     }
16 }
17 int main()
18 {
19   char s[80],c;
20   printf("\nInput a strinr:");
21   gets(s);
22   printf("\nInput a character:");
23   c=getchar();
24   fun(s,c);
25   printf("\nThe result is %s:\n",s);
26   return 0;
27 }
```

【实验提示】

① 程序第7行对字符数组进行循环遍历,只要字符串没有结束且当前字符和字符ch不相同两个条件同时满足时,说明ch和字符串中的字符不同,将字符ch加在字符串的尾部,否则,什么都不做。因此,遍历时while语句的循环条件很重要。

② 程序第12行将与字符串任一字符都不相同的字符ch插入到字符串的最后,代替了'\0',此时注意第14行要重新在插入ch字符后的字符串结尾赋'\0'。

(5) 程序设计一

从键盘任意输入一个字符串,将该字符串中的所有字符按其ASCII码值从小到大排序

后输出。

【实验提示】

① 根据前面所学的排序算法(冒泡排序法),将字符串中各字符的 ASCII 码值依次进行比较排序。

② 利用字符串处理 strlen 函数,求出输入的字符串所包含字符的个数,即排序对象的个数,即可采用冒泡法排序。

(6) 程序设计二

下面给定的程序中,fun 函数的功能是将字符串 s 中的所有数字字符移到所有非数字字符之后,并保持数字字符和非数字字符原有的次序。例如,字符串 s 为 def35abcg4jhdt7。执行结果为 def abcg jhdt3547。

部分源程序已给出,如下所示。请勿改动 main 函数和其他函数中的任何内容,仅在 fun 函数的空白处填入编写的语句。

```
#include <stdio.h>
#include <string.h>
void fun(char s[])
{
  int i,j=0,k=0;
  char t1[80],t2[80];
  for(i=0;s[i]!='\0';i++)
      if(s[i]>='0'&&s[i]<='9')
          /*****************BEGIN*****************/


          /*****************END*****************/
  for(i=0;i<k;i++)
       s[i]=t1[i];
  for(i=0;i<j;i++)
       s[k+i]=t2[i];
}
int main()
{
  char s[80];
  printf("\nInput a string:");
  gets(s);
  fun(s);
  printf("\nThe result is:");
  puts(s);
  return 0;
}
```

【实验提示】

① 将字符串 s 中的数字字符和非数字字符挑出,分别存储在字符数组 t2 和 t1 中。

② 数组 t1 用来存放非数字字符,变量 k 用来统计非数字字符的个数;数组 t2 存放数字字符,变量 j 用来统计数字字符的个数。

③ 最后分别将两个数组连接到数组 s 即可。

5.3.3 练习与思考

1. 单选题

(1) 下面选项中,叙述正确的是(　　)。

A. 在定义字符数组时不进行初始化,数组元素会被赋予默认初值'\0'

B. 可以在省略行下标和列下标的情况下,对二维字符数组进行初始化

C. 在定义字符数组时进行部分初始化,未初始化元素会被赋予默认初值'\0'

D. 用字符串常量初始化字符数组时,数组大小应至少等于字符串有效字符的个数

(2) 下面程序段的输出结果是(　　)。

```
char s[ ] = {"012xy"};
int i,n = 0;
for(i = 0;s[i]!= 0;i++)
    if(s[i]> = 'a'&& s[i]< = 'z')    n++;
printf(" % d\n",n);
```

A. 2　　B. 0　　C. 3　　D. 5

(3) 对两个数组 a 和数组 b 进行如下初始化,则下面选项中叙述正确的是(　　)。

```
char a[ ] = "ABCDEF";
char b[ ] = {'A','B','C','D','E','F'};
```

A. 数组 a 与数组 b 完全相同　　B. 数组 a 与数组 b 长度相同

C. 数组 a 和数组 b 中都存放字符串　　D. 数组 a 比数组 b 的长度长

(4) 下列选项中,不合法的数组定义语句是(　　)。

A. char a[9] = {'s','t','i','r','n','g'};

B. char a = "string";

C. char a[9] = {"string"};

D. char a[9] = "string";

(5) 下面程序段的输出结果是(　　)。

```
char ch[7] = {"12ab56"};
int i,s = 0;
for(i = 0;ch[i]> = '0'&&ch[i]< = '9';i += 2)
    s = 10 * s + ch[i] - '0';
printf(" % d\n",s);
```

A. 1　　B. 12　　C. 12ab56　　D. 1256

(6) 下面程序段的输出结果是(　　)。

```
char a[ ] = "morning", t;
int i, j;
for(i = 0;i < 7;i++)
    if(a[j]< a[i])    j = i;
t = a[j];    a[j] = a[7];    a[7] = t;
puts(a);
```

A. mogninr　　B. mo　　C. morning　　D. moring

(7) 下面程序段的输出结果是(　　)。

```
char a[20] = "ABCD\0EFG\0",b[ ] = "IJK";
strcat(a,b);
printf(" % s\n",a);
```

A. ABCD\0EFG\0IJK　　B. ABCDIJK

C. IJK　　D. EFGIJK

(8) 设有定义和语句char str[ ] = "Sunny";printf(" % 5.3s",str);则输出是(　　)。(注：␣代表一个空格)

A. Sunny　　B. Sun␣␣␣　　C. ␣␣␣nny　　D. ␣␣␣Sun

(9) 以下字符数组初始化语句中,不正确的是(　　)。

A. char c[ ] = 'goodmorning';　　B. char c[20] = "goodmorning";

C. char c[ ] = {'a','b','c','d'};　　D. char c[ ] = {"goodmorning"};

(10) 下面程序段的输出结果是(　　)。

```
char ch[3][5] = {"AAAA","BBB","CC"};
printf(" % s\n",ch[1]);
```

A. AAAA　　B. BBBCC　　C. BBB　　D. CC

2. 填空题

(1) printf(" % d\n",strlen("ATS\n012\1"))的输出结果是_____。

(2) 判断字符串 a 和字符串 b 是否相等,应当使用_____函数。

(3) 若有 char s[ ][20]={"one World!", "one Dream!"};则 strlen(s[1])=_____。

(4) 若有定义 char a[10]="abcd";则 strlen(a)=_____;sizeof(a)=_____。

(5) 有以下程序段,字母 A 的 ASCII 码值是 65,则程序输出后的结果是_____。

```
char s[ ] = "ABC";
int i = 0;
do
{
    printf(" % d",s[i] % 10);
    i++;
} while(s[i]!= '\0');
```

(6) 若有程序段 char x[ ] = "STRING";x[0] = 0;x[1] = '\0';x[2] = '0';则 sizeof(x)=_____, strlen(x)=_____。

(7) 若有以下定义和语句 char s1[ ] = "12345", s2[ ] = "1234"; printf(" % d\n", strlen(strcpy(s1,s2)));则输出结果是_____。

(8) 有以下程序段,若从键盘输入 55566 7777abc 后,y 的值为_____。

```
int j;
float y;
char n[50];
```

```
scanf("%2d%3f%s",&j,&y,n);
```

(9) 下面程序段的输出结果是_____。

```
char a[] = "computer" , t;
int i, j = 0;
for(i = 0;i < 8;i++)
      for(j = i + 1;j < 8;j++)
         if(a[i]< a[j])
         {  t = a[i];   a[i] = a[j];   a[j] = t;  }
printf("%s",a);
```

(10) 若有以下程序段：

```
char p[20] = {'a','b','c','d'},q[] = "abc",r[] = "abcde";
strcat(p,r);
strcpy(p + strlen(q),q);
```

则 strlen(p)＝_____。

5.3.4 综合应用

1. 程序填空一

功能：试在程序的下画线处填入正确的内容，使程序得到如图 5-3 所示的运行结果。

图 5-3 运行结果

```
1  #include <stdio.h>
2  int main()
3  {
4       int i,j;
5       char a[5][5];
6       for(i = 1;i <= 4;i++)
7        for(j = 1;j <= 4;j++)
8           if(j <= i)
9                a[i][j] = '*';
10           else
11               a[i][j] = ' ';          //放入一个空格
12      for(i = 1;i <= 3;i++)
13      {
14           for(j = 1;j <= 4;j++)
```

```
15              /*****************FILL*****************/
16              ________ ;
17          printf("\n");
18      }
19      /*****************FILL*****************/
20      for(________)
21          printf("%c",a[4][j]);
22      printf("\b");    //光标退一格
23      /*****************FILL*****************/
24      for(________)
25          printf("%c",a[4][j]);
26      printf("\n");
27      for(i=3;i>=1;i--)
28      {
29          printf("    "); //输出3个空格
30          /*****************FILL*****************/
31          for(________)
32              printf("%c",a[i][j]);
33          printf("\n");
34      }
35    return 0;
36  }
```

【实验提示】

(1) 程序是将图形分成三部分来考虑,前3行是第一部分,第4行是第二部分,后3行是第三部分。第一部分和第三部分形状类似,不同的是第三部分按第一部分的行和列相反的顺序输出图形。第二部分即第四行的7个'*'又分两次输出。

(2) 程序第6～11行是将字符'*'和空格字符放入字符数组a中应有的位置。程序第12～18行输出图形的第一部分,第20～25行输出图形的第二部分,第27～34行输出图形的第三部分。

(3) 程序中定义的数组是5行5列的二维字符数组,实际上只使用了后4行4列的元素,即没有使用0行与0列的元素。

2. 程序填空二

请补充main函数,该函数的功能是从字符串str中取出所有的数字字符,并分别统计字符0,1,2,3,4,…,9的个数,然后把计数结果保存在数组b中并输出,并把其他字符的个数保存在b[10]中。

```
1   #include<stdio.h>
2   int main()
3   {
4       int i,b[11];
5       char str[]="ab1234456789cde0990";
6       printf("\nThe original array: \n");
7       puts(str);
```

```
    for(i = 0;i < 11;i++)
       b[i] = 0;
    /**************** FILL ****************/
    ________;
    while(str[i]!= '\0')
    {
       /**************** FILL ****************/
        switch(________)
        {
            case '0':b[0]++;break;
            case '1':b[1]++;break;
            case '2':b[2]++;break;
            case '3':b[3]++;break;
            case '4':b[4]++;break;
            case '5':b[5]++;break;
            case '6':b[6]++;break;
            case '7':b[7]++;break;
            case '8':b[8]++;break;
            case '9':b[9]++;break;
             /**************** FILL ****************/
             default:________;
          }
          /**************** FILL ****************/
          ________;
     }
     printf("\n The statistical results:\n");
     for(i = 0;i < 10;i++)
        printf("\n %d: %d",i,b[i]);
     printf("\nThe others : %d\n",b[10]);
     return 0;
}
```

【实验提示】

(1) 对数组 str 中字符串的所有字符进行循环遍历并判断其种类,分别计数即可。注意,switch 语句的执行过程。

(2) 试用多分支的 if 语句替代 switch 语句。

3. 程序改错

下列给定的程序中,fun 函数的功能是从字符串 s 中找出子字符串 t 的个数。例如,当字符串 s 中的内容为 mnuvmnxmn,字符串 t 的内容为 mn,则求出的个数为 3。请改正程序中的错误。注意,不要改动 main 函数,不得增行或删行,不得更改程序的结构。

```
#include <stdio.h>
#include <string.h>
int fun(char s[],char t[])
{
```

```
5       /**************** ERROR ****************/
6       int i = 0,j = 0,n;
7       while(s[i]!= '\0')
8       {
9         while(t[j]!= '\0')
10           /**************** ERROR ****************/
11            if(s[i]!= t[j])
12            {
13                i++;
14                j++;
15            }
16            else
17            {
18                i++;
19                j = 0;
20                /**************** ERROR ****************/
21                continue;
22            }
23            if(t[j] == '\0')
24                n++;
25            j = 0;
26      }
27    return n;
28  }
29  int main()
30  {
31    char s[100],t[100];
32    int m;
33    printf("\nPlease enter string s:");
34    scanf("%s",s);
35    printf("\nPlease enter string t:");
36    scanf("%s",t);
37    m = fun(s,t);
38    printf("\nThe result is :m = %d\n",m);
39    return 0;
40  }
```

【实验提示】

(1) 程序第6行中定义的变量i是数组s的下标,j是数组t的下标,n则是存放字符串t在字符串s中存在的个数,充当计数器。

(2) 从字符串s中找出子字符串t的方法是分别循环遍历字符串s和字符串t,从字符串s的第一个字符开始,若字符串s中的当前字符与字符串t的第一个字符相同,则比较字符串和字符串t的下一个字符(程序第7～15行);若不相同,则字符串s继续下移,而字符串t重回第一个字符,内循环退出(程序第16～20行)。如果一直循环直到字符串t结束,则说明字符串t在字符串s中出现,计数器n累加(程序第22行),字符串t重回第一个字符,再次开始遍历字符串t进行比较。

(3) fun函数中变量n用来累计字符串t在字符串s中存在的个数,通过return语句返

回给 main 函数。

4. 程序设计一

请编写 fun 函数,其功能是将一个数字字符串转换为一个整数(不得调用由 C 语言提供的,将数字字符串转换为对应整数的函数)。例如:若输入字符串－1234,则调用 fun 函数把它转换为整数值－1234。

部分代码给出,如下所示。请勿改动 main 函数和其他函数中的任何内容,仅在 fun 函数的圆括号中填入编写的语句。

```
#include <stdio.h>
#include <string.h>
long fun(char p[ ])
{
    long n = 0;
    int flag = 1,i;
    /**************** BEGIN ****************/


    /**************** END ****************/
    return n;
}
int main()
{
    char s[6];
    long n;
    gets(s);
    n = fun(s);
    printf("%ld\n",n);
    return 0;
}
```

【实验提示】

(1) 该题考察数字字符串转化为对应整数的算法操作。

(2) 考虑符号问题:用 if 语句判断数字字符串的第一个字符是'－'还是'＋'确定转换的整数的正负,用变量 flag 放＋1 或－1 作为系数,调节最终转换成的整数的正负。

(3) 不得调用 C 语言提供的将字符串转换为整数的函数,则数字字符转换为相应的数字需要通过该数字字符减去'0'来实现,即 p[i]－'0'就可以得到与 p[i]这个字符对应的数字。可用下面的循环语句,在数字字符串中完成将每个数字字符转换为相应数字,并且将其组成为整数(不含符号)。

```
while(p[i]!= '\0')
{
   n = n * 10 + p[i] - '0';
   i++;
}
```

(4) 综合符号问题,最后 n=flag * n。

5. 程序设计二

规定输入的字符串中只包含字母和 * 号。编写 fun 函数,其功能是删除字符串中所有的 * 号。编写函数时,不得使用 C 语言提供的字符串函数。例如,字符串中的内容为 **** A * BC ** DEF * G ******,删除后,字符串中的内容应当是 ABCDEFG。

注意:部分代码给出如下。请勿改动 main 函数和其他函数中的任何内容,仅在 fun 函数的圆括号中填入编写的语句。

```
#include <stdio.h>
#include <conio.h>
void fun(char s[])
{
    int i,j=0;
    /**************** BEGIN ****************/


    /**************** END ****************/
}
int main()
{
   char s[81];
   gets(s);
   fun(s);
   printf("The string after deleted:\n");
   puts(s);
   return 0;
}
```

【实验提示】

(1) 欲删除字符串中所有 * 号,需用循环语句遍历字符串的每一个字符,依次判断该字符是否是 * 号。若不是 * 号,则将该元素在原数组中保留;若是 * 号,则继续判断下一个字符。

(2) 从字符串下标为 0 的元素开始循环逐个进行比较,if(s[i]!='*')则保留,注意原字符串的下标和删除 * 号后新字符串的下标的变化。

(3) 保留下来的新字符串最后要加字符串结束标志'\0'。

6. 程序设计三

下面给定的程序中,fun 函数的功能是对 5 个国家名字,按从小到大的顺序排序。从 main 函数中输入 5 个国家的名字,排序后的结果也在 main 函数中输出。

部分代码已给出,如下所示。请勿改动 main 函数和其他函数中的任何内容,仅在 fun 函数的空白处填入编写的语句。

```
#include <stdio.h>
#include <string.h>
int main()
{
    void fun(char name[][20]);
    char name[5][20];
    int i;
    printf("\n Input five  names:\n");
    for(i=0;i<5;i++)
        gets(name[i]);
    fun(name);
    printf("The result is:\n");
    for(i=0;i<5;i++)
    {
        puts(name[i]);
        putchar('\n');
    }
    return 0;
}
void fun(char name[5][20])
{
    int i, j;
    char temp[20];
    /********** BEGIN **********/


    /********** END **********/
}
```

【实验提示】

(1) 可以将第 5 行的二维字符数组看成是 5 个一维字符数组,每个一维字符数组存放一个国家名,共 5 个国家名,即 5 个字符串。对 5 个字符串排序,用冒泡排序法排序即可。程序第 22 行定义的数组 temp 作为排序时字符串交换用的备用数组。

(2) 使用字符串比较函数实现字符串的比较,使用字符串复制函数实现字符串的位置互换。

5.4 实践拓展

5.4.1 新个人所得税计算方法与实现

个人所得税与我们每个人的利益息息相关。2019 年 1 月 1 日新的《中华人民共和国个人所得税法》正式实施,个人需要缴纳的个人所得税也发生了变化,起征点从之前的 3500 元上调至 5000 元,还可以享受专享附加扣除以及社保扣除,工资超过 5000 元才需要纳税,个人综合所得税按全年累计计算。新个人所得税税率表见表 5-7。

表 5-7　新个人所得税税率表

级　数	全年应纳税所得	税率/%	速算扣除数/元
1	不超过 36 000 元的部分	3	0
2	超过 36 000 元至 144 000 元的部分	10	2520
3	超过 144 000 元至 300 000 元的部分	20	16 920
4	超过 300 000 元至 420 000 元的部分	25	31 920
5	超过 420 000 元至 660 000 元的部分	30	52 920
6	超过 660 000 元至 960 000 元的部分	35	85 920
7	超过 960 000 元的部分	45	181 920

请编程实现：计算个人每月应缴纳所得税额以及全年累计应缴纳所得税额。个人月工资数、三险一金每月扣除金额以及专项附加扣除金额由用户从键盘输入。

【实验提示】

(1) 计算个人全年累计应缴纳所得税额，定义两个含有 12 个元素的数组：

```
float a[12],tax[12],x;
```

数组 a 用来存放每个月税前工资，数组 tax 用来存放每个月应纳税金额。通过循环输入每个月的税前工资到数组 a，输入专项扣除额到变量 x 中。

(2) 累加数组 a 的各月工资数，再减去三险一金的金额和每月专项扣除的金额，即为应纳税额。

(3) 根据个人所得税税率表，判断应纳税额的纳税税率区间，计算应纳税额。

(4) 例如：如果个人 1 月工薪收入所得为 18 000 元，三险一金每月扣除 3200 元，专项附加扣除共 2000 元，个税起征点为 5000 元，则无须纳税额为 3200＋2000＋5000，则 1 月份应交税为：[18 000－(3200＋2000＋5000)]×3%＝234 元，放入 tax[0]中；2 月工薪收入 18 000 元，其他所得 1500 元，综合所得合计 19 500 元，则 1～2 月累积应纳税[(18 000＋19 500)－(3200＋2000＋5000)×2]×3%＝513 元，2 月应交税＝513－234＝279 元，放入 tax[1]中；……。依此方法，即可求出每月应纳税额及全年应纳税额。

5.4.2　模拟超市寄存柜存取操作

在大中型超市门口一般都放置有很多寄存柜，顾客可以将不能带入超市的物品暂时寄存在里面，购物结束后再取回。客户按动存包按钮，如果寄存柜中有空的箱子，则打开一个空箱子，并打印一张印有密码的纸条，客户购物结束凭密码取回个人的物品。如果没有空箱子，则显示“寄存柜已满，请耐心等待”。

本拓展项目要求编写一个程序模拟自动寄存柜的存包、取包功能。

【实验提示】

(1) 一般超市寄存柜都有若干排，每排都有若干个存包箱。构造两个二维数组，一个表示超市的寄存柜，二维数组的下标对应寄存柜中每个存包箱所在的位置；一个用来存放寄存柜中每个存包箱的密码。假设某超市的寄存柜共有 50 个存包箱，定义 box[6][11]的二维数组，数组行下标 1～5 表示存包箱所在的行数编号，列下标 1～10 表示存包箱所在的列数编号，定义二维数组 pas[6][11]存放对应存包箱的密码。

(2) 利用 rand 随机函数分别对两个二维数组进行初始化,其中一个初始化 0～1 的随机数,0 表示箱子为空,可以存包;1 表示箱子非空,当前不能存包。放密码的二维数组随机生成 6 位数密码。

使用 rand()%(b－a＋1)＋a 可生成 a～b 的随机数,含 a 和 b。

```
srand((unsigned)time(NULL));          //用系统时间作为生成随机数的种子
rand()%2;                             //随机生成 0～1 的随机数
```

同理,想要生成随机 6 位数密码:rand()%100000＋900000。

(3) 程序运行界面示例如图 5-4 所示,有带有编号的运行菜单及必要的文字提示,以便顾客操作。顾客可以循环地输入要进行的操作编号,进行相应的操作。当输入 3 时,结束程序运行。

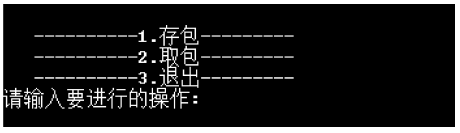

图 5-4 程序运行菜单界面

(4) 当顾客存包时,需要寻找可存包的空的存包箱(二维数组元素值为 0),在该二维数组中查找值为 0 的元素,确定该元素行下标和列下标,即可找到空的存包箱的位置。存包运行界面如图 5-5 所示。

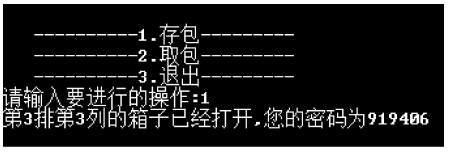

图 5-5 存包运行界面

(5) 当顾客需要取回已寄存的包,此时需用户输入密码,将用户输入的密码与该箱子应有的密码进行比较。如果密码正确则箱子打开取出包,同时箱子变回空箱状态;如果密码输入错误,则需重新输入密码,直到密码正确为止。取包运行界面如图 5-6 所示。

图 5-6 取包运行界面