5 数组实验

5.1实验目的

（1）掌握数组的说明、初始化和使用。

（2）掌握一维数组作为函数参数时实参和形参的用法。

（3）掌握字符串处理函数的设计，包括串操作函数及数字串与数之间转换函数实现算法。

（4）掌握基于分治策略的二分查找算法和选择法排序算法的思想，以及相关算法的实现。

5.2实验内容及要求

5.2.1源程序改错

下面是用来将数组a中元素按升序排序后输出的源程序。分析源程序中存在的问题，并对源程序进行修改，使之能够正确完成任务。

源程序

1 #include<stdio.h>

2 int main(void)

3 {

4 int a[10] = {27, 13, 5, 32, 23, 3, 17, 43, 55, 39};

5 void sort(int [],int);

6 int i;

7 sort(a[0],10);

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

9 printf("%6d",a[i]);

10 printf("\n");

11 return 0;

12 }

13 void sort(int b[], int n)

14 {

15 int i, j, t;

16 for (i = 0; i < n - 1; i++)

17 for ( j = 0; j < n - i - 1; j++)

18 if(b[j] < b[j+1])

19 t = b[j], b[j] = b[j+1], b[j+1] = t;

20 }

解答：

（1）错误改正：

1) 第7行中的调用函数不正确，应该为：

sort(a,10);

2) 第18行中的判断条件不正确，应该为：

if(b[j]>b[j+1])

（2）运行截图如下所示：



图5-1 程序修改题运行截图

5.2.2源程序完善、修改、替换

(1) 下面的源程序用于求解瑟夫问题：M个人围成一圈，从第一个人开始依次从1至N循环报数，每当报数为N时报数人出圈，直到圈中只剩下一个人为止。请在源程序中的下划线处填写合适的代码来完善该程序。

源程序：

#include<stdio.h>

#define M 10

#define N 3

int main(void)

{

int a[M], b[M]; /\* 数组a存放圈中人的编号，数组b存放出圈人的编号 \*/

int i, j, k;

for(i = 0; i < M; i++) /\* 对圈中人按顺序编号1—M \*/

a[i] = i + 1;

for(i = M, j = 0; i > 1; i--){

/\* i表示圈中人个数，初始为M个，剩1个人时结束循环；j表示当前报数人的位置 \*/

for(k = 1; k <= N; k++) /\* 1至N报数 \*/

if(++j > i - 1) j = 0;/\* 最后一个人报数后第一个人接着报，形成一个圈 \*/

b[M-i] = j? a[j-1] : a[j+1] ; /\* 将报数为N的人的编号存入数组b \*/

if(j)

for(k = --j; k < i; k++) /\* 压缩数组a，使报数为N的人出圈 \*/

a[k]=a[k+1] ;

}

for(i = 0;i < M – 1; i++) /\* 按次序输出出圈人的编号 \*/

printf(“%6d”, b[i]);

printf(“%6d\n”, a[0]); /\* 输出圈中最后一个人的编号 \*/

return 0;

}

(2) 上面的程序中使用数组元素的值表示圈中人的编号，故每当有人出圈时都要压缩数组，这种算法不够精炼。如果采用做标记的办法，即每当有人出圈时对相应数组元素做标记，从而可省掉压缩数组的时间，这样处理效率会更高一些。因此，请采用做标记的办法修改（1）中的程序，并使修改后的程序与（1）中的程序具有相同的功能。

解答：

1. 源程序代码如下所示：

#include "stdio.h"

int main()

{

int a[10],i,j,k=1;

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

a[i]=i+1;

while(k<31)

{

if(a[i%10]) k++,i++;

else

{i++;

continue;

}

if((k-1)%3==0)

{

printf("%d ",a[(i-1)%10]);

a[(i-1)%10]=0;

}

}

return 0;

}

3)实验思路：

使用两个计数器，一个负责调节程序的运行，另外一个负责计数，如果被取出，那么就记为0，计数器不自增。

1. 运行截图如下所示：



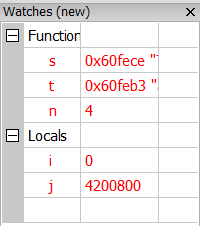
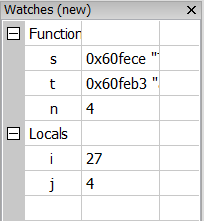
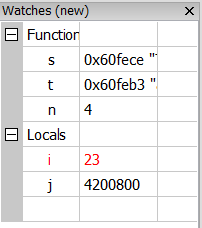
图5-2 程序替换题运行截图

5.2.3跟踪调试源程序

在下面所给的源程序中，函数strncat(s,t,n)本来应该将字符数组t的前n个字符连接到字符数组s中字符串的尾部。但函数strncat在定义时代码有误，不能实现上述功能。请按下面的要求进行操作，并回答问题和排除错误。

1. 单步执行源程序。进入函数strncat后观察表达式s、t和i。当光条落在for语句所在行时，i为何值？当光条落在strncat函数块结束标记（右花括号 }）所在行时, s、t分别为何值？

解答：

1. 分析函数出错的原因，排除错误，使函数正确实现功能，最后写出程序的输出结果。

解答：

1. 错误原因:由于调用函数时对于变量的理解不够，所以循环语句开始时，是s字符串已经结束了，导致不能在他后边加上理想中的字符

2）改正：在while语句之后，for语句之前加上i=i-1；使加上的字符串

3）运行截图如下所示：

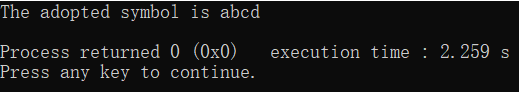


图5-3 跟踪调试题运行截图

源程序：

#include<stdio.h>

void strncat(char [],char [],int);

int main(void)

{

char a[50]="The adopted symbol is ",b[27]="abcdefghijklmnopqrstuvwxyz";

strncat(a, b, 4);

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

return 0;

}

void strncat(char s[],char t[], int n)

{

int i = 0, j;

while(s[i++]) ;

for(j = 0; j < n && t[j];)

s[i++] = t[j++];

s[i] = '\0';

}

5.2.4程序设计

编写并上机调试运行能实现以下功能的程序。

1. 编写一个程序,从键盘读取数据，对一个3×4矩阵进行赋值，求其转置矩阵，然后输出原矩阵和转置矩阵。

解答：

1. 源程序代码如下所示：

#include "stdio.h"

int main()

{

int a[3][4],b[4][3];

int i,j;

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

{

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

scanf("%d",&a[i][j]);

}

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

{

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

b[j][i]=a[i][j];

}

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

{

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

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

putchar('\n');

}

putchar('\n');

putchar('\n');

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

{

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

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

putchar('\n');

}

return 0;

}

1. 设计思路：

A：先定义两个数组，并且对原来的数组进行赋值。

B：然后进行循环，把已有的数组的行赋值给另一数组的列，列赋值给行，从而实现矩阵的转置。

C：接着用两个循环，分别打出转置前后的矩阵即可。

1. 运行截图如下所示：

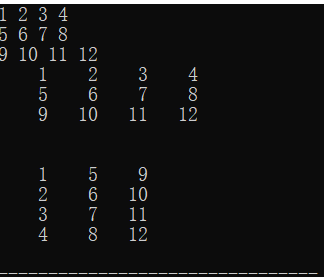


图5-4 程序设计题1运行截图

1. 编写一个程序, 其功能要求是：输入一个整数，将它在内存中二进制表示的每一位转换成为对应的数字字符，存放到一个字符数组中，然后输出该整数的二进制表示。

解答：

1. 源程序代码如下所示：

#include "stdio.h"

int main()

{

int n,i;

char c[32];

int mask=1<<31;

scanf("%d",&n);

for(i=0;i<32;i++)

{

c[i]=n&mask?'1':'0';

n=n<<1;

}

for(i=0;i<32;i++)

printf("%c",c[i]);

return 0;

}

1. 实验思路：

A：本题主要采用逻辑尺的相关知识，原理是计算机中的数字都是以二进制形式存储的。

B：首先定义一个32位的字符数组，同时定义一个逻辑尺，该逻辑尺是1左移31位后得到的

C；然后分别用该逻辑尺和输入的数字进行按位与，如果结果为0，则将0这个字符储存在数组中，否则储存1在这个数组中。

D：最后用一个循环就可以将这个数组中的字符输出。

1. 运行截图如下所示：

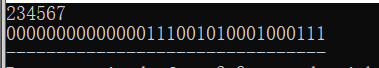


图5-5 程序设计题2运行截图

1. 编写一个程序, 其功能要求是：输入n个学生的姓名和C语言课程的成绩，将成绩按从高到低的次序排序，姓名同时作相应调整，输出排序后学生的姓名和C语言课程的成绩。然后，输入一个C语言课程成绩值，用二分查找进行搜索。如果查找到有该成绩，输出该成绩同学的姓名和C语言课程的成绩；否则输出提示“not found!”。

解答：

1. 源程序代码如下所示：

#include "stdio.h"

#include "string.h"

int binarysearch(int s[],int k,int o);

int main()

{

int i,k,m,t,o;

scanf("%d",&k);

char n[k][20],c[20];

int s[k];

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

scanf("%s%d",n[i],&s[i]);

int p,q;

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

{

for(q=0;q<k-p-q;q++)

{

if(s[q]<s[q+1])

{

s[q]=s[q]+s[q+1];

s[q+1]=s[q]-s[q+1];

s[q]=s[q]-s[q+1];

strcpy(c,n[q]);

strcpy(n[q],n[q+1]);

strcpy(n[q+1],c);

}

}

}

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

printf("%-20s%d\n",n[i],s[i]);

putchar('\n');

scanf("%d",&m);

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

{

scanf("%d",&o);

t=binarysearch(s,k,o);

if(t==-1)

printf("Not found!\n");

else

printf("%-20s%d\n",n[t],s[t]);

}

return 0;

}

int binarysearch(int s[],int k,int o)

{

int f=0,b=k-1,m;

for(;f<=b;)

{

m=(f+b)/2;

if(o<s[m]) f=m+1;

else if(o>s[m]) b=m-1;

else return m;

}

return -1;

}

1. 实验思路：

A：本体主要应用到了函数调用，冒泡排序，多维数组（二维数组），以及二分法查找。

B：首先定义一个二维数组，用来存储姓名，定义一个一维数组，用来存储成绩，定义一个函数，用来二分法查找成绩。

C：然后用一个循环来输入姓名，注意，姓名是以字符串形式输入的，因此在存储时不用加那一个&。

D：冒泡排序时要注意循环的次数，并且要用到字符串的复制，因此要在头文件中include，并且要注意成绩交换时姓名也要同步交换，保证成绩和姓名要一一对应。

F：调用函数时要注意调用的形式要正确。

1. 运行截图如下所示;

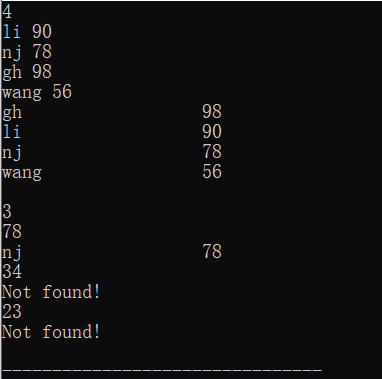


图5-6 程序设计题3运行截图

5.2.5选做题程序设计

编写并上机调试运行能实现以下功能的函数和程序。

1. 编写函数strnins(s,t,n),其功能是：可将字符数组 t中的字符串插入到字符数组 s中字符串的第n个字符的后面。

解答：

1. 源程序代码如下所示：

#include "stdio.h"

void strnins(int s[],int t[],int n);

int main()

{

int s[100],t[100],n;

int i=0;

int j=0;

while((s[i++]=getchar())!='\n');

s[i-1]='\0';

while((t[j++]=getchar())!='\n');

t[j-1]='\0';

scanf("%d",&n);

strnins(s,t,n);

return 0;

}

void strnins(int s[],int t[],int n)

{

int k;

char m[100];

int i=0;int j=0;

while(s[i]!='\0') i++;

while(t[j]!='\0') j++;

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

m[k]=s[n+k];

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

s[n+k]=t[k];

for(k=0;k<n+j;k++)

printf("%c",s[k]);

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

printf("%c",m[k]);

}

1. 解题思路：

A：首先，本题考查的是对于字符数组的使用，而且涉及到了函数调用的问题。

B：主要在于用另外一个数组暂时存储一部分字符，然后再将字符交换，从而实现实验的目的。

3）运行截图如下所示：

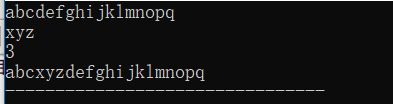


图5-7 选做题1运行截图

1. 编写一个实现八皇后问题的程序，即：在8\*8方格国际象棋盘上放置8个皇后，任意两个皇后不能位于同一行、同一列或同一斜线（正斜线或反斜线）上，并输出所有可能的放法。

解答：

1. 源程序代码如下所示：

#include "stdio.h"

int main()

{

int k1,k2,k3,k4,k5,k6,k7,k8;

int m=0;

for(k1=0;k1<8;k1++)

{

for(k2=0;k2<8;k2++)

{

if(k2==k1||k2==k1-1||k2==k1+1) continue;

for(k3=0;k3<8;k3++)

{

if(k3==k1||k3==k2||k3==k2-1||k3==k2+1||k3==k1-2||k3==k1+2) continue;

for(k4=0;k4<8;k4++)

{

if(k4==k1||k4==k2||k4==k3||k4==k3-1||k4==k3+1||k4==k1-3||k4==k1+3||k4==k2-2||k4==k2+2) continue;

for(k5=0;k5<8;k5++)

{

if(k5==k1||k5==k2||k5==k3||k5==k4||k5==k4-1||k5==k4+1||k5==k1+4||k5==k1-4||k5==k2-3||k5==k2+3||k5==k3-2||k5==k3+2) continue;

for(k6=0;k6<8;k6++)

{

if(k6==k1||k6==k2||k6==k3||k6==k4||k6==k5||k6==k5-1||k6==k5+1||k6==k1-5||k6==k1+5||k6==k2-4||k6==k2+4||k6==k3-3||k6==k3+3||k6==k4-2||k6==k4+2) continue;

for(k7=0;k7<8;k7++)

{

if(k7==k1||k7==k2||k7==k3||k7==k4||k7==k5||k7==k6||k7==k6-1||k7==k6+1||k7==k1-6||k7==k1+6||k7==k2-5||k7==k2+5||k7==k3-4||k7==k3+4||k7==k4-3||k7==k4+3||k7==k5-2||k7==k5+2) continue;

for(k8=0;k8<8;k8++)

{

if(k8==k1||k8==k2||k8==k3||k8==k4||k8==k5||k8==k6||k8==k7||k8==k7-1||k8==k7+1||k8==k1-7||k8==k1+7||k8==k2-6||k8==k2+6||k8==k3-5||k8==k3+5||k8==k4-4||k8==k4+4||k8==k5-3||k8==k5+3||k8==k6-2||k8==k6+2) continue;

m++;

printf("%d %d %d %d %d %d %d %d\n",k1,k2,k3,k4,k5,k6,k7,k8);

}

}

}

}

}

}

}

}

printf("%d",m);

return 0;

}

1. 程序设计思路如下所示：

A：首先用到了嵌套循环，并且用到了枚举法来求解。

B：主要是利用了一层层来判断是否满足条件，如果满足就向下一层进行，否则的话就返回后加一。

C：判断时主要利用行数于他们之间距离的关系来进行，如他们在同一条斜线，那么他们的差值绝对值是行数之差。因此来判断是否满足题意。

1. 运行截图如下所示：

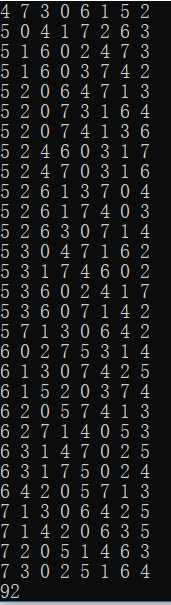
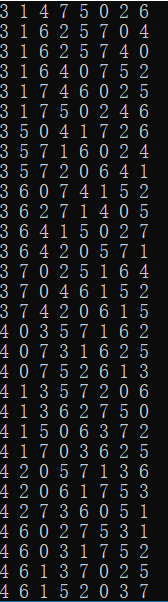
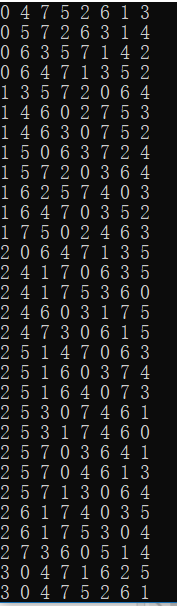


图5-8 选做题2运行截图

5.3 实验小结

1）：本次实验难度较大，而且我存在的问题也挺多的，真正去编程的时候发现自己好多地方都不会，有些知识点都记不起来了，导致错误一大堆，浪费了大量的时间。

2）不过本次实验也启示我要多注意细节，因为细节决定成败，而且好多小问题其实是不容易被发现的，就会浪费大量的时间。

3）通过本次实验我发现自己对于字符的处理不够熟悉，而且知识点也较多，自己记得也不是很熟练，运用起来就显得生疏。

4）这次试验让我看到了自己的不足，以后要在C语言中投入更多的时间才可以达到理想中的高度。加油！