

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级： CS17??**

**学 号： U2017HUSTERGS**

**姓 名： HUSTERGS**

**指导教师： 祝 建 华**

**报告日期： 2017/11/16**

**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 6](#_Toc503258041)

[1.1 实验目的 6](#_Toc503258042)

[1.2 实验内容 6](#_Toc503258043)

[1.2.1 源程序改错 6](#_Toc503258044)

[1.2.2 源程序修改替换 8](#_Toc503258045)

[1.2.3 程序设计 9](#_Toc503258046)

[1.3 实验小结 16](#_Toc503258047)

[实验2 流程控制实验 17](#_Toc503258048)

[2.1实验目的 17](#_Toc503258049)

[2.2实验内容 17](#_Toc503258050)

[2.2.1源程序改错题 17](#_Toc503258051)

[2.2.2源程序修改替换题 18](#_Toc503258052)

[2.2.3编程设计题 21](#_Toc503258053)

[2.2.4 选做题 31](#_Toc503258054)

[2.3实验小结 33](#_Toc503258055)

[实验3 函数与程序结构实验 34](#_Toc503258056)

[3.1实验目的 34](#_Toc503258057)

[3.2实验内容 34](#_Toc503258058)

[3.2.1源程序改错题 34](#_Toc503258059)

[3.2.2源程序修改替换题 35](#_Toc503258060)

[3.2.3跟踪调试题 37](#_Toc503258061)

[3.2.4编程设计题 40](#_Toc503258062)

[3.2.5选做题 46](#_Toc503258063)

[3.3实验小结 48](#_Toc503258064)

[实验4 编译预处理实验 49](#_Toc503258065)

[4.1实验目的 49](#_Toc503258066)

[4.2实验内容 49](#_Toc503258067)

[4.2.1源程序改错题 49](#_Toc503258068)

[4.2.2源程序修改替换题 50](#_Toc503258069)

[4.2.3跟踪调试题 53](#_Toc503258070)

[4.2.4编程设计题 55](#_Toc503258071)

[4.3实验小结 58](#_Toc503258072)

[实验5 数组实验 59](#_Toc503258073)

[5.1 实验目的 59](#_Toc503258074)

[5.2 实验内容 59](#_Toc503258075)

[5.2.1 源程序改错 59](#_Toc503258076)

[5.2.2 源程序完善、修改、替换 60](#_Toc503258077)

[5.2.3 跟踪调试源程序 62](#_Toc503258078)

[5.2.4 编程设计题 64](#_Toc503258079)

[5.2.5 选做题 71](#_Toc503258080)

[5.3 实验小结 75](#_Toc503258081)

[实验6 指针实验 76](#_Toc503258082)

[6.1实验目的 76](#_Toc503258083)

[6.2实验内容 76](#_Toc503258084)

[6.2.1源程序改错题 76](#_Toc503258085)

[6.2.2源程序完善、修改、替换题 77](#_Toc503258086)

[6.2.3跟踪调试题 78](#_Toc503258087)

[6.2.4编程设计题 79](#_Toc503258088)

[6.2.5选做题 88](#_Toc503258089)

[6.3 实验小结 90](#_Toc503258090)

[实验7 结构与联合实验 91](#_Toc503258091)

[7.1实验目的 91](#_Toc503258092)

[7.2实验内容 91](#_Toc503258093)

[7.2.1表达式求值的程序验证题 91](#_Toc503258094)

[7.2.2源程序修改替换题 91](#_Toc503258095)

[7.2.3编程设计题 94](#_Toc503258096)

[7.2.4选做题 104](#_Toc503258097)

[7.3 实验小结 121](#_Toc503258098)

[实验8 文件实验 122](#_Toc503258099)

[8.1实验目的 122](#_Toc503258100)

[8.2实验内容 122](#_Toc503258101)

[8.2.1．文件类型的程序验证题 122](#_Toc503258102)

[8.2.2．源程序修改替换题 123](#_Toc503258103)

[8.2.3编程设计题 125](#_Toc503258104)

[8.3 实验小结 127](#_Toc503258105)

# 实验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(#define PI 3.14159;) 利用#define预处理定义的符号常量后不应有分号，正确形式为：

#define PI 3.14159

2) 第3行(void main( void )) 主函数声明不应该使用void作为返回值类型，C标准规定，main函数应该使用int作为返回值类型，并且void和main连写了，正确形式为：

int main(void)

3) 第6行(short p, k;) 变量声明个数与后续使用的变量个数不一致，即newint变量未经过声明就进行使用，正确形式为：

short p, k, newint;

4) 第10行(scanf(“%d”, f ) ;) 使用scanf函数时，函数参数不是地址，即未对变量f进行取地址运算，正确形式为：

scanf(“%d”, &f ) ;

5)第11行(c = 5/9\*(f-32) ;) 5,9均为整形常量，在进行除法运算时，会截断小数部分，导致损失精度，应将其中任意一个改为浮点型常量，正确形式为：

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

6)第15行(scanf("%f", &r);) r为double类型，所以scanf的占位符应该为%lf，正确形式为：

scanf("%lf", &r);

7)第17行(printf("\nThe acreage is %.2f\n\n",&s);) 输出s的值应将s作为参数而不是s的地址，正确形式为：

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

8)第20行(scanf("%x %x", &k, &p );) k，p均为short型变量，以十六进制输出时，应该使用%hx而不是%x作为占位符，正确形式为：

scanf("%hx %hx", &k, &p );

9)第21行(newint = (p&0xff00)|(k&0xff00)<<8;) 要求将k的高字节作为结果的低字节，应该将k的高字节，提取出后右移至低字节，正确形式为:

newint=(p&0xff00)|(((k&0xff00)>>8)&0x00ff);

（2）错误修改后运行结果如图1-1所示。

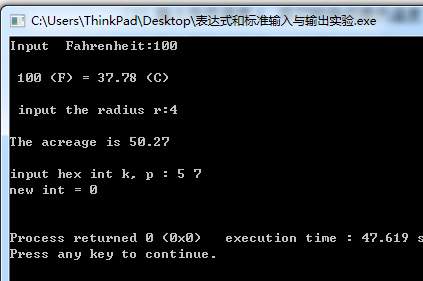


图1-1 源程序改错题运行截图

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：**

1. 思路：把a、b看做数轴上的点，围绕两点间的距离来进行计算。  
   具体过程：第一句“a=b-a”求出ab两点的距离，并且将其保存在a中；第二句“b=b-a”求出a到原点的距离（b到原点的距离与ab两点距离之差），并且将其保存在b中；第三句“a=b+a”求出b到原点的距离（a到原点距离与ab两点距离之和），并且将其保存在a中。
2. 替换后的程序如下所示：

#include<stdio.h>

void main( )

{

int a, b;

printf(“Input two integers:”);

scanf(“%d %d”,&a,&b);

a=b-a;

b=b-a;

a=b+a;

prinf(“\na=%d,b=%d”,a,b);

}

1. 输出实例如图1-2所示。

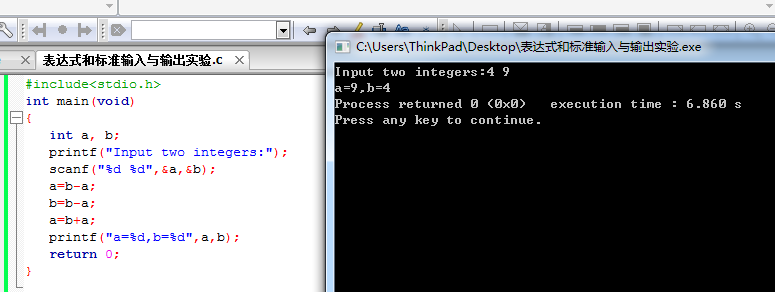


图1-2源程序修改替换题输出实例

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。

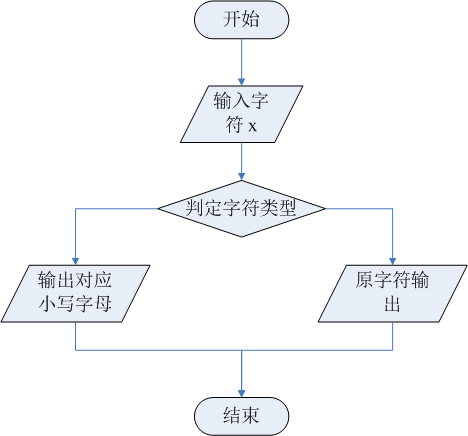


图1-3 编程题1的程序流程图

2）源程序清单

#include<stdio.h>

int main(void)

{

char x;

printf("Input your char x:\n");

scanf("%c", &x);

if (x >= 'a' && x <= 'z')//如果X是小写字母

printf("%c", x);//就打印X

else if (x >= 'A' && x <= 'Z')//如果X是大写字母

printf("%c", x+32);//那就输出X对应的小写字母

else //如果X不在范围内

printf("Please input the right char:\n");//就输出错误信息并结束

return 0;

}

3）测试

（a） 测试数据：

第一组：输入G 理论输出：g

第二组：输入r， 理论输出：r

第三组：输入2，理论输出：Please input the right char：

（b） 对应测试数据的运行结果如图1-4至1-6所示

第一组：

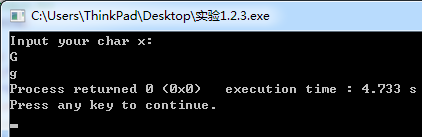


图1-4 编程设计题1第一组测试数据运行结果

第二组：

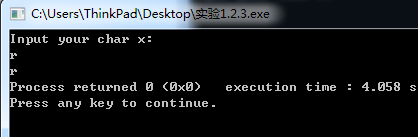


图1-6 编程设计题1第二组测试数据运行结果

第三组：

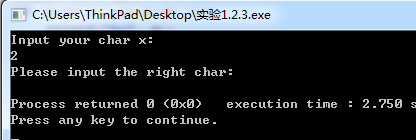


图1-6 编程设计题1第三组测试数据运行结果

**（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. 算法流程如图1-7所示。



图1-7 编程题2的程序流程图

2）程序清单

#include<stdio.h>

int main(void)

{

unsigned short x, m, n;

printf("Please input x(十六进制), m, n:\n");

scanf("%hx%hu%hu", &x, &m, &n);//以十六进制进行输入输出，便于检查

if (m >= 0 && m <= 15 && n >= 1 && n <= 16-m)

{

x >>= m;//将所要截取的位数段移到最右端

x <<= (16 - n);//将所要截取的位数段移到最左端

printf("%hx\n", x);//以十六进制输出X

}

else

printf("m, n不在规定的范围内\n");//输出错误信息

return 0;

}

3）测试

（a） 测试数据：

表1-1 编程题3的测试数据

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试  用例 | 程 序 输 入 | | | 理 论 结 果 |
| X(十六进制) | m | N |
| 用例1 | 0100 0110 1000 0000（4680） | 7 | 4 | 计算结果1101 0000 0000 0000 即D000 |
| 用例2 | 1101 0101 1000 0011（D583） | 16 | 1 | m, n不在规定的范围内（m值超范围） |
| 用例3 | 1101 0101 1000 0011（D583） | 13 | 5 | m, n不在规定的范围内（n值超范围） |

（b） 对应测试测试用例1的运行结果如图1-8所示。

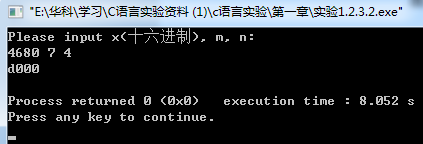


图1-8 编程题2的测试用例一的运行结果

对应测试测试用例2的运行结果如图1-9所示。

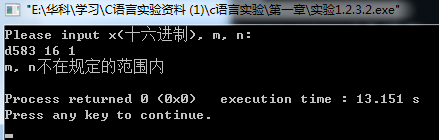


图1-9 编程2的测试用例二的运行结果

对应测试测试用例3的运行结果如图1-10所示。

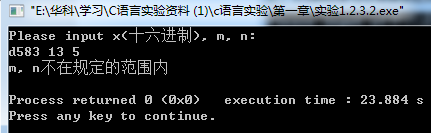


图1-10 编程题2的测试用例三的运行结果

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。

**解答：**

1. 算法流程如图1-11所示。



图1-11 编程题3的程序流程图

1. 程序清单

/\*

\*定义了四个逻辑尺

\*用于截取IP地址的四段

\*/

#include<stdio.h>

int main(void)

{

unsigned long int ip;

scanf("%lu", &ip);

unsigned short first = ip&0x000000ff;

unsigned short second = (ip&0x0000ff00)>>8;

unsigned short third = (ip&0x00ff0000)>>16;

unsigned short forth = (ip&0xff000000)>>24;

printf("%hu.%hu.%hu.%hu", first, second, third, forth);

//反向输出

return 0;

}

3）

a) 测试数据:

输入：676879571（00101000 01011000 01011100 11010011）

理论输出：211.92.88.40；

b)程序输出截图如图1-12所示：

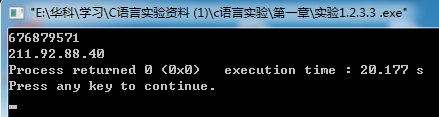


图1-12 编程题3的测试用例的运行结果

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。

## 1.3 实验小结

问题：

1. 在源程序纠错题中，由于short int的特殊性，忘记了在引用时，占位符都应使用h前缀，后来才想起应该要将x改为hx，对应变量的short int类型。

2．在第2道编程题中，一开始输入输出都是十进制，导致检验结果是否正确时十分麻烦，之后想到将输入输出都换为16进制以便于检查结果的正确性。因此，在第三道编程题中，我也采取了先转为16进制检验结果是否正确后再以十进制输出的方法。

体会：

1. 要注意输入输出的细节部分，例如占位符与变量类型是否匹配，变量在引用时是否已经有值等等。

2. 在调试程序是否正确时应该适当地调整输入输出模式，以达到最佳效果，而最终的输出格式可以在确认结果正确后再进行调试。如进制转化。

# 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. 错误修改
2. 第2行，将void改为int，同时在程序尾部添加return 0;因为C标准规定main函数返回值为int，而不是void.
3. 第6行，将n改为&n，使用scanf函数时传递的是要打印数的地址
4. 第7行，将逗号改为分号，由于for循环中要用分号间隔
5. 修改结果：

源程序代码：

#include <stdio.h>

int main(void)//此处原来为void,相应的在结尾添加return 0;

{

int i,n,s=1;

printf("Please enter n:");

scanf("%d",&n);//改为&n

for(i=1;i<=n;i++)//原来为逗号，改为分号

s=s\*i;

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

return 0;

}

运行结果如图2-1所示：

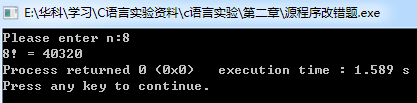


图2-1 源程序改错题运行结果

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。

**解题思路：**

用while和do-while替换for语句，就是将for循环的第一语句放在while循环外，将第二句作为while循环的判断条件，将第三句放在while循环尾部。

源代码：

1. do-while型

#include <stdio.h>

int main(void)

{

int n, i = 1, s=1;

printf("Please enter n:");

scanf("%d",&n);

do

{

s \*= i;

i++;

}while(i <= n);

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

return 0;

}

运行结果如图2-2所示：

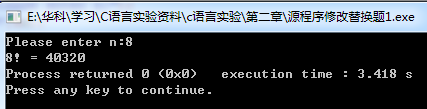


图2-2 程序修改替换题do-while型

1. while型

#include <stdio.h>

int main(void)

{

int n, i = 1, s=1;

printf("Please enter n:");

scanf("%d",&n);

while(i <= n)

{

s \*= i;

i++;

}

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

return 0;}

运行结果如2-3所示：

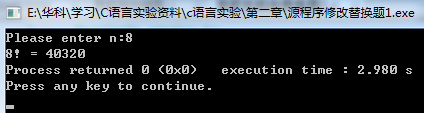


图2-4程序修改替换题while型

（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。

1）流程图如图2-5所示。

图2-5 源程序修改替换题流程图

2）源程序代码：

#include<stdio.h>

int main(void)

{

int S;

int n = 1;

scanf("%d", &S);

while(f(n) < S)

n++;

printf("%d", n);

return 0;

}

int f(int n)//创建函数，便于判断大小

{

int sum = 1, i = 1;

while(i <= n)

{

sum \*= i;

i++;

}

return sum;

}

1. 运行结果如图2-6所示：

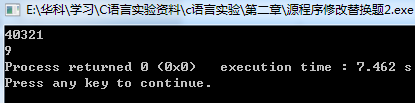


图2-7 程序修改替换题2运行结果

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. If语句程序流程图如图2-8所示， switch语句程序流程图如图2-9所示

‘’图2-8程序设计题1 if语句的流程图

图2-9程序设计题1 switch语句的流程图

1. 程序清单

**If型：**

#include<stdio.h>

int main(void)

{

int x;

printf("请输入工资:\n");

scanf("%d", &x);//输入x的值，然后根据他的范围来决定进行什么操作

if (x < 1000 && x >= 0)

printf("税金额度为0\n");

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

printf("税金额度为%.1f\n", (x -1000)\* 0.05);

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

printf("税金额度为%.1f\n", (x-2000) \* 0.1 + 50);

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

printf("税金额度为%.1f\n", (x -3000)\* 0.15 + 100+ 50);

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

printf("税金额度为%.1f\n", (x-4000) \* 0.2 + 150 + 100 + 50);

else if (x > 5000)

printf("税金额度为%.1f\n", (x -5000)\* 0.25 + 200 + 150 + 100 + 50);

else

printf("工资不能为负\n");//如果输入为负数，则提示并结束程序

return 0;

}

Switch型：

#include<stdio.h>

int main(void)

{

int x;

printf("请输入工资:\n");

scanf("%d", &x);

if(x < 0)//由于后期switch函数的default要用于处理大于5000的部分，所以需要在开始就排除掉x小于0的情况

{

printf("工资不能为负\n");

return 0;

}

switch(x / 1000)//根据x/1000的范围来判断进行的操作

{

case 0: printf("税金额度为0\n");break;

case 1: printf("税金额度为%.1f\n", (x -1000)\* 0.05);break;

case 2: printf("税金额度为%.1f\n", (x-2000) \* 0.1 + 50);break;

case 3: printf("税金额度为%.1f\n", (x -3000)\* 0.15 + 100+ 50);break;

case 4: printf("税金额度为%.1f\n", (x-4000) \* 0.2 + 150 + 100 + 50);break;

default : printf("税金额度为%.1f\n", (x -5000)\* 0.25 + 200 + 150 + 100 + 50);break;

}

return 0;

}

1. 测试
2. 用例：

此处只以if为例

用例1：输入 -1000 输出： 工资不能为负

用例2：输入 900 输出： 税金额度为0

用例3：输入 1500 输出：税金额度为25.0

1. 输出验证：

用例1到3的输出截图如图2-10到2-12所示：

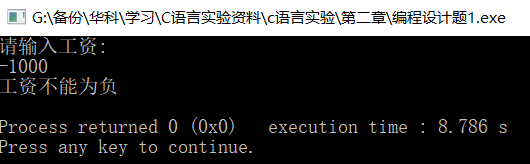


图2-10编程设计题1用例1

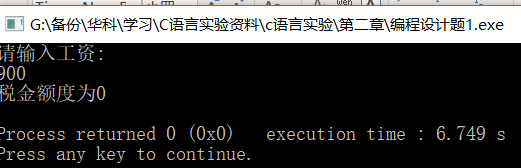


图2-11编程设计题1用例2

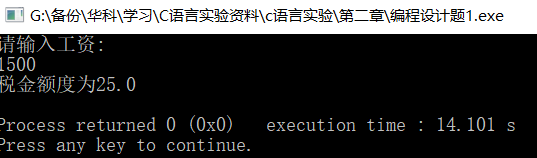


图 2-12 编程设计题1用例3

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。

1. 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. 程序流程图如图2-13所示：



图2-13程序设计题2流程图

1. 程序清单：

#include<stdio.h>

#include<string.h>

int main(void)

{

char c;

while((c = getchar()) != '\n')//读取字符到一行结束

{

if (c == ' ')//如果c是空格

{

putchar(c);//就先把c打印出来

while((c = getchar()) == ' ')//然后持续读取直到碰到第一个不是空格的字符

continue;

}

If (c != ‘\n’)//由于有可能c此时为换行符，所以要进行一次判断后再输出c

putchar(c);

}

return 0;

}

1. 测试输出

用例1 输入：“ aaa aa a ” 理论输出：“ aaa aa a ”

此输入包含了开头连续空格，中间连续空格，尾部连续空格三种情况

输出截图如图2-14所示：

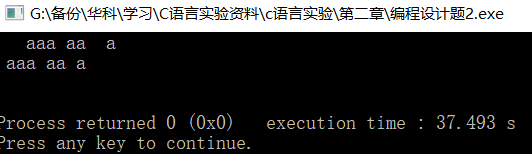


图2-14程序设计题2用例1输出

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）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. 由于程序流程图很复杂，此处以大概思路与程序注释替代程序流程图

思路：

第一层for循环循环column = 10次，用于打印每一行的内容。

第二层有两个for循环，第一个循环用于打印每一行前的空格，循环次数为总行数减去当前行数，每次打印两个空格。

第二个循环用于打印每一个数字，利用固定输出字段宽度为4并向左靠齐来满足题目中所说的数字间距要求，循环次数为当前行数

同时定义一个函数number用于打印i行j列的数字，使用递归。

1. 程序清单：

#include<stdio.h>

#define COLUMN 10//使用宏定义行数，便于后期修改

int number(int i, int j);

int main(void)

{

int i, j, m;

for(i = 0; i < COLUMN; i++)//用于打印每一行的内容。

{

for(m = COLUMN-i-1; m > 0; m--)//用于打印每行前的空行

printf(" ");

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

printf("%-4d", number(i, j));/\* 固定输出字段宽度为4并向左靠齐\*/

printf("\n");

}

return 0;

}

int number(int i, int j)

{

int sum;

if (!j)

sum = 1;

else

sum = number(i, j-1)\*(i - j + 1)j;/\*利用题目中的提示编写递归，计算i行j列的内容\*/

return sum;

}

1. 测试截图

截图如图2-15所示

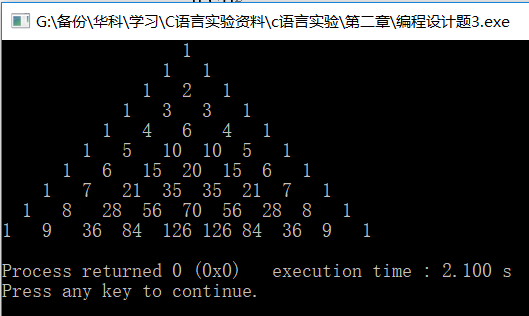


图2-15编程设计题3输出截图

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。

（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。

程序流程图如图2-16所示



图2-16程序设计题4 流程图

1. 源程序清单：

#include<stdio.h>

int main(void)

{

int number;

printf("请输入一个正整数:\n");

scanf("%d", &number);//读取number

if (number > 0)//如果number大于0

{

while(number)

{

printf("%d", number%10);打印number的各位

number /= 10;//去掉number的个位，便于进行下一轮循环

}

}

else

printf("输入的值不是正数");

return 0;

}

1. 测试

用例1：输入4321 输出1234

用例2：输入2510 输出0152

用例1,2截图如图2-17与2-18所示

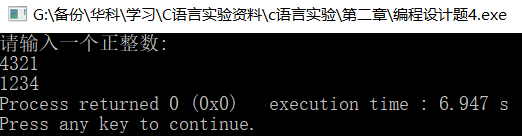


图2-17编程设计题4用例1

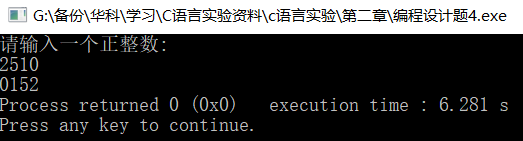


图2-18编程设计题4用例2

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。

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. 程序流程图如图2-19所示：



图2-19选做题程序流程图

1. 源程序清单：

#include <stdio.h>

#include <math.h>

#define f(x) (3\*(x)\*(x)\*(x)-4\*(x)\*(x)-5\*(x)+13)//宏定义函数及其参数，便于后期修改

#define f1(x) (9\*(x)\*(x)-8\*(x)-5)

#define A 1

int main(void)

{

double x1 = A, x2, result;

x2 = x1 - f(x1)/f1(x1);

while(fabs(x2-x1) > 1e-6)

{

x1 = x2;

x2 = x1 - f(x1)/f1(x1);

}

printf("%lf\n", x2);

printf("x2 - x1 = %lf\n", x2-x1);

return 0;

}

1. 测试结果：

由于只有输出，故直接给出运行截图

截图如图2-20所示：

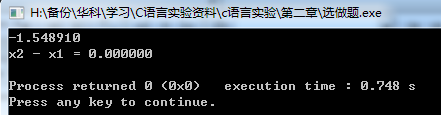


图2-10选做题测试截图

与预期结果一致。证明结果正确

## 2.3实验小结

问题：在编程设计题1中最初没有理解分段收取税金的含义，导致整个程序错误，说明每次在写程序时，最好是确保已经完全弄懂题目的意思再进行操作

体会：在此次实验中，我切实感受到了程序设计时所要注意的一些问题，尤其是程序的可修改性，比如打印金字塔一题，我使用了宏定义column = 10，后期如果需要对打印的行数进行修改就很容易，同样在选做题中，f(x)与其导数均为宏定义，也是方便函数后期修改的

# 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++)

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

8. }

9. long sum\_fac(int n)

10. {

11. long s=0;

12. int i;

13. long fac;

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

15. fac\*=i;

16. s+=fac;

17. return s;

18. }

**解答：**

1. 错误修改：
2. 第1行，将”stdio.h”改为<stdio.h>双引号表示在本地查找，尖括号表示在C标准库中查找
3. 第2行，将void改为int，同时在程序尾添加return 0，由于C标准规定main函数的返回值为int型
4. 第13行，应给fac赋初值，否则后面fac将会以垃圾值进行计算，并得到错误的结果
5. 第16行，应该将s += fac;这一句放在for循环内部，否则会导致计算错误。
6. 修改结果：

源程序代码：

#include <stdio.h> //应该改为<stdio> "双引号代表在本地查找"

long sum\_fac(int n);

int main(void)//改为int,同时在尾部添加return 0;

{

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)

{

long s=0;

int i;

long fac=1;//应对fac赋初值，否则在接下来的计算中会出现错误

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

{

fac\*=i;

s+=fac;//此句应该放在for循环中

}

return s;

}

程序运行结果如图3-1所示：

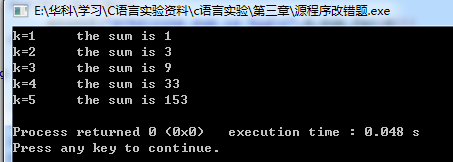


图3-1 源程序改错题截图

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. 思路：由于该程序为顺序输出，故，可以利用这一特性使用static静态变量，以减小运算量。
2. 源程序代码：

#include <stdio.h>//利用该题按顺序打印的特点可以通过创建static变量来减少计算量

long sum\_fac(int n);

int main(void)

{

int k;

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

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

return 0;

}

static long s = 0; fac = 1; //声明静态变量

long sum\_fac(int n)

{

fac \*= n;

s += fac;

return s;

}

1. 运行结果如图3-2所示：

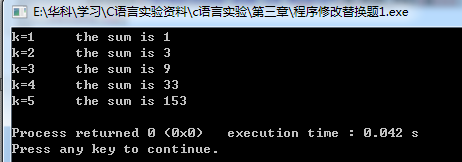


图3-2源程序修改替换题1运行结果

1. 修改第1题中sum\_fac函数，计算。

**解答：**

1. 思路：通过修改变量类型，即将int型改为double，然后再对负责计算的函数进行少量修改。
2. 源程序代码：

#include <stdio.h>

double sum\_fac(int n);

int main(void)

{

int k;

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

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

return 0;

}

static double s = 0, fac = 1;//修改变量类型

double sum\_fac(int n)

{

fac\*=n;

s+=(1/fac);//修改计算方式

return s;

}

1. 程序运行截图如图3-3所示：

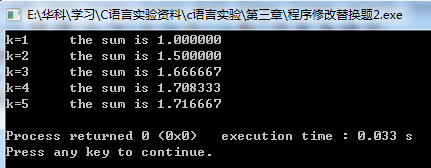


图3-3 程序修改替换题2运行截图

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：**

在本机上：

p = 0x28ff10 i = 2

截图如图3-4所示：

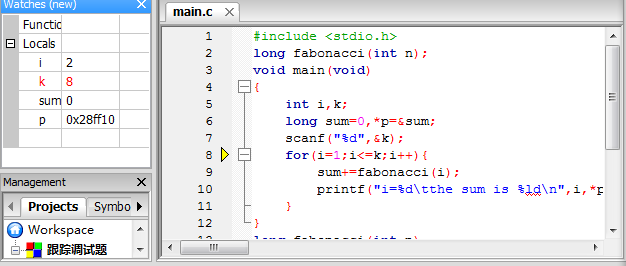


图3-4 程序跟踪调试题1截图

1. 从fabonacci函数返回后光条停留在哪个语句上？

**解答：**

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

截图如图3-5所示：

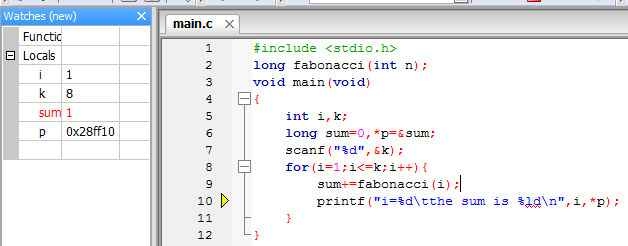
****

图3-5 源程序跟踪调试题2截图

1. 进入fabonacci函数时，watch窗口显示的是什么？

**解答：**

显示的是fabonacci函数中形参的值

截图如图3-6所示：

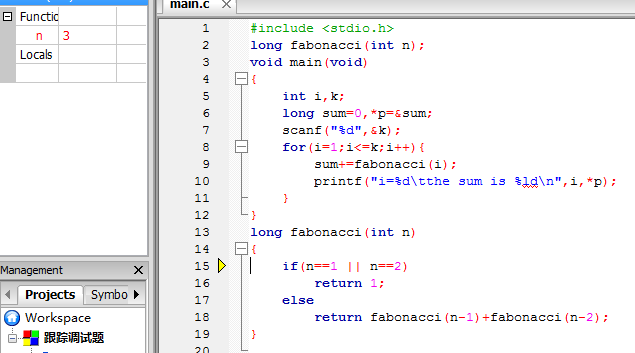


图3-6 程序跟踪调试题3

1. 当i=3时，从调用fabonacci函数到返回，n值如何变化？

**解答：**

n 从3变为2

**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. 程序流程图如图3-7所示：



图3-7 编程设计题1流程图

1. 源程序代码：

#include <stdio.h>

int f(int m, int n);

int main(void)

{

int a, b;

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

a > b ? printf("最大公约数为%d ", f(a, b)) :

printf("最大公约数为%d ", f(b, a)); //根据a, b大小的不同来分别处理

return 0;

}

int f(int m, int n)

{

m %= n;

if(m)//如果m不为零

f(n, m);//进行下一轮迭代

else

return n;//否则返回被除数，即最大公约数

}

1. 测试：

测试用例1：输入5 8 理论输出：1

测试用例2：输入14 7 理论输出：7

测试用例3：输入9 9 理论输出：9

用例截图如图3-8到3-10所示：

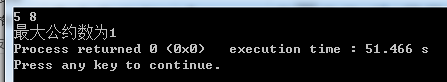


图3-8 编程设计题1用例1截图

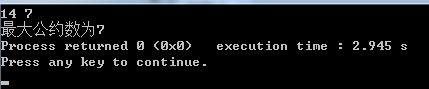


图3-9 编程设计题1用例2截图

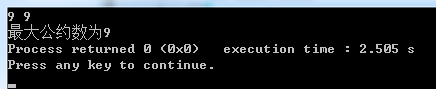


图3-10 编程设计题1用例3截图

（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. 程序流程图如图3-11所示：

图3-11 编程设计题2 流程图

1. 程序清单：

#include <stdio.h>

int main(void)

{

int num, i = 2;

printf("请输入一个正偶数\n");

scanf("%d", &num);

while(!(f(i) && f(num-i)))//当两个有一个不是质数时

i++;//测试下一个数

printf("%d = %d + %d", num, i, num - i);//输出结果

return 0;

}

int f(int n)

{

int i, j;

for (i = 2, j = (int)sqrt(n); i <= j; i++)//为了减小计算量，以sqrt(n)为界

if (n % i)

continue;

else

return 0;

return 1;

}

1. 测试：

测试截图如图3-12所示：

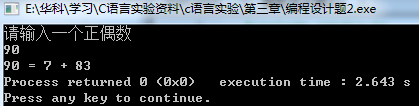


图3-12 程序设计题2测试截图

（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. 程序流程图如图3-13所示：（此处num应该为num += 2 而不是num++）

图3-13 编程设计题3 流程图

1. 程序清单：

#include <stdio.h>

#include <math.h>

#define BEGIN 20

#define END 50

int main(void)

{

int num = BEGIN, i;

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

while(num <= END)//使用while循环对每一个数进行判断

{

i = 2;

while(!(f(i) && f(num-i)))

i++;

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

num += 2;

}

return 0;

}

int f(int n)

{

int i, j;

for (i = 2, j = (int)sqrt(n); i <= j; i++)//与上一题基本相同

if (n % i)

continue;

else

return 0;

return 1;

}

1. 测试（begin为20 ，end为50）

测试截图如图3-14所示：

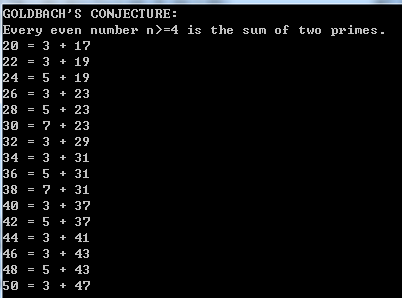


图3-14 程序设计题3 测试截图

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：**

根据上述代码，成功生成可执行文件，直接放上截图（如图3-15与图3-16所示）：

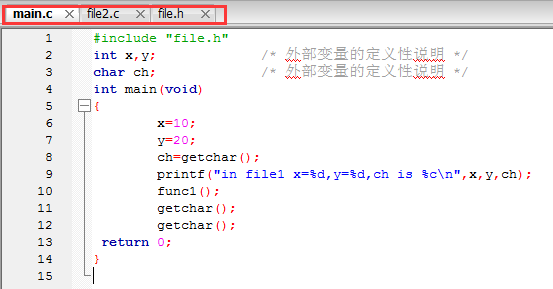


图3-15选做题截图1

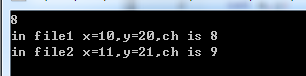


图3-16选做题截图2

## 3.3实验小结

体会：在设计程序时，可以通过分析程序特征来改进计算方法，以减小计算量，缩短计算时间，如源程序修改替换题中，利用顺序打印的特点，使用了static静态变量，大大减小了计算量。故在程序设计题2和3中判断一个数是否为质数时，都是以sqrt为界，以减小计算量。

# 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. 错误修改：
2. 第二行，将a + b 改为 ((a) + (b))，由于不清楚具体计算时符号，也就不能判定优先级，同理
3. 第三行，将a – b 改为 ((a) – (b))
4. 第四行，将SWAP(a,b) a=b,b=a 改为SWAP(a,b,t) {t = a; a = b; b = t;}，交换a, b时需要一个变量进行过度
5. 第七行，声明变量a, 由于在后面程序使用了a，而并没有进行声明，需要在此声明。
6. 第十二行，将SWAP(a,b);改为SWAP(a,b,t)，由于在之前定义时，就为三个变量，前后需要保持一致。
7. 修改结果：
8. 程序清单：

#include "stdio.h"

#define SUM ((a) + (b))//应该改为((a) + (b))由于不知道具体的运算先后顺序

#define DIF ((a) - (b))//同理改为((a) - (b))

#define SWAP(a,b,t) {t = a; a = b; b = t;}//改为三个参数

void main()

{

int b, t, a;

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

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

printf("\nSUM=%d\n the difference between square of a and square of b is:%d",SUM, SUM\*DIF);

SWAP(a,b,t);//添加参数t

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

}

1. 运行结果如图4-1所示：

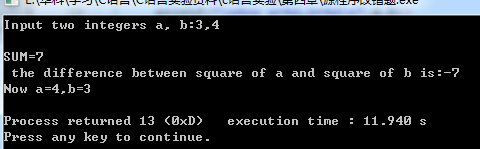


图4-1 源程序改错题运行截图

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. 第一问：
2. 程序清单：

#include <stdio.h>//此处添加了头文件

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

float sum(float x, float y);//此处添加了两个函数的声明

int main(void)//修改main函数返回值类型，同时在程序尾添加return 0;

{

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));

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;

}

float sum(float x, float y)

{

return x+y;

}

1. 运行结果如图4-2所示：

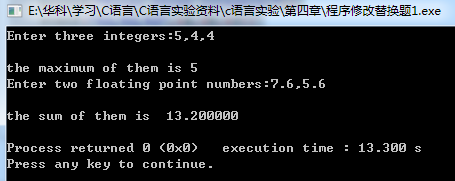


图4-2 源程序修改替换题第一小问运行截图

1. 第二问：
2. 程序清单：

#include <stdio.h>

#define max(x,y,z) (((x)>(y)?(x):(y))>(z))?((x)>(y)?(x):(y)):(z)//此处用宏定义替换了后面的max函数

float sum(float x, float y);

int 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));

return 0;

}

float sum(float x, float y)

{

return x+y;

}

1. 程序运行截图如图4-3所示：

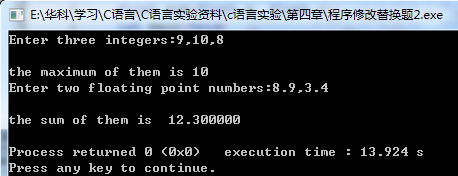


图4-3 程序修改替换题第二小问运行截图

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

}

int integer\_fraction(float x)

{

int i=x;

return i;

}

**解答：**

1. 程序清单：

#include <stdio.h>

#include <limits.h>

#include <assert.h>//添加了三个头文件

#define R

int integer\_fraction(float x);//提前声明函数

int main(void)//将main函数返回值改为了int 并在程序尾添加了return 0;

{

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);

if(s<=INT\_MAX)//在s不溢出时才打印

{

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(float x)

{

int i=x;

return i;

}

运行截图如图4-4所示：

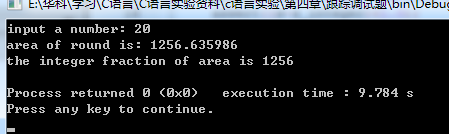


图4-4 跟踪调试题 修改后运行截图

1. 运行截图如图4-5与4-6所示：

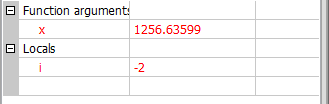


图4-5程序跟踪调试题 运行截图1

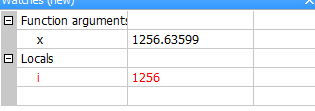


图4-6 程序跟踪调试题 运行截图2

1. 请看注释

4.2.4编程设计题

（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. 程序流程图如图4-7所示：



图4-7 编程设计题1 程序流程图

1. 源程序清单：

#include <stdio.h>//该程序默认输入的三边可以构成三角形

#include <math.h>//由于用到了sqrt函数，添加math.h头文件

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

#define AREA(a,b,c) (sqrt(((S(a,b,c))-(a))\*((S(a,b,c))-b)\*((S(a,b,c))-c)\*(S(a,b,c))))

//分别定义计算S和AREA的宏

int main (void)

{

float a,b,c;

printf("请输入三角形的三边：\n");

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

printf("面积为%f", AREA(a,b,c));

return 0;

}

1. 测试：

测试用例1:输入：3，4，5 理论输出：6

测试用例2:输出：6，6，6 理论输出：15.588

运行结果如图4-8与图4-9所示

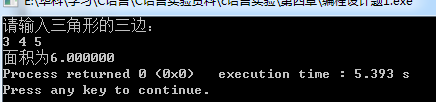


图4-8 编程设计题1 用例1截图

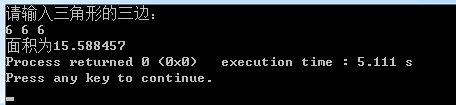


图4-9 编程设计题1 用例2截图

（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。

**解答：**

1. 程序流程图如图4-10所示。



图4-10 编程设计题2 程序流程图

1. 源程序清单：

#define CHANGE 0

#include <stdio.h>

int main(void)

{

printf("请输入一行电报文字\n");

char c;

#if CHANGE == 1

while((c = getchar()) != '\n')

{

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

putchar(c+32);

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

putchar(c-32);

else

putchar(c);//判断c的类型并采取相应的操作

}

#elif CHANGE == 0

while((c = getchar()) != '\n')

putchar(c);

#endif // CHANGE

return 0;

}

1. 测试：（分别令CHANGE为1和0）

测试一：0：输入：abd\*7& 理论输出：abd\*7&

测试二：1：输入：abdFgY&$ 理论输出：ABDfGy&$

测试截图如图4-11与4-12所示：

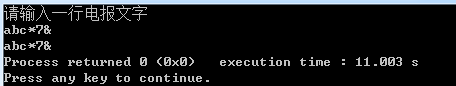


图4-11 程序设计题2 测试用例1截图

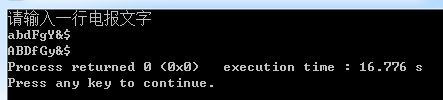


图4-12 程序设计题2 测试用例2截图

## 4.3实验小结

体会：体会到了宏定义在程序中的重要性。宏定义一般来说有两个作用，一个是通过定义明示常量来减小后期程序移植时对程序所做的修改。令一个是通过定义带参数的宏来减少对应函数被调用时的内存损耗。但同时在使用宏定义替换函数时，要注意优先级的问题，所以需要在必要的地方加上括号，同时要注意，宏定义毕竟不是函数，所以要尽量避免使用具有副作用的参数。

# 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;

1. }

**解答：**

1. 错误清单与源程序：

#include<stdio.h>

void sort(int [],int);//函数声明放在main函数外面

int main(void)

{

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

int i;

sort(a,10);//此处传递的应该是a而不是a[0]

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

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

printf("\n");

return 0;

}

void sort(int b[], int n)

{

int i, j, t;

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

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

if(b[j] > b[j+1])//由于是按升序排列，所以需要改变不等号方向

{

t = b[j];

b[j] = b[j+1];

b[j+1] = t;//将分隔语句的逗号改为分号，同时加上括号

}

}

1. 运行截图如图5-1所示：

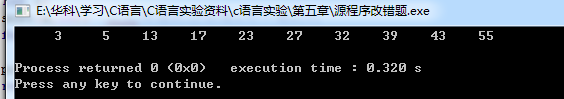


图5-1 源程序改错题运行结果截图

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;

}

(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. 填空：
2. b[M-1] = j ? a[j-1]:a[i-1];
3. for(k = --j; k < i; k++)

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

运行截图如图5-2所示：

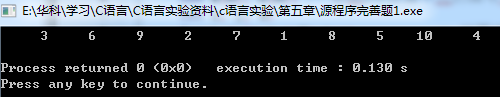


图5-2 程序完善题第一问运行截图

1. 优化：

程序清单：

#include<stdio.h>

#define M 10

#define N 3

int a[M];//初始化数组a为0

int main(void)

{

int b[M];

int i, j, k;

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

{

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

{

if (j == M)//如果数到了最后一个人

j = 0;//就回到第一个人继续数

if (a[j])//如果a[j]为1,即被标记，则跳过他，进行下一轮

k--;

j++;

}

a[j-1] = 1;//标记数3的人

b[i] = j;//记录出圈的人的编号

}

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

printf("%6d", b[i]);//打印结果

return 0;

}

1. 运行截图如图5-3所示：

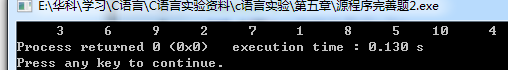


图5-3 源程序完善题第二问运行截图

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？

（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. 单步执行截图如图5-4与图5-5所示:

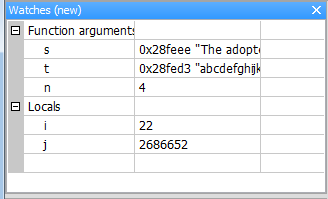


图5-4 跟踪调试题单步执行截图1

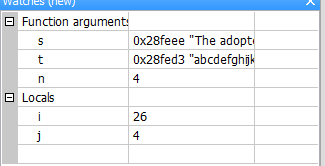


图5-5 跟踪调试题单步执行截图2

2. 在strncat函数中，寻找字符串中的空字符时出错，应将 while(s[i++]) ;改为while(s[i])

i++;使得在找到空字符后i能够准确定位。

运行结果如图5-6所示：

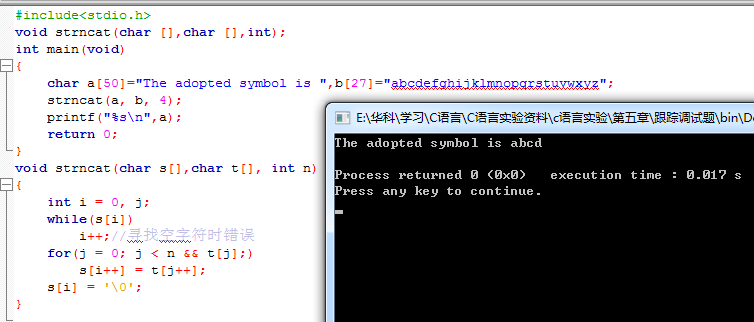


图5-6 跟踪调试题运行结果截图

5.2.4 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）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. 程序流程图如图5-7所示：



图5-7 程序设计题1流程图

1. 程序清单：

#include <stdio.h>

#define M 3

#define N 4

void scanarray(int m, int n, int (\*array)[n]);

void printarray(int m, int n, int (\*array)[n]);

int main(void)

{

int i, j;

int a[M][N];//创建需要进行计算的数组

int b[N][M];//与存放结果的数组

scanarray(M, N, a);//用于读入数据的函数

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

for (j = 0; j < N; j++)//循环，将a中的值赋给b,即转置矩阵

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

printf("原矩阵为：\n");

printarray(M, N, a);

printf("现矩阵为：\n");

printarray(N, M, b);

return 0;

}

void scanarray(int m, int n, int (\*array)[n])

{

int i, j;

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

{

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

scanf("%d", (\*(array+i)+j));

}

}

void printarray(int m, int n, int (\*array)[n])//用于打印矩阵的函数

{

int i, j;

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

{

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

if (j + 1 == n)

printf("%d", \*(\*(array+i)+j));

else

printf("%d ", \*(\*(array+i)+j));

printf("\n");

}

}

1. 运行截图如图5-8所示：

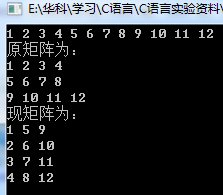


图5-8 编程设计题1运行截图

（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. 程序流程图如图5-9所示：



图5-9 编程设计题2程序流程图

1. 程序清单：

#include <stdio.h>

int main(void)

{

int num;

char string[33];

int m, i;

printf("请输入要计算的数字：\n");

scanf("%d", &num);

for (i = 0, m = 1; i < 32; i++)

{

string[31-i] = m & num ? '1' : '0';//将结果存入一个字符数组中

m <<= 1;//对每一位用逻辑尺得到结果

}

string[32] = '\0';//将字符数组转变为字符串

printf("%s", string);//打印结果

return 0;

}

1. 程序运行截图如图5-10所示：

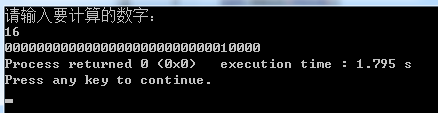


图5-10 编程设计题2 运行截图

（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. 程序流程图如图5-11所示：



图5-11 程序设计题3 程序流程图

1. 程序清单：

#include <stdio.h>

#include <string.h>

int main(void)

{

char tempstring[10];

int n, i, j, temp, findscore;

printf("请输入输入的数据组数：\n");

scanf("%d", &n);

char names[n][10];

int scores[n];

for (i = 0; i < n; i++)//循环读取数据

{

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

scanf("%d", scores+i);

}

for (i = 0; i < n; i++)//冒泡排序

{

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

{

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

{

temp = scores[j];

scores[j] = scores[j+1];

scores[j+1] = temp;

strcpy(tempstring, names[j]);

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

strcpy(names[j+1], tempstring);

}

}

}

/\*打印表\*/

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

{

printf("%s:", names[i]);

printf(" %d\n", scores[i]);

}

printf("请输入查找的分数：\n");

scanf("%d", &findscore);

int low = 0, high = n - 1, mid;//二分查找，寻找目标分数

int flag = 0;

while (low <= high)

{

mid = (low + high + 1)/2;

if (scores[mid] > findscore)

high = mid - 1;

else if (scores[mid] < findscore)

low = mid + 1;

else

{

findscore = mid;

flag = 1;

break;

}

}

if (flag)//如果找到的话

{//打印名字与分数

printf("%s", names[findscore]);

printf(" %d\n", scores[findscore]);

}

else//如果没有找到

printf("not found!\n");//就提示没有找到

return 0;

}

1. 程序运行截图如图5-12与图5-13所示：

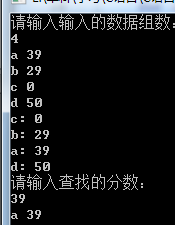


图5-12 程序设计题3运行截图1

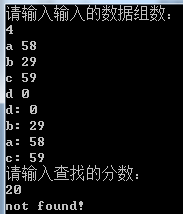


图5-13 程序设计题3 运行截图2

5.2.5 选做题

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. 思路：

从s中的第n+1个字符开始一直到最后的空字符，一个个移动到length(s)+length(t)-n个之后，为t的插入腾出空间，然后调用strcpy函数将t中的字符插入到s中第n个字符后

1. 程序清单：

#include <stdio.h>

#include <string.h>

void strnins(char \*s, char \*t, int n);

int main(void)

{

char string1[20] = "jaodcjsi";

char string2[] = "dcs";

strnins(string1, string2, 3);

printf("%s", string1);

return 0;

}

void strnins(char \*s, char \*t, int n)

{

int j;

int length1 = strlen(t);

int length2 = strlen(s);

for (j = 0; j < length2 - n + 1; j++)

s[length1 + length2 - j] = s[length2 - j];//移动字符串，为t腾出空间

strncpy(s+n, t, length1);//覆盖字符串

}

1. 运行截图如图5-14所示：

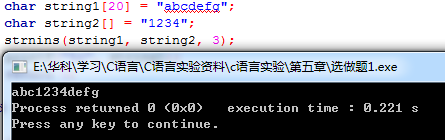


图5-14 选做题运行截图

(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. 思路：

用数组a[8][8]表示棋盘，初始化为0，从第一行开始，调用函数findposition()寻找每一行的皇后。在第i行中，从a[i][0]到a[i][8],若a[i][j]的值为0，则说明可以放置皇后，改行找到皇后之后，对棋盘做出相应变化，即调用函数markposition()，以该皇后为中心，将其所能影响到的每一行，每一列，以及，每一斜列的值都加一，表示该位置不能防止皇后，在调用函数findposition寻找下一行的皇后位置，如果到第8行找到了位置就将结果打印出来，如果在某一行没有找到可以放置皇后的位置，就调用regainposition撤销上一行放置棋子的影响，即将其所能影响的位置的数值都减一，并回到上一行，重新寻找位置。创建一个数组b[8]用于记录每一行放置棋子的位置，方便操作。

1. 程序清单：

#include <stdio.h>

void markposition(int i, int j);//用于对皇后周边的棋子造成影响

int findposition(int i, int j);//用于在i行j列中寻找位置

void regainposition(int i, int j);//用于消除i行j列皇后的影响

void printresult(void);//用于打印最后的结果

int num[8][8];

int record[8];

int sum = 0;

int main(void)

{

int i = 0;

while (record[0] < 8)

{

if (findposition(i, record[i]))//如果找到了位置

{

markposition(i, record[i]);//进行标记

if (i == 7)//如果到了最后一行

{

printresult();//就打印结果

printf("\n");

regainposition(i, record[i]);//进行回溯

record[i]++;

if (record[i] == 8)

goto part1;

continue;

}

i++;

}

else

{

part1:

if (i == 0)

break;

regainposition(i-1, record[i-1]);//进行回溯

record[i] = 0;

record[--i]++;

if (record[i] == 8)//如果上一行也没有找到

goto part1;//就回到上上行

}

}

printf("sum = %d", sum);

return 0;

}

void markposition(int i, int j)

{

int m;

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

{

num[i][m]++;//一行

num[m][j]++;//一列

if (i - m >=0 && j - m >= 0)

num[i-m][j-m]++;//左上

if (i + m < 8 && j + m < 8)

num[i+m][j+m]++;//右下

if (i - m >= 0 && j+m < 8)

num[i-m][j+m]++;//左下

if (i + m < 8 && j - m >= 0)

num[i+m][j-m]++;//右下

}

num[i][j] = 1;//重新将皇后标为1

}

int findposition(int i, int j)

{

for (; j < 8; j++)

if(num[i][j] == 0)

{

num[i][j] = 1;

record[i] = j;//记录当前行的操作位置

return 1;

}

return 0;

}

void regainposition(int i, int j)//同findposition

{

int m;

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

{

num[i][m]--;

num[m][j]--;

if (i - m >=0 && j - m >= 0)

num[i-m][j-m]--;

if (i + m < 8 && j + m < 8)

num[i+m][j+m]--;

if (i - m >= 0 && j+m < 8)

num[i-m][j+m]--;

if (i + m < 8 && j - m >= 0)

num[i+m][j-m]--;

}

num[i][j] = 0;

}

void printresult(void)

{

int i, j;

for (i = 0; i < 8; i++)//对每一行进行搜索

{

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

if (num[i][j] == 1)//如果发现皇后，就打印坐标

{

printf("<%d,%d> ", i, j);

break;//并直接转到下一行

}

}

sum++;

}

1. 运行结果截图如图5-15所示:

<a,b>代表每一种做法的每一行的皇后的坐标(此图不是完整截图)

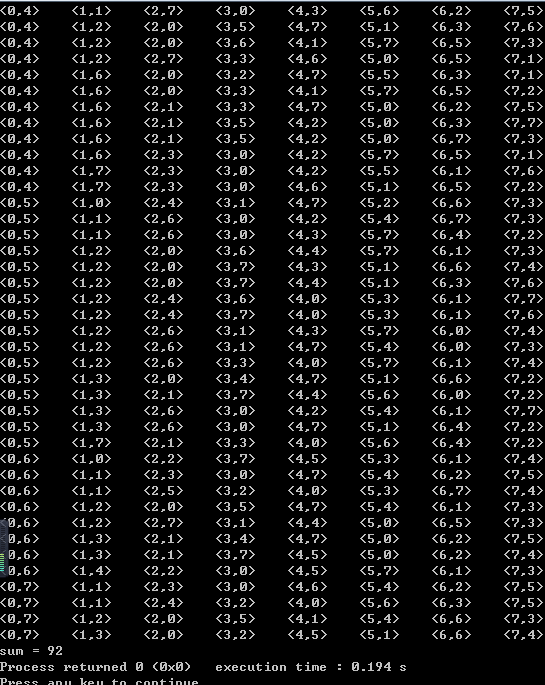


图5-15 选做题3运行截图

## 5.3 实验小结

体会：通过第五章的实验，我深刻体会到了数组下标的重要性，特别是在循环控制的时候，对于数组下标的控制就尤为重要。还有就是在做“八皇后”问题时，接触到了“回溯算法”, 通过回溯算法实际上一个类似枚举的搜索尝试过程，主要是在搜索尝试过程中寻找问题的解，当发现已不满足求解条件时，就“回溯”返回，尝试别的路径。这样就可以通过回溯来大大减少运算次数，以提高程序运行速度，当然，我最终的做法还是比较复杂，针对这个题其实还有一些更加好的方法，所以说编写程序就是一个错误和优化中不断提高的过程。

# 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。

#include "stdio.h"

void main(void)

{

float \*p;

scanf("%f",p);

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

}

解答：

1. 错误清单以及修改后的程序：

#include "stdio.h"

void main(void)

{

float x, \*p = &x;//加入一个变量x用于给p指

scanf("%f",p);

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

}

1. 运行截图如图6-1所示：

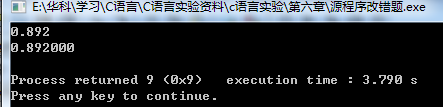


图6-1 源程序改错题运行截图

6.2.2源程序完善、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：

1. 补充空白：

第一空：char\* (\*p)(char \*,const char \*);//声明函数指针p

第二空：gets(a);//读入数据

第三空：gets(b);//读入数据

第四空：result = p(a,b);//处理数据

1. 运行截图如图6-2所示：

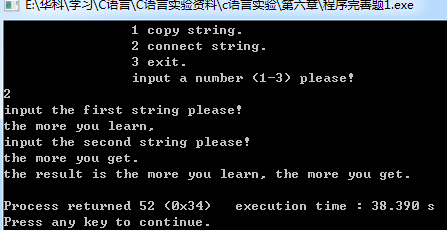


图6-2 源程序完善题运行截图

6.2.3跟踪调试题

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？

（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. 进入strcpy时的截图与返回main时的截图如图6-3与6-4所示。



图6-3 进入strcpy时s的值



图6-4 返回main时s的值

1. 排除错误：

修改后代码与错误如下：

#include "stdio.h"

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

void main(void)

{

char a[60],b[60]="there is a boat on the lake.";//修改a的长度使其能够接收b

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

}

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

{

char \*p = s;

while(\*p++=\*t++)//创建临时变量，避免在循环时改变s的值

;

return (s);

}

程序运行截图如图6-5所示。

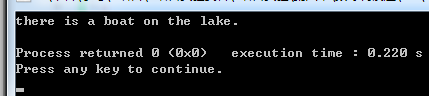


图6-5 跟踪调试题运行截图

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. 流程图如图6-6所示：



图6-6 程序设计题1的流程图

1. 程序清单：

#include <stdio.h>

int main(void)

{

long x=0x1234ABCD, k;

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

char up\_half, low\_half;

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

{

low\_half = (\*p)&0x0f;

if(low\_half<10)

low\_half += '0';

else

low\_half = (low\_half-10)+'A';

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

if(up\_half<10)

up\_half += '0';

else

up\_half = (up\_half-10)+'A';

p++;

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

}

return 0;

}

1. 程序运行截图如图6-7所示：

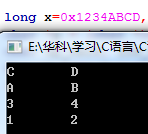


图6-7程序设计题1运行截图

（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. 程序流程图如图6-8所示：

其中change函数的想法为一旦遇到空格，就开始向后读取，直到不是空格为止，然后将后边的字符复制到第一个空格之后，如果遇到空行就break，在处理完后，将结果输出，并进行下一轮的操作。



图6-8 编程设计题2的程序流程图

1. 程序清单：

#include <stdio.h>

#include <string.h>

void change(char \*s[], int n);

int main(void)

{

int n, i;

printf("请输入计算的组数：\n");

scanf("%d", &n);

getchar();

printf("请输入每一行的数据：\n");

char strings[n][80];

char \* p[n];

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

{

gets(strings[i]);

p[i] = strings[i];

}

change(p, n);

return 0;

}

void change(char \*s[], int n)

{

int i, j, temp;

int flag = 0;

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

{

for (j = 0;;j++)

{

if (\*(s[i]+j) == '\0')

break;

if (\*(s[i]+j) == ' ')

{

temp = j;

while(\*(s[i]+j+1) == ' ')

j++;

if (temp != j)

{

strcpy(s[i]+temp+1, s[i]+j+1);

flag = 1;

}

}

if (flag)

j = temp + 1;

flag = 0;

}

if (strlen(s[i]))

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

}

}

1. 测试：

用例：5

jk lk o o oo p l

J o sss ww q p e q

\n

Ji ko o o o

K o o

理论输出：jk lk o o oo p l

J o sss ww q p e q

Ji ko o o o

K o o

实际运行截图如图6-9所示：

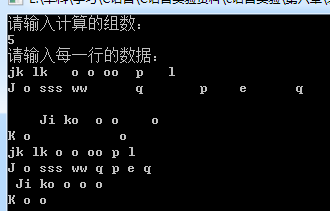


图6-9 编程设计题2 测试截图

（4）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。）

1. 具体思路（此题为单纯的计算，故没有使用流程图，直接写出思路）

根据M，N定义一个二维数组，用于储存成绩表（包含科目与学生）再分别定义五个全局数组，便于函数操作，分别储存每个学生的平均分，每个科目的平均分，每个科目低于平均分的人数，每个科目高于90分的人数，每个科目不及格的人数，再定义两个字符串二维数组，用于储存学生姓名与科目名称，待全部数据输入完毕后，调用函数依题意进行计算，最后打印结果

1. 程序清单：

#include <stdio.h>

#define M 3

#define N 4

double averagescores[M];

double averagescores2[N];

int lowercount[M];

int bad[M];

int higher[M];

void average(char (\*p)[10], char (\*q)[10], int (\*scores)[N]);

void average2(char (\*q)[10], int (\*scores)[N]);

void lowercount2(char (\*p)[10], int (\*scores)[N]);

void bad\_good(char (\*p)[10], int (\*scores)[N]);

int main(void)

{

int i, j;

char subj[M][10];

char name[N][10];

int scores[M][N];

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

{

printf("请输入第%d门课程的名字：", i+1);

scanf("%s", \*(subj+i));

}

printf("\n");

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

{

printf("请输入第%d个学生的名字： ", i+1);

scanf("%s", \*(name+i));

}

printf("\n");

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

{

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

{

printf("请输入%s的%s成绩： ", name[i], subj[j]);

scanf("%d", \*(scores+j)+i);

}

printf("\n");

}

printf("\n");

average2(name, scores);

average(subj, name, scores);

lowercount2(subj, scores);

bad\_good(subj, scores);

return 0;

}

void average(char (\*p)[10], char (\*q)[10], int (\*scores)[N])

{

int i, j;

double sum = 0.0;

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

{

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

sum += \*(\*(scores+i)+j);

averagescores[i] = sum/N;

printf("课程%s的平均成绩为: %.2f\n", \*(p+i), sum/N);

sum = 0;

}

printf("\n");

}

void average2(char (\*q)[10], int (\*scores)[N])

{

int i, j;

double sum = 0.0;

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

{

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

sum += \*(\*(scores+j)+i);

averagescores2[i] = sum/M;

printf("%s的平均成绩为：%.2f\n", \*(q+i), sum/M);

sum = 0;

}

printf("\n");

}

void lowercount2(char (\*p)[10], int (\*scores)[N])

{

int i, j;

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

{

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

{

if (\*(averagescores+i) > \*(\*(scores+i)+j))

(\*(lowercount+i))++;

}

printf("对于课程%s，低于平均分的人数为：%d\n", \*(p+i), lowercount[i]);

}

printf("\n");

}

void bad\_good(char (\*p)[10], int (\*scores)[N])

{

int i, j;

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

{

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

if (\*(\*(scores+i)+j) < 60

)

(\*(bad+i))++;

printf("对于课程%s, 不及格的人数为：%d\n", \*(p+i), bad[i]);

}

printf("\n");

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

{

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

if (\*(\*(scores+i)+j) >= 90)

(\*(higher+i))++;

printf("对于课程%s, 成绩在90分及以上的人数为：%d\n", \*(p+i), higher[i]);

}

printf("\n");

}

1. 运行截图如图6-10所示：

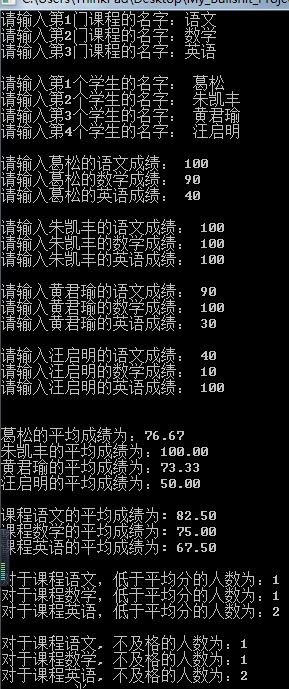


图6-10 编程设计题3 程序运行截图

6.2.5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

1. 思路：

和书上的例题相似，由于各个部分长度固定，可以通过从后向前不断相加的方法，即每次将字符转化为数字进行计算，计算完完成后再将数字转化为字符，跳过小数点，最后输出结果

1. 程序清单：

#include <stdio.h>

#define N 20

#define M 10

int main(void)

{

char a[N+M+2];

char b[N+M+2];

char result[N+M+3];

int carry, i, A, B;

printf("请输入第一个数字：\n");

scanf("%s", a);

printf("请输入第二个数字：\n");

scanf("%s", b);

result[N+M+2] = '\0';

result[N+1] = '.';

for (i = N+M, carry = 0; i >= 0; i--)

{

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

continue;

A = a[i] - '0';

B = b[i] - '0';

result[i+1] = (A + B + carry)%10 + '0';

carry = (A + B + carry) >= 10 ? 1 : 0;

}

result[0] = carry + '0';

if (result[0] != '0')

printf("结果为：\n%s", result);

else

printf("结果为：\n%s", result+1);

return 0;

}

1. 测试：

用例1：输入：12345678912345678912.1234567891

98765432109876543210.0123456789

理论输出：111111111022222222122.1358024680

用例2：输入：10000000000000000000.9999999999

10000000000000000000.0000000001

理论输出：20000000000000000001.0000000000

实际运行截图如图6-11与6-12所示：

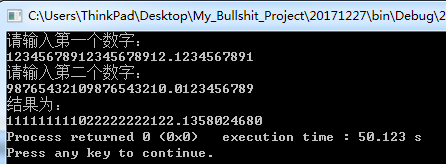


图6-11 选做题1 用例1运行截图

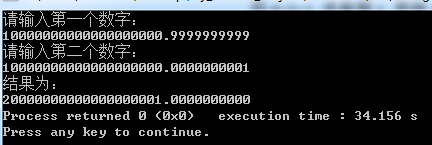


图6-12 选做题1 用例2运行截图

（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。

1. 思路：（由于所给的例子中strcmp的返回值为int 而不是char\* 故我将题目中指针所指的函数的参数表中的第一个参数改为了char\*）由于数组p含有两个元素，每个元素都是一个指向形如char \*f(char \*, const char\*)的函数的指针，故可以直接对数组中成员赋值，如p[0] = strcat;p[1] = strcpy;调用时直接使用(p[n])(char\*,const char\*);即可
2. 程序清单:

#include <stdio.h>

#include <string.h>

int main(void)

{

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

p[0] = strcat;

p[1] = strcpy;

char string1[20] = "ABCDEFG";

char string2[20] = "abcdefg";

char string3[40];

printf("%s\n", (p[0])(string1, string2));

(p[1])(string3, string2);

printf("%s", string3);

return 0;

}

1. 测试：

char string1[20] = "ABCDEFG";

char string2[20] = "abcdefg";

char string3[40];

操作为将string2通过strcat函数接在string1之后，调用strcmp函数将string2的内容复制到string3中

运行截图如图6-13所示：

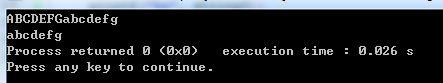


图6-13 选做题2运行截图

**(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。

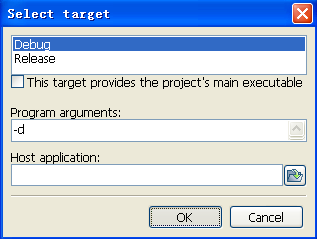
****

图6.？ 输入main函数的参数

## 6.3 实验小结

体会：这次实验感受最深的就是高精度计算那道题，通过将每一项先从字符转化为数字，计算完毕之后再转化为字符，储存在一个数组中，最后将其输出，十分巧妙。

# 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 | ‘B’ | ‘B’ |
| 3 | \*p++->t,\*p->t | ‘x’ | ‘x’ |
| 4 | \*(++p)->t | ‘x’ | ‘x’ |
| 5 | \*++p->t | ‘V’ | ‘V’ |
| 6 | ++\*p->t | ‘V’ | ‘V’ |

结果如表格

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 \*/

}

1. 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. 原因：将一个指针传入函数时，在函数中只能改变指针所指向的内容而不能改变指针本身，所以要通过传入指针的地址来改变指针本身。

运行截图如图7-1所示。

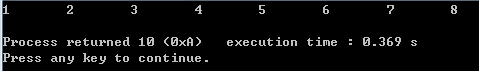


图7-1 程序改错题运行截图1

1. 程序清单：

#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 \*/

loc\_head->next = NULL;

tail=loc\_head;

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

tail->next = loc\_head;

loc\_head = tail;

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

}

}

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

}

运行截图如图7-2所示。

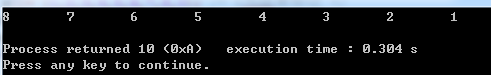


图7-2 程序改错题运行截图2

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. 程序流程图如图7-3所示。



图7-3 编程设计题1流程图

1. 程序清单：

#include <stdio.h>

struct bits{

unsigned int bit0 : 1;

unsigned int bit1 : 1;

unsigned int bit2 : 1;

unsigned int bit3 : 1;

unsigned int bit4 : 1;

unsigned int bit5 : 1;

unsigned int bit6 : 1;

unsigned int bit7 : 1;

}x = {1,0,1,0,0,1,0,0};

void f0(struct bits b){printf("the function 0 %d is called!\n",b.bit0);}

void f1(struct bits b){printf("the function 1 %d is called!\n",b.bit1);}

void f2(struct bits b){printf("the function 2 %d is called!\n",b.bit2);}

void f3(struct bits b){printf("the function 3 %d is called!\n",b.bit3);}

void f4(struct bits b){printf("the function 4 %d is called!\n",b.bit4);}

void f5(struct bits b){printf("the function 5 %d is called!\n",b.bit5);}

void f6(struct bits b){printf("the function 6 %d is called!\n",b.bit6);}

void f7(struct bits b){printf("the function 7 %d is called!\n",b.bit7);}

int main(void)

{

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

p[0] = f0;

p[1] = f1;

p[2] = f2;

p[3] = f3;

p[4] = f4;

p[5] = f5;

p[6] = f6;

p[7] = f7;

if (x.bit0)(p[0])(x);

if (x.bit1)(p[1])(x);

if (x.bit2)(p[2])(x);

if (x.bit3)(p[3])(x);

if (x.bit4)(p[4])(x);

if (x.bit5)(p[5])(x);

if (x.bit6)(p[6])(x);

if (x.bit7)(p[7])(x);

return 0;

}

1. 运行截图如图7-4所示。

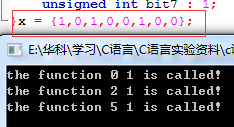


图7-4 编程设计题1 运行截图

（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. 思路：

每次打印一个菜单，根据用户所输入的数据来进行相应的操作，分别编写(1)~(5)所要求的函数

1. 程序清单:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct scores

{

char card\_num[20];

char name[10];

int math;

int english;

int physics;

int c\_lan;

struct scores \*next;

};//定义全局结构

void newstudent(struct scores \*\*head);//添加学生

void print\_info(struct scores \*head);//打印信息

void print\_average(struct scores \*head);//打印平均分

void change\_info(struct scores \*head);//改变分数

struct scores \* find\_student(struct scores \*head, char\* target);//寻找学生

void print\_all(struct scores \*head);//打印所有信息

int main(void)

{

int choice;//用于记录用户的选择

int flag;//用于判断是否添加了学生

struct scores \*head=NULL, \*\*tail = &head;

do

{

flag = 0;

printf("Please choose:\n");

printf("1. add student\n");

printf("2. print information\n");

printf("3. print average scores\n");

printf("4. change information\n");

printf("5. print information and average scores\n");

printf("6. quit\n");

scanf("%d", &choice);

switch(choice)//根据用户的选择调用相应的函数

{

case 1: newstudent(tail);flag = 1;break;

case 2: print\_info(head);break;

case 3: print\_average(head);break;

case 4: change\_info(head);break;

case 5: print\_all(head);break;

default: return 0;

}

if (head == NULL)

head = \*tail;

if (flag)

tail = &(\*tail)->next;

}

while(choice != 6);

return 0;

}

void newstudent(struct scores \*\*head)

{

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

printf("Please enter the card number of student:\n");

scanf("%s", p->card\_num);

printf("Please enter the student's name:\n");

scanf("%s", p->name);

printf("Please enter the score of math:\n");

scanf("%d", &p->math);

printf("Please enter the score of English:\n");

scanf("%d", &p->english);

printf("Please enter the score of physics:\n");

scanf("%d", &p->physics);

printf("Please enter the score of c\_language:\n");

scanf("%d", &p->c\_lan);

p->next = NULL;

\*head = p;

}

void print\_info(struct scores \*head)

{

while(head)

{

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n\n", head->c\_lan);

head = head->next;

}

}

void print\_average(struct scores \*head)

{

int sum = 0;

while (head)

{

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("The average score of %s is %.2f\n", head->name, sum/4.0);

head = head->next;

}

}

void change\_info(struct scores \*head)

{

char target[10];

int choice;

int data;

printf("Please enter the name of the wanted student:\n");

scanf("%s", target);

struct scores \*temp = find\_student(head, target);

if (temp)

{

printf("Please choose the subject you want to change:\n");

printf("1. Math\n");

printf("2. Physics\n");

printf("3. English\n");

printf("4. c language\n");

scanf("%d", &choice);

switch(choice)

{

case 1: printf("Please enter the new score of math:\n");

scanf("%d", &data);

temp->math = data;

break;

case 2: printf("Please enter the new score of physics:\n");

scanf("%d", &data);

temp->physics = data;

break;

case 3: printf("Please enter the new score of English:\n");

scanf("%d", &data);

temp->english = data;

break;

case 4: printf("Please enter the new score of c language:\n");

scanf("%d", &data);

temp->c\_lan = data;

break;

}

}

else

printf("Student not found\n");

}

struct scores \* find\_student(struct scores \*head, char\* target)

{

while(head && strcmp(head->name, target))

head = head->next;

return head;

}

void print\_all(struct scores \*head)

{

int sum = 0;

while(head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n", head->c\_lan);

printf("sum: %d\n", sum);

printf("The average score of %s is %.2f\n\n", head->name, sum/4.0);

head = head->next;

}

}

1. 运行截图如图7-5至7-8所示。

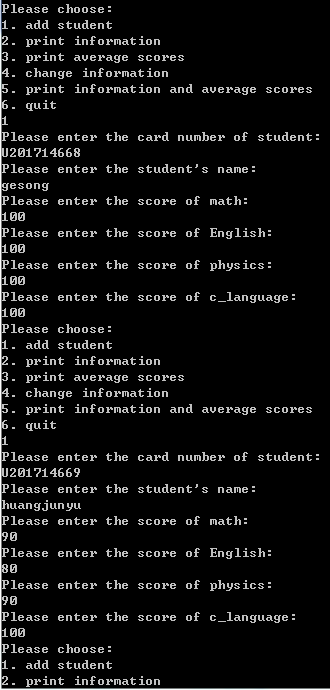


图7-5 编程设计题运行截图1

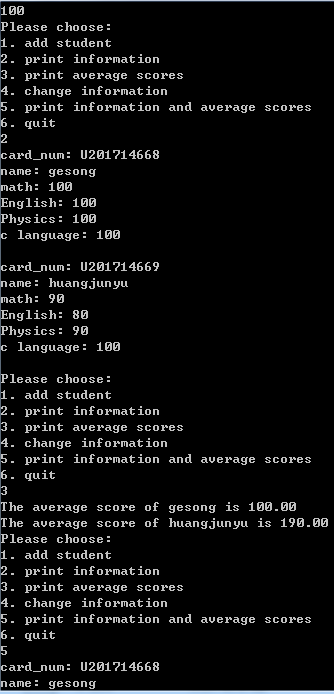


图7-6 编程设计题运行截图2

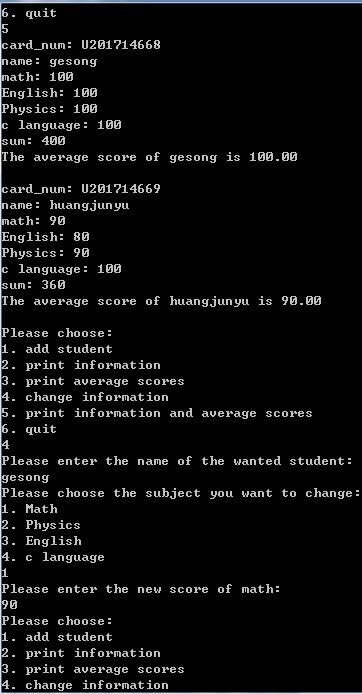


图7-7 编程设计题运行截图3

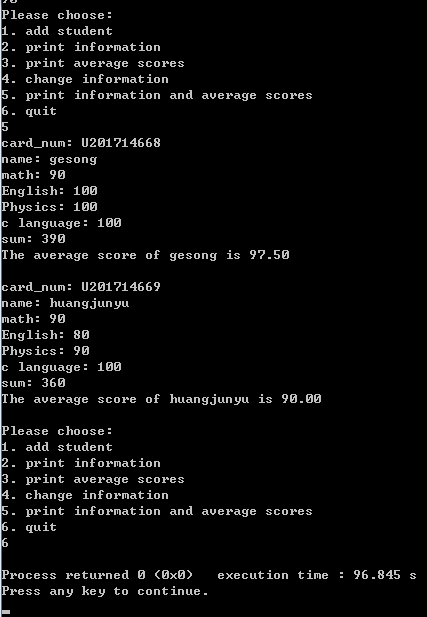


图7-6 编程设计题运行截图4

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. 源程序清单：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct scores

{

char card\_num[20];

char name[20];

int math;

int english;

int physics;

int c\_lan;

struct scores \*next;

};

void newstudent(struct scores \*\*head);

void print\_info(struct scores \*head);

void print\_average(struct scores \*head);

void change\_info(struct scores \*head);

struct scores \* find\_student(struct scores \*head, char\* target);

void print\_all(struct scores \*head);

void sortdata(struct scores \*head);//添加排序函数

int main(void)

{

int choice;

int flag;

struct scores \*head=NULL, \*\*tail = &head;

do

{

flag = 0;

printf("Please choose:\n");

printf("1. add student\n");

printf("2. print information\n");

printf("3. print average scores\n");

printf("4. change information\n");

printf("5. print information and average scores\n");

printf("6. sort data\n");

printf("7. quit\n");

scanf("%d", &choice);

switch(choice)

{

case 1: newstudent(tail);flag = 1;break;

case 2: print\_info(head);break;

case 3: print\_average(head);break;

case 4: change\_info(head);break;

case 5: print\_all(head);break;

case 6: sortdata(head);break;

default: return 0;

}

if (head == NULL)

head = \*tail;

if (flag)

tail = &(\*tail)->next;

}

while(choice != 7);

return 0;

}

void newstudent(struct scores \*\*head)

{

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

printf("Please enter the card number of student:\n");

scanf("%s", p->card\_num);

printf("Please enter the student's name:\n");

scanf("%s", p->name);

printf("Please enter the score of math:\n");

scanf("%d", &p->math);

printf("Please enter the score of English:\n");

scanf("%d", &p->english);

printf("Please enter the score of physics:\n");

scanf("%d", &p->physics);

printf("Please enter the score of c\_language:\n");

scanf("%d", &p->c\_lan);

p->next = NULL;

\*head = p;

}

void print\_info(struct scores \*head)

{

while(head)

{

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n\n", head->c\_lan);

head = head->next;

}

}

void print\_average(struct scores \*head)

{

int sum;

while (head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("The average score of %s is %.2f\n", head->name, sum/4.0);

head = head->next;

}

printf("\n");

}

void change\_info(struct scores \*head)

{

char target[10];

int choice;

int data;

printf("Please enter the name of the wanted student:\n");

scanf("%s", target);

struct scores \*temp = find\_student(head, target);

if (temp)

{

printf("Please choose the subject you want to change:\n");

printf("1. Math\n");

printf("2. Physics\n");

printf("3. English\n");

printf("4. c language\n");

scanf("%d", &choice);

switch(choice)

{

case 1: printf("Please enter the new score of math:\n");

scanf("%d", &data);

temp->math = data;

break;

case 2: printf("Please enter the new score of physics:\n");

scanf("%d", &data);

temp->physics = data;

break;

case 3: printf("Please enter the new score of English:\n");

scanf("%d", &data);

temp->english = data;

break;

case 4: printf("Please enter the new score of c language:\n");

scanf("%d", &data);

temp->c\_lan = data;

break;

}

}

else

printf("Student not found\n");

}

struct scores \* find\_student(struct scores \*head, char\* target)

{

while(head && strcmp(head->name, target))

head = head->next;

return head;

}

void print\_all(struct scores \*head)

{

int sum = 0;

while(head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n", head->c\_lan);

printf("sum: %d\n", sum);

printf("The average score of %s is %.2f\n\n", head->name, sum/4.0);

head = head->next;

}

}

void sortdata(struct scores \*head)

{

int flag = 1;

struct scores \*p = head;

struct scores \*ptr, \*q, temp;

int sum1, sum2;

while (flag)

{

q = p;

flag = 0;

while (q->next != NULL)

{

sum1 = sum2 = 0;

sum1 += q->c\_lan;

sum1 += q->english;

sum1 += q->math;

sum1 += q->physics;

sum2 += q->next->c\_lan;

sum2 += q->next->english;

sum2 += q->next->math;

sum2 += q->next->physics;

if (sum1 > sum2)

{

flag = 1;

temp = \*q->next;

ptr = q->next->next;

\*q->next = \*q;

q->next->next = ptr;

ptr = q->next;

\*q = temp;

q->next = ptr;

}

q = q->next;

}

}

}

1. 运行截图如图7-7所示。（此题是在编程设计题的基础上修改，故只截取出重要部分）

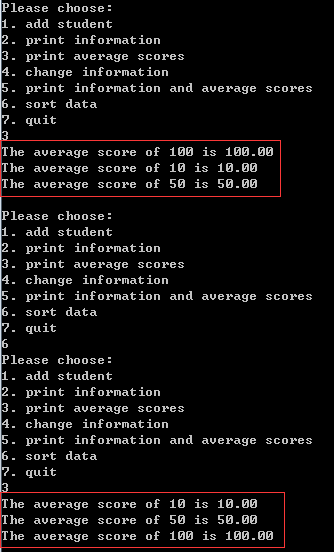


图7-7 选做题1 运行截图

（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. 程序清单：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct scores

{

char card\_num[20];

char name[20];

int math;

int english;

int physics;

int c\_lan;

struct scores \*next;

};

void newstudent(struct scores \*\*head);

void print\_info(struct scores \*head);

void print\_average(struct scores \*head);

void change\_info(struct scores \*head);

struct scores \* find\_student(struct scores \*head, char\* target);

void print\_all(struct scores \*head);

void sortdata(struct scores \*\*head);

int main(void)

{

int choice;

int flag;

struct scores \*head=NULL, \*\*tail = &head;

do

{

flag = 0;

printf("Please choose:\n");

printf("1. add student\n");

printf("2. print information\n");

printf("3. print average scores\n");

printf("4. change information\n");

printf("5. print information and average scores\n");

printf("6. sort data\n");

printf("7. quit\n");

scanf("%d", &choice);

switch(choice)

{

case 1:

newstudent(tail);

flag = 1;

break;

case 2:

print\_info(head);

break;

case 3:

print\_average(head);

break;

case 4:

change\_info(head);

break;

case 5:

print\_all(head);

break;

case 6:

sortdata(&head);

break;

default:

return 0;

}

if (head == NULL)

head = \*tail;

if (flag)

tail = &(\*tail)->next;

}

while(choice != 7);

return 0;

}

void newstudent(struct scores \*\*head)

{

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

printf("Please enter the card number of student:\n");

scanf("%s", p->card\_num);

printf("Please enter the student's name:\n");

scanf("%s", p->name);

printf("Please enter the score of math:\n");

scanf("%d", &p->math);

printf("Please enter the score of English:\n");

scanf("%d", &p->english);

printf("Please enter the score of physics:\n");

scanf("%d", &p->physics);

printf("Please enter the score of c\_language:\n");

scanf("%d", &p->c\_lan);

p->next = NULL;

\*head = p;

}

void print\_info(struct scores \*head)

{

while(head)

{

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n\n", head->c\_lan);

head = head->next;

}

}

void print\_average(struct scores \*head)

{

int sum;

while (head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("The average score of %s is %.2f\n", head->name, sum/4.0);

head = head->next;

}

printf("\n");

}

void change\_info(struct scores \*head)

{

char target[10];

int choice;

int data;

printf("Please enter the name of the wanted student:\n");

scanf("%s", target);

struct scores \*temp = find\_student(head, target);

if (temp)

{

printf("Please choose the subject you want to change:\n");

printf("1. Math\n");

printf("2. Physics\n");

printf("3. English\n");

printf("4. c language\n");

scanf("%d", &choice);

switch(choice)

{

case 1:

printf("Please enter the new score of math:\n");

scanf("%d", &data);

temp->math = data;

break;

case 2:

printf("Please enter the new score of physics:\n");

scanf("%d", &data);

temp->physics = data;

break;

case 3:

printf("Please enter the new score of English:\n");

scanf("%d", &data);

temp->english = data;

break;

case 4:

printf("Please enter the new score of c language:\n");

scanf("%d", &data);

temp->c\_lan = data;

break;

}

}

else

printf("Student not found\n");

}

struct scores \* find\_student(struct scores \*head, char\* target)

{

while(head && strcmp(head->name, target))

head = head->next;

return head;

}

void print\_all(struct scores \*head)

{

int sum = 0;

while(head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n", head->c\_lan);

printf("sum: %d\n", sum);

printf("The average score of %s is %.2f\n\n", head->name, sum/4.0);

head = head->next;

}

}

void sortdata(struct scores \*\*head)

{

struct scores \*prior1, \*prior2, \*p1, \*p2, \*t;

int sum1, sum2;

p1 = \*head;

for (p2 = p1->next, prior2 = p1; p2 != NULL; prior2 = p2, p2 = p2->next) //这一个循环用于先处理头结点

{

sum1 = sum2 = 0;

sum1 += p1->c\_lan;

sum1 += p1->english;

sum1 += p1->math;

sum1 += p1->physics;

sum2 += p2->c\_lan;

sum2 += p2->english;

sum2 += p2->math;

sum2 += p2->physics;

if (sum1 > sum2)

{

prior2->next = p1;

t = p1->next;

p1->next = p2->next;

(\*head) = p2;

p2->next = t;

p1 = p2;

}

}

for (prior1 = \*head, p1 = prior1->next; p1->next != NULL; prior1 = p1, p1 = p1->next)//这个循环用于处理剩余所有部分

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

{

sum1 = sum2 = 0;

sum1 += p1->c\_lan;

sum1 += p1->english;

sum1 += p1->math;

sum1 += p1->physics;

sum2 += p2->c\_lan;

sum2 += p2->english;

sum2 += p2->math;

sum2 += p2->physics;

if (sum1 > sum2)

{

t = p2->next;

prior1->next = p2;

prior2->next = p1;

p2->next = p1->next;

p1->next = t;

p1 = p2;

}

}

}

1. 运行截图如图7-8所示。（同上，只截取重要部分）

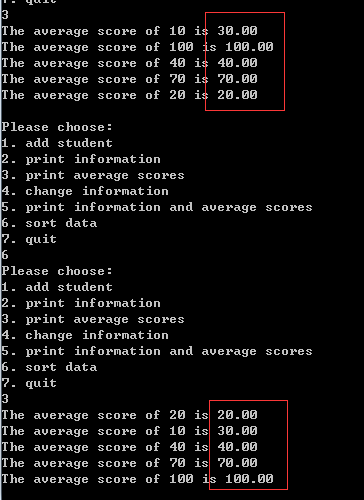


图7-8选做题2运行截图

1. 采用双向链表重做编程设计题第（2）题。
2. 程序清单：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct scores

{

char card\_num[20];

char name[10];

int math;

int english;

int physics;

int c\_lan;

struct scores \*next;

struct scores \*prior;

};

void newstudent(struct scores \*\*head);

void print\_info(struct scores \*head);

void print\_average(struct scores \*head);

void change\_info(struct scores \*head);

struct scores \* find\_student(struct scores \*head, char\* target);

void print\_all(struct scores \*head);

int main(void)

{

int choice;

int flag;

struct scores \*head=NULL, \*\*tail = &head;

do

{

flag = 0;

printf("Please choose:\n");

printf("1. add student\n");

printf("2. print information\n");

printf("3. print average scores\n");

printf("4. change information\n");

printf("5. print information and average scores\n");

printf("6. quit\n");

scanf("%d", &choice);

switch(choice)

{

case 1: newstudent(tail);flag = 1;break;

case 2: print\_info(head);break;

case 3: print\_average(head);break;

case 4: change\_info(head);break;

case 5: print\_all(head);break;

default: return 0;

}

if (head->next == NULL)

tail = &head;

if (head->next != NULL && flag)

tail = &(head->next);

}

while(choice != 6);

return 0;

}

void newstudent(struct scores \*\*head)

{

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

printf("Please enter the card number of student:\n");

scanf("%s", p->card\_num);

printf("Please enter the student's name:\n");

scanf("%s", p->name);

printf("Please enter the score of math:\n");

scanf("%d", &p->math);

printf("Please enter the score of English:\n");

scanf("%d", &p->english);

printf("Please enter the score of physics:\n");

scanf("%d", &p->physics);

printf("Please enter the score of c\_language:\n");

scanf("%d", &p->c\_lan);

if (\*head == NULL)

{

\*head = p;

p->next = NULL;

p->prior = NULL;

}

else

{

(\*head)->next = p;

p->next = NULL;

p->prior = (\*head);

}

}

void print\_info(struct scores \*head)

{

while(head)

{

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n\n", head->c\_lan);

head = head->next;

}

}

void print\_average(struct scores \*head)

{

int sum = 0;

while (head)

{

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("The average score of %s is %.2f\n", head->name, sum/4.0);

head = head->next;

}

}

void change\_info(struct scores \*head)

{

char target[10];

int choice;

int data;

printf("Please enter the name of the wanted student:\n");

scanf("%s", target);

struct scores \*temp = find\_student(head, target);

if (temp)

{

printf("Please choose the subject you want to change:\n");

printf("1. Math\n");

printf("2. Physics\n");

printf("3. English\n");

printf("4. c language\n");

scanf("%d", &choice);

switch(choice)

{

case 1: printf("Please enter the new score of math:\n");

scanf("%d", &data);

temp->math = data;

break;

case 2: printf("Please enter the new score of physics:\n");

scanf("%d", &data);

temp->physics = data;

break;

case 3: printf("Please enter the new score of English:\n");

scanf("%d", &data);

temp->english = data;

break;

case 4: printf("Please enter the new score of c language:\n");

scanf("%d", &data);

temp->c\_lan = data;

break;

}

}

else

printf("Student not found\n");

}

struct scores \* find\_student(struct scores \*head, char\* target)

{

while(head && strcmp(head->name, target))

head = head->next;

return head;

}

void print\_all(struct scores \*head)

{

int sum = 0;

while(head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n", head->c\_lan);

printf("sum: %d\n", sum);

printf("The average score of %s is %.2f\n\n", head->name, sum/4.0);

head = head->next;

}

}

1. 运行截图如图7-9所示。

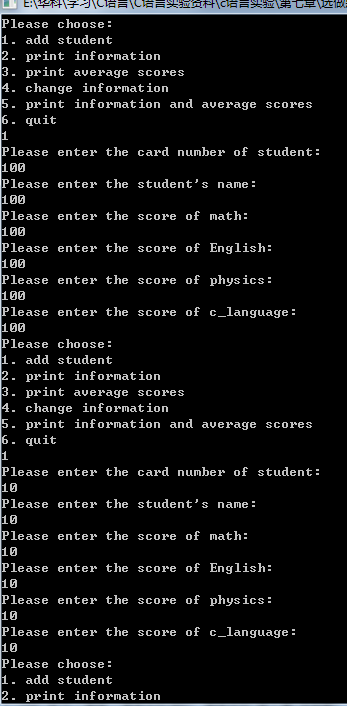


图7-9 选做题3 运行截图

## 7.3 实验小结

体会：这次实验让我深刻体会到了自己的不足。花费了大量的时间在最后几个以链表为主题的选做题中，最后还是通过模仿书上的例题才最终解决。还有一点就是运算符的优先级很重要，没有弄清楚运算符的优先级就开始计算可能会导致非常严重的错误。所以要么就在不清楚的时候加上必要的括号，要么就去查表，并在旁边写上注释，以便后期修改。

# 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实验内容

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。
2. 将两处sizeof(short)均改为sizeof(char)结果有什么不同，为什么？
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：**

1. 运行截图如图8-1所示。

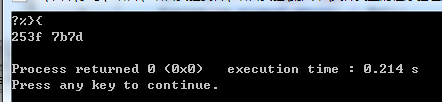


图8-1 验证题运行截图

1. 输入结果如图8-2所示。

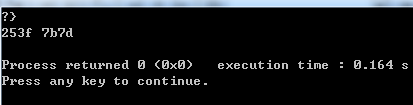


图8-2 验证题修改后运行截图1

原因：

Short：以short指向a和b，会在文件中依次写入a，b的低高字节，即‘？’，‘%’‘}’‘}’

Char：以char指向a和b，会在文件中依次写入a的低字节，b的低字节，即‘？’‘}’

1. 输出结果如图8-3所示。

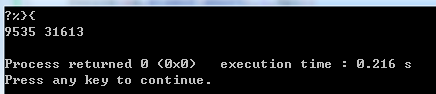


图8-3 验证题修改后运行截图2

16进制变为了10进制

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;

}

（2）用输入输出重定向freopen改写main函数。

1. 错误分析：

由于 != 运算符优先级高于 = 故ch=fgetc(fp)!=EOF就相当于ch=（fgetc(fp)!=EOF）即ch只有两个值，0或1，所以应该加一个括号,即(ch=fgetc(fp))!=EOF

1. 源程序清单

#include<stdio.h>

#include<stdlib.h>

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

{

char ch;

FILE \*fp, \*fp2;

if(argc!=2)

{

printf("Arguments error!\n");

exit(-1);

}

if((fp=freopen(argv[1],"r",stdin))==NULL)//此处改为使用freopen即可

{

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

exit(-1);

}

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

putchar(ch);

fclose(fp); /\* ¹Ø±Õfilename \*/

return 0;

}

运行截图如图8-4所示。(由于Windows命令行自带type命令，故将命令改为了type2)



图8-4源程序修改题运行截图

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. 流程图如图8-5所示。



图8-5 编程设计题流程图

1. 程序清单：

#include <stdio.h>

int main(void)

{

FILE \*fp = fopen("b.txt", "a");//打开文件

char string[100];

char \*p = string;

gets(string);//读入数据

while('\n' != \*p)

{

while(' ' == \*p)//读取到第一个不是空格的符号

p++;

if (\*p >= 'a' && \*p <= 'z')//判断p的类型，作出修改

\*p += 'A' - 'a';

while(' ' != \*p && '\n' != \*p)//继续读取

p++;

}

fprintf(fp, "%s", string);//将修改好的数据写入文件

fclose(fp);

return 0;

}

1. 运行截图如图8-6所示。

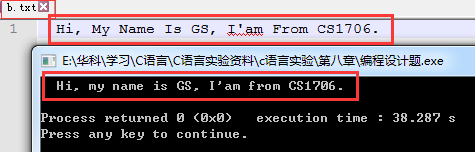


图8-6 编程设计题运行就截图

## 8.3 实验小结

体会：必须承认还是对文件的操作不是特别熟练，在实验的过程中由于不知道如何使用一些特定的文件函数，在书本上和网上都查阅了一些资料，最终完成了实验，同时也学到了很多。