

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级： 计算机科学与技术**

**学 号： U201714726**

**姓 名： 王 明 明**

**指导教师： 甘 早 斌**

**报告日期： 2018.1.9**

**计算机科学与技术学院**

**目 录**

[**实验1 表达式和标准输入与输出实验 1**](#_Toc503202938)

[1.1 实验目的 1](#_Toc503202939)

[1.2 实验内容 1](#_Toc503202940)

[1.3 实验小结 11](#_Toc503202944)

[**实验2 流程控制实验 12**](#_Toc503202945)

[2.1 实验目的 12](#_Toc503202946)

[2.2 实验内容 12](#_Toc503202947)

[2.3 实验小结 32](#_Toc503202952)

[**实验3 函数与程序结构实验 34**](#_Toc503202953)

[3.1 实验目的 34](#_Toc503202954)

[3.2 实验内容 34](#_Toc503202955)

[3.3 实验小结 50](#_Toc503202961)

[**实验4 编译预处理实验 52**](#_Toc503202962)

[4.1 实验目的 52](#_Toc503202963)

[4.2 实验内容 52](#_Toc503202964)

[4.3 实验小结 65](#_Toc503202969)

[**实验5 数组实验 66**](#_Toc503202970)

[5.1 实验目的 66](#_Toc503202971)

[5.2 实验内容 66](#_Toc503202972)

[5.3 实验小结 94](#_Toc503202978)

[**实验6 指针实验 94**](#_Toc503202979)

[6.1 实验目的 94](#_Toc503202980)

[6.2 实验内容 95](#_Toc503202981)

[6.3 实验小结 120](#_Toc503202986)

[**实验7 结构与联合实验 121**](#_Toc503202987)

[7.1 实验目的 121](#_Toc503202988)

[7.2 实验内容 121](#_Toc503202989)

[7.3 实验小结 145](#_Toc503202994)

[**实验8 文件实验 146**](#_Toc503202995)

[8.1 实验目的 146](#_Toc503202996)

[8.2 实验题目及要求 146](#_Toc503202997)

[8.3 实验小结 152](#_Toc503203000)

参考文献 153

# 

# 1 表达式和标准输入与输出实验

## 1.1 实验目的

(1)熟练掌握各种运算符的运算功能，操作数的类型，运算结果的类型及运算过程中的类型转换，重点是C语言特有的运算符，例如位运算符，问号运算符，逗号运算符等；熟记运算符的优先级和结合性。

（2）掌握getchar, putchar, scanf 和printf 函数的用法。

（3）掌握简单C程序（顺序结构程序）的编写方法。

1.2 实验内容

### 1.2.1 源程序改错

下面给出了一个简单C语言程序例程，用来完成以下工作：

（1）输入华氏温度f，将它转换成摄氏温度c后输出；

（2）输入圆的半径值ｒ，计算并输出圆的面积ｓ；

（3）输入短整数ｋ、ｐ，将ｋ的高字节作为结果的低字节，ｐ的高字节作为结果的高字节，拼成一个新的整数，然后输出；

在这个例子程序中存在若干语法和逻辑错误。要求参照1.3和1.4的步骤对下面程序进行调试修改，使之能够正确完成指定任务。

1 #include<stdio.h>

2 #define PI 3.14159;

3 voidmain( void )

4 {

5 int f ;

6 short p, k ;

7 double c , r , s ;

8 /\* for task 1 \*/

9 printf(“Input Fahrenheit:” ) ;

10 scanf(“%d”, f ) ;

11 c = 5/9\*(f-32) ;

12 printf( “ \n %d (F) = %.2f (C)\n\n ”, f, c ) ;

13 /\* for task 2 \*/

14 printf("input the radius r:");

15 scanf("%f", &r);

16 s = PI \* r \* r;

17 printf("\nThe acreage is %.2f\n\n",&s);

18 /\* for task 3 \*/

19 printf("input hex int k, p :");

20 scanf("%x %x", &k, &p );

21 newint = (p&0xff00)|(k&0xff00)<<8;

22 printf("new int = %x\n\n",newint);

}

**解答：**

（1）错误修改：

1) 第2行的符号常量定义后不能有分号，正确形式为：

#define PI 3.14159

2) 第3行的voidmain，正确形式为：

void main

3) 第9行的“” ，正确形式为：

“ ”

4) 第10行的scanf(“%d”, f ) ，正确形式为：

scanf(“%d”, &f )

5) 第11行的c = 5/9\*(f-32) ，正确形式为：

c = (5.0/9)\*(f-32)

6) 第15行%f ，正确形式为：%lf

7) 第17行&s ，正确形式为：s

8) 第20行%x %x，正确形式为：%hd %hd

9) 第21行<<8 ，正确形式为：>>8

10)第22行newint，正确形式为：在第6行声明 short p,k,newint;

（2）错误修改后运行结果如图1-1所示：

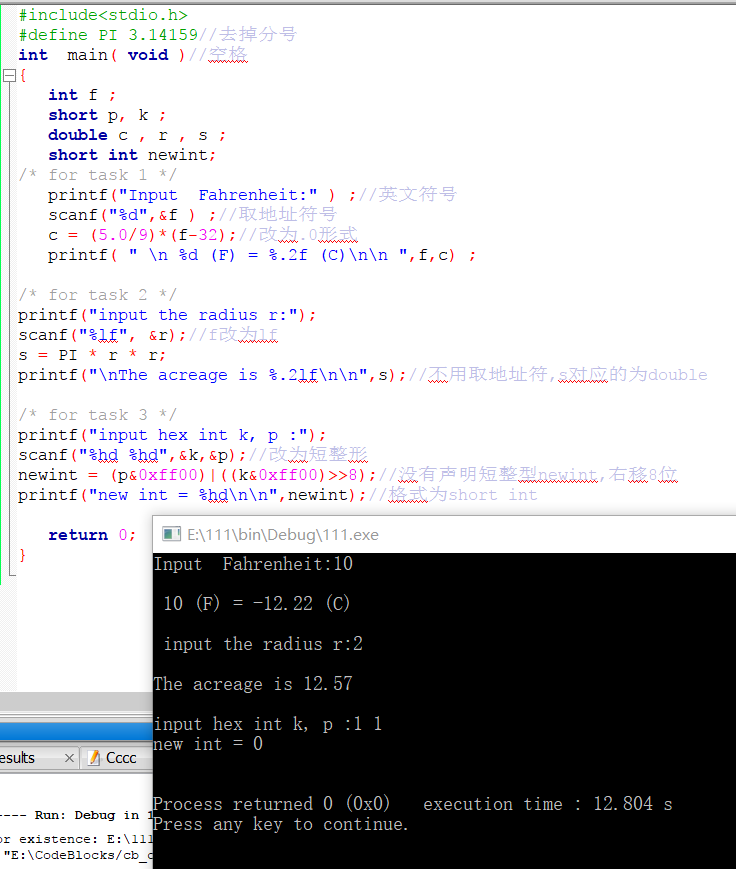
****

图1-1 程序修改测试截图

### 1.2.2 源程序修改替换

下面的程序利用常用的中间变量法实现两数交换，请改用不使用第3个变量的方法实现。该程序中t是中间变量，要求将定义语句中的t删除，修改下划线处的语句，使之实现两数对调的操作。

#include<stdio.h>

void main( )

{

int a, b, t;

printf(“Input two integers:”);

scanf(“%d %d”,&a,&b);

t=a ；a=b；b=t；

prinf(“\na=%d,b=%d”,a,b);

}

**解答：**

可通过加减法来进行替换，替换后的程序如下所示：

#include<stdio.h>

int main( )

{

int a,b;

printf("Input two integers:");

scanf("%d %d",&a,&b);

a=a+b;

b=a-b;

a=a-b;

printf("\na=%d,b=%d",a,b);

return 0;

}

运行结果如图1-2所示

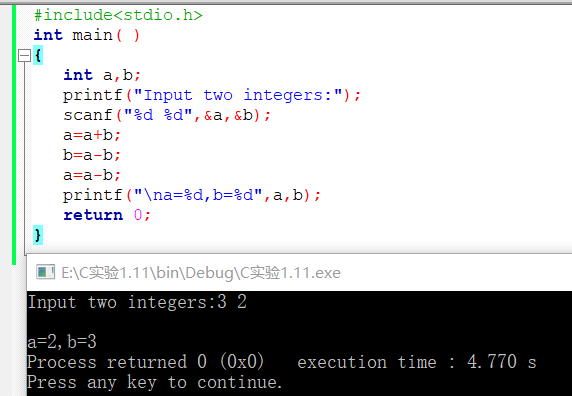


图1-2程序替换运行截图

### 1.2.3 程序设计

**（1）**编写一个程序，输入字符ｃ，如果ｃ是大写字母，则将ｃ转换成对应的小写，否则ｃ的值不变，最后输出ｃ。

**解答：**

1） 算法流程如图1-3所示。

请输入字母c

否

小写？

输出该字母

c=c+32

图1-3 编程题1的程序流程图

2）源程序清单

#include <stdio.h>

#include <stdlib.h>

int main()

{

char c;

scanf("%c",&c);

if((c>='a')&&(c<='z'))

printf("%c\n",c);

if((c>='A')&&(c<='Z'))

{

c=(c+32);

printf("%c\n",c);

}

return 0;

}

3）测试

（a） 测试数据：

小写字母a 输出小写字母a

大写字母A 输出小写字母a

（b） 对应测试数据的运行结果如图1-4和图1-5所示

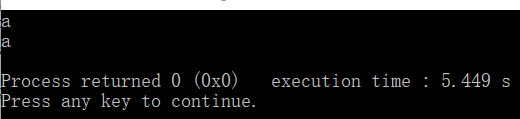


图1-4 测试数据截图

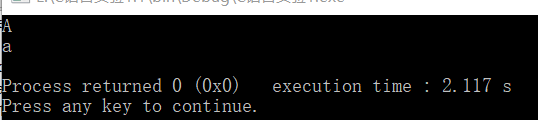


图1-5测试数据截图

（2）编写一个程序，输入无符号短整数x，ｍ，ｎ（0 ≤ｍ≤ 15, 1 ≤ ｎ≤ 16-ｍ）,取出x从第ｍ位开始向左的ｎ位（ｍ从右至左编号为0～15），并使其向左端（第15位）靠齐。

**解答：**

1) 解题思路如图1-6所示：

输入x,m,n

输入正确？

否

是

将x右移m位

再将x左移（16-n）位

提示输入错误

输出变换后的x

图1-6编程题（2）的程序流程图

2）程序清单

#include <stdio.h>

#include <stdlib.h>

int main()

{

unsigned short int x,m,n;

printf("输入x、m(0~15)和n(1~16-m):\n");

scanf("%hd %hd %hd",&x,&m,&n);

if((m<0||m>15)||(n<1||n>(16-m))) //输入的m,n要满足条件

printf("输入错误!\n");

else

{

x=x>>m; //右移m位

x=x<<(16-n); //左移(16-n)位

printf("ans=%hu\n",x); //%hu正确

}

return 0;

}

3）测试

（a） 测试数据：

叙述选择测试数据的方法。。。如表1-1所示。

表1-1 编程题3的测试数据

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试  用例 | 程 序 输 入 | | | 理 论 结 果 | 运 行 结 果 |
| X | m | N |
| 用例1 | 0001 0011 1000 1000即  十进制5000 | 3 | 4 | 计算结果0001 0000 0000 0000即4096 |  |
| 用例2 | 十进制4800 | 16 | 1 | 输入错误（m值超范围） |  |
| 用例3 | 十进制6156 | 13 | 5 | 输入错误（n值超范围） |  |

（b） 对应测试测试用例1的运行结果如图1-7所示。

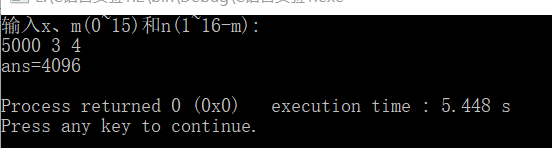


图1-7 测试结果截图

图1-6 编程题（3）的测试用例一的运行结果

对应测试测试用例2的运行结果如图1-8所示。

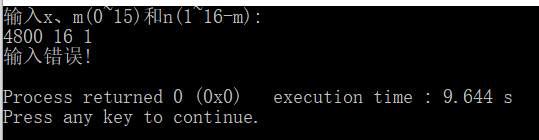


图1-8 测试结果截图

图1-6 编程题3的测试用例二的运行结果

对应测试测试用例3的运行结果如图1-9所示。

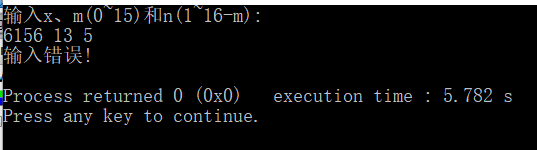


图1-9 测试结果截图

图1-6 编程题3的测试用例三的运行结果

说明上述的运行结果与理论分析吻合，验证了程序的正确性。

（3）IP地址通常是4个用句点分隔的小整数，如32.55.1.102。这些地址在机器中用无符号长整形表示。编写一个程序，以机器存储的形式读入一个32位的互联网IP地址，对其译码，然后用常见的句点分隔的4部分的形式输出。

1) 流程图如图1-10所示

输入无符号长整型数

设计屏蔽码，分别得到4个二进制数

移动相应的位数

转换为十进制输出

图1-10 编程题3的程序流程图

2) 程序如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

printf("请输入无符号长整型\n");

unsigned long int x,x1,x2,x3,x4,mask1,mask2,mask3,mask4;

scanf("%lu",&x);

mask1 = 0xff000000;

mask2 = 0x00ff0000;

mask3 = 0x0000ff00;

mask4 = 0x000000ff;

x1 = (x&mask1)>>24;

x2 = (x&mask2)>>16;

x3 = (x&mask3)>>8;

x4 = x&mask4;

printf("%lu.%lu.%lu.%lu\n",x4,x3,x2,x1);

return 0;

}

3) 测试数据

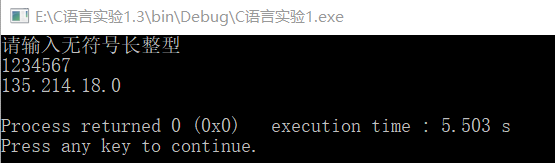


图1-11 测试结果截图

图1-10 编程题（3）的运行结果

## 1.3 实验小结

主要叙述实验过程中遇到的问题，如何解决的，通过分析、结果问题后的体会。

问题：

改错实验中，在第一部分中，错误很多都是中文符号造成的。拷贝之后，一编译发现错误很多。

编程题第二个，刚开始以十六进制写的是正确的，改成十进制时出错，虽然知道错在哪，但不知道怎么改。

编程题第三个，IP地址输出时是反向输出的，当时并不知道。

解决方法：

拷贝之后，先把标点符号改正确，再找语法错误，最后编译寻找一些逻辑错误，把问题分步骤解决。

编程题第二个，请教老师后豁然开朗，可以用%hu来输出无符号短整型。

编程题第三个，先是自行百度，对百度的答案并不放心，又询问了助教。

体会：编程序不仅仅是编程，还涉及很多数学知识，多个学科交叉。英文符号和汉语符号不能乱用，因为汉文英文的符号差别较小，比较难分辨，以后一定要细心。遇到不会的一定要思考，之后再询问他人，这样才有效果。没有什么学科是多余的，画流程图的时候就意识到了大学计算机基础的重要性。

# 实验2 流程控制实验

2.1 实验目的

（1）掌握复合语句、if语句、switch语句的使用，熟练掌握for、while、do-while三种基本的循环控制语句的使用，掌握重复循环技术，了解转移语句与标号语句。

（2）练习循环结构for、while、do-while语句的使用。

（3）练习转移语句和标号语句的使用。

（4）使用Turbo C 2.0集成开发环境中的调试功能：单步执行、设置断点、观察变量值。

2.2 实验内容

### 2.2.1 源程序改错题

下面是计算s=n!的源程序，在这个源程序中存在若干语法和逻辑错误。要求在计算机上对这个例子程序进行调试修改，使之能够正确完成指定任务。例如，8！=40320。

1 #include <stdio.h>

2 void main(void)

3 {

4 int i,n,s=1;

5 printf("Please enter n:");

6 scanf("%d",n);

7 for(i=1,i<=n,i++)

8 s=s\*i;

9 printf("%d! = %d",n,s);

10 }

**解答：**

(1)程序修改：

1）第6行 scanf("%d",n) ，正确形式为：scanf("%d",&n)

2）第7行 for(i=1,i<=n,i++)，正确形式为：for(i=1;i<=n;i++)

（2）程序修改后运行结果：

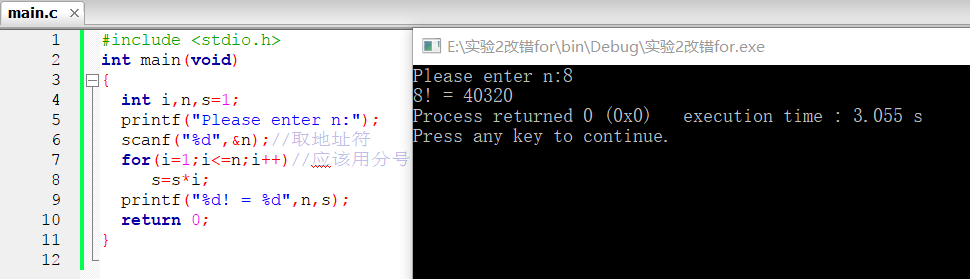


图2-1 测试结果截图

### 2.2.2 源程序修改替换题

（1）修改第1题，分别用while和do-while语句替换for语句。

解答：

1）程序用while语句替换如下：

#include <stdio.h>

#include <stdlib.h>

int main(void)

{

int i=1,s=1,n;

printf("Please enter n:");

scanf("%d",&n);

while(i<=n)

{

s = s \* i;

i++;

}

printf("%d ! = %d\n",n,s);

return 0;

}

程序运行结果如图2-2所示：

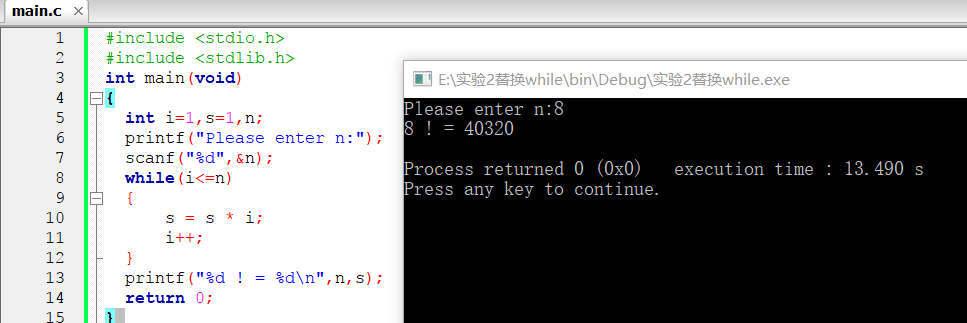


图2-2 测试结果截图

2）程序用do-while语句替换如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

int n,s = 1,i = 1;

printf("Please enter n:");

scanf("%d",&n);

do{

s = s \* i;

i++;

}while(i <= n);

printf("%d! = %d\n",n,s);

return 0;

}

运行结果如图2-3所示：

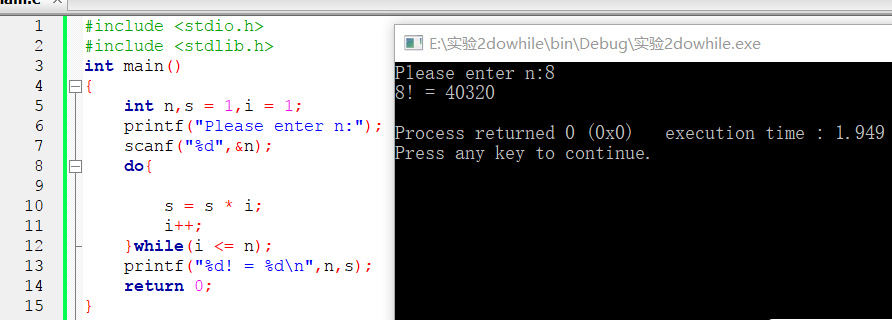


图2-3 测试结果截图

（2）修改第1题，输入改为“整数S”，输出改为“满足n！≥S的最小整数n”。例如输入整数40310，输出结果为n=8。

解答：

1）程序修改如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

int n=0,s,i;

printf("整数s:");

scanf("%d",&s);

if(s==1||s==0) printf("最小n为：%d\n",n);

if(s>1)

{

int n=1;

for(i=1;n<s;i++)

{

n=n\*i;

}

i--;

printf("最小的n为：%d\n",i);

}

return 0;

}

2）运行结果如图2-4所示：

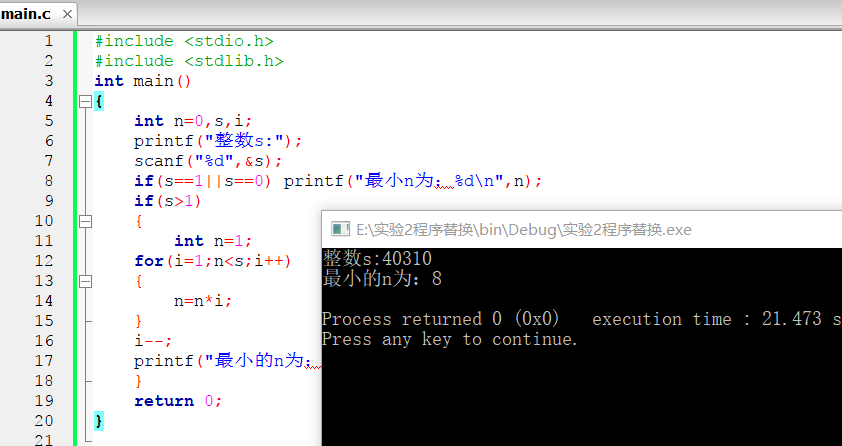


图2-4 测试结果截图

### 2.2.3 编程设计题

编写并上机调试运行能实现以下功能的程序。

（1） 假设工资税金按以下方法计算：x ＜ 1000元，不收取税金；1000 ≤ x ＜ 2000，收取5%的税金；2000 ≤ x ＜ 3000，收取10%的税金；3000 ≤ x ＜ 4000，收取15%的税金；4000 ≤ x ＜ 5000，收取20%的税金；x＞5000，收取25%的税金。编写一个程序,输入工资金额，输出应收取税金额度，要求分别用if语句和switch语句来实现。

解答：

1）使用if语句的流程图如图2-5所示：

请输入工资

<1000?

是

X=0

否

是

X=(x-1000)\*0.05

1000<x<=2000?

否

是

X=(x-2000)\*0.1+50

2000<x<=3000?

否 是

X=(x-3000)\*0.15+150

3000<x<=4000?

否

4000<x<=5000?

是

X=(x-4000)\*0.2+300

否

X=(x-5000)\*0.25+500

输出X

图2-5 编程题（1）用if语句流程图

2）用if语句实现如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

float x,s;

printf("Please input salary:\n");

scanf("%f",&x);

if(x<1000)

{

s = 0.0;

printf("收取的税金为：%f\n",s);

}

else if((x>=1000)&&(x<2000))

{

s = (x-1000)\* 0.05;

printf("收取的税金为：%f\n",s);

}

else if((x>=2000)&&(x<3000))

{

s = (x-2000)\* 0.1;

s = s + 50.0;

printf("收取的税金为：%f\n",s);

}

else if((x>=3000)&&(x<4000))

{

s = (x-3000) \* 0.15;

s = s + 150.0;

printf("收取的税金为：%f\n",s);

}

else if((x>=4000)&&(x<5000))

{

s = (x-4000) \* 0.2;

s = s + 300.0;

printf("收取的税金为：%f\n",s);

}

else

{

s = (x-5000)\* 0.25;

s = s + 500.0;

printf("收取的税金为：%f\n",s);

}

return 0;

}

3）测试数据如下：

输入500，结果为0；

输入1400，结果为20；

输入2020，结果为52；

输入3060，结果为159；

输入4200，结果为340；

输入6030，结果为757.5；

4）运行结果如图2-6所示：

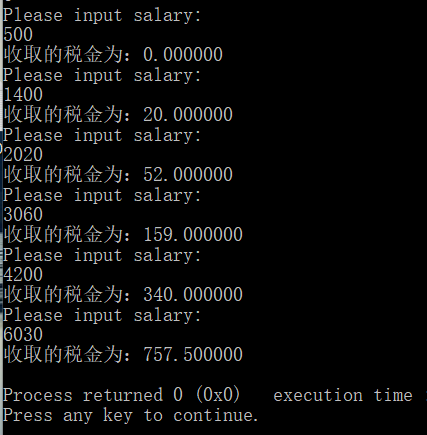


图2-6 测试结果截图

1）使用switch语句流程图如图2-7所示：

输入X,S，M

M=X/1000

是

M=0?

S=0

否 是

M=1?

S=(X-1000)\*0.05

否 是

M=2?

S=(X-2000)\*0.1+50

否 是

M=3?

S=(X-3000)\*0.15+150

否

是

M=4?

S=(X-4000)\*0.2+300

是

S=(X-5000)\*0.25+500

输出S

图2-7 编程题（1）用switch语句流程图

2）用switch语句实现如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

float x,s;

printf("Please input salary:\n");

scanf("%f",&x);

switch((int)(x/1000))

{

case 0:

{

s = 0.0;

printf("缴纳税金为:%f\n",s);

}break;

case 1:

{

s = (x-1000)\* 0.05;

printf("缴纳税金为:%f\n",s);

}break;

case 2:

{

s = (x - 2000) \* 0.1;

s = s + 50.0;

printf("缴纳税金为:%f\n",s);

}break;

case 3:

{

s = (x-3000) \* 0.15;

s = s + 150.0;

printf("缴纳税金为:%f\n",s);

}break;

case 4:

{

s = (x-4000) \* 0.2;

s = s + 300.0;

printf("缴纳税金为:%f\n",s);

}break;

default :

printf("缴纳税金为:%f\n",((x - 5000) \* 0.25) + 500);

}

return 0;

}

3）测试数据：

输入500，结果为0；

输入1400，结果为20；

输入2020，结果为52；

输入3060，结果为159；

输入4200，结果为340；

输入6030，结果为757.5；

4）运行结果如图2-8所示：

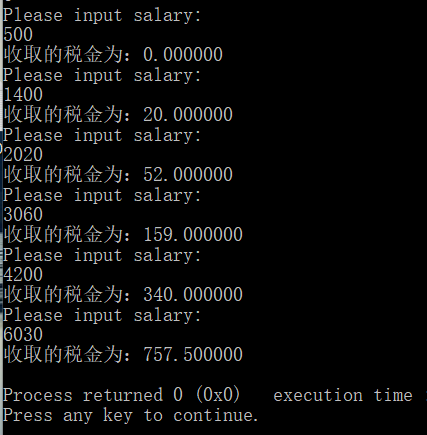


图2-8 运行结果截图

（2）编写一个程序,将输入的一行字符复制到输出，复制过程中将一个以上的空格字符用一个空格代替。

解答：

1）流程如下：

1设置三种状态，copy,start,comment

2 判断输入的字符，如果字符不是文件结束标志，状态置为copy

3 如果遇到字符是空格，状态变为start,如果不是空格就输出字符

4 如果start状态下遇到空格，状态变为comment，执行步骤5.如果不是空格，就输出一个空格。如果下一个字符是空格，状态为start,继续执行步骤4.否则状态为copy,执行步骤3，并且输出该字符

5 在comment状态下，如果遇到的字符不是空格，则输出一个空格。如果下一个字符是空格，则状态变为start，执行步骤4.否则输出该字符，且状态变为copy，执行步骤3。如果在comment 状态下遇到的为空格，则状态依旧是comment,继续执行步骤5

2）程序如下：

#include <stdio.h>

#include <stdlib.h>

enum {copy,start,comment};

int main()

{

char c;

int state = copy;

while((c=getchar())!=EOF)

{

switch(state)

{

case copy:

if (c == ' ') state = start;

else putchar(c);

break;

case start:

if (c == ' ') state= comment;

else

{

putchar(' ');

state = (c == ' ') ? start :(putchar(c),copy);

}

break;

case comment:

if(c!=' ')

{

putchar(' ');

state = (c == ' ') ? start :(putchar(c),copy);

}

else state = comment;

}

}

return 0;

}

3）运行截图如图2-9所示：

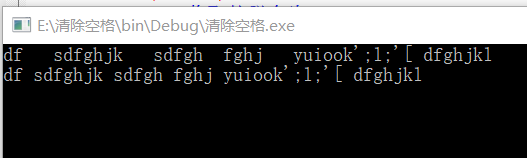


图2-9 运行结果截图

（3）编写一个程序，打印如下的杨辉三角形。

1 /\*第0行 \*/

1 1 /\*第1行 \*/

1 2 1 /\*第2行 \*/

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

1 6 15 20 15 6 1

1 7 21 35 35 21 7 1

1 8 28 56 70 56 28 8 1

1 9 36 84 126 126 84 36 9 1

每个数据值可以由组合计算（表示第i行第j列位置的值），而的计算如下：

 (i=0,1,2,…)

 (j=0,1,2,3,…,i)

本程序中为了打印出金字塔效果，要注意空格的数目。一位数之间是3个空格，两位数之间有2个空格，3位数之间只有一个空格，程序编制过程中要注意区分。

解答：

1）编程题（3）的流程如下：

1 设计i,j;数组a[10][10]

2 i初值为1，如果此时i<10,则执行步骤3，i的值加1。如果不成立，则执行步骤6

3 将a[i][0],a[i][i]均设为1，并判断i的值是否大于1，如果大于1，执行步骤4，否则返回步骤2

4 j的初值为1，当j的值不等于i时，j的值加1，执行步骤5.否则返回步骤2

5 将a[i-1][j-1]+a[i-1][j]赋值给a[i][j]

6 i的初值为1，如果i的值小于10，则执行步骤7，且i的值加1，否则程序结束

7 设置u，且当u的值小于19-2\*i时，输出空格，且u的值加1.否则执行步骤6

8 j的初值为1，当j小于或等于i时执行步骤9，j的值加1.否则开始执行步骤6

9 判断j的值与i是否相等，相等则执行步骤10，否则执行步骤11

10 输出a[i][j],且换行

11 输出a[i][j]

2）编程题（3）的程序如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

int i,j;

int a[10][10];

a[0][0]=1;

for(i=1;i<10;i++)

{

a[i][0]=1;

a[i][i]=1;

if(i>1)

{

for(j=1;j!=i;j++)

{

a[i][j]=a[i-1][j-1]+a[i-1][j];

}

}

}

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

{

int u;

for(u=1;u<19-i\*2;u++)

{

printf(" ");

}

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

{

if(i==j)

{

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

printf("\n");

}

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

}

}

return 0;

}

3）运行结果截图如图2-10所示：

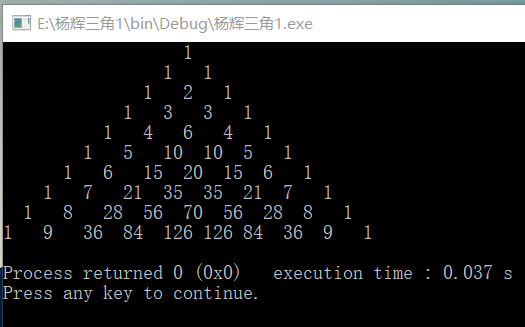


图2-10 运行截图

（4）编写一个程序，将用户输入的任意正整数逆转，例如，输入1234，输出4321。

解答：

1）编程题（4）的流程图如图2-11所示：

输入正整数x

设置中间量t

X！=0 ?

否

是

t=x%10;

x=x/10;

输出t

图2-11 编程题（4）的流程图

2）编程题（4）的程序如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

unsigned int x;

scanf("%u",&x);

while(x!=0)

{

int t;

t=x%10;

x=x/10;

printf("%d",t);

}

printf("\n");

return 0;

}

3）编程题（4）的测试数据：

输入12354 则应该输出45321

4）编程题（4）的运行结果如图2-12所示：

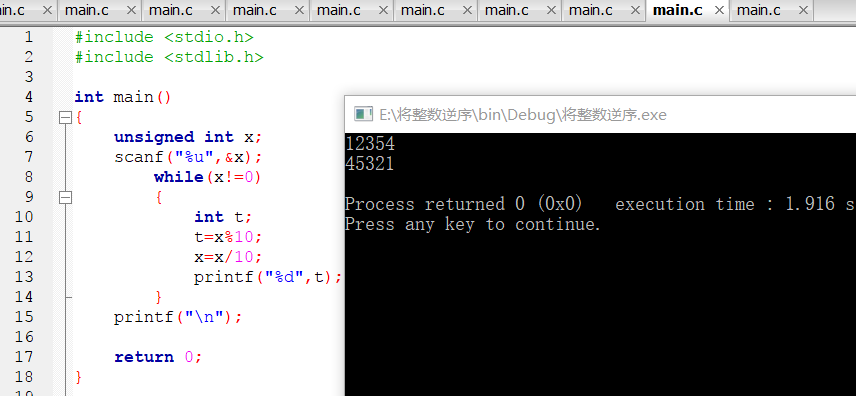


图2-12 运行截图

### 2.2.4 选做题

编写并上机调试运行能实现以下功能的程序。

编写一个程序，用牛顿迭代法求方程*f(x)=*3*x3-*4*x3-5x+*13*=0*满足精度e=10-6的一个近似根，并在屏幕上输出所求近似根。

牛顿迭代法求方程近似根的迭代公式为：

，

其中, *f '(x)*是函数*f(x)*的导函数。牛顿迭代法首先任意设定的一个实数来作为近似根的迭代初值*x*0，然后用迭代公式计算下一个近似根*x*1。如此继续迭代计算*x*2, *x*3, …, *x*n, 直到 |*x*n- *x*n-1|≤精度e，此时值*x*n即为所求的近似根。

解答：

（1）流程如下：

1）f(x)=(((3\*x-4)\*x)-5)\*x+13 f’(x)=((9\*x-8)\*x-5)

2) 设置一个中间量d,且d=-((((3\*x-4)\*x)-5)\*x+13)/((9\*x-8)\*x-5)

3) x = x + d

4) 检查d，如果|d|>e,转到步骤2),否则转到3)

（2）程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define EPS 1e-6

int main()

{

double x,d;

printf("请输入初始值\n");

scanf("%lf",&x);

do

{

d = - ((((3 \* x - 4) \* x) - 5) \* x + 13)/((9 \* x - 8) \* x - 5);

x = x + d;

} while(fabs(d) > EPS);

printf("x=%lf\n",x);

return 0;

}

（3）测试数据：

输入1，对应结果为：-1.548910

（4）运行结果如图2-13所示：

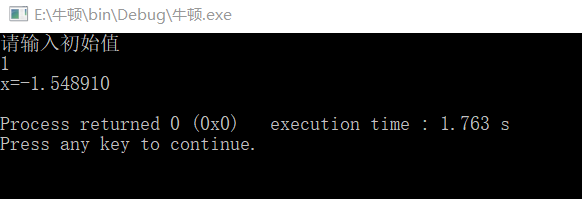


图2-13 运行结果截图

2.3 实验小结

实验中遇到的问题，如何解决的，以及体会

问题：

程序改错题并不困难，困难的是用while语句和do-while语句进行替换，因为0的阶乘和1的阶乘均是1，由于开始并没有意识到这个问题，在程序写完之后还以为自己是正确的。后来，在测试0和1的时候发现了问题。

在这个题的基础上判断大于S值的最小n的阶乘对应的n的值，我在这个问题上犯的错是：n最后多加了一个1。

编程设计题第一个，用if和switch语句来编写，我的问题是不知道收税的机制，也就是只乘了一个税率，而不是每个工资段都进行收税。

编程设计题第二个清除多余的空格和第三个题打印杨辉三角，没有思路。

如何解决的：

将0的阶乘这种特殊情况单独列出来，单独考虑，单独打印。

对于判断大于S值的最小n的阶乘对应的n的值，注意加1的次数和>=符号的运用就解决了。

编程题第一个是听到同学不一样的观点开始在网上查阅资料，最后的将程序修改为正确的，这个步骤并没有遇到很大的难题。

清除多余空格的这道题，计算思维老师在课堂上曾介绍过图灵机的思想，用状态和几个基本动作来实现问题的解决。而且，在C语言程序与设计的教材上有一个清除注释的问题，和这个问题的思想很相近，在参考教材的情况下，先打印出教材上的程序，理解关键步骤后，运用switch语句最终解决了这个难题。

打印杨辉三角这个题，我首先考虑的使用循环来写，但是，仅仅运用循环我无法解决数值的变换，空格的多少，以及何时归换行。最终，我参考教材第七章的数组，结合第四章的for语句，以及在网课上看到的打印特定图形时控制输出空格的方法，最终得以解决。

体会：

很多看似简单的问题，也有着一些神奇之处，如果稍不注意就会出错，也许这个错误很小，但是却可能导致整个程序不能正常运行。

学习不仅仅可以在课堂上听老师讲课，知识的获取途径是多种多样的，多学广益，学习涉及到的知识面的宽度决定了思路的宽度。

厚积才能薄发，我应该坚持学习C语言，不懂的问题就要搞懂，随着积累，我相信我可以学好这门课程。

# 实验3 函数与程序结构实验

3.1 实验目的

（1）熟悉和掌握函数的定义、声明；函数调用与参数传递方法；以及函数返回值类型的定义和返回值使用。

（2）熟悉和掌握不同存储类型变量的使用。

（3）熟悉多文件编译技术。

3.2 实验内容

### 3.2.1 源程序改错题

下面是计算s=1!+2!+3!+…+n!的源程序，在这个源程序中存在若干语法和逻辑错误。要求在计算机上对这个例子程序进行调试修改，使之能够正确完成指定任务。

1 #include "stdio.h"

2 void main(void)

3 {

4 int k;

5 for(k=1;k<6;k++)

6 printf("k=%d\tthe sum is %ld\n",k,sum\_fac(k));

7 }

8 long sum\_fac(int n)

9 {

10 long s=0;

11 int i;

12 long fac;

13 for(i=1;i<=n;i++)

14 fac\*=i;

15 s+=fac;

16 return s;

17 }

解答：

1）在第1行和第2行之间添加函数声明long sum\_fac(int n);

2）将第2行void main(void)，改为int main(),并且在16行和17行之间添加return 0;

3）第12行，正确形式为：long fac=1;

4）第13行的for语句应有大括号，包括第14和15行

运行结果如图3-1所示：

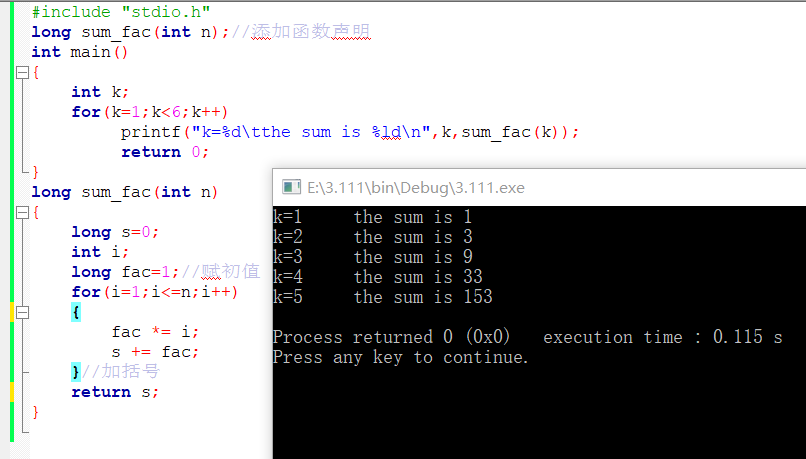


图3-1 运行结果截图

### 3.2.2 源程序修改替换题

（1）修改第1题中sum\_fac函数，使其计算量最小。

解答：

1）源程序如下：

#include "stdio.h"

long sum\_fac(int n);

int main()

{

int k;

for(k=1;k<6;k++)

printf("k=%d\tthe sum is %ld\n",k,sum\_fac(k));

return 0;

}

long sum\_fac(int n)

{

static long s=0;

static long fac=1;

fac \*= n;

s += fac;

return s;

}

2）运行结果如图3-2所示：

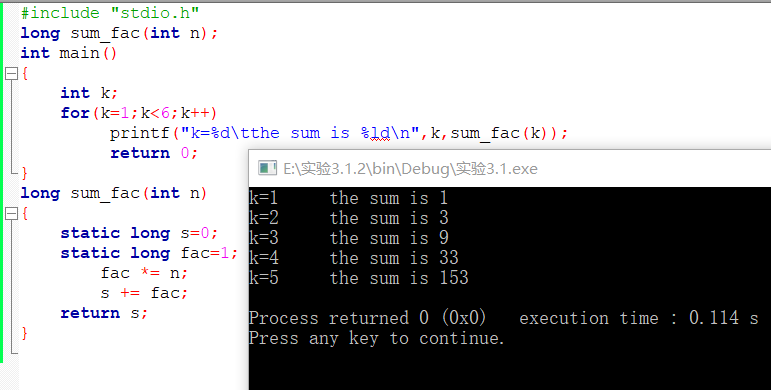


图3-2 运行结果截图

（2）修改第1题中sum\_fac函数，计算。

解答：

1）源程序如下：

#include "stdio.h"

double sum\_fac(int n);

int main()

{

int k;

for(k=1;k<6;k++)

printf("k=%d\tthe sum is %lf\n",k,sum\_fac(k));

return 0;

}

double sum\_fac(int n)

{

double s=0;

int i;

double fac=1;

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

{

fac \*= i;

s = s +((1.0)/fac);

}

return s;

}

2）运行结果如图3-3所示：

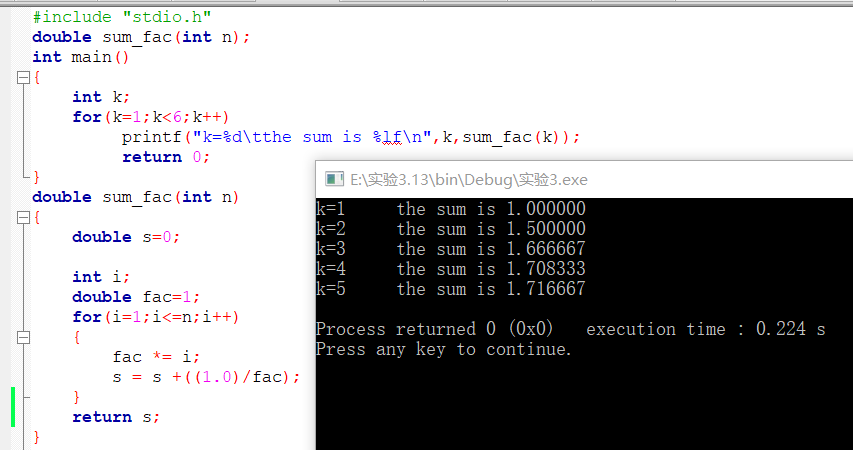


图3-3运行结果截图

### 3.2.3 跟踪调试题

下面是计算fabonacci数列前n项和的源程序，现要求单步执行该程序，观察p,i,sum,n值，即：

（1）刚执行完scanf("%d",&k);语句，p,i值是多少？

解答:

P值为0x60ff00,i的值为1

如图3-4所示

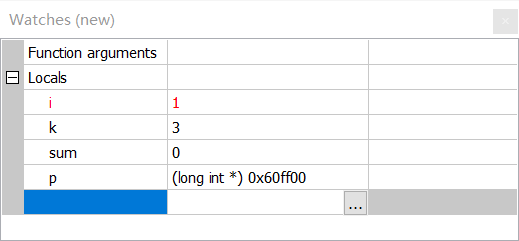


图3-4 结果截图

（2）从fabonacci函数返回后光条停留在哪个语句上？

解答：

TIM图片20171127210303

图3-5 结果截图

（3）进入fabonacci函数时，watch窗口显示的是什么？

解答：

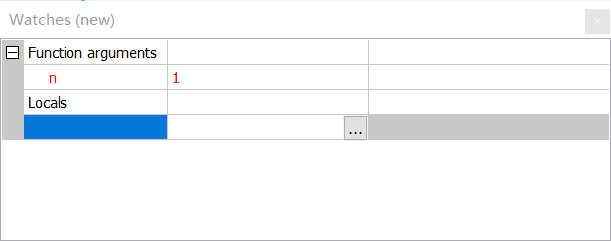


图3-6 结果截图

（4）当i=3时，从调用fabonacci函数到返回，n值如何变化？

解答：

n的值从3变为1

如图3-7和图3-8所示

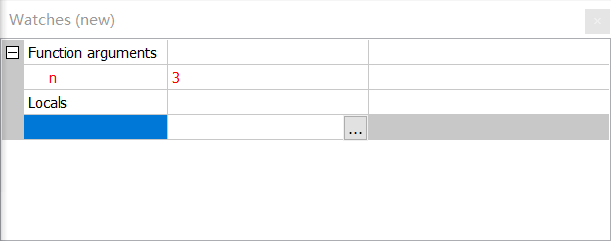


图3-7 结果截图

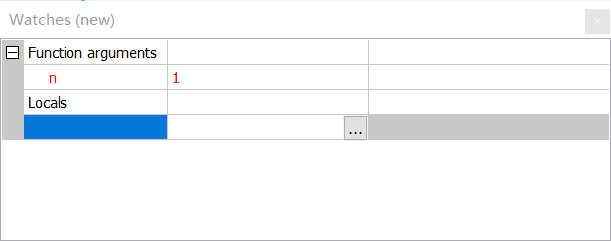


图3-8 运行截图

**源程序**

void main(void)

{

int i,k;

long sum=0,\*p=&sum;

scanf("%d",&k);

for(i=1;i<=k;i++){

sum+=fabonacci(i);

printf("i=%d\tthe sum is %ld\n",i,\*p);

}

}

long fabonacci(int n)

{

if(n==1 || n==2)

return 1;

else

return fabonacci(n-1)+fabonacci(n-2);

}

其中，long sum=0,\*p=&sum;声明p为长整型指针并用&sum取出sum的地址对p初始化。\*p表示引用p所指的变量（\*p即sum）。

### 3.2.4 编程设计题

（1）编程让用户输入两个整数，计算两个数的最大公约数并且输出之（要求用递归函数实现求最大公约数）。同时以单步方式执行该程序，观察递归过程。

解答：

1）流程图如图3-9和3-10所示：

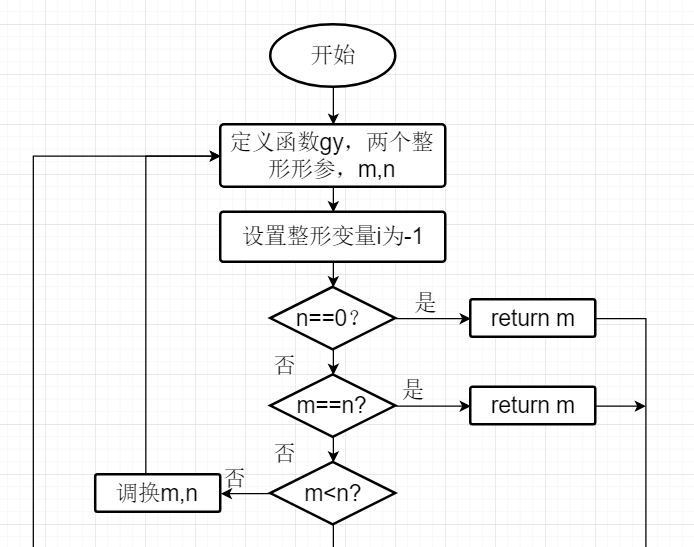


图3-9 编程题（1）流程图

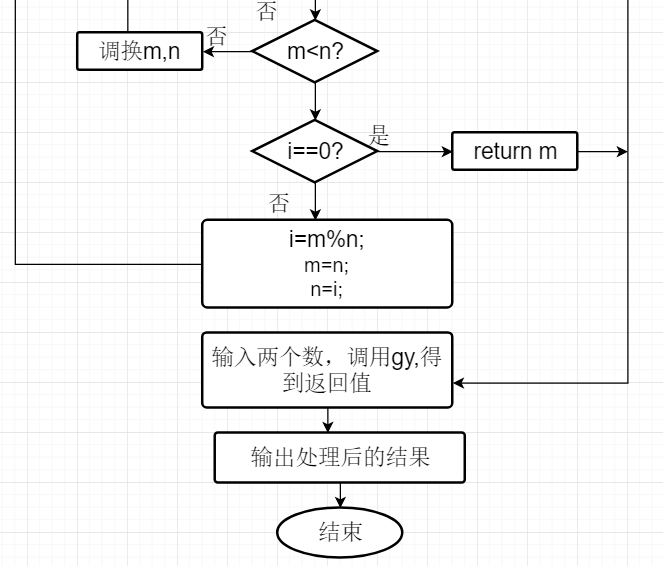


图3-10 编程题（1）流程图

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

int gy(int m,int n)

{

int i=-1;

if(m<n) gy(n,m);

if(n==0) return m;

if(m==n) return m;

else

{

for(;i!=0;)

{

i= m%n;

m=n;

n=i;

gy(m,n);

}

}

return m;

}

int main()

{

printf("请输入两个整数\n");

int j,k;

scanf("%d %d",&j,&k);

printf("%d\n",gy(j,k));

return 0;

}

3）测试数据：

输入30 20，结果为10

输入45 27，结果为9

4）运行结果如图3-11和图3-12所示：

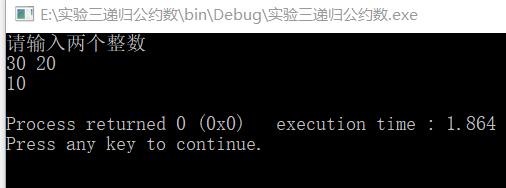


图3-11 编程题（1）运行截图

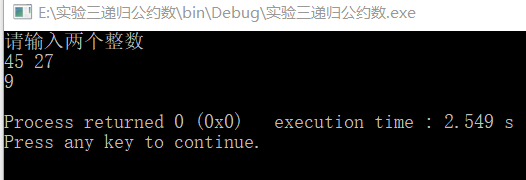


图3-12 编程题（1）运行截图

（2）编程验证歌德巴赫猜想：一个大于等于4的偶数都是两个素数之和。

解答：

1）流程如下：

1.设置函数GD,主函数设置n的值为4，当n<=100,执行步骤2，否则程序结束。

2.设置变量i，i的初值为2，如果i<=n/2,执行步骤3，否则i的值加1后，再次执行步骤2

3.调用GD函数两次，参数分别为i和n-i，如果返回值均为1，打印出等式。否则n的值加2，返回步骤1，判断n<=100? 满足条件则执行步骤2，否则程序结束。.

4.函数GD，形参为整型n;如果n<2,返回0.如果n==2，返回1.如果n>2，执行步骤5；

5.设置变量i，i的初值为2，如果i\*i<=n，执行步骤6；否则返回值为1.

6.如果n%i==0，则返回0否则返回值为1；之后i的值加1，返回步骤5，进行判断。

2）源程序如下：

#include <stdio.h>

int GD(int);

int main() {

int n;

for (n = 4; n <= 100; n += 2) {

int i;

for (i = 2; i <= n / 2; i++)

if (GD(i) && GD(n - i)) {

printf("%d = %d + %d\n", n, i, n - i);

break;

}

}

return 0;

}

int GD(int n) {

int i;

if (n < 2)

return 0;

if (n == 2)

return 1;

else

for (i = 2; i \* i <= n; i++)

if (n % i == 0)

return 0;

return 1;

}

3）运行结果如图3-13所示：



图3-13 编程题（2）运行截图

（3）编写一个程序，证明对于在符号常量BEGIN和END之间的偶数这一猜测成立。例如，如果BEGIN为10，END为20，程序的输出应为：

GOLDBACH'S CONJECTURE:

Every even number n>=4 is the sum of two primes.

10=3+7

12=5+7

……

20=3+17

解答：

1）流程如下：

1.设置函数GD,主函数，输入两个数BEGIN和END，如果BEGIN为奇数，则n=BEGIN+1.否则n=BEGIN。，当n<=END,执行步骤2，否则程序结束。

2.设置变量i，i的初值为2，如果i<=n/2,执行步骤3，否则i的值加1后，再次执行步骤2

3.调用GD函数两次，参数分别为i和n-i，如果返回值均为1，打印出等式。否则n的值加2，返回步骤1，判断n<=END? 满足条件则执行步骤2，否则程序结束。.

4.函数GD，形参为整型n;如果n<2,返回0.如果n==2，返回1.如果n>2，执行步骤5；

5.设置变量i，i的初值为2，如果i\*i<=n，执行步骤6；否则返回值为1.

6.如果n%i==0，则返回0否则返回值为1；之后i的值加1，返回步骤5，进行判断。

2）源程序如下：

#include <stdio.h>

int GD(int);

int main()

{

int n,m,BEGIN,END;

scanf("%d %d",&BEGIN,&END);

n=BEGIN;

m=END;

printf("GOLDBACH'S CONJECTURE:\nEvery even number n>=4 is the sum of two primes.\n");

for(n = 4; n <= m; n += 2) {

int i;

for(i = 2; i <= n / 2; i++)

if(GD(i) && GD(n - i))

{

printf("%d = %d + %d\n", n, i, n - i);

break;

}

}

return 0;

}

int GD(int n)

{

int i;

if (n < 2)

return 0;

if (n == 2)

return 1;

else

for (i = 2; i \* i <= n; i++)

if (n % i == 0)

return 0;

return 1;

}

3）测试数据：

输入10 20，结果为

4=2+2；

6=3+3；

8=3+5；

10=3+7；

12=5+7；

14=3+11；

16=3+13；

18=5+13；

20=3+17；

4）运行结果如图3-14所示：

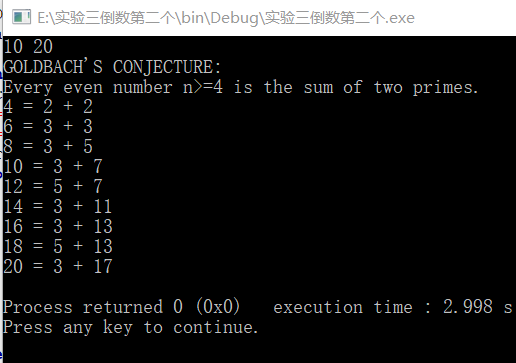


图3-14 编程题（3）运行截图

### 3.2.5 选做题

假设一个C程序由file1.c和file2.c两个源文件及一个file.h头文件组成，file1.c、file2.c和file.h的内容分别如下所述。试编辑该多文件C程序，并编译和链接。然后运行生成的可执行文件。

源文件file1.c的内容为：

#include "file.h"

int x,y; /\* 外部变量的定义性说明 \*/

char ch; /\* 外部变量的定义性说明 \*/

int main(void)

{

x=10;

y=20;

ch=getchar();

printf("in file1 x=%d,y=%d,ch is %c\n",x,y,ch);

func1();

return 0;

}

源文件file2.c的内容为：

#include "file.h"

void func1(void)

{

x++;

y++;

ch++;

printf("in file2 x=%d,y=%d,ch is %c\n",x,y,ch);

}

头文件file.h的内容为：

#include <stdio.h>

extern int x,y; /\* 外部变量的引用性说明 \*/

extern char ch; /\* 外部变量的引用性说明 \*/

void func1(void); /\* func1函数原型 \*/

解答：

1）流程图如图3-15所示：

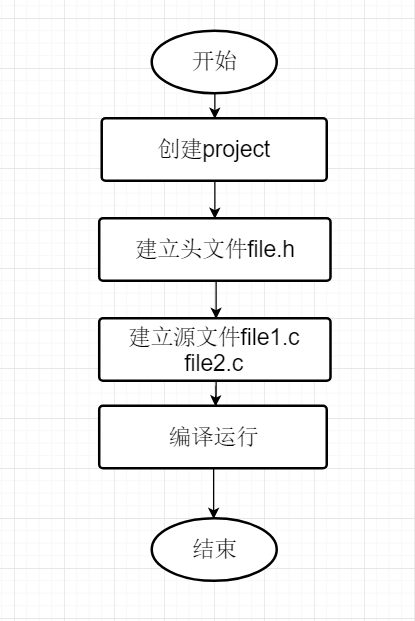


图3-15 选做题流程图

2）运行结果如图3-16和图3-17所示：

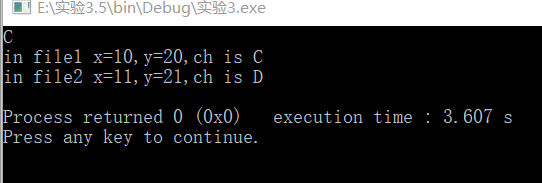


图3-16 运行结果截图

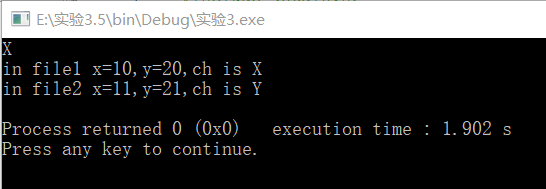


图3-17 运行结果截图

3.3 实验小结

**3.3.1 遇到的问题**：

1）由于自己写函数时，都习惯性的在main函数与头文件之间写，所以并未意识到函数声明的问题，结果编译运行时一直报错。

2）源程序替换第一题，使其求n的阶乘和的运算量最小。需要使用Static静态变量，使用后并没有去除函数中的for语句，而当时并未意识到这个问题。

3）用递归方法求两个整数的最小公约数，写程序的时候出了些错误，输入两个整数结果不正确，上机实验时才发现。

4）验证哥德巴赫猜想那道题没有思路。

**3.3.2 解决方法**：

1）查阅课本中关于函数的那一节，函数声明和函数定义都写上后，最终该问题被解决。

2）在上机实验时请助教检查，助教指出了static静态变量使用不正确的问题，在助教的指导下，最终解决了该问题。

3）递归函数未写正确，在将for语句改为if判断语句，并且将return n;改为return m;后，最终正确。

4）自己搜索资料，如何判断一个数是否为素数，用C语言如何判断一个数是素数，最后通过定义函数，最终解决了该问题。

**3.3.3、有何体会**：

1）要多看课本，课本上的例题很多都是经典且很基础的内容，多练习例题有助于增强对C语言的学习，以及发现程序错误，减化程序的能力。

2）流程图难画，将思路化成流程图需要技巧和耐心。

3）面对困难问题首先要思考，之后尝试去做，遇到难解的问题要查阅课本，请教老师同学。

# 实验4 编译预处理实验

**4.1 实验目的**

（1）掌握文件包含、宏定义、条件编译、assert宏的使用；

（2）练习带参数的宏定义、条件编译的使用；

（3）练习assert宏的使用；

（4）使用Turbo C 2.0集成开发环境中的调试功能：单步执行、设置断点、观察变量值。

**4.2 实验内容**

### 4.2.1 源程序改错题

下面是用宏来计算平方差、交换两数的源程序，在这个源程序中存在若干语法和逻辑错误。要求在计算机上对这个例子程序进行调试修改，使之能够正确完成指定任务。

1 #include "stdio.h"

2 #define SUM a+b

3 #define DIF a-b

4 #define SWAP(a,b) a=b,b=a

5 void main

6 {

7 int b, t;

8 printf("Input two integers a, b:");

9 scanf("%d,%d", &a,&b);

10 printf("\nSUM=%d\n the difference between square of a and square of b

11 is:%d",SUM, SUM\*DIF);

12 SWAP(a,b);

13 Printf("\nNow a=%d,b=%d\n",a,b);

14 }

解答：

（1）程序修改如下：

1）第2行，正确形式为：#define SUM ((a)+(b))

2）第3行，正确形式为：#define DIF ((a)-(b))

3）第4行，正确形式为：#define SWAP(a,b) {int t;t=a;a=b;b=t;}

4）第7行，t为多余的变量，未声明a的变量

（2）错误修改后运行结果如图4-1所示：

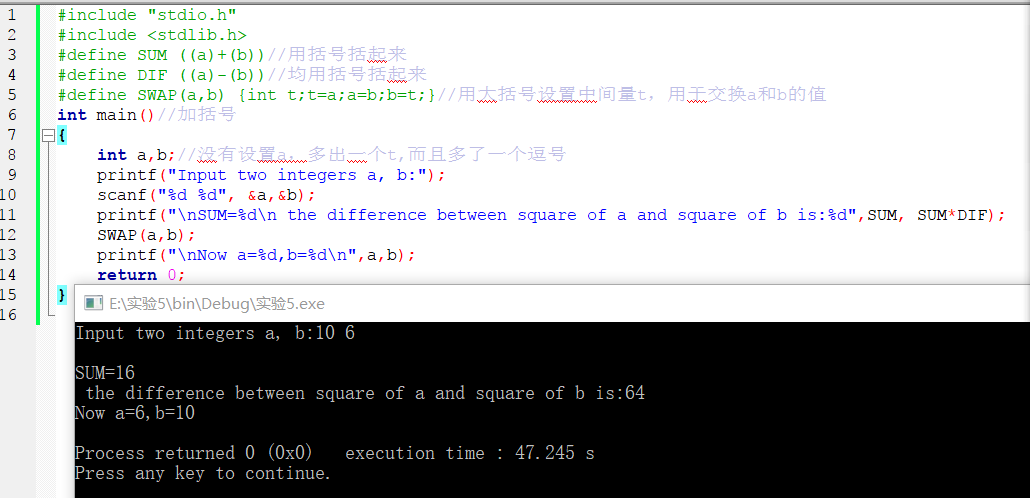


图4-1 程序修改运行截图

### 4.2.2 源程序修改替换题

下面是用函数实现求三个数中最大数、计算两数之和的程序，在这个源程序中存在若干语法和逻辑错误。

要求：（1）对这个例子程序进行调试修改，使之能够正确完成指定任务；

（2）用带参数的宏替换函数max，来实现求最大数的功能。

void main(void)

{

int a, b, c;

float d, e;

printf("Enter three integers:");

scanf("%d,%d,%d",&a,&b,&c);

printf("\nthe maximum of them is %d\n",max(a,b,c));

printf("Enter two floating point numbers:");

scanf("%f,%f",&d,&e);

printf("\nthe sum of them is %f\n",sum(d,e));

}

int max(int x, int y, int z)

{

int t;

if (x>y)

t=x;

else

t=y;

if (t<z)

t=z;

return t;

}

float sum(float x, float y)

{

return x+y;

}

解答：

1）程序修改后源程序如下：

#include <stdio.h>

#include <stdlib.h>//加上头文件

double sum(double x, double y);

int max(int x, int y, int z);

int main()

{

int a, b, c;

double d, e;

printf("Enter three integers:");

scanf("%d %d %d",&a,&b,&c);

printf("\nthe maximum of them is %d\n",max(a,b,c));

printf("Enter two floating point numbers:");

scanf("%lf %lf",&d,&e);

printf("\nthe sum of them is %f\n",sum(d,e));

return 0;

}

int max(int x, int y, int z)

{

int t;

if (x>y)

t=x;

else

t=y;

if (t<z)

t=z;

return t;

}

double sum(double x, double y)

{

return x+y;

}

2）修改后运行结果如图4-2所示：

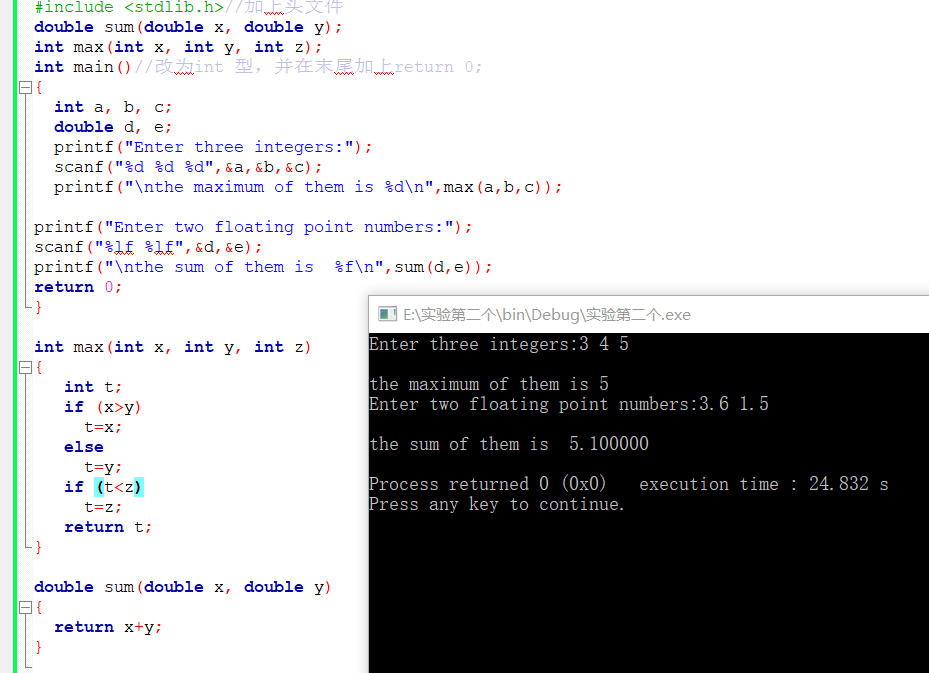


图4-2 运行截图

3）替换后源程序如下：

#include <stdio.h>

#include <stdlib.h>

#define max(a,b,c) ((a>b)?((a>c)?a:c):((b>c)?b:c))

double sum(double x, double y);

int main()

{

int a, b, c;

double d, e;

printf("Enter three integers:");

scanf("%d %d %d",&a,&b,&c);

printf("\nthe maximum of them is %d\n",max(a,b,c));

printf("Enter two floating point numbers:");

scanf("%lf %lf",&d,&e);

printf("\nthe sum of them is %f\n",sum(d,e));

return 0;

}

double sum(double x, double y)

{

return x+y;

}

4）程序替换后结果如图4-3所示：

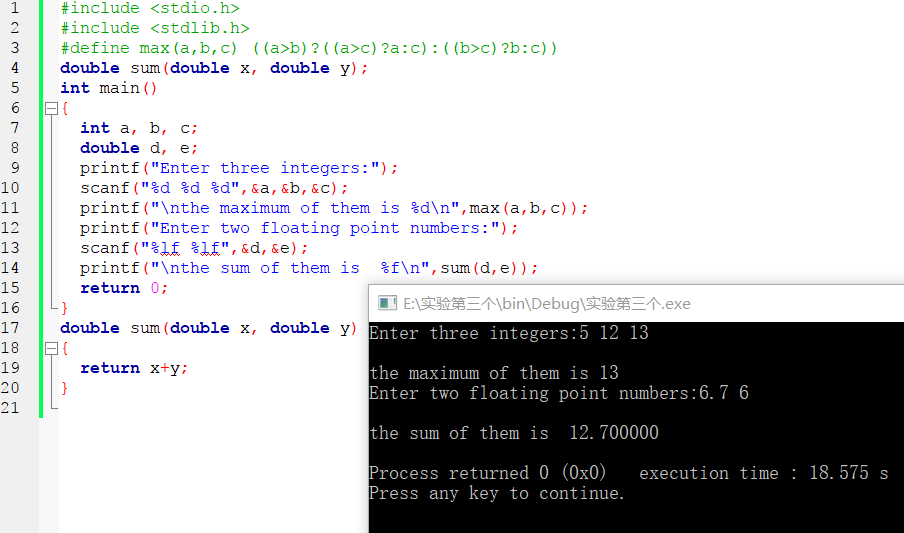


图4-3 运行结果截图

### 4.2.3 跟踪调试题

下面程序利用R计算圆的面积s，以及面积s的整数部分。现要求：

（1）修改程序，使程序编译通过且能运行；

（2）单步执行。进入函数decimal\_fraction时watch窗口中x为何值？在返回main时, watch窗口中i为何值？

（3）排除错误，使程序能正确输出面积s值的整数部分，不会输出错误信息assertion failed。

#define R

void main(void)

{

float r, s;

int s\_integer=0;

printf ("input a number: ");

scanf("%f",&r);

#ifdef R

s=3.14159\*r\*r;

printf("area of round is: %f\n",s);

s\_integer= integer\_fraction(s);

printf("the integer fraction of area is %d\n", s\_integer);

assert((s-s\_integer)<1.0);

#endif

}

解答：

1）修改后的源程序为：

#include <stdio.h>

#include <stdlib.h>

#include <assert.h>

#define R

int integer\_fraction(int x);

int main()

{

float r,s;

int s\_integer=0;

printf ("input a number: ");

scanf("%f",&r);

#ifdef R

s=3.14159\*r\*r;

printf("area of round is: %f\n",s);

s\_integer= integer\_fraction(s);

printf("the integer fraction of area is %d\n", s\_integer);

assert((s-s\_integer)<1.0);

#endif

return 0;

}

int integer\_fraction(int x)

{

int i=x;

return i;

}

int integer\_fraction(float x)

{

int i=x;

return i;

}

2）程序修改后运行结果如图4-4所示：

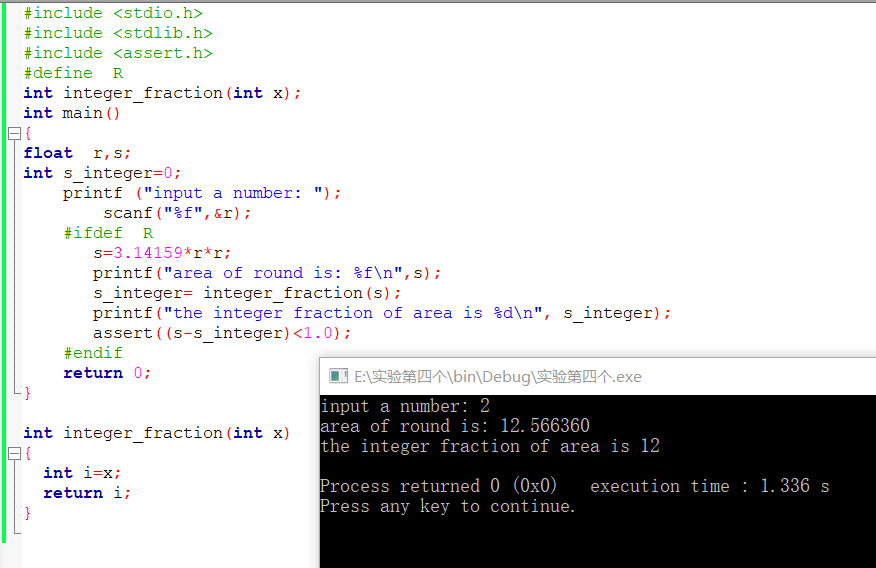


图4-4 程序修改后运行截图

3）该步骤值为：

进入时为3.14159

返回时I为3

### 4.2.4 程序设计题

（1）三角形的面积是，其中，a,b,c为三角形的三边，定义两个带参数的宏，一个用来求s，另一个用来求area。编写程序，用带参数的宏来计算三角形的面积。

解答：

1）流程图如图4-5所示：

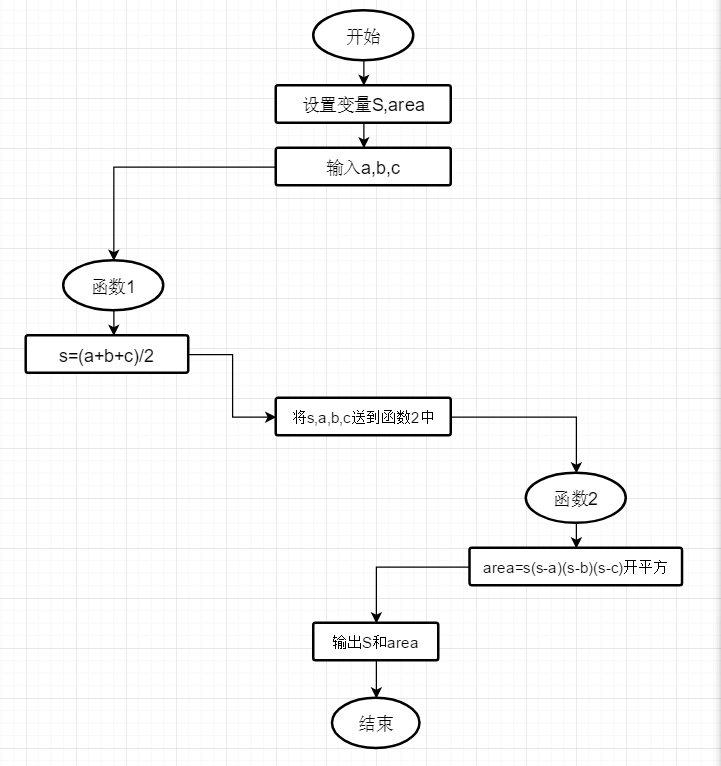


图4-5 编程题（1）流程图

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define s(a,b,c) ((a)+(b)+(c))/2

#define area(t,a,b,c) sqrt(t\*(t-a)\*(t-b)\*(t-c))

int main()

{

printf("请输入三角形的三条边\n");

double a,b,c,t;

scanf("%lf %lf %lf",&a,&b,&c);

t=s(a,b,c);

printf("三角形的周长为%lf\n",2\*t);

printf("三角形面积为%lf\n",area(t,a,b,c));

return 0;

}

3）运行结果如图4-6所示：

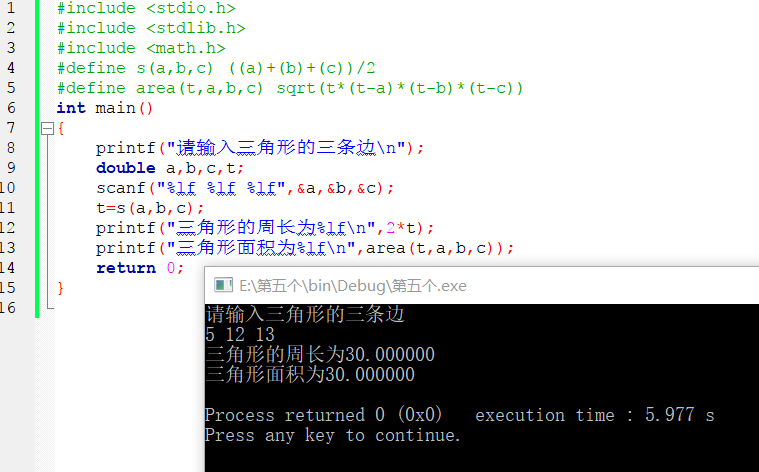


图4-6 运行结果截图

（2）用条件编译方法来编写程序。输入一行电报文字，可以任选两种输出：一为原文输出；二为变换字母的大小写（如小写‘a’变成大写‘A’，大写‘D’变成小写‘d’），其他字符不变。用#define命令控制是否变换字母的大小写。例如，#define CHANGE 1 则输出变换后的s文字，若#define CHANGE 0则原文输出。

1）流程图如图4-7所示：

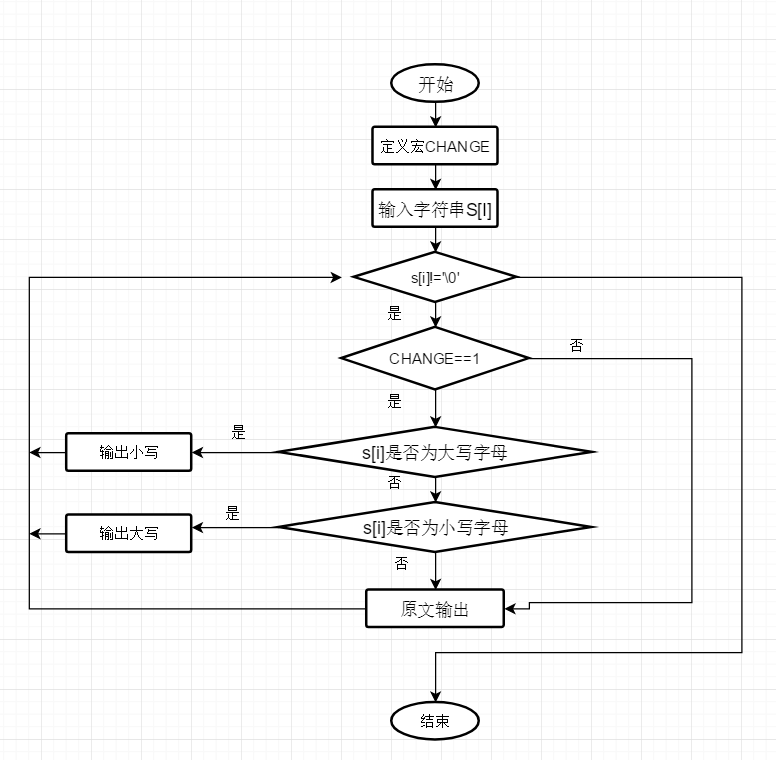


图4-7 编程题（2）流程图

2）源程序如下：

#include <stdlib.h>

#include <stdio.h>

#define CHANGE 1

int main()

{

char s[80];

int i=0;

scanf("%[^\n]",s);

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

{

#if CHANGE

if(s[i]>='A' && s[i]<='Z')

s[i]=s[i]+'a'-'A';

else if(s[i]>='a' && s[i]<='z')

s[i]=s[i]-'a'+'A';

#endif // CHANGE

i++;

}

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

return 0;

}

3）运行结果如图4-8和图4-9所示：

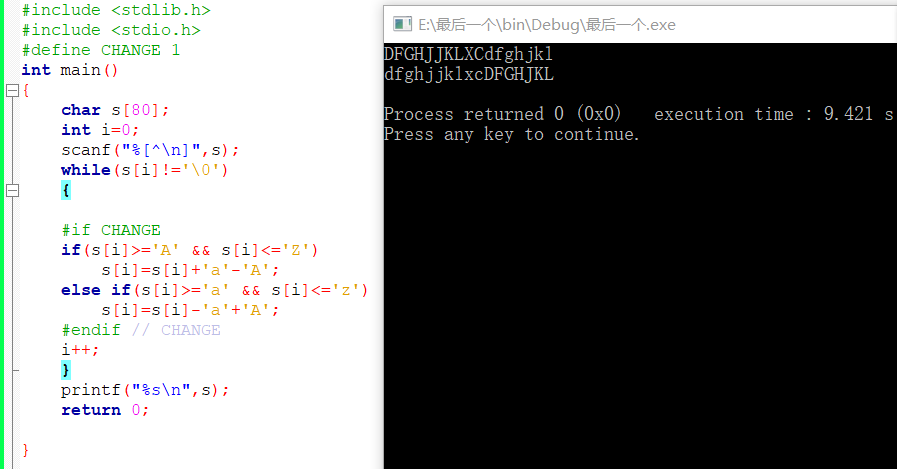


图4-8 变换大小写运行截图

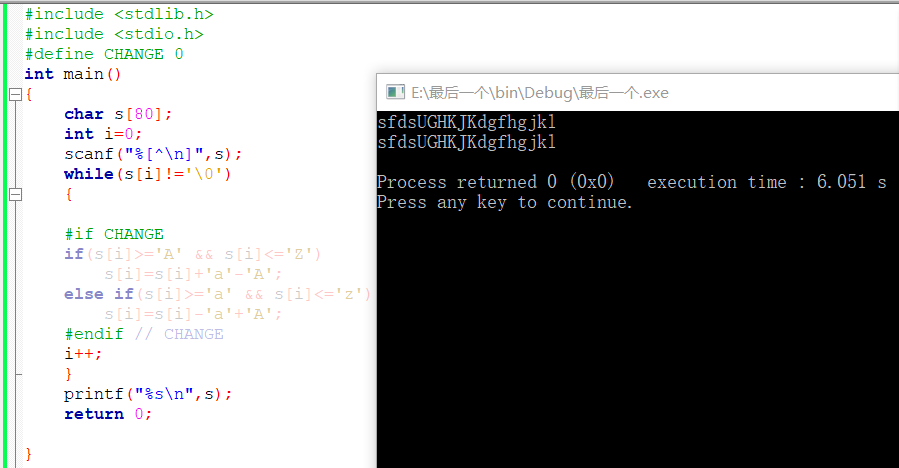


图4-9 不变换大小写截图

**4.3 实验小结**

遇到的问题：

1）对条件编译理解不够深刻，对#if #ifdef之间的区别不够了解

2）不知道还有#include<assert.h>

如何解决：

1）在网上看CSDN博客，有对该问题的解答

2）询问同学关于assert.h

有何体会：

1）C语言的学习不会一蹴而就，它是一个循序渐进的过程，每天都会学到新知识。

2）遇到不会的一定要积极解决，寻求帮助，不能将问题积累。

3）在实验中遇到的大大少少的问题与所犯过的错误将是我以后的要加以注意的地方与认真审查的部分，为我奠定坚固的基石。

4）伴随着学习的深入，我发现高深的东西还有很多很多，不懂不熟的知识还有很多，所以我不能停步。

**实验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

**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）程序改错：

1. 在1行和第2行之间对函数添加函数声明

正确形式为：void sort(int [],int);

2. 第7行正确形式为： sort(&a[0],10);

3. 第13行改为：void sort(int \*b, int n)

4. 第18行改为 ：if(b[j] > b[j+1])

（2）程序运行如图5-1所示：

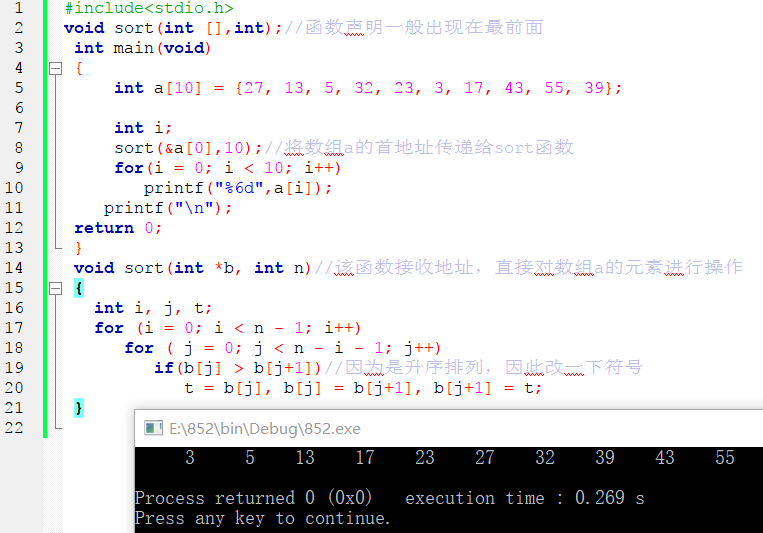


图5-1 运行截图

**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? : ; /\* 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 \*/

;

}

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;

}

解答：

1）程序如下：

#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] ; /\* 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）运行结果如图5-2所示：

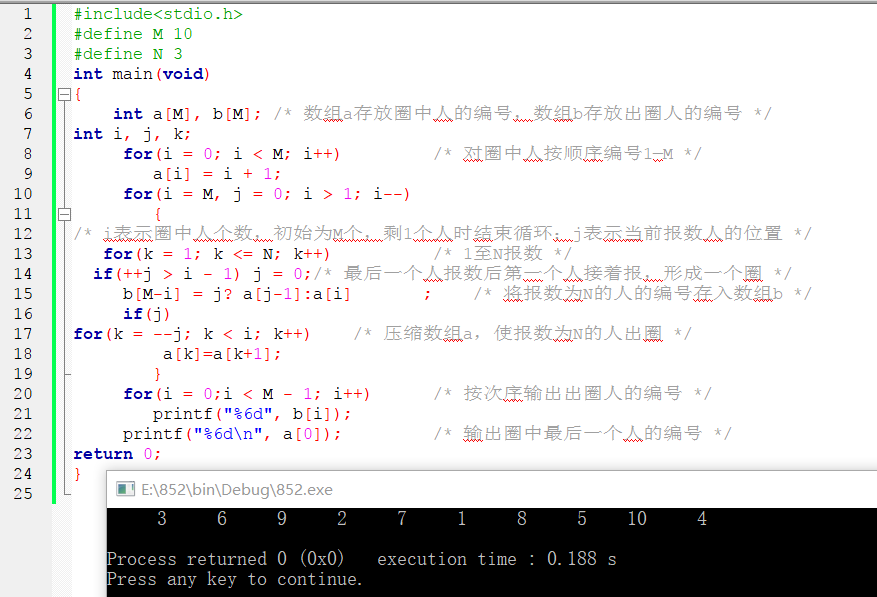


图5-2 运行截图

(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）程序如下：

#include<stdio.h>

#define M 10

#define N 3

int main(void)

{

int a[M],b[M];

int i,j,k=M,l=0;

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

a[i]=i+1;

i=-1;

while(k>1)

{

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

{

do i=(i+1)%M;

while(a[i]==0);

}

b[l]=a[i];

l++;

a[i]=0;

k--;

}

for(j=0;j<M-1;j++)

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

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

if(a[j])

printf("%6d\n",a[j]);

return 0;

}

2）运行结果如图5-3所示：

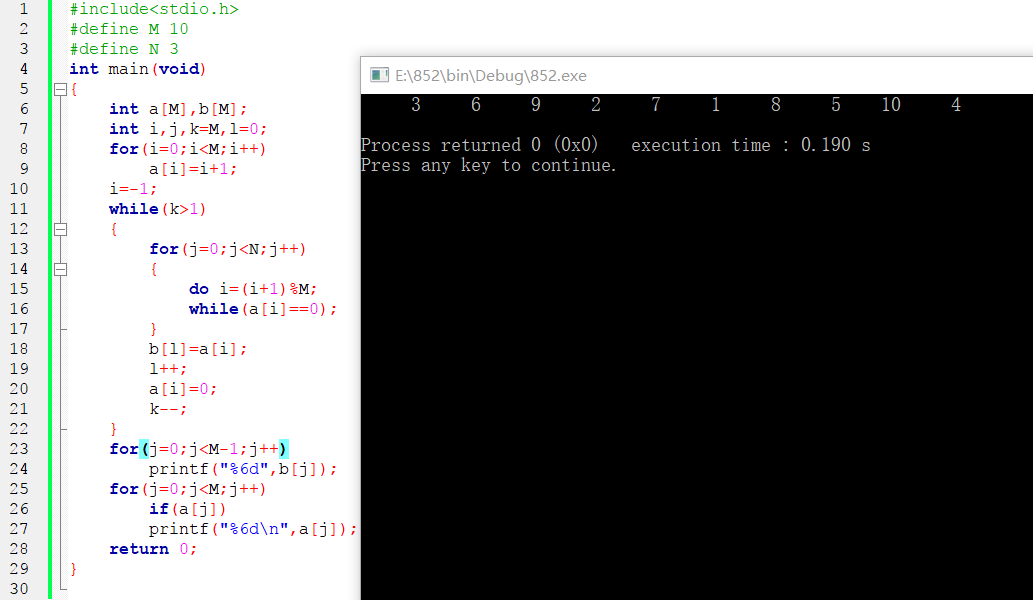


图5-3 运行结果截图

**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：

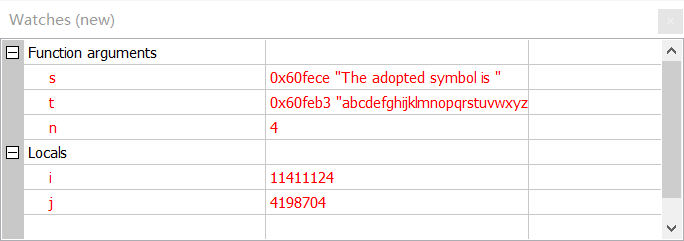


图5-4 运行截图（a）

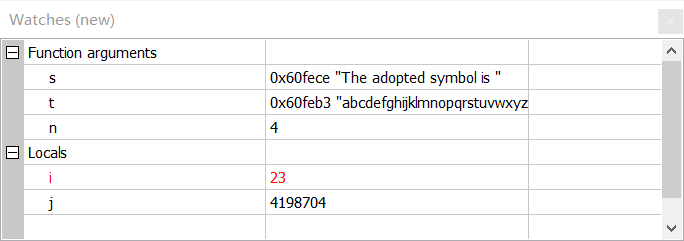


图5-5 运行截图(b)

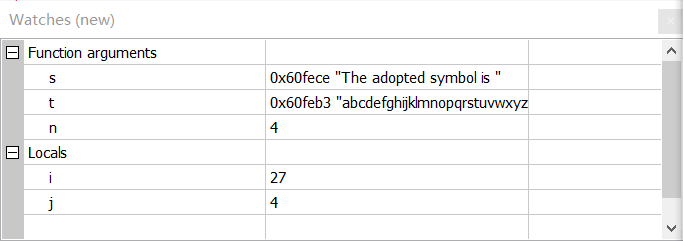


图5-6 运行截图(c)

（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';

}

解答：

1）修改后程序如下：

#include<stdio.h>

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

{

int i = 0, j;

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

i--;

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

{

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

}

s[i+n] = '\0';

}

int main(void)

{

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

trncat(a, b, 4);

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

return 0;

}

2）运行结果如图5-7所示：

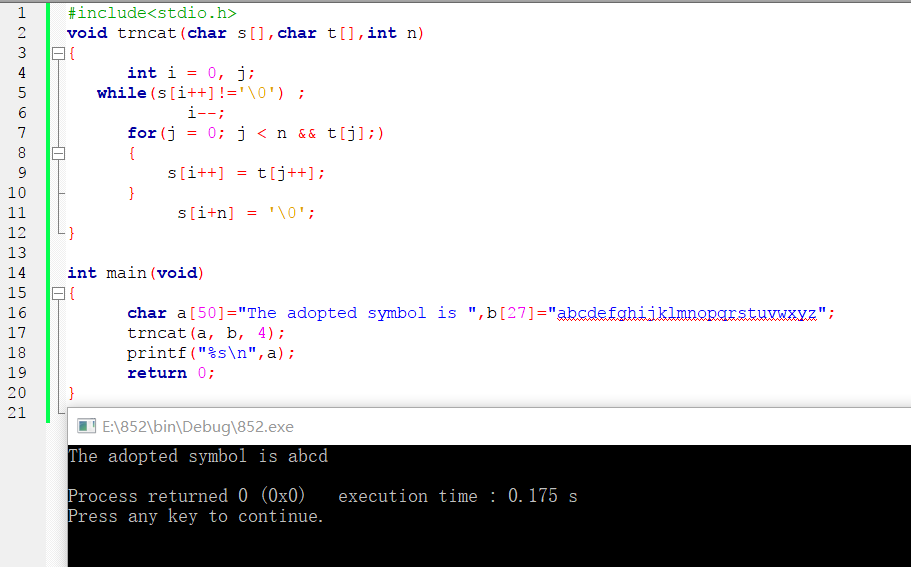


图5-7 运行截图

**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）流程图如图5-8和图5-9所示

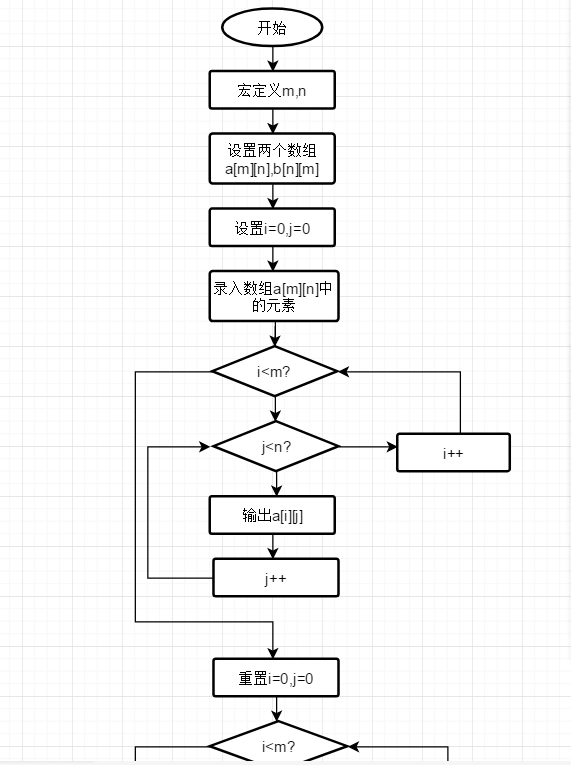


图5-8 编程题（1）流程图

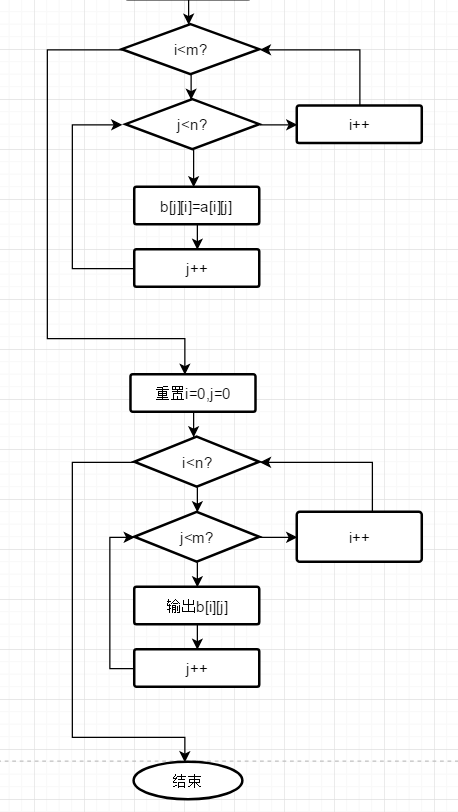


图5-9 编程题（1）流程图

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

#define m 3

#define n 4

int main()

{

int i,j;

int a[m][n],b[n][m];

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

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

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

getchar();

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

{

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

{

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

}

printf("\n");

}

printf("\n");

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

{

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

{

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

}

}

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

{

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

{

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

}

printf("\n");

}

return 0;

}

3）测试数据如下：

1 2 3 8

55 6 9 2

5 6 9 8

输出形式应为：

1 2 3 8

55 6 9 2

5 6 9 8

1 55 5

2 6 6

3 9 9

8 2 8

4）运行结果如图5-10所示：

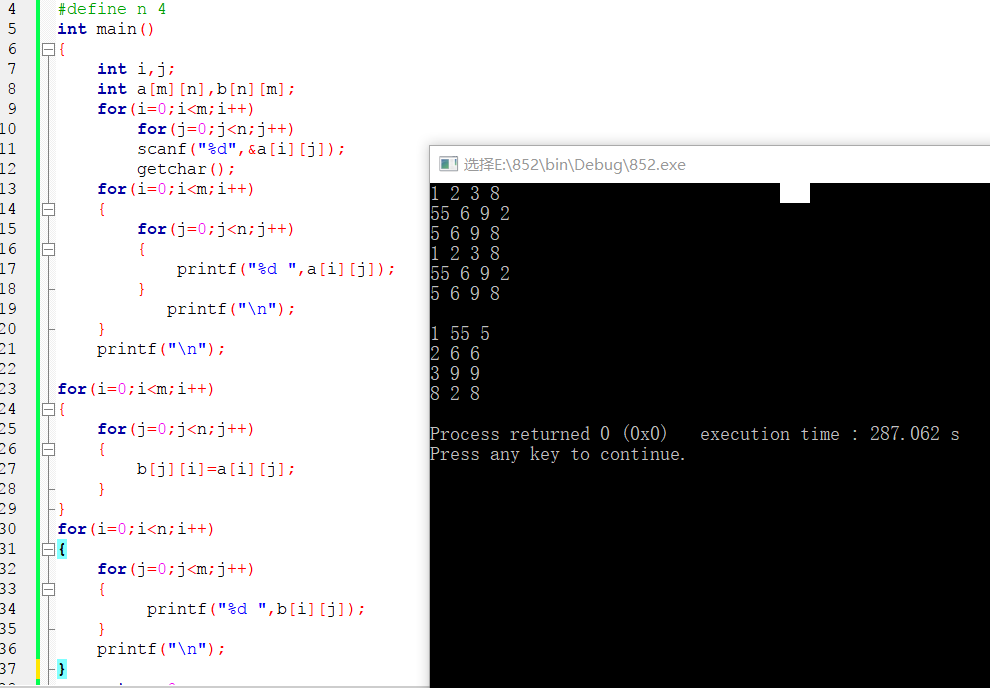


图5-10 运行截图

（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）流程图如图5-11和图5-12所示

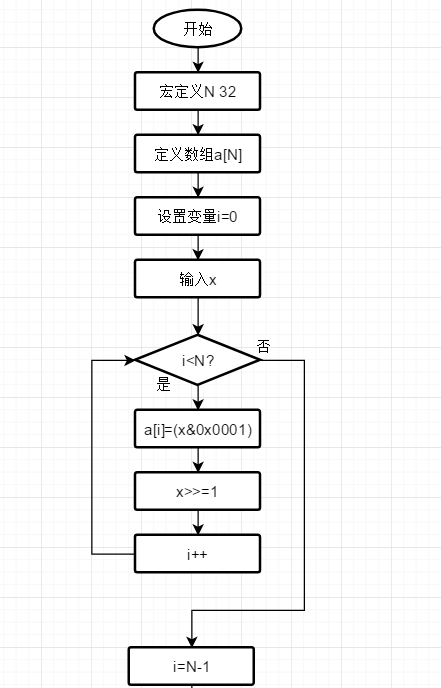


图5-11 编程题（1）流程图

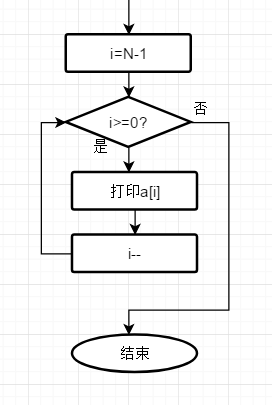


图5-12 编程题（1）流程图

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

#define N 32

int main()

{

unsigned x,a[N];

int i;

scanf("%u",&x);

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

{

a[i]=(x&0x0001);

x>>=1;

}

for(i=N-1;i>=0;i--)

{

printf("%u",a[i]);

}

printf("\n");

return 0;

}

3）测试数据：

输入8 则输出结果为0000 0000 0000 0000 0000 0000 0000 1000

4）运行结果如图5-13所示：

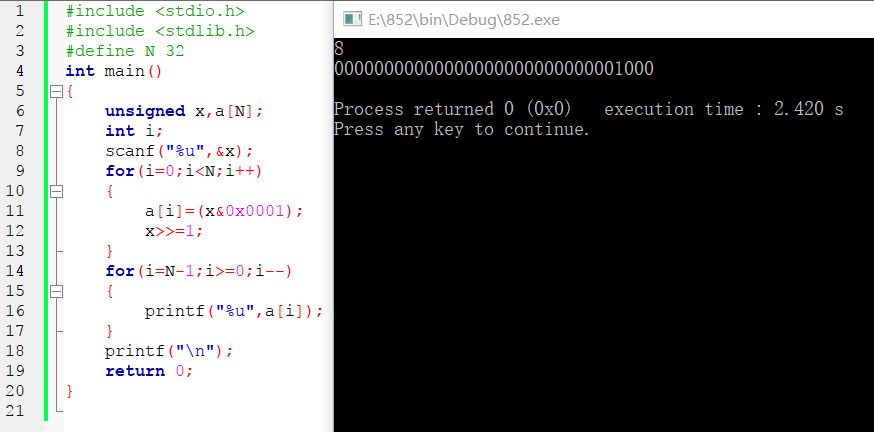


图5-13 运行截图

（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）步骤如下：

1.定义宏N,M，整形变量I,j,temp,b,sco;char类型数组 name[N][M],和a[M],整型数组score[N]

2.录入姓名和成绩

3.i的初值为0，判断i是否小于N-1，如果是，则执行步骤4，否则执行7

4.j的初值为0，判断j是否小于N-i-1，如果是执行步骤5，否则i的值加1，返回步骤3

5.判断score[j]是否小于score[j+1],如果是则执行步骤6，否则j的值加1，返回步骤4

6. temp=score[j];score[j]=score[j+1]；score[j+1]=temp; strcpy(a,name[j]);strcpy(name[j],name[j+1]);

strcpy(name[j+1],a);调用函数strcpy,之后j的值加1，返回步骤4

7. 设置i的初值为0，判断i是否小于N，如果是则执行步骤8，否则执行9

8. 输出name[i],score[i]

9. 录入要查找的成绩sco，将数组score和sco，N作为形参调用函数B，即执行步骤11。设置b为其返回值，得到返回值执行步骤10

10.如果b不等于-1，则输出name[b],score[b]。否则输出not found!

11.设置front=0,back=N-1,middle.当front小于等于back时，执行步骤12，否则返回-1到步骤9.

12.middle= (front+back)/2;判断sco是否大于score[middle],如果是则back=middle-1;如果sco小于score[middle],则front=middle+1;否则返回值middle到步骤9.每次执行完后判断front与back的大小，即步骤11中的判断条件。

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define N 4

#define M 20

int B(int a[],int x,int n)

{

int front=0,back=n-1,middle;

while(front<=back)

{

middle=(front+back)/2;

if(x>a[middle])

back=middle-1;

else if(x<a[middle])

front=middle+1;

else return middle;

}

return -1;

}

int main()

{

int i,j,temp,b,sco;//b为调用函数得到的返回值,sco为所要查询的分数

char name[N][M],a[M]={'\0'};

int score[N];

printf("请分别输入姓名和成绩\n");

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

{

scanf("%[^\n]",name[i]);

getchar();

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

getchar();

}

for(i=0;i<N-1;i++)

{

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

{

if(score[j]<score[j+1])

{

temp=score[j];

score[j]=score[j+1];

score[j+1]=temp;

strcpy(a,name[j]);

strcpy(name[j],name[j+1]);

strcpy(name[j+1],a);

}

}

}

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

{

printf("%s,%4d\n",name[i],score[i]);

}

printf("请输入要查找的成绩\n");

scanf("%d",&sco);

b=B(score,sco,N);

if(b!=-1)

printf("姓名%s,分数%4d\n",name[b],score[b]);

else printf("not found!\n");

return 0;

}

2）测试数据如下：

str 1

88

str2

55

str3

99

str4

66

要查找的数据为66

输出结果为：

Str3 99

Str1 88

Str4 66

Str2 55

姓名str4,分数66

3）运行结果如图5-14所示：

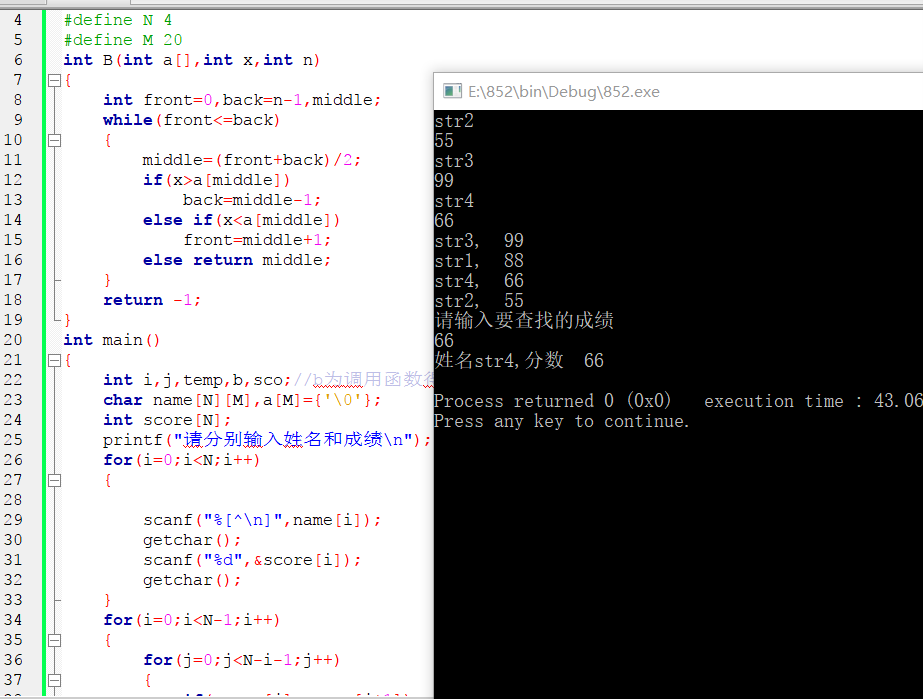


图5-14 运行截图

**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）流程图如图5-15和图5-16所示

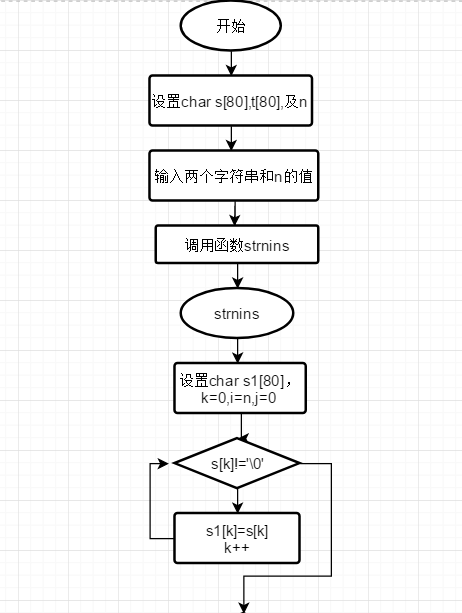


图5-15 选做题（1）流程图

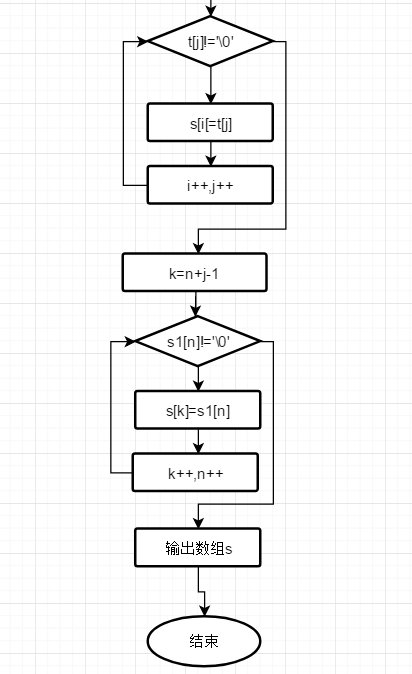


图5-16 选做题（1）流程图

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

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

int main()

{

char s[80],t[80];

int i,j,n;

printf("请输入字符串s和字符串t,以及n的值\n");

scanf("%[^\n]",s);

getchar();

scanf("%[^\n]",t);

getchar();

scanf("%d",&n);

strnins(s,t,n);

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

return 0;

}

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

{

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

char s1[80];

while(s1[k]=s[k++]);

while(s[i++]=t[j++]);

k=n+j-1;

while(s[k++]=s1[n++]);

}

3）测试数据如下：

输入：sfghhjkftufk/&8\*999;;''

mmm

5

输出: sfghhmmmjkftufk/&8\*999;;''

4)运行结果如图5-16所示：

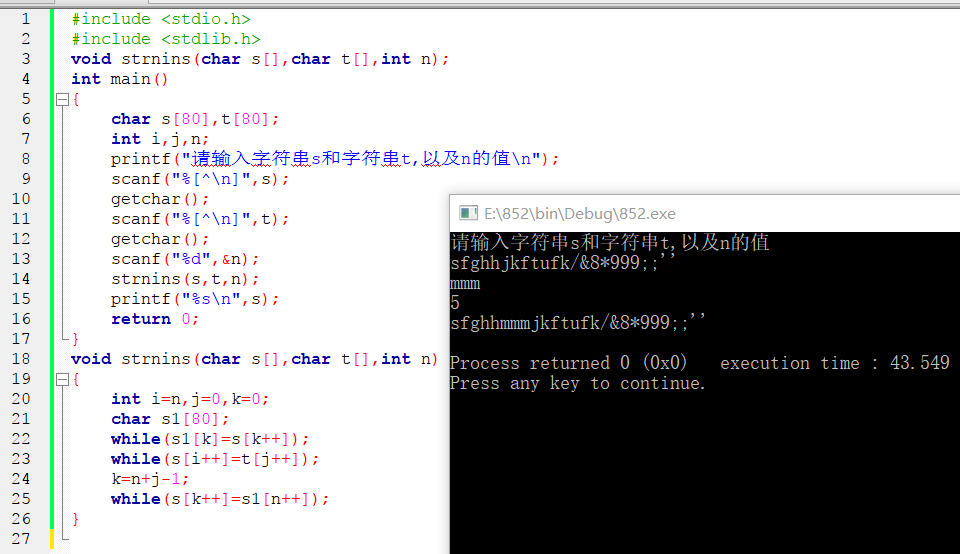


图5-17 运行截图

(2) 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）源程序如下：

#include<stdio.h>

#define TRUE 1

#define FALSE 0

#define MAX 8

void print(char queen[]);

int judge(int index);

static char queen[MAX] = {};

static int counter = 0;

int main(void){

judge(0);

printf("there are totally %d solutions\n", counter);

return 0;

}

void print(char queen[]){

printf("The %dth solution:\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n", counter);

int i,j;

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

for (j = 0; j < queen[i]; j++)

printf("0 ");

printf("1 ");

for (j = queen[i] + 1; j < 8; j++)

printf("0 ");

printf("\n");

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

int judge(int index){

if(index == MAX){

counter++;

print(queen);

return TRUE;

}

int i,j;

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

int flag = TRUE;

queen[index] = i;//将index行的后移动到第i列位置，(index, i)

for (j = 0; j < index; j++){//j是行数

if(queen[j] == queen[index] ||

(index - j) == (queen[index] - queen[j]) ||

(j - index) == (queen[index] - queen[j])){

flag = FALSE; break;

}

}

if(flag)

judge(index + 1);

}

}

2）运行结果如图5-18所示：

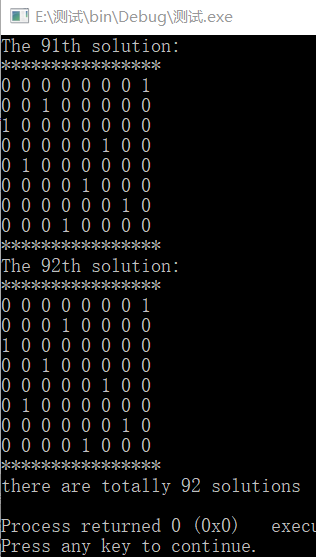


图5-18 选做题（2）运行截图

**5.3 实验小结**

遇到的问题：

（1）瑟夫问题，刚开始自己并不明白，琢磨很久，以至于在草稿纸上手动计算。

（2）跟踪调试题遇到一些小问题，改了几次没有得到正确结果。

（3）八皇后问题完全摸不着头脑。

如何解决的：

（1）瑟夫问题，在演算之后看网上他人的解法，自己琢磨步骤对应的含义，把程序的意思搞清楚后解决了该问题。之后助教上机检查的时候，提醒我可以用标记的方法减少运算量。

（2）跟踪调试题，自己就一直尝试，并不是不会，而是由于粗心没有发现问题所在，最后解决了该问题。

（3）八皇后问题，自己求教两位大佬，折腾了好多天才弄明白。

体会：

（1）C语言的学习由浅入深，关键在于与数学思想的交汇融合，C语言的实质是为了解决现实问题，因此更为注重实践。

（2）要对数学加强学习，不然最后空有技巧而没有实际的解决问题的能力。

（3）实验中我深刻意识到完成程序的编写，决不意味着万事大吉。

（4）在今后学习中我要更多的动脑，综合运用所学，多看相关东西，多上机练习，提高自己动手能力。

# 实验6 指针实验

6.1 实验目的

（1）．熟练掌握指针的说明、赋值、使用。

（2）．掌握用指针引用数组的元素，熟悉指向数组的指针的使用。

（3）．熟练掌握字符数组与字符串的使用，掌握指针数组及字符指针数组的用法。

（4）．掌握指针函数与函数指针的用法。

（5）．掌握带有参数的main函数的用法。

6.2 实验内容

### 6.2.1 源程序改错题

下面程序是否存在错误？如果存在，原因是什么？如果存在错误，要求在计算机上对这个例子程序进行调试修改，使之能够正确执行。

1 #include "stdio.h"

2 void main(void)

3 {

4 float \*p;

5 scanf("%f",p);

6 printf("%f\n",\*p);

7 }

解答：

1）第2行，改为：int main()

2）第4行，未对指针p赋初值，为空指针。正确形式为：float x, \*p=&x;

3）第6行和第7行之间添加：return 0;

修改后的运行结果如图6-1所示：

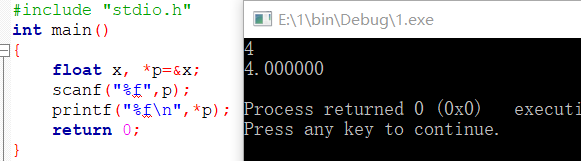


图6-1 程序修改运行截图

### 6.2.2 源程序修改替换题

（1）下面的程序通过函数指针和菜单选择来调用字符串拷贝函数或字符串连接函数，请在下划线处填写合适的表达式、语句、或代码片段来完善该程序。

#include "stdio.h"

#include "string.h"

void main(void)

{

char a[80],b[80],c[160],\*result=c;

int choice,i;

do{

printf("\t\t1 copy string.\n");

printf("\t\t2 connect string.\n");

printf("\t\t3 exit.\n");

printf("\t\tinput a number (1-3) please!\n");

scanf("%d",&choice);

}while(choice<1 || choice>5);

switch(choice){

case 1:

p=strcpy;

break;

case 2:

p=strcat;

break;

case 3:

goto down;

}

getchar();

printf("input the first string please!\n");

i=0;

printf("input the second string please!\n");

i=0;

result= (a,b);

printf("the result is %s\n",result);

down:

;

}

（2）请上机运行第（1）题程序，使之能按下面要求输出结果：（（输入）表示该数据是键盘输入数据）

1 copy string.

2 connect string.

3 exit.

input a number (1-3) please!

2 （输入）

input the first string please!

the more you learn, （输入）

input the second string please!

the more you get. （输入）

the result is the more you learn,the more you get.

解答：

源程序如下：

#include "stdio.h"

#include "string.h"

int main()

{

char \*(\*p)(char \*,char \*);

char a[80],b[80],c[160],\*result=c;

int choice,i;

do{

printf("\t\t1 copy string.\n");

printf("\t\t2 connect string.\n");

printf("\t\t3 exit.\n");

printf("\t\tinput a number (1-3) please!\n");

scanf("%d",&choice);

}while(choice<1 || choice>5);

switch(choice){

case 1:

p=strcpy;

break;

case 2:

p=strcat;

break;

case 3:

goto down;

}

getchar();

printf("input the first string please!\n");

i=0;

gets(a);

printf("input the second string please!\n");

i=0;

scanf("%[^\n]",b);

result= p(a,b);

printf("the result is %s\n",result);

down:

;

return 0;

}

运行结果如图6-2所示

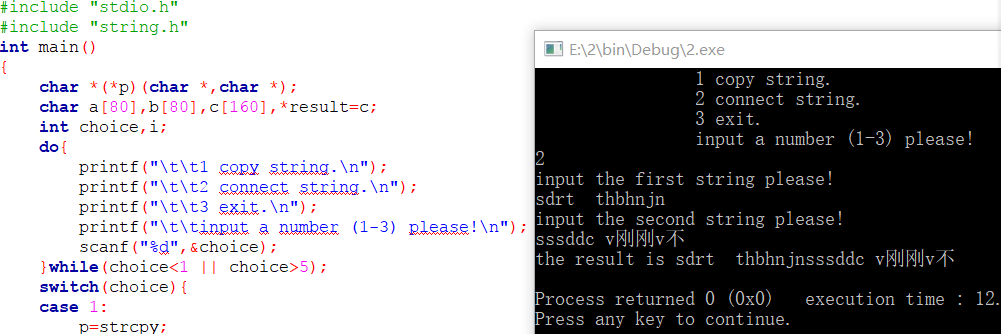


图6-2 运行结果截图

### 6.2.3 程序调试题

请按下面的要求对源程序进行操作，并回答问题和排除错误。

（1）单步执行。进入strcpy时watch窗口中s为何值？返回main时, watch窗口中s为何值？

解答：

进入strcpy时s的值如图6-3 所示

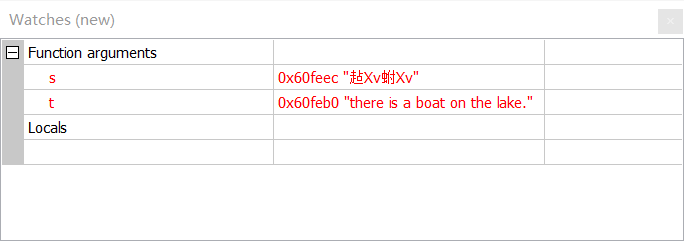


图6-3 进入strcpy时s的值截图

返回main函数时s值如图6-4所示

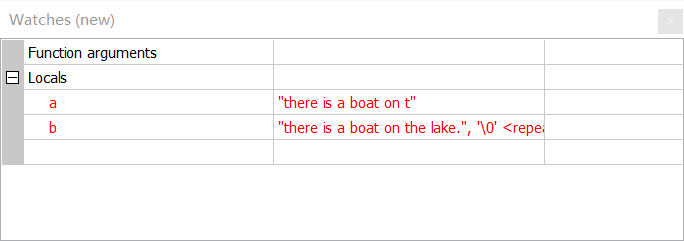


图6-4 返回返回main时s值的截图

（2）排除错误，使程序输出结果为：there is a boat on the lake.

#include "stdio.h"

char \*strcpy(char \*,char \*);

void main(void)

{

char a[20],b[60]="there is a boat on the lake.";

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

}

char \*strcpy(char \*s,char \*t)

{

while(\*s++=\*t++)

;

return (s);

}

**解答：**

1）修改后的程序如下：

#include "stdio.h"

#include <string.h>//添加

char \*strcpy1(char \*,char \*);//与库函数重名字

int main()

{

char a[80],b[60]="there is a boat on the lake.";//数组a太小

printf("%s\n",strcpy1(a,b));

return 0;

}

char \*strcpy1(char \*s,char \*t)

{

char \*p=s;//因为指针变化之后会改变位置

while((\*s++=\*t++));

return p;

}

2）运行结果如图6-5 所示

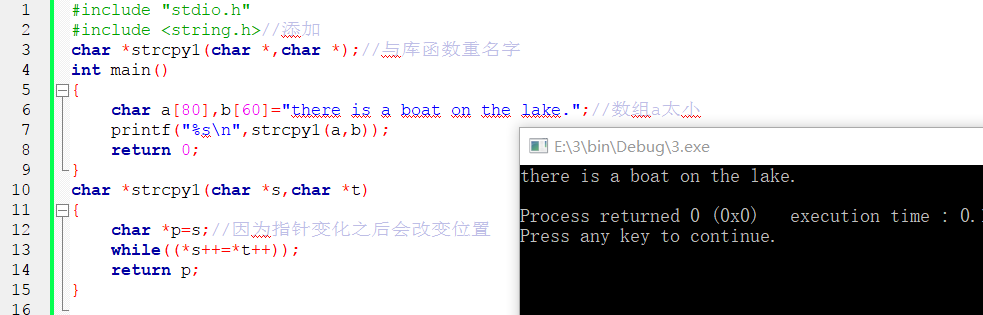


图6-5 运行结果截图

### 6.2.4 编程设计题

（1）一个长整型变量占4个字节，其中每个字节又分成高4位和 低4位。试从该长整型变量的高字节开始，依次取出每个字节的高4位和低4位并以数字字符的形式进行显示。

解答：

1）流程图如6-6所示：

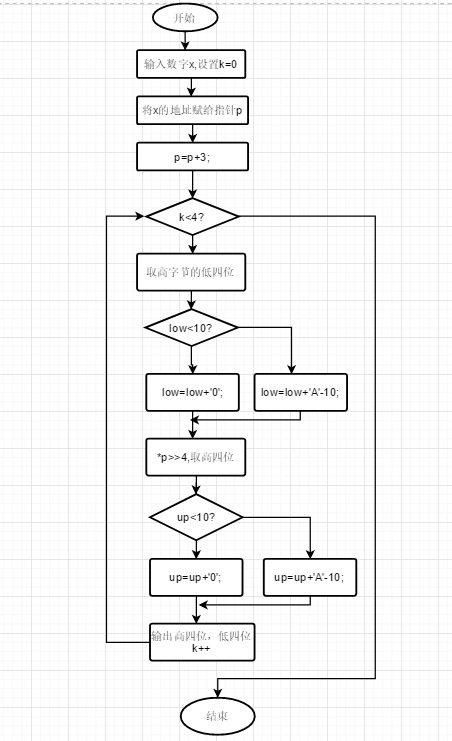


图6-6 编程题（1）流程图

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

long int x,k;

scanf("%lx",&x);

char \*p=(char \*)&x;

p=p+3;

char up,low;

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

{

low=(\*p)&0x0f;

if(low<10)

low=low+'0';

else

low=low+'A'-10;

up=(\*p>>4)&0x0f;

if(up<10)

up=up+'0';

else

up=up+'A'-10;

printf("%c %c\n",up,low);

p--;

}

return 0;

}

3）运行结果如图6-7 所示

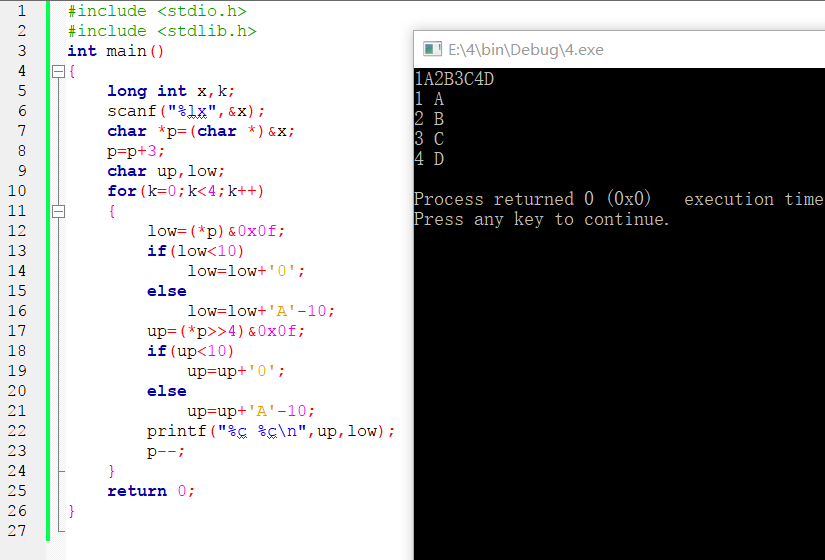


图6-7 编程题（1）运行截图

（2）利用大小为n的指针数组指向用gets函数输入的n行，每行不超过80个字符。编写一个函数，它将每一行中连续的多个空格字符压缩为一个空格字符。在调用函数中输出压缩空格后的各行，空行不予输出。

解答：

1）流程图如图6-8和图6-9所示：

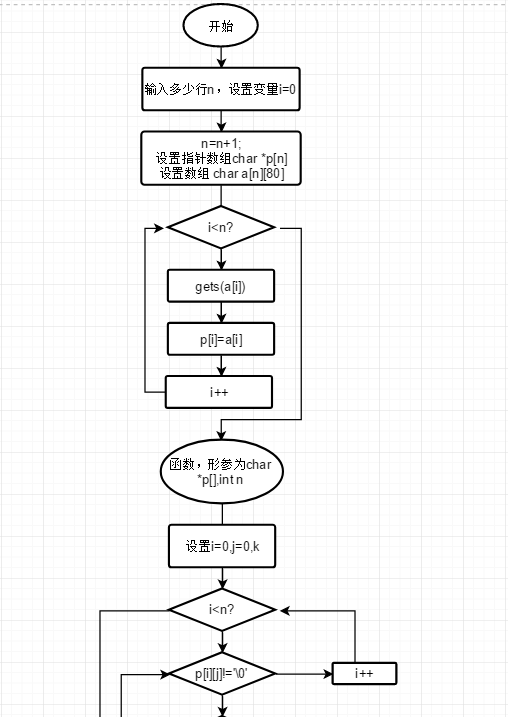


图6-8 编程题（2）流程图

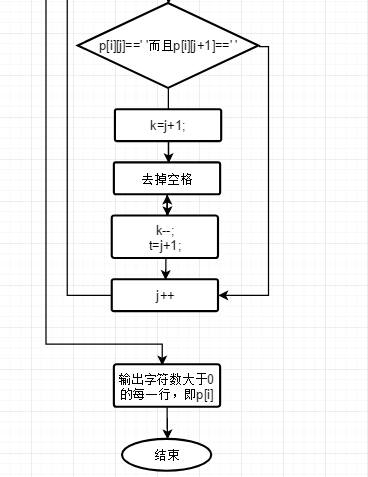


图6-9 编程题（2）流程图

2）源程序如下:

#include <stdio.h>

#include <string.h>

#define N 80//每行80字符

void zha(char \*p[],int n)//函数，形参为指针数组和整数

{

int i,j,k,t;

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

{

for(j=0;p[i][j]!='\0';j++)//只要每行没有遇到结束符

{

if(p[i][j]==' '&&p[i][j+1]==' ')//如果连续遇到空格

{

k=j+1;

while(p[i][k++]==' ');//去掉空格，找到不是空格时下标

k--;

t=j+1;

while((p[i][t++]=p[i][k++]));//将k之后的字符拼接到空格起始的地方

}

}

if(strlen(p[i])>0)//如果每一行除空格外的字符数大于0，即只要不是空行就输出

printf("%s\n",p[i]);

}

}

int main()

{

int n,i;

printf("请输入数字，多少行\n");

scanf("%d",&n);

n=n+1;

char \*p[n];

char a[n][N];

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

{

gets(a[i]);//得到每一行的字符

p[i]=a[i];//将每一行的首地址赋给指针变量

}

zha(p,n);//调用函数

return 0;

}

3)运行结果如图6-10 所示

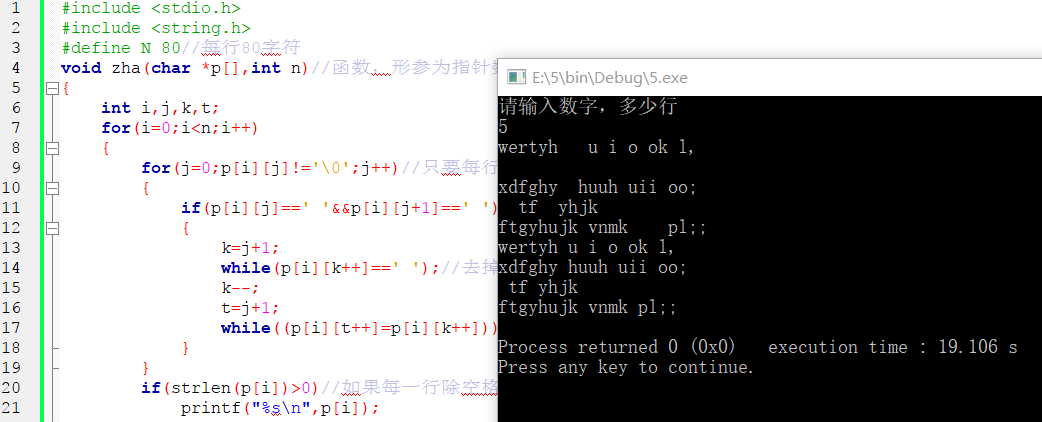


图6-10 编程题（2）运行截图

（3）设某个班有N个学生，每个学生修了M门课程（用#define定义N、M）。输入M门课程的名称，然后依次输入N个学生中每char个学生所修的M门课程的成绩并且都存放到相应的数组中。编写下列函数：

a.计算每个学生各门课程平均成绩；

b.计算全班每门课程的平均成绩；

c.分别统计低于全班各门课程平均成绩的人数；

d.分别统计全班各门课程不及格的人数和90分以上（含90分）的人数。

在调用函数中输出上面各函数的计算结果。（要求都用指针操作，不得使用下标操作。）

解答：

1）解题思路：

首先设置学生姓名、学科、分数的二维数组，之后分别用相应的指针指代，分别录入数据，最终通过指针来完成计算、比较和输出。

2）源程序如下：

#include <stdio.h>

#define M 2

#define N 2

int main()

{

char course[M][20],student[N][20];

float score[N][M],sum,sum1;

char \*c=&course[0][0];

char \*s=&student[0][0];

float \*sc=&score[0][0];

int i,j;

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

{

printf("请输入第%d门课程名称\n",i+1);

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

}

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

{

printf("请输入对于第%d个同学的名字\n",j+1);

scanf("%s",student[j]);

}

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

{

printf("对于同学%s：\n",student[i]);

printf("输入对应课程的成绩\n");

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

{

printf("请输入%s课程的成绩\n",course[j]);

scanf("%f",&score[i][j]);

}

}

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

{

sum=0;

printf("学生%s课程的平均成绩为：\n",s+i\*20);

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

sum = sum+\*(sc+i\*M+j);

printf("%f\n",sum/M);

}

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

{

sum1=0;

printf("对于%s课程，全班的平均成绩为：\n",c+i\*20);

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

{

sum1=sum1 + \*(sc+j\*M+i);

}

printf("%f\n",sum1/N);

}

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

{

int count=0;

printf("对于%s课程，低于平均分人数为：\n",c+i\*20);

sum1=0;

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

{

sum1=sum1 + \*(sc+j\*M+i);

}

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

{

if(\*(sc+j\*M+i)<sum1/N)

count++;

}

printf("%d\n",count);

count=0;

}

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

{

printf("对于%s课程，不及格人数为：\n",c+i\*20);

int count=0;

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

{

if(\*(sc+j\*M+i)<60)

count++;

}

printf("%d\n",count);

count=0;

}

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

{

printf("对于%s课程，高于90分人数为：\n",c+i\*20);

int count=0;

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

{

if(\*(sc+j\*M+i)>90)

count++;

}

printf("%d\n",count);

count=0;

}

return 0;

}

3）运行结果如图6-11 所示

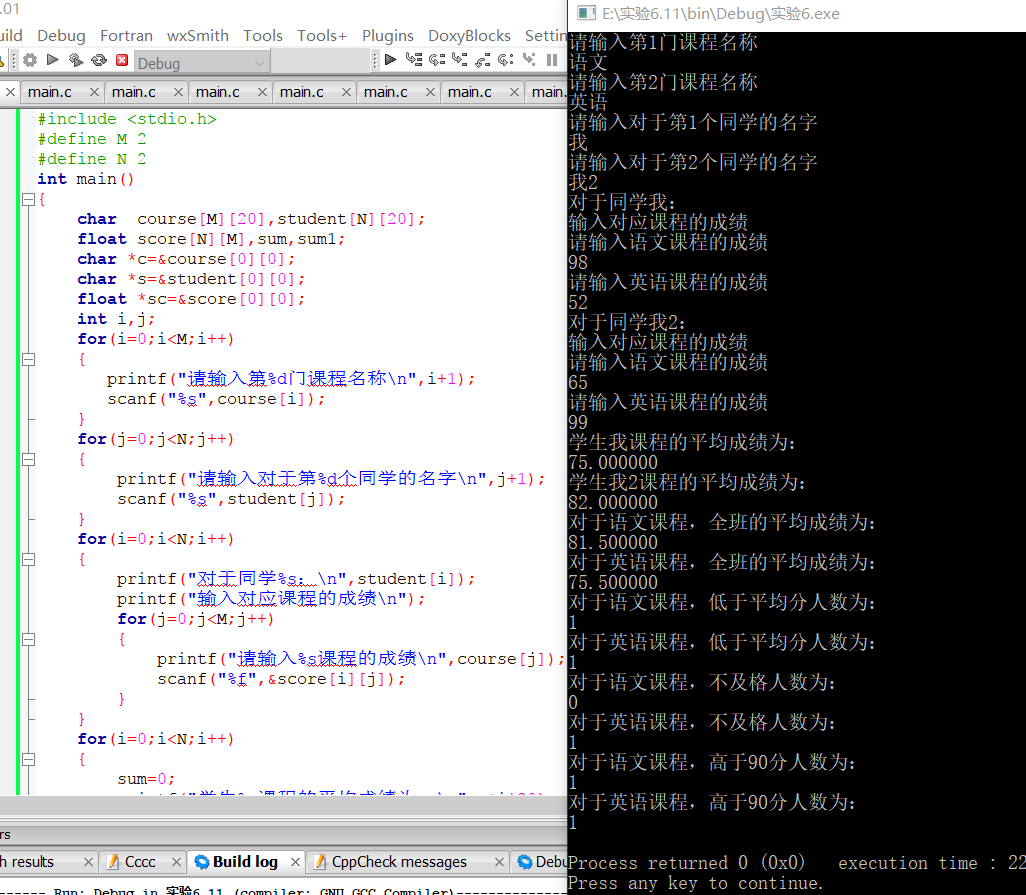


图6-11 编程题（3）运行截图

5. 选做题

（1）设有N位整数和M位小数（N=20，M=10）的数据a,b。编程计算a+b并输出结果。

如：12345678912345678912.1234567891 + 98765432109876543210.0123456789

解答：

1）思路如下：

设置两个字符数组用于储存字符串，从从小数部分进行加法运算，并要判断是否进位，并且以小数点为分界。

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

int main()

{

int i,x=0;

char a[50],b[50];

int s[51]={};

gets(a);

gets(b);

for(i=30;i>=0;i--)

{

int m,n,k;

if(a[i]=='.')

i--;

m=a[i]-'0'+x;

n=b[i]-'0';

k=m+n;

if(k>=10)

{

x=k/10;

s[i+1]=k%10;

}

else

{

s[i+1]=k%10;

x=0;

}

}

s[0]=x;

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

{

if(i==21)

printf(".");

else printf("%d",s[i]);

}

return 0;

}

3）运行结果如图6-12 所示

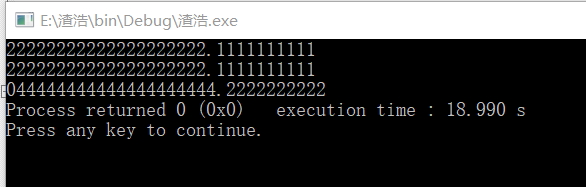


图6-12 选做题（2）运行截图

（2）编写使用复杂声明char \*(\*p[2])(const char \*,const char \*);的程序。

提示：p中元素可为strcmp、strstr等函数名。

解答：

1）流程图如图6-13 和6-14 所示：

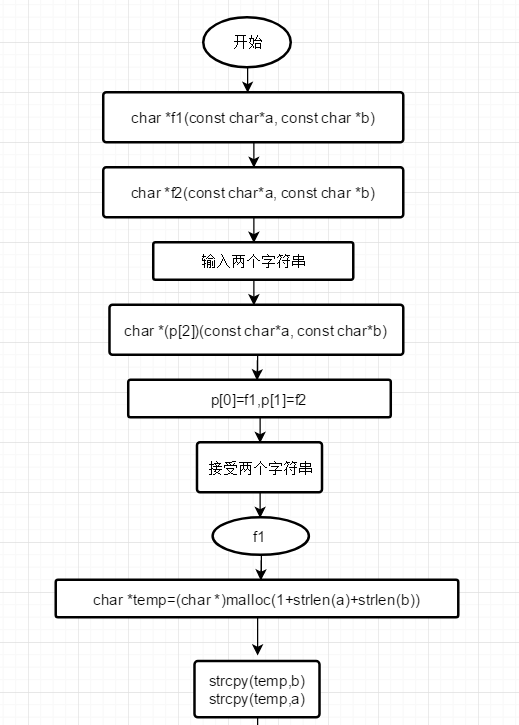


图6-13 选做题（2）流程图

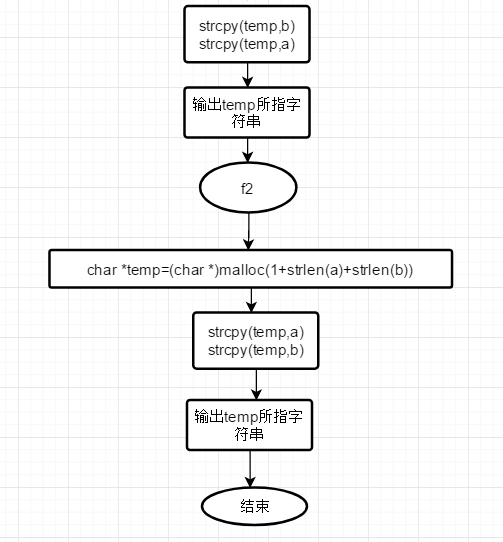


图6-14 选做题（2）流程图

2）源程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char \*f1(const char \*a,const char \*b);

char \*f2(const char \*a,const char \*b);

char \*f1(const char \*a,const char \*b)

{

char \*temp=(char \*)malloc(1+strlen(a)+strlen(b));

strcpy(temp,a);

strcat(temp,b);

return temp;

}

char \*f2(const char \*a,const char \*b)

{

char \*temp=(char \*)malloc(1+strlen(a)+strlen(b));

strcpy(temp,b);

strcat(temp,a);

return temp;

}

int main()

{

printf("请输入两组字符串\n");

char \*(\*p[2])(const char \*,const char \*);

p[0]=f1;

p[1]=f2;

char t[80],s[80];

scanf("%[^\n]",t);

getchar();

scanf("%[^\n]",s);

getchar();

printf("%s\n",p[0](t,s));

printf("%s\n",p[1](t,s));

return 0;

}

3）运行结果如图6-15 所示

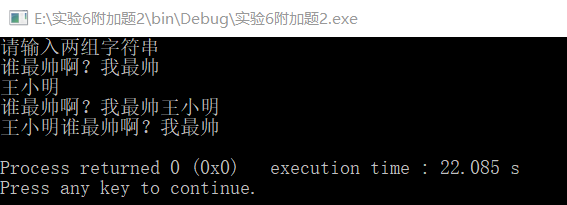


图6-15 选做题（2）运行截图

**(三) 指定main函数的参数**

选择“project/ set programs' arguments…”菜单命令，即可打开图2.12所示的对话框，在“Program arguments”文本框中输入main函数的参数。注意只输入命令行中文件名后的参数，文件名不输人。

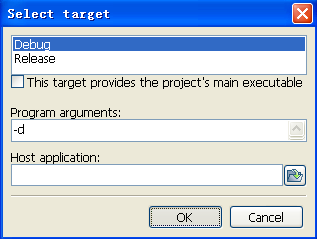
****

图6-16 输入main函数的参数

解答：

（1）思路如下：

先写好带参数的main函数，之后打开CodeBlocks上的project,然后点击set programs’arguments,之后输入字符串，返回CodeBlocks界面进行编译运行。

（2）程序如下：

#include <stdio.h>

#include <stdlib.h>

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

{

int i;

printf("文件名和输入的名称：\n");

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

{

printf("%s\n",argv[i]);

}

return 0;

}

（3）输入数据如图6-17 所示

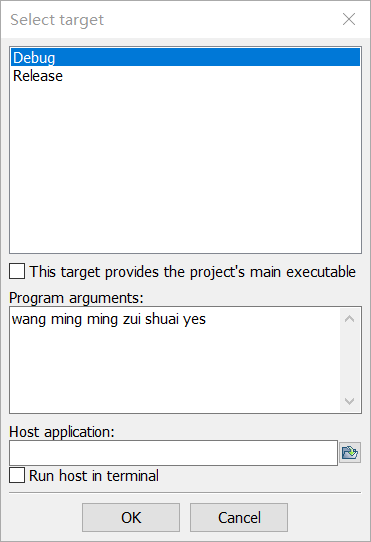


图6-17 输入数据截图

（3）运行结果如图6-18 所示

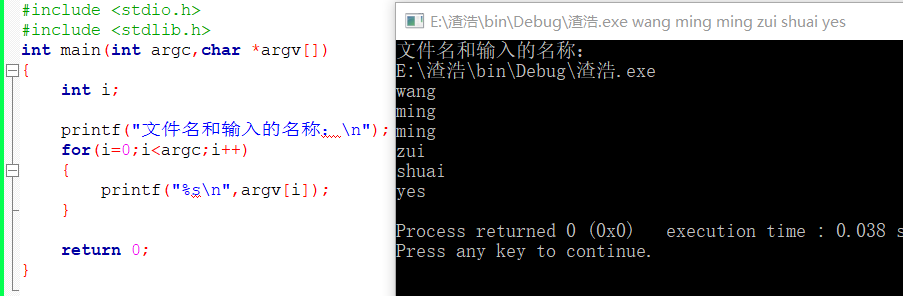


图6-18 运行截图

6.3 实验小结

问题：

（1）超长小数的加法不太会。

（2）实际使用带参数的main函数不太会。

解决方法：

（1）书上有超长整数的加法例子，参考着书，又问了问同学。

（2）上次的纸质作业有带参数的main函数这类的题，我写错了，思考之后写出来了。

体会：

（1）学习要温故知新，时常巩固前面学到的知识，熟能生巧。

（2）知识不够广，有些内容没有学好，不能要用到时及时反映出来，认识程度不够深刻。

（3）基本功不够，有些函数的表达不太精通，需要看书来核实，以致耗时较多。

（4）编程思想不够发散，看着题目有时想不出解答的方法，更不用说编写程序来解题了。

# 实验7 结构与联合实验

**7.1 实验目的**

1．通过实验，熟悉和掌握结构的说明和引用、结构的指针、结构数组、以及函数中使用结构的方法。

2．通过实验，掌握动态储存分配函数的用法，掌握自引用结构，单向链表的创建、遍历、结点的增删、查找等操作。

3．了解字段结构和联合的用法。

**7.2 实验内容**

### 7.2.1 表达式求值的程序验证题

设有说明：

char u[]="UVWXYZ";

char v[]="xyz";

struct T{

int x;

char c;

char \*t;

}a[]={{11,ˊAˊ,u},{100, ˊBˊ,v}},\*p=a;

请先自己计算下面表达式的值，然后通过编程计算来加以验证。(各表达式相互无关)

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **表达式** | **计算值** | **验证值** |
| 1 | (++p)->x | 100 | 100 |
| 2 | p++,p->c | a [1],B | B |
| 3 | \*p++->t,\*p->t | U,V | U,x |
| 4 | \*(++p)->t | x | x |
| 5 | \*++p->t | V | V |
| 6 | ++\*p->t | V | V |

### 7.2.2 源程序修改替换题

给定一批整数，以0作为结束标志且不作为结点，将其建成一个先进先出的链表，先进先出链表的指头指针始终指向最先创建的结点（链头），先建结点指向后建结点，后建结点始终是尾结点。

（1）源程序中存在什么样的错误（先观察执行结果）？对程序进行修改、调试，使之能够正确完成指定任务。

源程序如下：

#include "stdio.h"

#include "stdlib.h"

struct s\_list{

int data; /\* 数据域 \*/

struct s\_list \*next; /\* 指针域 \*/

} ;

void create\_list (struct s\_list \*headp,int \*p);

void main(void)

{

struct s\_list \*head=NULL,\*p;

int s[]={1,2,3,4,5,6,7,8,0}; /\* 0为结束标记 \*/

create\_list(head,s); /\* 创建新链表 \*/

p=head; /\*遍历指针p指向链头 \*/

while(p){

printf("%d\t",p->data); /\* 输出数据域的值 \*/

p=p->next; /\*遍历指针p指向下一结点 \*/

}

printf("\n");

}

void create\_list(struct s\_list \*headp,int \*p)

{

struct s\_list \* loc\_head=NULL,\*tail;

if(p[0]==0) /\* 相当于\*p==0 \*/

;

else { /\* loc\_head指向动态分配的第一个结点 \*/

loc\_head=(struct s\_list \*)malloc(sizeof(struct s\_list));

loc\_head->data=\*p++; /\* 对数据域赋值 \*/

tail=loc\_head; /\* tail指向第一个结点 \*/

while(\*p){ /\* tail所指结点的指针域指向动态创建的结点 \*/

tail->next=(struct s\_list \*)malloc(sizeof(struct s\_list));

tail=tail->next; /\* tail指向新创建的结点 \*/

tail->data=\*p++; /\* 向新创建的结点的数据域赋值 \*/

}

tail->next=NULL; /\* 对指针域赋NULL值 \*/

}

headp=loc\_head; /\* 使头指针headp指向新创建的链表 \*/

}

解答：

1）修改后程序如下：

#include "stdio.h"

#include "stdlib.h"

struct s\_list{

int data; /\* 数据域 \*/

struct s\_list \*next; /\* 指针域 \*/

} ;

void create\_list (struct s\_list \*\*headp,int \*p);

int main()

{

struct s\_list \*head=NULL,\*p;

int s[]={1,2,3,4,5,6,7,8,0}; /\* 0为结束标记 \*/

create\_list(&head,s); /\* 创建新链表 \*/

p=head; /\*遍历指针p指向链头 \*/

while(p){

printf("%d\t",p->data); /\* 输出数据域的值 \*/

p=p->next; /\*遍历指针p指向下一结点 \*/

}

printf("\n");

return 0;

}

void create\_list(struct s\_list \*\*headp,int \*p)

{

struct s\_list \* loc\_head=NULL,\*tail,\*temp;

if(p[0]==0) /\* 相当于\*p==0 \*/

;

else { /\* loc\_head指向动态分配的第一个结点 \*/

loc\_head=(struct s\_list \*)malloc(sizeof(struct s\_list));

loc\_head->data=\*p++; /\* 对数据域赋值 \*/

tail=loc\_head; /\* tail指向第一个结点 \*/

while(\*p){ /\* tail所指结点的指针域指向动态创建的结点 \*/

tail->next=(struct s\_list \*)malloc(sizeof(struct s\_list));

tail=tail->next; /\* tail指向新创建的结点 \*/

tail->data=\*p++; /\* 向新创建的结点的数据域赋值 \*/

}

tail->next=NULL; /\* 对指针域赋NULL值 \*/

}

\*headp=loc\_head; /\* 使头指针headp指向新创建的链表 \*/

}

2）运行结果如图7-1所示

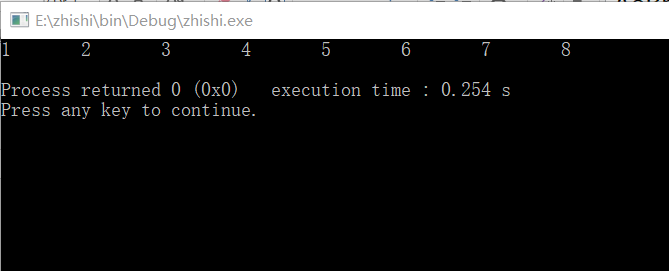


图7-1 运行结果截图

（2）修改替换create\_list函数，将其建成一个后进先出的链表，后进先出链表的头指针始终指向最后创建的结点（链头），后建结点指向先建结点，先建结点始终是尾结点。

解答：

1）程序如下：

#include "stdio.h"

#include "stdlib.h"

struct s\_list{

int data; /\* 数据域 \*/

struct s\_list \*next; /\* 指针域 \*/

} ;

void create\_list (struct s\_list \*\*headp,int \*p);

int main()

{

struct s\_list \*head=NULL,\*p;

int s[]={1,2,3,4,5,6,7,8,0}; /\* 0为结束标记 \*/

create\_list(&head,s); /\* 创建新链表 \*/

p=head; /\*遍历指针p指向链头 \*/

while(p){

printf("%d\t",p->data); /\* 输出数据域的值 \*/

p=p->next; /\*遍历指针p指向下一结点 \*/

}

printf("\n");

return 0;

}

void create\_list(struct s\_list \*\*headp,int \*p)

{

struct s\_list \* loc\_head=NULL,\*tail,\*temp;

if(p[0]==0) /\* 相当于\*p==0 \*/

;

else { /\* loc\_head指向动态分配的第一个结点 \*/

loc\_head=(struct s\_list \*)malloc(sizeof(struct s\_list));

loc\_head->data=\*p++; /\* 对数据域赋值 \*/

tail=loc\_head; /\* tail指向第一个结点 \*/

while(\*p)

{

temp=(struct s\_list\*)malloc(sizeof(struct s\_list));

temp->next=loc\_head;

loc\_head=temp;

loc\_head->data=\*p++;

}

tail->next=NULL; /\* 对指针域赋NULL值 \*/

}

\*headp=loc\_head; /\* 使头指针headp指向新创建的链表 \*/

}

2）运行结果如图7-2所示

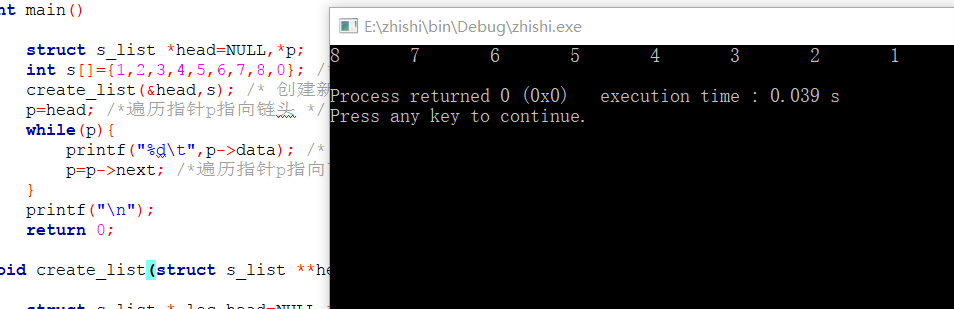


图7-2运行截图

### 7.2.3 编程设计题

（1）设计一个字段结构struct bits，它将一个8位无符号字节从最低位向最高位声明为8个字段，各字段依次为bit0, bit1, …, bit7，且bit0的优先级最高。同时设计8个函数，第i个函数以biti(i=0,1,2,…,7)为参数，并且在函数体内输出biti的值。将8个函数的名字存入一个函数指针数组p\_fun。如果bit0为1，调用p\_fun[0]指向的函数。如果struct bits中有多位为1，则根据优先级从高到低依次调用函数指针数组p\_fun中相应元素指向的函数。8个函数中的第0个函数可以设计为：

void f0(struct bits b)

{

Printf(“the function %d is called!\n”,b);

}

解答：

1）程序如下：

#include <stdio.h>

#include <stdlib.h>

struct bits{

unsigned char bit0:1,bit1:1,bit2:1,bit3:1,bit4:1,bit5:1,bit6:1,bit7:1;}a;

union w{

unsigned char t;

struct bits a;

}m;

void f0(struct bits b)

{

printf("the 1 function %d is called!\n",b.bit0);

}

void f1(struct bits b)

{

printf("the 2 function %d is called!\n",b.bit1);

}

void f2(struct bits b)

{

printf("the 3 function %d is called!\n",b.bit2);

}

void f3(struct bits b)

{

printf("the 4 function %d is called!\n",b.bit3);

}

void f4(struct bits b)

{

printf("the 5 function %d is called!\n",b.bit4);

}

void f5(struct bits b)

{

printf("the 6 function %d is called!\n",b.bit5);

}

void f6(struct bits b)

{

printf("the 7 function %d is called!\n",b.bit6);

}

void f7(struct bits b)

{

printf("the 8 function %d is called!\n",b.bit7);

}

int main()

{

void (\*p\_fun[8])(struct bits b);

printf("input char:");

m.t=getchar();

m.a.bit0=1;

p\_fun[0]=f0;

p\_fun[1]=f1;

p\_fun[2]=f2;

p\_fun[3]=f3;

p\_fun[4]=f4;

p\_fun[5]=f5;

p\_fun[6]=f6;

p\_fun[7]=f7;

if(m.a.bit0) p\_fun[0](m.a);

if(m.a.bit1) p\_fun[1](m.a);

if(m.a.bit2) p\_fun[2](m.a);

if(m.a.bit3) p\_fun[3](m.a);

if(m.a.bit4) p\_fun[4](m.a);

if(m.a.bit5) p\_fun[5](m.a);

if(m.a.bit6) p\_fun[6](m.a);

if(m.a.bit7) p\_fun[7](m.a);

return 0;

}

2）运行结果如图7-3所示

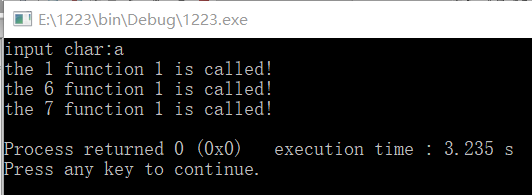


图7-3 编程题（1）运行截图

（2）用单向链表建立一张班级成绩单，包括每个学生的学号、姓名、英语、高等数学、普通物理、C语言程序设计四门课程的成绩。用函数编程实现下列功能：

(1) 输入每个学生的各项信息。

(2) 输出每个学生的各项信息。

(3) 修改指定学生的指定数据项的内容。

(4) 统计每个同学的平均成绩（保留2位小数）。

(5) 输出各位同学的学号、姓名、四门课程的总成绩和平均成绩。

解答：

1）解题思路：

设置相应函数，每个函数完成相应功能，之间以指针作为形参。

先创建链表，输入信息并判断信息是否为结束标志，在主函数中调用函数从而顺序正确。

2)源程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct stu{

long num;

char s[10];

int c1,c2,c3,c4;

struct stu \*next;

};

void creatlist(struct stu \*\*headp);

void outputlist(struct stu \*\*headp);

void correctlist(struct stu \*\*headp);

void avstu(struct stu \*\*headp);

void sumstu(struct stu \*\*headp);

int main()

{

struct stu \*head=NULL;

creatlist(&head);

outputlist(&head);

correctlist(&head);

avstu(&head);

sumstu(&head);

return 0;

}

void creatlist(struct stu \*\*headp)

{

\*headp=(struct stu \*)malloc(sizeof(struct stu));

struct stu \*tail=\*headp,\*p=\*headp;

printf("please input the following item devided by square:\n");

printf("ID\tname\tEnglish grade\tmath grade\tphysics grade\tc language grade\n");

printf("end with 0\n");

for(;;)

{

scanf("%ld",&p->num);

if(p->num==0) break;

scanf("%s %d %d %d %d",p->s,&p->c1,&p->c2,&p->c3,&p->c4);

p=(struct stu \*)malloc(sizeof(struct stu));

tail->next=p;

tail=p;

}

tail->next=NULL;

}

void outputlist(struct stu \*\*headp)

{

struct stu \*p=\*headp;

while(p->next!=NULL)

{

printf("ID %ld NAME %s ENG %d MATH %d phy %d c %d\n",p->num,p->s,p->c1,p->c2,p->c3,p->c4);

p=p->next;

}

}

void correctlist(struct stu \*\*headp)

{

char c;

struct stu \*p=\*headp;

printf("1:look up by name\n");

printf("2:look up by number\n");

c=getchar();

c=getchar();

switch(c)

{

case '1':

printf("please input the name\n");

char t[10];

scanf("%s",t);

for(;p->next!=NULL&&strcmp(p->s,t);p=p->next);

if(!strcmp(p->s,t))

printf("FIND ID %ld NAME %s ENG %d MATH %d PHY %d C %d\n",p->num,p->s,p->c1,p->c2,p->c3,p->c4);

else if(p->next == NULL)

{

printf("NOT FOUND\n");

return ;

}

break;

case '2':

printf("please input the number\n");

long tmp;

scanf("%ld",&tmp);

for(;p->next!=NULL&&tmp!=p->num;p=p->next);

if(tmp==p->num)

printf("FIND ID %ld NAME %s ENG %d MATH %d PHY %d C %d\n",p->num,p->s,p->c1,p->c2,p->c3,p->c4);

else if(p->next==NULL)

{

printf("NOT FOUND\n");

return ;

}

break;

}

printf("which one to change?\n");

printf("1:number\n");

printf("2:name\n");

printf("3:Grade of English\n");

printf("4:Grade of Math\n");

printf("5:Grade of Physics\n");

printf("6:Grade of C\n");

printf("7:I dont want to change anything\n");

c=getchar();

c=getchar();

printf("please reinput\n");

switch(c)

{

case '1':scanf("%ld",&p->num);break;

case '2':scanf("%s",p->s);break;

case '3':scanf("%d",&p->c1);break;

case '4':scanf("%d",&p->c2);break;

case '5':scanf("%d",&p->c3);break;

case '6':scanf("%d",&p->c4);break;

default : return ;

}

}

void avstu(struct stu \*\*headp)

{

struct stu \*p=\*headp;

for(;p->next!=NULL;p=p->next)

{

double tp;

tp=(p->c1+p->c2+p->c3+p->c4)/4.0;

printf("ID %ld Name %s average %.2lf\n",p->num,p->s,tp);

}

}

void sumstu(struct stu \*\*headp)

{

struct stu \*p;

p=\*headp;

for(;p->next!=NULL;p=p->next)

{

double tp;

int sum;

sum=p->c1+p->c2+p->c3+p->c4;

tp=sum/4.0;

printf("ID %ld Name %s sum is %d average is %.2lf\n",p->num,p->s,sum,tp);

}

}

3）运行结果如图7-4所示

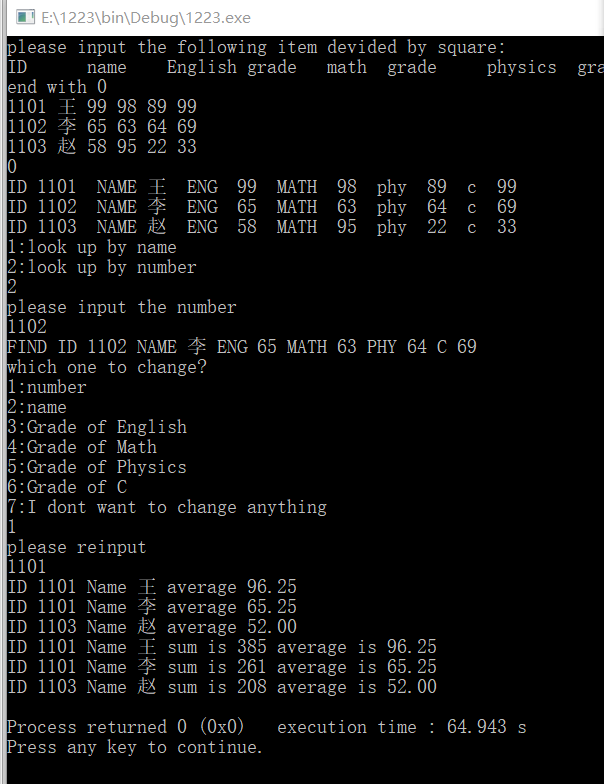


图7-4 编程题（2）运行截图

### 7.2.4 选做题

（1）对编程设计题第（2）题的程序，增加按照平均成绩进行升序排序的函数，写出用交换结点数据域的方法升序排序的函数，排序可用选择法或冒泡法。

解答：

1）修改后的程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct stu{

long num;

char s[10];

int c1,c2,c3,c4;

double t;

struct stu \*next;

};

void creatlist(struct stu \*\*headp);

void avstu(struct stu \*\*headp);

void sort\_lists(struct stu \*\*headp);

int main()

{

struct stu \*head=NULL;

creatlist(&head);

avstu(&head);

sort\_lists(&head);

return 0;

}

void creatlist(struct stu \*\*headp)

{

\*headp=(struct stu \*)malloc(sizeof(struct stu));

struct stu \*tail=\*headp,\*p=\*headp;

printf("please input the following item devided by square:\n");

printf("ID\tname\tEnglish grade\tmath grade\tphysics grade\tc language grade\n");

printf("end with 0\n");

for(;;)

{

scanf("%ld",&p->num);

if(p->num==0) break;

scanf("%s %d %d %d %d",p->s,&p->c1,&p->c2,&p->c3,&p->c4);

p=(struct stu \*)malloc(sizeof(struct stu));

tail->next=p;

tail=p;

}

tail->next=NULL;

}

void avstu(struct stu \*\*headp)

{

struct stu \*p=\*headp;

for(;p->next!=NULL;p=p->next)

{

double tp;

tp=(p->c1+p->c2+p->c3+p->c4)/4.0;

p->t=tp;

printf("ID %ld Name %s average %.2lf\n",p->num,p->s,tp);

}

}

void sort\_lists(struct stu \*\*headp)

{

struct stu \*p=\*headp,\*p1,\*p2;

int len=0,i,j;

double k;

long num1;

char s1[10];

int c11,c22,c33,c44;

p2=\*headp;

while(p){len++,p=p->next;}

for(i=0,p=\*headp;i<len-1;i++,p=p->next)

{

for(j=0,p1=\*headp;j<len-i-1;j++,p1=p1->next)

{

if(p1->t>p1->next->t)

{

c11=p1->c1;

p1->c1=p1->next->c1;

p1->next->c1=c11;

c22=p1->c2;

p1->c2=p1->next->c2;

p1->next->c2=c22;

c33=p1->c3;

p1->c3=p1->next->c3;

p1->next->c3=c33;

c44=p1->c4;

p1->c4=p1->next->c4;

p1->next->c4=c44;

k=p1->t;

p1->t=p1->next->t;

p1->next->t=k;

num1=p1->num;

p1->num=p1->next->num;

p1->next->num=num1;

strcpy(s1,p1->s);

strcpy(p1->s,p1->next->s);

strcpy(p1->next->s,s1);

}

}

}

while(p2->next!=NULL)

{

printf("%ld\t%s\t%d\t%d\t%d\t%d\t%.2lf\n",p2->num,p2->s,p2->c1,p2->c2,p2->c3,p2->c4,p2->t);

p2=p2->next;

}

}

2）运行截图如图7-5 所示

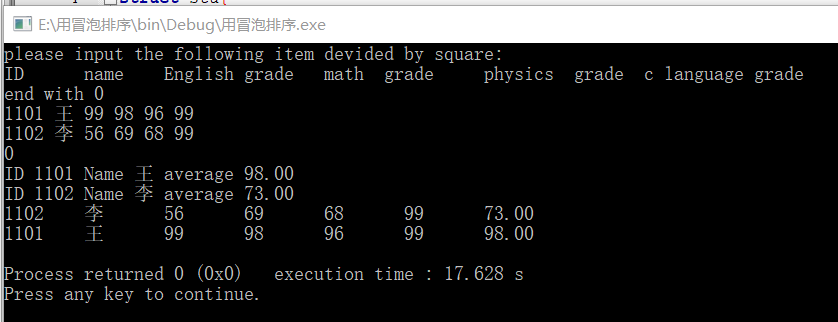


图7-5 选做题（1）运行截图

（2）对选做题第（1）题，进一步写出用交换结点指针域的方法升序排序的函数。

解答：

1）修改后的程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct stu{

long num;

char s[10];

int c1,c2,c3,c4;

double t;

struct stu \*next;

};

void creatlist(struct stu \*\*headp);

void avstu(struct stu \*\*headp);

void sort\_lists\_by\_pointer(struct stu \*\*headp);

int main()

{

struct stu \*head=NULL;

creatlist(&head);

avstu(&head);

sort\_lists\_by\_pointer(&head);

return 0;

}

void creatlist(struct stu \*\*headp)

{

\*headp=(struct stu \*)malloc(sizeof(struct stu));

struct stu \*tail=\*headp,\*p=\*headp;

printf("please input the following item devided by square:\n");

printf("ID\tname\tEnglish grade\tmath grade\tphysics grade\tc language grade\n");

printf("end with 0\n");

for(;;)

{

scanf("%ld",&p->num);

if(p->num==0) break;

scanf("%s %d %d %d %d",p->s,&p->c1,&p->c2,&p->c3,&p->c4);

p=(struct stu \*)malloc(sizeof(struct stu));

tail->next=p;

tail=p;

}

tail->next=NULL;

}

void avstu(struct stu \*\*headp)

{

struct stu \*p=\*headp;

for(;p->next!=NULL;p=p->next)

{

double tp;

tp=(p->c1+p->c2+p->c3+p->c4)/4.0;

p->t=tp;

printf("ID %ld Name %s average %.2lf\n",p->num,p->s,tp);

}

}

void sort\_lists\_by\_pointer(struct stu \*\*headp)

{

struct stu \*p1,\*p2,\*prior1,\*prior2,\*k,\*l;

p1=\*headp;

for(p2=p1->next,prior2=p1;p2!=NULL;prior2=p2,p2=p2->next)

{

if(p1->t>p2->t)

{

prior2->next=p1;

k=p1->next;

p1->next = p2->next;

(\*headp)=p2;

p2->next=k;

p1=p2;

}

}

for(prior1=\*headp,p1=prior1->next;p1->next!=NULL;prior1=p1,p1=p1->next)

{

for(p2=p1->next,prior2=p1;p2!=NULL;prior2=p2,p2=p2->next)

{

if(p1->t>p2->t)

{

k=p2->next;

prior1->next=p2;

prior2->next=p1;

p2->next=p1->next;

p1->next=k;

p1=p2;

}

}

}

l=\*headp;

while(l->next)

{

printf("%ld\t%s\t%d\t%d\t%d\t%d\t%.2lf\n",l->num,l->s,l->c1,l->c2,l->c3,l->c4,l->t);

l=l->next;

}

}

2）运行结果如图7-6 所示

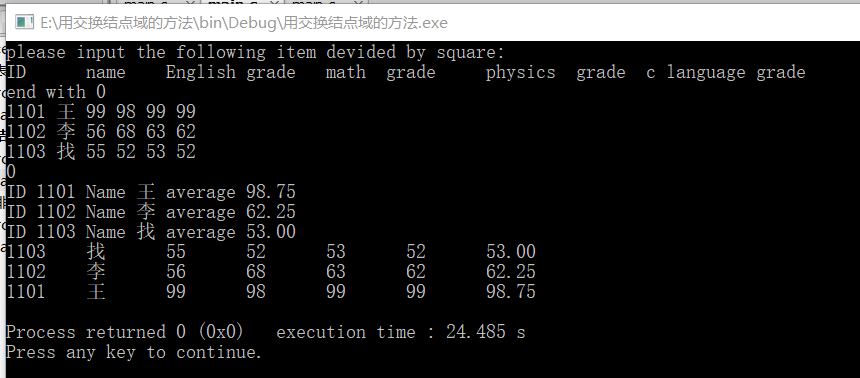


图7-6 选做题（2）运行截图

（3）采用双向链表重做编程设计题第（2）题。

**7.3 实验小结**

问题：对链表的理解不够，双向链表不会

方法：在中国大学MOOC上看视频，理解了单向链表

体会：

（1）我不懂的东西还很多，要坚持学习。

（2）认为万无一失的程序，实际上机运行时可能会出现很多意想不到的问题。

（3）演算过程中注意初始化指针的位置，防止前面的运算对后续的运算结果产生影响。

（4）锻炼编写代码的能力和思维的灵活性。

实验8 文件实验

8.1 实验目的

（1）熟悉文本文件和二进制文件在磁盘中的存储方式；

（2）熟练掌握流式文件的读写方法。

8.2 实验题目及要求

### 8.2.1 文件类型的程序验证题

设有程序：

#include <stdio.h>

int main(void)

{

short a=0x253f,b=0x7b7d;

char ch;

FILE \*fp1,\*fp2;

fp1=fopen("d:\\abc1.bin","wb+");

fp2=fopen("d:\\abc2.txt","w+");

fwrite(&a,sizeof(short),1,fp1);

fwrite(&b,sizeof(short),1,fp1);

fprintf(fp2,"%hx %hx",a,b);

rewind(fp1); rewind(fp2);

while((ch = fgetc(fp1)) != EOF)

putchar(ch);

putchar('\n');

while((ch = fgetc(fp2)) != EOF)

putchar(ch);

putchar('\n');

fclose(fp1);

fclose(fp2);

return 0;

}

1. 请思考程序的输出结果，然后通过上机运行来加以验证。

如图8-1 所示：

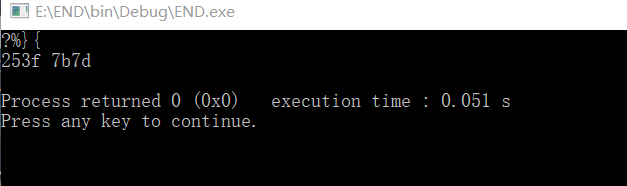


图8-1 程序运行截图

1. 将两处sizeof(short)均改为sizeof(char)结果有什么不同，为什么？

解答：

由于类型的变化，相当于一个字符型指针去读取一个数字，会按照字节进行划分，而不是作为一个数据两个字节一起输出。

运行结果如图8-2 所示

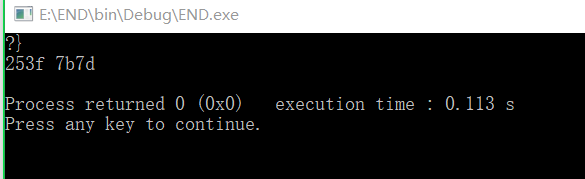


图8-2 运行截图

（3）将fprintf(fp2,"%hx %hx",a,b) 改为 fprintf(fp2,"%d %d",a,b)结果有什么不同。

解答：

运行结果如图8-3 所示

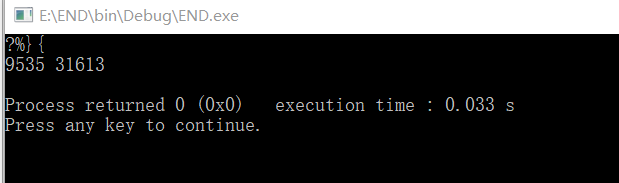


图8-3 运行截图

### 8.2.2 源程序修改替换题

将指定的文本文件内容在屏幕上显示出来，命令行的格式为：

type filename

1. 源程序中存在什么样的逻辑错误（先观察执行结果）？对程序进行修改、调试，使之能够正确完成指定任务。

#include<stdio.h>

#include<stdlib.h>

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

{

char ch;

FILE \*fp;

if(argc!=2){

printf("Arguments error!\n");

exit(-1);

}

if((fp=fopen(argv[1],"r"))==NULL){ /\* fp 指向 filename \*/

printf("Can't open %s file!\n",argv[1]);

exit(-1);

}

while(ch=fgetc(fp)!=EOF) /\* 从filename中读字符 \*/

putchar(ch); /\* 向显示器中写字符 \*/

fclose(fp); /\* 关闭filename \*/

return 0;

}

解答：

1）修改后的程序如下：

#include<stdio.h>

#include<stdlib.h>

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

{

char ch;

FILE \*fp;

if(argc!=2){

printf("Arguments error!\n");

exit(-1);

}

if((fp=fopen(argv[1],"r"))==NULL){ /\* fp 指向 filename \*/

printf("Can't open %s file!\n",argv[1]);

exit(-1);

}

while((ch=fgetc(fp))!=EOF) /\* 从filename中读字符 \*/

putchar(ch); /\* 向显示器中写字符 \*/

fclose(fp); /\* 关闭filename \*/

return 0;

}

2）运行结果如图8-4 所示

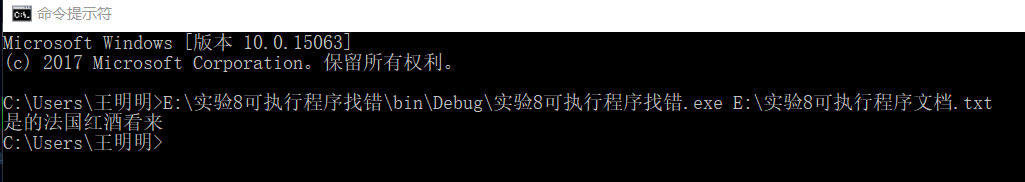


图8-4 运行截图

（2）用输入输出重定向freopen改写main函数。

解答：

1）源程序如下：

#include<stdio.h>

#include<stdlib.h>

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

{

char ch;

FILE \*fp;

if(argc!=2){

printf("Arguments error!\n");

exit(-1);

}

if((fp=freopen(argv[1],"r",stdin))==NULL){ /\* fp 指向 filename \*/

printf("Can't open %s file!\n",argv[1]);

exit(-1);

}

while((ch=fgetc(fp))!=EOF) /\* 从filename中读字符 \*/

putchar(ch); /\* 向显示器中写字符 \*/

fclose(fp); /\* 关闭filename \*/

return 0;

}

2）运行结果如图8-5 所示

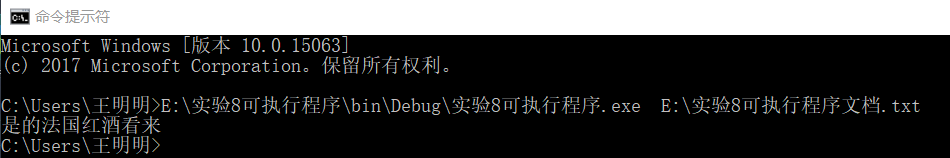


图8-5 运行截图

**8.2.3、编程设计题**

（1）从键盘输入一行英文句子，将每个单词的首字母换成大写字母，然后输出到一个磁盘文件“test”中保存。

解答：

1）源程序如下：

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

int main()

{

FILE \*fp;

char a[40];

int i=0;

fp=fopen("test.txt","w");

gets(a);

a[0]=toupper(a[0]);

while(\*(a+i)!='\0')

{

if(islower(\*(a+i))&&\*(a+i-1)==' ')

\*(a+i)=toupper(\*(a+i));

i++;

}

puts(a);

fputs(a,fp);

fclose(fp);

return 0;

}

2）运行结果如图8-6和8-7所示：

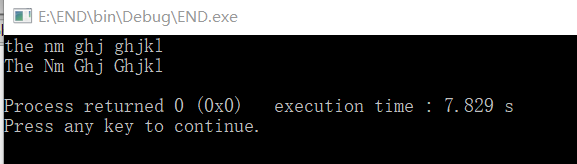


图8-6 运行截图



图8-7 运行截图

## 8.3 实验小结

遇到的问题：

（1） 对文件存在的形式理解不够深刻，打开文件和关闭文件的形式不是很理解。

（2）实验题的最后一题，对于运行程序后，生成的text.txt文件位置不够理解。

解决方法：

（1）根据老师课堂上的大致讲解，自己在中国大学MOOC上进行自学，解决了w、r、r+、w+等形式问题。

（2）自己找助教询问最后一题生成的文件位置问题。

体会：

（1）文件这一章节虽是学期末尾的知识，但却依旧重要。

（2）虽然这已经是最后一次C语言实验报告了，但对于我来说，C语言上机实验却是永远不会从我的学习中淡出的。

（3）虽然这个学期的上机实验课已经结束，但是我学到的东西却可以伴随我一生。

参考文献

[1] 曹计昌,卢萍,李开. C语言程序设计,北京： 科学出版社,2013

[2] 李开,卢萍,曹计昌. C语言实验与课程设计, 北京：科学出版社,2011