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++)

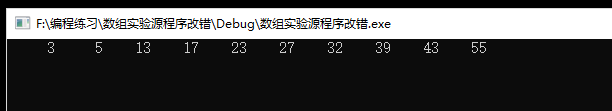
18 if(b[j] < b[j+1])

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

20 }

**解答：**

1. 错误修改：
2. 第5行，函数声明最好放在main函数之外声明
3. 第7行，参数传递有误，正确形式为：void sort(a,10);
4. 第18行，比较出错，正确形式为：if(b[j] > b[j+1])
5. 错误修改后运行结果：



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[i-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：**

替换后程序如下：

#include<stdio.h>

#define M 10

#define N 3

int main(void)

{

int a[M] = { 1,1,1,1,1,1,1,1,1,1 };

int b[M];

int i, j = 0, k, q = 0;

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

{

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

{

if (j > M - 1)

j = 0;

if (a[j] == 0)

continue;

k++;

if (k % N == 0)

{

a[j] = 0;

break;

}

}

b[q++] = j + 1;

}

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

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

printf("\n");

return 0;

}



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？

（2）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。

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];)

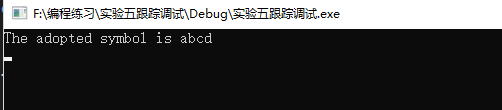
s[i++] = t[j++];

s[i] = '\0';

}

1. i=23 s=” The adopted symbol is ” t=” abcdefghijklmnopqrstuvwxyz”
2. 在strncat函数的while循环后加入i--

结果：



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. 解题思路：
2. 读入3\*4矩阵，两层for循环为二维数组arr[M][N]赋值；
3. 两层for循环将arr[M][N]输出，最外层循环每循环一次输出一个换行符
4. 调用函数zhuan，传入数组指针以及行数；
5. 以两层for循环的方式将转置矩阵输出；
6. 结束。
7. 程序清单：

#include<stdio.h>

#define M 3

#define N 4

void zhuan(int a[][N],int n);

int main(void)

{

int arr[M][N];

int i,j;

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

{

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

{

printf("第%d行，第%d列：", i + 1, j + 1);

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

}

}

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

{

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

{

printf("%4d", arr[i][j]);

}

putchar('\n');

}

putchar('\n');

zhuan(arr, M);

return 0;

}

void zhuan(int a[][N], int n)

{

int i, j;

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

{

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

{

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

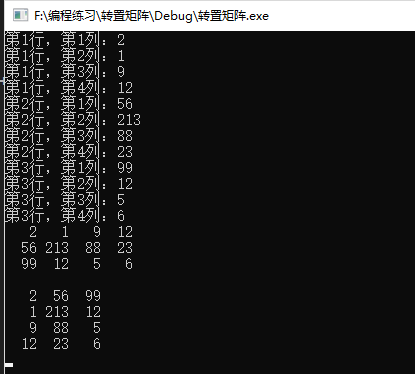
}

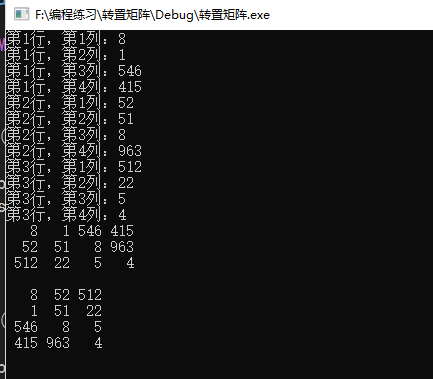
putchar('\n');

}

}

1. 测试：





（2）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. 解题思路：
2. 读入整数，创建大小为sizeof(int)\*8+1的字符数组；
3. i= sizeof(int) \* 8-1，若i小于0，跳转步骤5；
4. 以逻辑尺获取整数二进制最后一位，转化为数字字符赋值给arr[i]；
5. 数字按位右移一位，i--，转步骤二；
6. arr[i]=’\0’；
7. 输出字符串；
8. 结束。
9. 程序清单：

#include<stdio.h>

int main(void)

{

char arr[sizeof(int)\*8+1];

int num,i,ch;

scanf("%d", &num);

int mask = 0x1;

for (i = sizeof(int) \* 8-1; i >=0; i--)

{

ch = num & mask;

arr[i] = ch + 48;

num >>= 1;

}

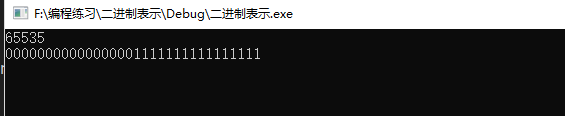
arr[sizeof(int) \* 8] = '\0';

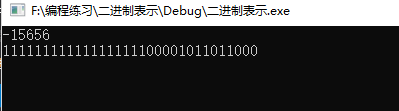
printf("%s", arr);

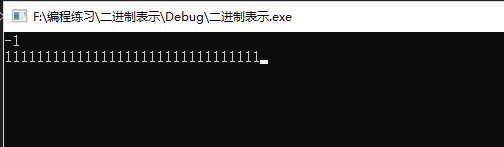
return 0;

}

1. 测试：







（3）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. 解题思路：
2. 读取学生数；
3. 以二维字符数组形式储存姓名，以一维数组的形式储存成绩，数组下标一一对应；
4. 调用sort函数以成绩进行排序，排序方法使用选择排序；
5. 从高到低输出每个同学的名字；
6. 读取输入的成绩直至输入非数字，若为非数字转步骤9；
7. 调用find函数，以二分查找的方式查找成绩；
8. 若查到成绩，函数返回对应下标，否则返回-1；
9. 若返回值为-1，输出not found，否则以返回的下标值输出姓名和成绩，转步骤5；
10. 结束。
11. 程序清单：

#include<stdio.h>

#include<string.h>

void sort(int grade[], char name[][40], int n);

int findgrade(int grade[], int num, int begin, int end);

int k = -1,l=0;

int main(void)

{

int grade[20];

char name[20][40];

int i = 0, n, find, j;

printf("how many:");

scanf("%d", &n);

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

{

printf("name:");

scanf("%s", name[i]);

printf("grades:");

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

}

sort(grade, name, n);

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

printf("%6s : %6d\n", name[i], grade[i]);

while (scanf("%d", &find) == 1)

{

j = findgrade(grade, find, 0, n);

if (j == -1)

printf("not found!\n");

else

printf("%-6s : %-6d\n", name[j], grade[j]);

k = -1;

l = 0;

}

return 0;

}

void sort(int grade[], char name[][40], int n)

{

if (n <= 0)

return;

int temp, i, min = 0;

char tempname[40];

strcpy(tempname, name[0]);

temp = grade[0];

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

if (grade[i] < temp)

{

temp = grade[i];

strcpy(tempname, name[i]);

min = i;

}

grade[min] = grade[n - 1];

grade[n - 1] = temp;

strcpy(name[min], name[n - 1]);

strcpy(name[n - 1], tempname);

sort(grade, name, n - 1);

}

int findgrade(int grade[], int num, int begin, int end)

{

if (begin == end)

l++;

if (begin == end && l > 1)

return k;

if (begin > end)

return k;

int n = (begin + end) / 2;

int x = grade[n];

if (x == num)

{

k = n;

return k;

}

else if (num > x)

{

findgrade(grade, num, begin, n);

}

else if (num < x)

{

findgrade(grade, num, n+1 , end);

}

}

1. 测试：

