

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级：**

**学 号：**

**姓 名：**

**指导教师： 祝 建 华**

**报告日期：**

**计算机科学与技术学院**

**目□□录**（黑体小2号加粗居中这里□代表空格）

**[1□□□表达式和标准输入输出实验 1](#_Toc404837920)**

[1.1□□实验目的 1](#_Toc404837921)

[1.2□□实验内容 1](#_Toc404837922)

[1.3□□实验小结 1](#_Toc404837923)

**[2□□□流程控制实验 2](#_Toc404837924)**

[2.1□□实验目的 1](#_Toc404837921)

[2.2□□实验内容 1](#_Toc404837922)

[2.3□□实验小结 1](#_Toc404837923)

**[3□□□函数与程序结构实验 3](#_Toc404837929)**

[3.1□□实验目的 1](#_Toc404837921)

[3.2□□实验内容 1](#_Toc404837922)

[3.3□□实验小结 1](#_Toc404837923)

**[4□□□编译预处理实验 4](#_Toc404837934)**

[4.1□□实验目的 1](#_Toc404837921)

[4.2□□实验内容 1](#_Toc404837922)

[4.3□□实验小结 1](#_Toc404837923)

**[5□□□数组实验 5](#_Toc404837938)**

[5.1□□实验目的 1](#_Toc404837921)

[5.2□□实验内容 1](#_Toc404837922)

[5.3□□实验小结 1](#_Toc404837923)

**[6□□□指针实验 6](#_Toc404837943)**

[6.1□□实验目的 1](#_Toc404837921)

[6.2□□实验内容 1](#_Toc404837922)

[6.3□□实验小结 1](#_Toc404837923)

**[7□□□结构与联合实验 7](#_Toc404837948)**

[7.1□□实验目的 1](#_Toc404837921)

[7.2□□实验内容 1](#_Toc404837922)

[7.3□□实验小结 1](#_Toc404837923)

**[8□□□文件实验 8](#_Toc404837953)**

[8.1□□实验目的 1](#_Toc404837921)

[8.2□□实验内容 1](#_Toc404837922)

[8.3□□实验小结 1](#_Toc404837923)

**[参考文献 9](#_Toc404837957)**

（目录部分的章为宋体小4号加粗，其余宋体小4号，字母、阿拉伯数字为Time New Roman小4号。此处目论仅说明格式规范，最后全部报告完成后，可先自动生成目录，再按规范字体格式整理即可）

**1.表达式和标准输入与输出实验**

* 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行函数返回值类型应与函数之间应该有空格，正确形式为：

void main( void )

3）第6行没有声明下面用到的变量newint，正确形式为：

short p, k , newint;

4）第9行格式化输出函数里面表示输出内容不能用汉语双引号，正确形 式为：

printf("Input Fahrenheit:" ) ;

5）第10行格式化输入函数里面运用了汉语双引号并且未使用指针将数值 赋给整形变量，正确形式为：

scanf("%d", &f ) ;

6）第11行未添加强制类型转换，正确形式为：

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

7)第12行double类型的变量输出不应用浮点类型输出，正确形式为：

printf( " \n %d (F) = %.2lf (C)\n\n ", f, c ) ;

8）第15行变量r属于double类型，不应用%.f，正确形式为：

scanf("%lf", &r);

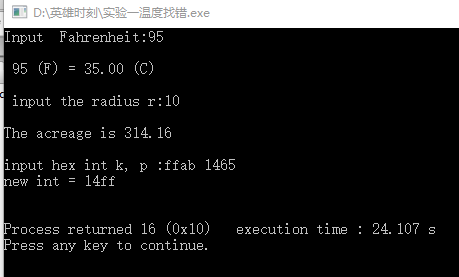
9）第17行double类型的变量输出不应用浮点类型输出并且不需要指针， 正确形式为

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

10）第21行不应该 先与运算向左移动8位，正确形式为：

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

1. 错误修改后运行结果：



**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：**

将b与a的差赋给b；a（实则为原来b的值）=a+b；b（实则为原来a的值）=a-b；替换后的程序如下所示：

#include<stdio.h>

void main(void)

{

int a,b,t;

printf("Input two integers:");

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

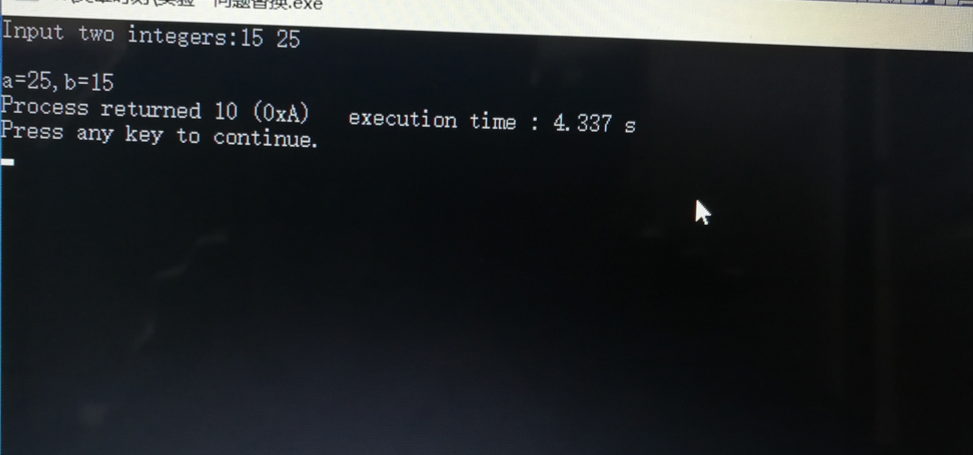
a=b-a;

b=b-a;

a=a+b;

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

}



**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1.1所示。

若输入的字母的ASCⅡ值在65~90之间，为大写字母，就将其的ASCⅡ值加上32后以字符的形式输出：若所输入字母的ASCⅡ值在97~122之间，将原字母的ASCⅡ值不变以字符型形式输出。

2) 源程序清单

#include<stdio.h>

int main(void)

{

int a;

a=getchar();

if(a>=65&&a<=90)

putchar(a+32);

else

{

putchar(a);

printf("\n");

}

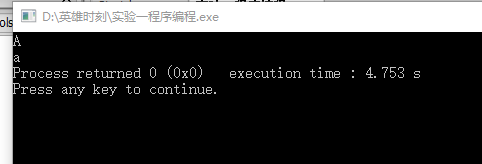
return 0;

}

3）测试

（a） 测试数据:

（b） 对应测试数据的运行结果截图



**（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：

1.输入x，m，n，为了方便分析测试结果，x的输入采用16进制

2.如果0 ≤ｍ≤ 15, 1 ≤ ｎ≤ 16-ｍ，转2.1，否则转3.

2.1 首先x>>m，将要处理的n位移动到最右；

2.2 再将上一步的结果左移n-1位，即：x<<(n-1);

2.3 用16进制输出结果并转4.

1. 显示输入错误信息；
2. 结束
3. 程序清单

#include<stdio.h>

void main( void )

{

unsigned short int x,m,n,q;

printf("输入x（16进制）、m（0~15）和n（1~16-m）：\n");

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

if (m>=0&&m<=15&&n>=1&&n<=(16-m))/\*判断m、n的值是否在合理范围内\*/

printf("ans=%hx\n",(x>>m)<<(16-n));

else

printf("输入错误!\n");

return 0;

}

3）测试

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 | 0100 0110 1000 0000（4680） | 7 | 4 | 计算结果1101 0000 0000 0000 即D000 | D000 |
| 用例2 | 1101 0101 1000 0011（D583） | 16 | 1 | 输入错误（m值超范围） |  |
| 用例3 | 1101 0101 1000 0011（D583） | 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1-2所示。

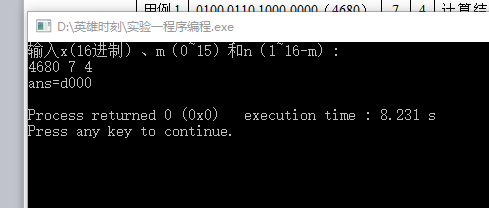


图1-2 编程题3的测试用例一的运行结果

对应测试测试用例2的运行结果如图1-3所示。

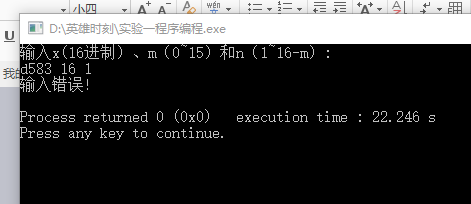


图1-3 编程题3的测试用例二的运行结果

对应测试测试用例3的运行结果如图1-4所示。

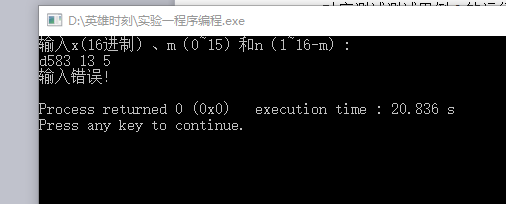


图1-4 编程题3的测试用例三的运行结果

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）解题思路：

1. 输入一个无符号长整形常数

2. 规定尺子mask为255

3. 将常数与尺子进行位运算后输出第一个数，之后向右移动8位后再次 与尺子进行位运算，连续进行同样的步骤三次后输出结果。

4. 结束

2）程序清单

#include<stdio.h>

int main(void)

{

unsigned long int b,c;

c=255;

scanf("%lu",&b);

printf("%lu.",b&c);

printf("%lu.",(b>>8)&c);

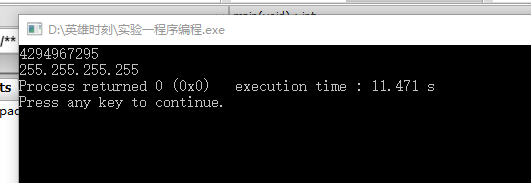
printf("%lu.",(b>>16)&c);

printf("%lu",(b>>24)&c);

return 0;

}

1. 测试



## 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。

答：实验中，在给源代码纠错的过程中，纠错后，运行程序后，发现结果与样例不同。无论怎样的输入，输出结果总是与最终答案不同。就一步步进行了debug，发现了第21行位运算不正确，需要将向左移动八位改成向右移动八位，于是进行了改正，程序正常运行。因此，我觉得无论是给代码挑错误，还是打代码，都需要画一下流程图，分析程序是怎样运行的。

# 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。

2.1、实验内容

**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：**

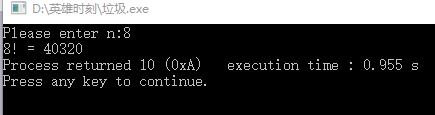
1. 错误修改：
2. 第6行缺少指针，不能将输入的值赋给变量n，正确形式为：

scanf("%d",&n);

2)第7行for循环里面的条件之间不应该用逗号分开，正确形式为：

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

(2）错误修改后运行结果：



**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. while语句的程序如下：

#include <stdio.h>

void main(void)

{

int i,n,s=1;

printf("Please enter n:");

scanf("%d",&n);

i=1;

while(i<=n)

{

s=s\*i;

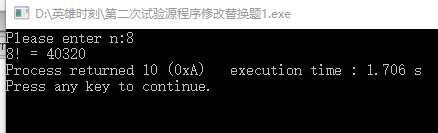
i++;

}

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

}

运行截图：



1. do while语句的程序如下：

#include <stdio.h>

void main(void)

{

int i,n,s=1;

printf("Please enter n:");

scanf("%d",&n);

i=1;

do

{

s=s\*i;

i++;

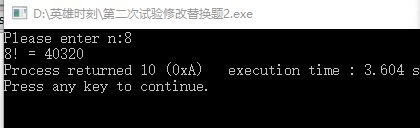
}

while(i<=n);

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

}

运行截图：



**（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。

修改后的程序如下：

#include <stdio.h>

void main(void)

{

int i,S,n=1,x=1;

printf("Please enter S:");

scanf("%d",&S);

while(x<S)

{

++n;

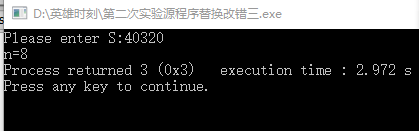
x=x\*n;

}

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

}

运行截图：



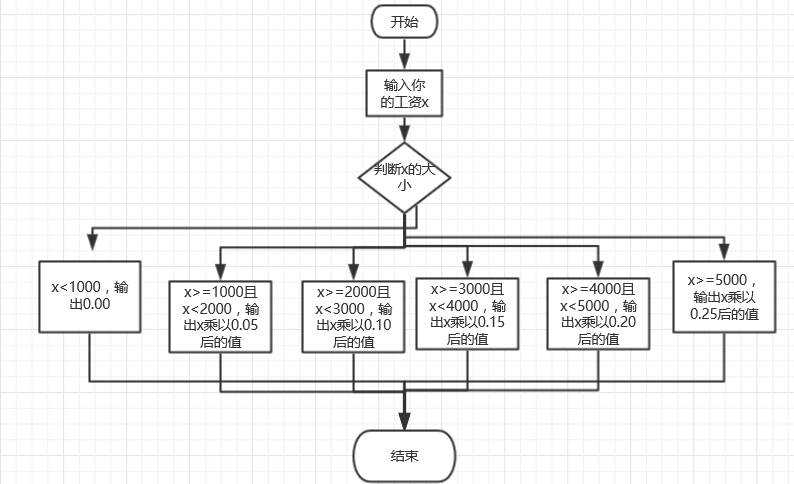
**3．编程设计题**

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. 算法流程图如下：



1. 源程序清

if语句：

#include<stdio.h>

int main(void)

{

double x;

printf("请输入您的月收入:\n");

scanf("%lf",&x);

if(x<1000)

printf("你所应缴纳的的税为:0.00元\n");

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

printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.05);

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

printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.10);

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

printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.15);

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

printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.20);

if(x>=5000)

printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.25);

return 0;

}

switch语句：#include<stdio.h>

int main(void)

{

double x;

int y;

printf("请输入您的月收入:\n");

scanf("%lf",&x);

y=x/1000;

switch(y){

case 0:printf("你所应缴纳的的税为:0.00元\n");break;

case 1:printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.05);break;

case 2:printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.10);break;

case 3:printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.15);break;

case 4:printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.20);break;

default:printf("你所应缴纳的的税为:%0.2lf元\n",x\*0.25);

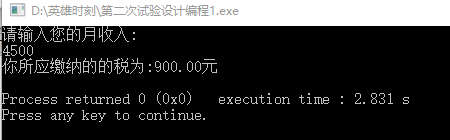
}

}

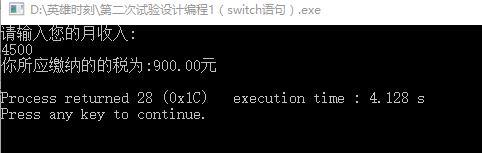
3）测试

（a） 测试数据：4500

（b） 对应测试数据的运行结果截图：

if语句： 

switch语句：



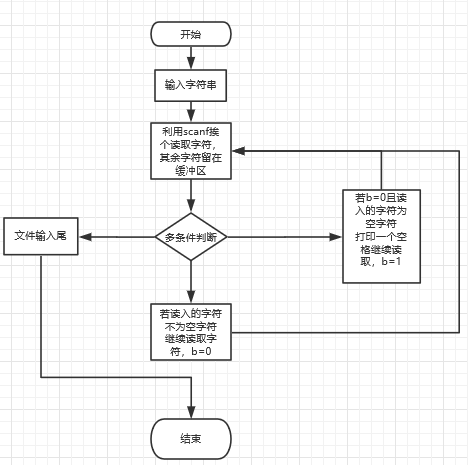
1. 编写一个程序,将输入的一行字符复制到输出，复制过程中将一个以上的空格字符用一个空格代替。

**解答：**

1) 解题思路：

1. 首先利用scanf输入字符，并且引入变量b=0
2. 开始输入字符的时候，设置条件，若输入的字符为空格且b=0，打印 一个空格后重新给b赋值，b=1；
3. 缓冲区的字符不断地进入getchar，若输入的字符为字母的话，输出 这个字母并重新给b赋值，b=0；
4. 遇文件尾输入结束
5. 输出结束
6. 程序结束

算法流程图如下：



1. 程序清单：

#include<stdio.h>

int main(void)

{

int b=0;

char a;

while(scanf("%c",&a)!=EOF)

{

if(a==' '&&b==0){

putchar(a);

b=1;}

if(a!=' '){

putchar(a);

b=0;}

}

return 0;

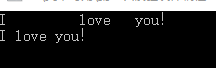
}

1. 测试
2. 测试数据： 用例1：I love you！

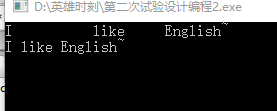
用例2：I like English~

（b）测试截图：

用例1：



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。

（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,…)

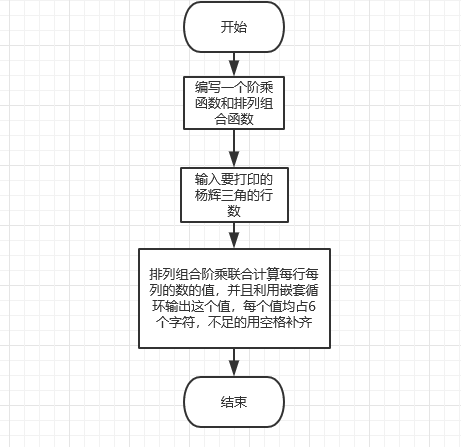
 (j=0,1,2,3,…,i)

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. 解题思路：
2. 输入要打印的杨辉三角的行数
3. 编写两个函数，分别为阶乘函数和排列组合函数
4. 运用两个嵌套循环分别输出行和列
5. 控制好每一行前面输出的空格数
6. 每个数字占用6个字符，达不到6位用空格补齐
7. 结束

算法流程图如下：



1. 程序清单：

#include<stdio.h>

unsigned long long int number(int a);//阶乘

unsigned long long int outcome(int a,int b);//排列组合

int main(void)

{

int a,b,c,d;//a是要打印几行

printf("请输入你想打印多少行杨辉三角:\n");

scanf("%d",&a);

b=0;

while(b<=(a-1))

{

c=0;

for(d=0;d<=(37-b\*3);d++)

putchar(' ');//表示前面要打印多少个空格

while(c<=b)

{

number(c);

printf("%6llu",outcome(b,c));

c++;

}//重复打印每行的数字

printf("\n");

b++;

}

return 0;

}

unsigned long long int number(int a)

{

int b,c=1;

if(a==0)

b=1;

else

for(b=1;b<=a;b++)

c=c\*b;

return c;

}

unsigned long long int outcome(int a,int b)

{

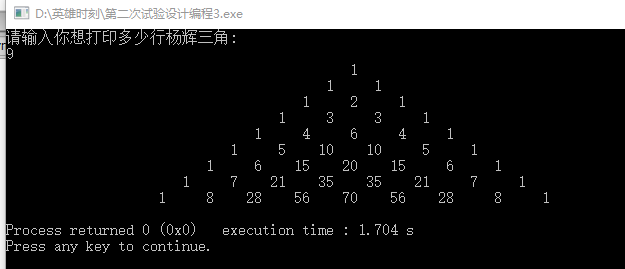
unsigned long long int c;

c=(number(a)/number(b))/number(a-b);

return c;

}

1. 测试
2. 测试数据： 9
3. 测试结果如图：



1. 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. 解题思路：
2. 输入一个整数
3. 对整数对10求模后输出
4. 整数除以10后求模输出直至整数为零（中间整数为整形除以10 后会去掉小数部分）
5. 结束
6. 程序清单：

#include<stdio.h>

int main(void)

{

int a,b;

printf("请输入你想反转的数字:\n");

scanf("%d",&a);

while(a!=0)

{

printf("%d",a%10);

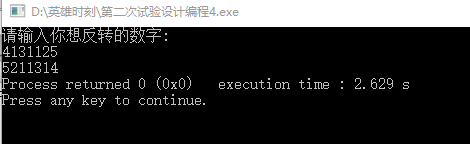
a=a/10;

}

return 0;

}

1. 测试
2. 测试数据： 4131125
3. 测试结果如图：



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. 解题思路：
2. 赋给a值；
3. 使用迭代轮回赋给a值；
4. 直至这次a的值与上次a的值的差在误差范围内
5. 结束
6. 程序清单：

#include<stdio.h>

int main(void)

{

double a,b;

a=1;

b=0;

while((a-b>0.000001)||(a-b<-0.000001))

{

b=a-(3\*a\*a\*a-4\*a\*a-5\*a+13)/(9\*a\*a-8\*a-5);

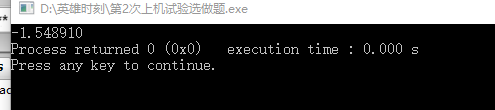
a=b-(3\*b\*b\*b-4\*b\*b-5\*b+13)/(9\*b\*b-8\*b-5);

}

printf("%lf",a);

return 0;

1. 测试
2. 测试数据： 1
3. 测试结果如图



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。

## 3.3 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。

小结：在试验中switch语句中，case与后面的字符应该有格，但是却忘记了，导致程序崩溃，查找课本后修改了问题，基础的知识要记牢；在杨辉三角阶乘函数的设计中，由于忘记0的阶乘仍然是1，导致打不出来杨辉三角，后来一步步debug发现了问题，进行了改正，在编程之前，最好要事先写一下伪代码。防止程序出现难以觉察的错误。

# 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

**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）错误修改：

1. 第二行没有对新函数进行定义，正确形式为：

long sum\_fac(int n);

1. 第十二行没有赋给fac应该具有的值，正确形式为：

long fac=1;

1. 第十四行，第十五行不能在for循环外面，正确形式为：

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

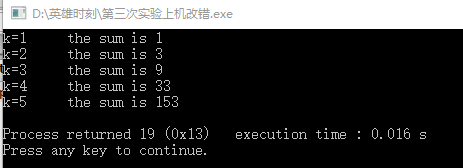
{

fac\*=i;

s+=fac;

}

（2）错误修改后运行结果：



**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：**将s的储存类型换成static，这样只对s进行一次复制，程序的运 算量变小，替换后的程序如下：

#include "stdio.h"

long sum\_fac(int n);

void main(void)

{

int k;

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

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

}

long sum\_fac(int n)

{

long static s=0;

int i;

long fac=1;

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

{

fac\*=i;

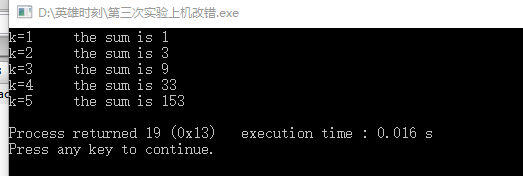
}

s+=fac;

return s;

}

运行截图如下：



1. 修改第1题中sum\_fac函数，计算。

**解答：**

将修改后的源程序使得s+=1/fac，同时将sum\_fac(int n)的返回值声明为 double类型，将fac和s变量声明为double类型，替换后的程序如下：

#include "stdio.h"

double sum\_fac(int n);

void main(void)

{

int k;

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

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

}

double sum\_fac(int n)

{

double s=0;

int i;

double fac=1;

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

{

fac\*=i;

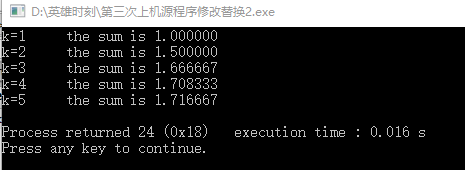
s+=1/fac;

}

return s;

}

运行截图如下：



**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？

i的值是所在内存中储存的一个当前值

p的值是一个地址

1. 从fabonacci函数返回后光条停留在哪个语句上？

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

1. 进入fabonacci函数时，watch窗口显示的是什么？

n的值

1. 当i=3时，从调用fabonacci函数到返回，n值如何变化？

n的值先为3，再变为2，再变为1，最后变为3

**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）。

**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. 解题思路：
2. 定义一个新函数，新函数有两个参数
3. 函数内部使用递归，首先交换大的值和小的值，使大的数值在前 面，当大数对小数求模为零时，返回被除数
4. 如果不是零，再次调用递归，直至两数求模为零
5. 输出最大公约数
6. 结束
7. 程序清单

#include<stdio.h>

int maths(int a,int b);

int main(void)

{

int a,b;

printf("请输入两个整数以求他们的最大公约数:\n");

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

printf("%d",maths(a,b));

return 0;

}

int maths(int a,int b)

{

int c,d;

if(a<b)

{

c=a;

a=b;

b=c;

}

if(a%b==0)

return b;

else

{

d=a%b;

a=b;

b=d;

maths(a,b);

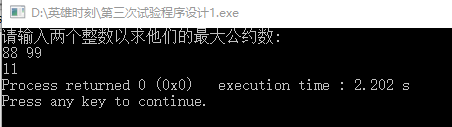
}

return maths(a,b);

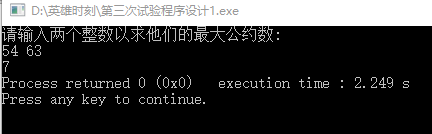
}

1. 测试数据 例1：88 99 例2： 54 63

对应的例1测试结果如图：



对应的例2测试结果如图：



1. 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。

1）解题思路：

1. 输入一个不小于4的偶数
2. 那么这个数最小就是6，由于1是个特殊值，而且这个输入的偶数的 一个素数不可能是2，那么测试的数据要从3开始，如果这俩个数都 为素数输出；否则，其中两个数分别加一减一，若到最后输出两个 素数，则猜想成立；否则不成立
3. 结束
4. 程序清单

#include<stdio.h>

int main(void)

{

int a,b,c,d,e,f=0,g=0;

printf("请连续输入你想验证的不小于4的偶数(停止测试停止请输入非数字):\n");

while(scanf("%d",&a)==1){

d=2;

b=3;

if(a<4||a%2!=0){

printf("请输入不小于4的偶数:\n");

continue;

}

else

for(;;){

if(a==4){

printf("满足条件的素数中其中的两个素数为:1 3\n");

break;

}

if(b%d!=0)

d++;

else{

b++;

d=2;

continue;

}

if(d==b){

c=a-b;

f=1;

e=2;

for(;;){

if(c%e!=0)

e++;

else{

b++;

d=2;

break;

}

if(e==c){

g=1;

}

if(g==1&&f==1)

break;

}

}

if(f==1&&g==1)

break;

}

if(f==1&&g==1){

printf("满足条件的素数中其中的两个素数为:%d %d\n",b,c);

f=0;

g=0;

}

}

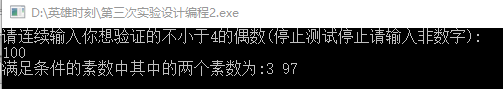
return 0;

}

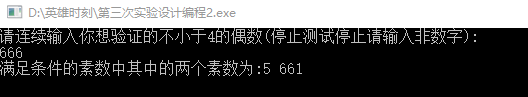
1. 测试

a）测试数据：例1：100 例2：666

b）对应的测试结果如图：

例1：

例2：



（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

1. 解题思路：
2. 首先赋给两个变量值
3. 将其中一个变量的组成这个变量的任意两个素数输出，并将这个变 量的值执行完一次就加2
4. 直至这个变量的值等于另一个变量
5. 程序结束
6. 程序清单

#include<stdio.h>

int main(void)

{

int a,b,c,d,e,f=0,g=0;

int BEGIN=10,END=20;

printf("GOLDNACH'S CONJECTURE:\n");

printf("Every even number n>=4 is the sum of two primes\n");

while(BEGIN<=END){

d=2;

b=3;

a=BEGIN;

for(;;){

if(a==4){

printf("满足条件的素数中其中的两个素数为:1 3\n");

break;

}

if(b%d!=0)

d++;

else{

b++;

d=2;

continue;

}

if(d==b){

c=a-b;

f=1;

e=2;

for(;;){

if(c%e!=0)

e++;

else{

b++;

d=2;

break;

}

if(e==c){

g=1;

}

if(g==1&&f==1)

break;

}

}

if(f==1&&g==1)

break;

}

if(f==1&&g==1){

printf("%d=%d+%d\n",BEGIN,b,c);

f=0;

g=0;

}

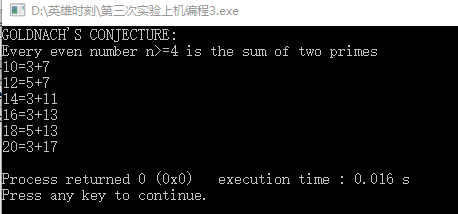
BEGIN=BEGIN+2;

}

return 0;

}

1. 运行截图



**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>

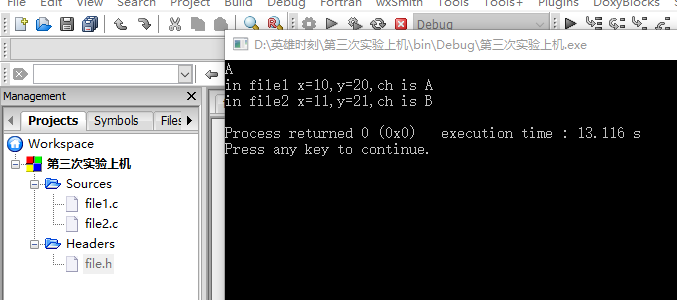
extern int x,y; /\* 外部变量的引用性说明 \*/

extern char ch; /\* 外部变量的引用性说明 \*/

void func1(void); /\* func1函数原型 \*/

1）解题思路：

1. 分别打出这三个文件的源代码
2. 将这三个文件放入同一个项目中
3. 项目将这三个文件自动链接生成可执行文件
4. 程序结束
5. 程序运行结果如图：



**3.3 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。

观察程序中的值变化时，设置断点依次进行，并注意值的变化。在编写哥德巴赫猜想的源程序过程中，一开始没有想到1既不是合数也不是素数，导致程序在循环中永远也不能出来，同时在一次循环结束将变换的值重新赋成最初的值，导致结果不正确。一步步debug发现了问题并进行改正。以后在编写的过程中可以写一些注释来表明这些数字的作用，这样不容易出现错误，就算出现了错误也能够很快的改正过来。

# 实验4 编译预处理实验

**一、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。

**二、实验题目及要求**

**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4行中没有设置中间变量，a和b的值不能交换，正确形式为：

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

1. 第5行没有说明主函数没有参数，正确形式为：

void main(void)

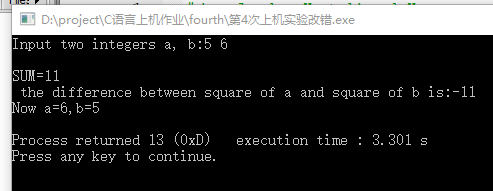
1. 第9行格式化输入里面不应该有逗号，正确形式为：

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

1. 第13行格式化输出函数应该小写，正确形式为：

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

1. 错误修改后运行结果：



**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;

}

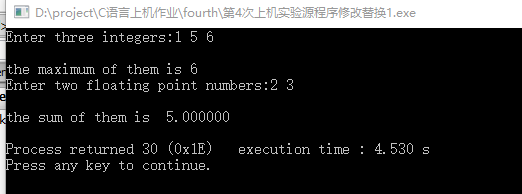
float sum(float x, float y)

{

return x+y;

}

1. 运行截图如下：



1. **解答：**

此题相当于定义一个宏函数，只需让此函数进行条件运算即可，替换后的源程序如下：

#include<stdio.h>

#define max(x,y,z) (((x>y)?x:y)>z)?((x>y)?x:y):z

float sum(float x, float y);

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));

}

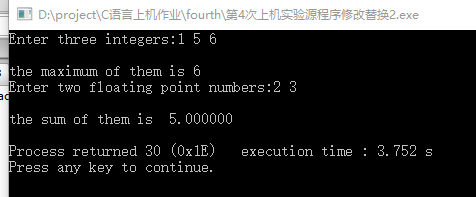
float sum(float x, float y)

{

return x+y;

}

运行截图如下：



**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;

}

1. 修改后的源程序清单：

#include<stdio.h>

#include<assert.h>

#define R

int integer\_fraction(float x);

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;

}

1. x为所求圆的面积，是一个浮点数；i在返回时是一个整数，其值为所求圆面积的整数部分
2. 错误修改过来

**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. 解题思路：首先进行宏定义，定义三个函数，分别代表对三角形面积求值的步骤，然后输入三条边的值，输出，程序结束。
2. 程序清单：

#include<stdio.h>

#include<math.h>

#define S(x,y,z) (x+y+z)/2

#define area(p,x,y,z) sqrt(p\*(p-x)\*(p-y)\*(p-z))

int main(void)

{

double a,b,c,d;

printf("请输入三角形的三条边的边长:\n");

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

d=S(a,b,c);

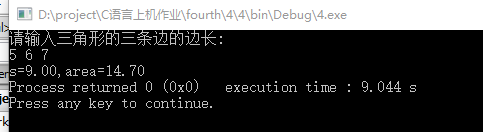
printf("s=%0.2lf,area=%0.2lf",d,area(d,a,b,c));

return 0;

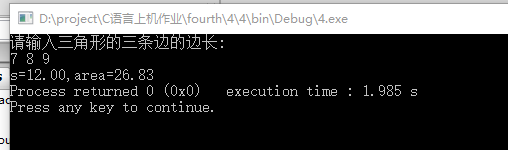
}

测试数据： 例1：5 6 7 例2：7 8 9

对应例1的测试结果如图：



对应例2的测试结果如图：



1. 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. 解题思路：

首先定义CHANCE,然后创建一个足够大的数组，在下面的输入字母并将字母储存到这个数组中，如果定义CHANCE为1，将其中的小写字母的ASCI码减去相应的值，如果定义CHANCE为0，将数组输出，程序结束。

1. 源程序清单：

#include<stdio.h>

#define CHANCE 1

int main(void)

{

int i=0,g=0,a;

char words[1000];

while(a=getchar()){

if(a=='\n')

break;

words[i]=a;

i++;

}

words[i]='\0';

#if CHANCE == 0

puts(words);

#elif CHANCE == 1

for(;;){

if(g==i)

break;

if(words[g]>=97&&words[g]<=122)

words[g]=words[g]-32;

g++;

}

puts(words);

#endif

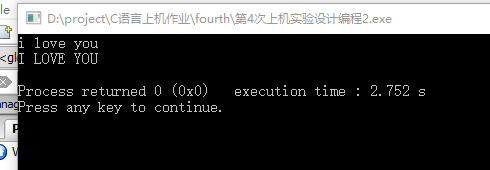
return 0;

}

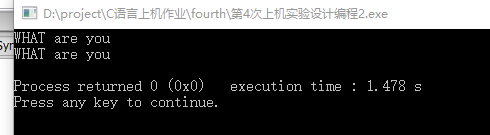
1. 测试：

测试数据：例1： i love you CHANCE=1; 例2：WHAT are you CHANCE=0;

对应例1测试结果如图：



对应例2测试结果如图：



## 5 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。

小结：本次上机实验在宏定义那里掌握的不熟练，通过查找课本解决了问题，宏定义里面的实参不需要分配空间，所以不需要进行声明。

5 数组实验

5.1 实验目的

（1）掌握数组的说明、初始化和使用。

（2）掌握一维数组作为函数参数时实参和形参的用法。

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

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

5.2 实验内容及要求

5.2.1 源程序改错

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

源程序

1 #include<stdio.h>

2 int main(void)

3 {

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

5 void sort(int [],int);

6 int i;

7 sort(a[0],10);

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

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

10 printf("\n");

11 return 0;

12 }

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

14 {

15 int i, j, t;

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

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

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

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

20 }

**解答：**

1. 错误修改：

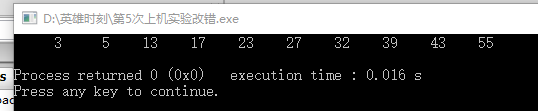
1）第7行sort函数的形参a[0]不是一个地址，正确形式为：

sort(a,10);

2）第18行因为函数按升序排列，小于号是错误的，正确形式为：

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

1. 错误修改后运行结果：



5.2.2 源程序完善、修改、替换

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

源程序：

#include<stdio.h>

#define M 10

#define N 3

int main(void)

{

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

int i, j, k;

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

a[i] = i + 1;

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

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

for(k = 1; k <= N; k++) /\* 1至N报数 \*/

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

b[M-i] = j? : ; /\* 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;

}

1. 填补后的源程序清单：

#include<stdio.h>

#define M 10

#define N 3

int main(void)

{

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

int i, j, k;

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

a[i] = i + 1;

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

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

for(k = 1; k <= N; k++) /\* 1至N报数 \*/

if(++j > i - 1)

j = 0;/\* 最后一个人报数后第一个人接着报，形成一个圈 \*/

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

if(j)

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

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

}

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

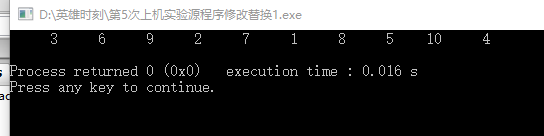
printf("%6d", b[i]);

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

return 0;

}

2）运行截图如下：



1. 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. 解题思路：

只需要将轮到人的编号设为0，当数组元素的值为0时，数数的时候跳过此人即可。但是数组并不应该压缩。

1. 源程序清单：

#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(a[j]==0){

while(a[j]==0){

j++;

if(j>9)

j=0;

}

k--;

}

else{

j++;

if(j>9)

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[9]; /\* 将报数为N的人的编号存入数组b \*/

if(j)

a[j-1]=0 ;

}

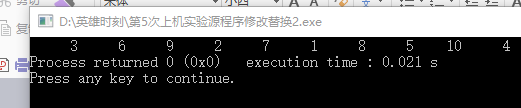
for(i = 0;i <= M - 1; i++) /\* 按次序输出出圈人的编号 \*/

printf("%6d", b[i]); /\* 输出圈中最后一个人的编号 \*/

return 0;

}

1. 程序运行截图如下：



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：**

i=23;s是第一个数组的首元素所在的地址;t是第二个数组的首元素所在的地址。

（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. 解题思路：

由于i所代表的值是第一个数组空字符在数组中的位置，第二个数组只接在了第一个数组空字符的后面，所以不能将两个数组连接起来。

1. 修改后的源程序清单：

#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++]) ;

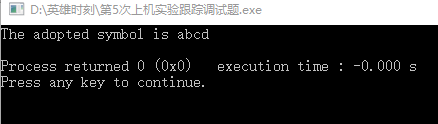
for(j = 0,-- i; j < n && t[j];)

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

s[i] = '\0';

}

1. 程序运行截图如下：



5.2.4 程序设计

编写并上机调试运行能实现以下功能的程序。

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

**解答：**

1. 解题思路：首先设置一个二维数组分别按照矩阵的形式储存数据，然后利用嵌套循环依次输出；之后将矩阵反转，改变输出的顺序，得到反转后的矩阵，程序结束。
2. 源程序清单：

#include<stdio.h>

int main(void)

{

int a,b;

int number1[3][4];

int number2[4][3];

for(a=0;a<3;a++){

for(b=0;b<4;b++){

scanf("%d",&number1[a][b]);

}

}

for(a=0;a<3;a++){

for(b=0;b<4;b++){

printf("%4d",number1[a][b]);

}

printf("\n");

}

printf("\n");

for(b=3;b>=0;b--){

for(a=2;a>=0;a--){

printf("%4d",number1[a][b]);

}

printf("\n");

}

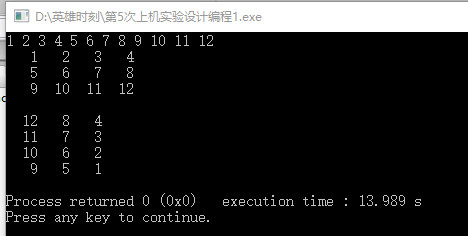
return 0;

}

1. 测试：

测试数据：1 2 3 4 5 6 7 8 9 10 11 12

测试结果如图：



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

**解答：**

1. 解题思路：建立一个数组，将输入的整数从第零位开始与1进行与运算，然后倒着储存到数组中，最后把数组从第一个元素开始依次输出，程序结束。
2. 源程序清单；

#include<stdio.h>

int main(void)

{

int a = 1;

int b ,c ,d;

scanf("%d",&b);

int number[32];

for(c=0;c<32;c++){

b=b>>c;

number[c]=(b&a);

}

for(d=31;d>=0;d--){

printf("%d ",number[d]);

}

return 0;

}

3）测试数据：2

对应的测试结果如图：



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

**解答：**

1. 解题思路：

建立两个数组按照顺序分别储存学生的姓名和成绩，然后输入10个学生的姓名和成绩。然后通过冒泡法排列成绩，使成绩最大的排在前面。按照二分法的方式查找，查找出相应的成绩和学生分数，输出学生姓名和成绩，程序结束。

1. 程序清单：

#include<stdio.h>

#include<string.h>

int search(int a[],int c,int d);

int main(void)

{

int a ,b ,c ,n=10,x ,k=0 ,l;

int score[10]={[0]=0};

char name[10][10] , wait[10];

for(a=0;a<n;a++){

scanf("%s %d",name[a],&score[a]);

}

for(;;){

for(a=n-1;a>=k;a--){

if(score[a-1]<score[a]){

c=score[a];

score[a]=score[a-1];

score[a-1]=c;

strcpy(wait,name[a]);

strcpy(name[a],name[a-1]);

strcpy(name[a-1],wait);

}

}

k++;

if(k==n-1)

break;

}

for(l=0;l<n;l++){

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

}

printf("下面开始查找学生成绩\n");

scanf("%d",&b);

x=search(score,b,n);

if(x!=-1){

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

for(++x;score[x] == b;x++)

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

}

else

printf("not found!\n");

return 0;

}

int search(int a[],int c,int d)

{

int low=0,high=d-1,mid;

while(low<=high){

mid=(low+high)/2;

if(c>a[mid])

high=mid-1;

else if(c<a[mid])

low=mid+1;

else

return (mid);

}

return -1;

}

1. 测试： 测试数据：a 1 b 2 c 3 d 4 e 5 f 6 g 7 h 7 i 7 j 10 7

测试结果如图：



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. 解题思路：设置两个数组，输入两串数组，分别储存到两个数组中，之后输入想要接到第一个数组的哪个字符后面，将两串字符拼接并且加上空字符，之后输出，程序结束。
2. 程序清单：

#include<stdio.h>

int strnins(char words1[],char words2[],int n);

char \* s\_gets(char \* st, int n);

int main(void)

{

int a;

char words1[31];

char words2[11];

s\_gets(words1,11);

s\_gets(words2,11);

scanf("%d",&a);

strnins(words1,words2,a);

puts(words1);

return 0;

}

char \* s\_gets(char \* st, int n)

{

char \* ret\_val;

int i=0;

ret\_val = fgets(st, n, stdin);

if (ret\_val)

{

while (st[i] != '\n' && st[i] != '\0')

i++;

if(st[i] == '\n')

st[i] = '\0';

else

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

continue;

}

return ret\_val;

}

int strnins(char words1[],char words2[],int i)

{

int q;

for(q=0;;i++,q++){

words1[i] = words2[q];

if(words2[q] == '\0')

break;

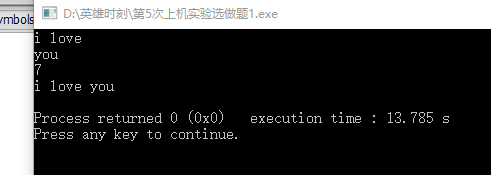
}

return 0;

}

1. 测试： 测试数据：i love you 7

测试结果如图：



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

## 5.3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。

小结：在源程序填空那道题，由于我不清楚条件选择语句前面的=与？谁的优先级更高，导致我在那道题上面花费了过多的时间，因此，学C语言打代码和编程固然很重要，但也要记住课本上的一些公式和优先级之类的基础知识。

# 实验6 指针实验

**一、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。

**二、实验题目及要求**

**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：**

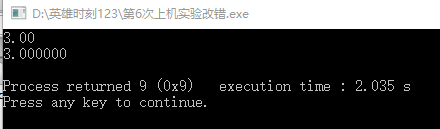
1. 错误修改：
2. 第5行不能直接在悬挂指针所指向的地址上储存值，正确形式为：

float \*p, a;

scanf("%f",&a);

p=&a;

1. 错误修改后运行结果：



**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:

;

}

**解答：**

填空后的源程序如下：

#include "stdio.h"

#include "string.h"

void main(void)

{

char \* (\*p)(char\*,const 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>3);

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;

while((a[i]=getchar())!='\n') i++;a[i]='\0';

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

i=0;

while((b[i]=getchar())!='\n') i++;b[i]='\0';

result=p(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!

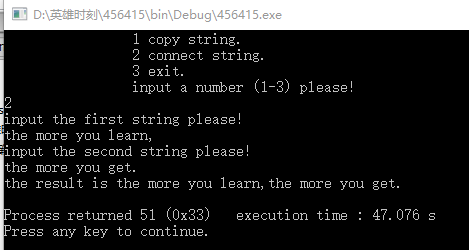
the more you learn, （输入）

input the second string please!

the more you get. （输入）

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

运行结果如图：



**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);

}

1. s的值为数组a第一个元素的地址；返回时s的值为数组a中空字符所在的地址。
2. 修改后的源程序如下：

#include "stdio.h"

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

void main(void)

{

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

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

}

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

{

char \*p;

p=s;

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

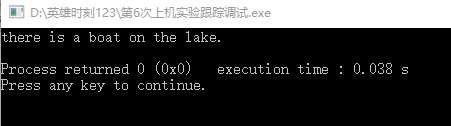
;

s=p;

return (s);

}

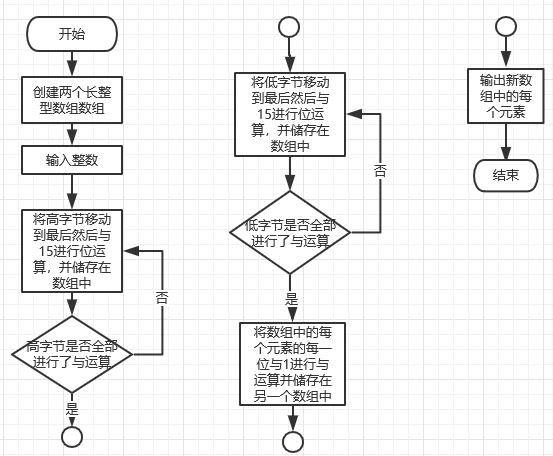
运行截图如下：



**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。

1）算法流程图如图所示：



2）源程序清单：

#include<stdio.h>

int main(void)

{

long int a, b, c=1, d, e, f, g=0;

long int number[8],all[32];

scanf("%ld",&b);

for(d=0,e=28;e>=4;d++,e=e-8)

{

a=15;

number[d]=(b>>e)&a;

}

for(e=24;e>=0;d++,e=e-8)

{

a=15;

number[d]=(b>>e)&a;

}

for(a=1,c=0;c<8;c++)

{

for(f=3;f>=0;f--,g++)

{

all[g]=(number[c]>>f)&a;

}

}

for(a=0;a<g;a++)

printf("%ld",all[a]);

return 0;

}

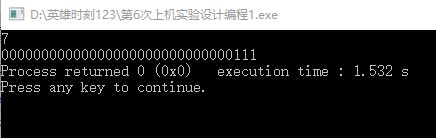
3）测试：

（a）测试数据：

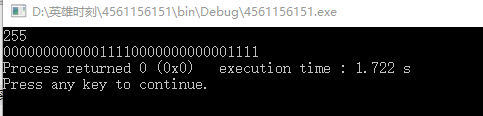
例1： 7 例2：255

（b）对应测试数据的运行结果如图：

例1：

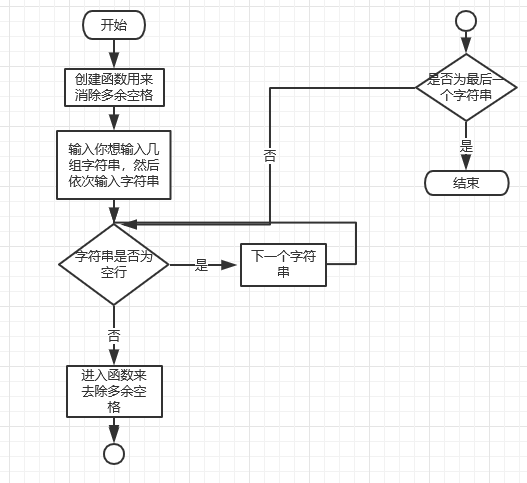


例2：



（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。

1)算法流程图如下：



2）源程序清单：

#include<stdio.h>

int change(char \*);

int main(void)

{

int n, i;

printf("输入你想输入字符的行数:\n");

scanf("%d",&n);

getchar();

printf("下面开始输入字符：\n");

char number[n][80];

char \*p[n];

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

p[i]=number[i];

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

{

gets(number[i]);

}

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

{

if(p[i][0]=='\n')

continue;

change(p[i]);

puts(p[i]);

}

return 0;

}

int change(char \*s)

{

int a, i, q, e;

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

{

q=i;

a=0;

if(s[q]==' ')

{

e=q;

for(;s[q+1]==' ';q++)

a=1;

for(;a==1,s[e+1]!='\0';e++,q++)

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

}

}

return 0;

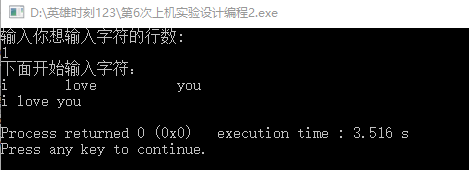
}

3）测试：

（a）测试数据：

例1：i love you

1. 测试结果如图：



（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；

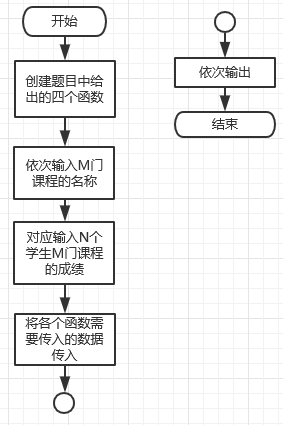
b.计算全班每门课程的平均成绩；

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. 算法流程图如图所示：



1. 源程序清单：

#include<stdio.h>

#define N 4

#define M 3

int student\_every\_average(float \*q,int m,int n);

int subject\_every\_average(float \*q,float \*n,char (\*p)[10],int x,int y);

int below(int \*m,float \*n,float \*q,char (\*p)[10],int x,int y);

int statistics(float \*q,int \*l,int \*h,char (\*p)[10],int x,int y);

int main(void)

{

char subject[M][10];

float grade[N][M];

float average\_subject[M];

char (\*p)[10];

float \*q,\*n=average\_subject;

int i,low[M]={[0]=0},high[M]={[0]=0}, Elow[M]={[0]=0};

int \*m=Elow,\*l=low,\*h=high;

p=subject;

q=grade[0];

for(i=1;i<=M;i++,p++)

{

scanf("%s",\*p);

}

p=subject;

for(i=0;i<N\*M;i++,q++)

{

scanf("%f",q);

}

q=grade[0];

printf("每个学生各门课程的平均成绩：\n");

student\_every\_average(q,M,N);

printf("全班每门课程的平均成绩：\n");

subject\_every\_average(q,n,p,M,N);

printf("\n");

printf("低于全班各门课程平均成绩的人数：\n");

below(m,n,q,p,M,N);

printf("\n");

printf("全班各门课程不及格的人数和90分以上（含90分）的人数：\n");

statistics(q,l,h,p,M,N);

printf("\n");

return 0;

}

int student\_every\_average(float \*q,int m,int n)

{

int i,b;

float a;

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

{

for(b=0,a=0;b<m;b++)

{

a=a+\*(q+i\*m+b);

}

printf("%.2f ",(float)(a/m));

}

printf("\n");

return 0;

}

int subject\_every\_average(float \*q,float \*n,char (\*p)[10],int x,int y)

{

int b,i;

float a;

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

{

for(b=0,a=0;b<y;b++)

{

a=a+\*(q+b\*x+i);

}

\*(n+i)=(float)(a/y);

printf("%s:%.2f ",\*(p+i),\*(n+i));

}

return 0;

}

int below(int \*m,float \*n,float \*q,char (\*p)[10],int x,int y)

{

int i,b;

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

{

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

{

if(\*(q+i\*M+b)<\*(n+b))

\*(m+b)=\*(m+b)+1;

}

}

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

printf("%s:%d ",\*(p+i),\*(m+i));

return 0;

}

int statistics(float \*q,int \*l,int \*h,char (\*p)[10],int x,int y)

{

int i,b;

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

{

for(b=0;b<x;b++)

{

if(\*(q+i\*x+b)<60)

(\*(l+b))++;

if(\*(q+i\*x+b)>=90)

(\*(h+b))++;

}

}

printf("不及格的人：\n");

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

printf("%s: %d ",\*(p+i),\*(l+i));

printf("\n");

printf("90分以上的人：\n");

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

printf("%s: %d ",\*(p+i),\*(h+i));

return 0;

}

3)测试：

1. 测试数据：

Chinese math English

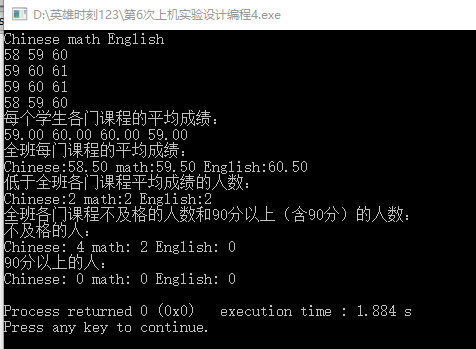
58 59 60

59 60 61

59 60 61

58 59 60

1. 测试结果如图：



**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

1. 解题思路：
2. 声明两个字符数组用来储存用户输入的两组字符串
3. 第三个数组每个元素初始化为0，储存前两个数组对应字符串的和。
4. 若前两个数组对应的字符相加并减去0的ASCII的码大于9的ASCII 码，则第三个数组前一个元素为1。
5. 若前两个数组中字符为“.”，跳过这一元素。
6. 直至加到第一位，若第三个数组第一个元素为1，则按顺序输出第三 个十足的所以元素，否则从第二个人元素开始输出。
7. 程序结束。
8. 源程序清单：

#include<stdio.h>

#define N 20

#define M 10

int main(void)

{

int i;

char number1[M+N+2],number2[M+N+2],number[M+N+2]={[0]=0};

printf("输入你想计算的两个数：\n");

scanf("%s",number1);

scanf("%s",number2);

for(i=M+N+1;i>=1;i--)

{

number1[i]=number1[i-1];

number2[i]=number2[i-1];

}

for(i=M+N+1;i>=1;i--)

{

if(number1[i]=='.')

{

number[i]='.';

continue;

}

number[i]=number[i]+number1[i]+number2[i]-48;

if(number[i]>57)

{

if(number1[i-1]=='.')

number[i-2]=1;

else

{

number[i-1]=1;

}

number[i]=number[i]-10;

}

}

if(number[0]=='1')

for(i=0;i<M+N+2;i++)

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

else

for(i=1;i<M+N+2;i++)

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

return 0;

}

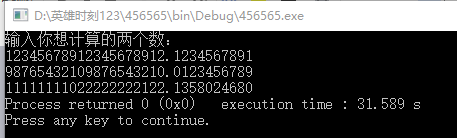
3）测试：

测试数据：

12345678912345678912.1234567891

98765432109876543210.0123456789

测试结果如图：



（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. 解题思路;
2. 声明两个字符串数组储存用户输入的字符串。
3. 然后声明两个函数数组指针分别带表strcmp函数和strstr函数。
4. 比较两个字符串，若相同，输出same，反之，not name
5. 比较两个字符串，若第二组是第一组的子集，输出A与B的交集为B， 反之A与B的交集不为B。
6. 源程序清单：

#include<stdio.h>

#include<string.h>

int main(void)

{

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

int i;

p[0]=strcmp;

p[1]=strstr;

char words1[61],words2[31];

printf("Please input words A\n");

for(i=0;(words1[i]=getchar())!='\n';i++);

words1[i]='\0';

printf("Please input words B\n");

for(i=0;(words2[i]=getchar())!='\n';i++);

words2[i]='\0';

if(p[0](words1,words2)==0)

printf("They are same\n");

else

printf("They are not same\n");

if(p[1](words1,words2)!=NULL)

printf("B∩A=B\n");

else

printf("B∩A≠B\n");

return 0;

}

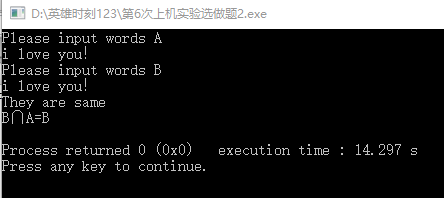
1. 测试：

a）测试数据： 例1：i love you！ i love you！

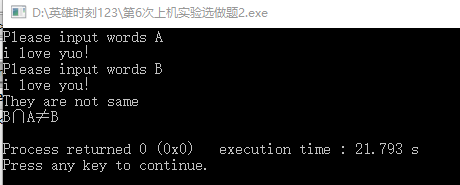
例2：i love yuo！ I love you！

b）测试结果如图：

例1：



例2：



**(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。

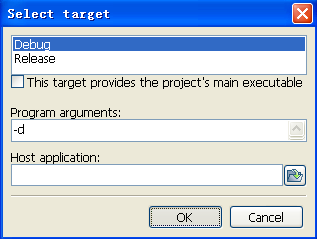
****

图6.？ 输入main函数的参数

# 实验7 结构与联合实验

**一、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。

**二、实验题目及要求**

**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的地址 | B的地址 |
| 3 | \*p++->t,\*p->t | U,x | U,x |
| 4 | \*(++p)->t | x | x |
| 5 | \*++p->t | V | V |
| 6 | ++\*p->t | V | V |

**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 \*/

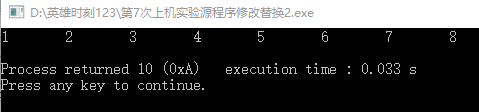
}

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

}

**解答：**

1. 解题思路：\*headp应该改为双重指针\*\*headp，传入的实参为&head。
2. 程序运行截图：



（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. 解题思路：

只需要让后继节点的指针指向前驱结点即可，第一个结点不指向任何位置，为空指针。

2）源程序清单：

#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,\*q,\*l;

if(p[0]==0);

else{

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

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

l=loc\_head;

while(\*p){

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

q->data=\*p++;

q->next=l;

l=q;

}

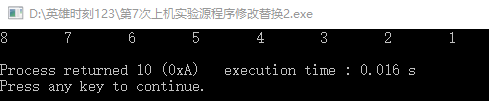
loc\_head->next=NULL;

}

\*headp=q;

}

3）程序运行截图如下：



**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)

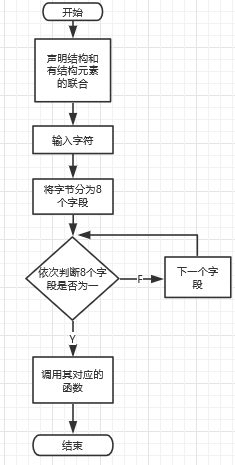
{

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

}

**解答：**

1. 算法流程图如下：



1. 程序清单：

#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;

};

union w8{

unsigned char i;

struct bits s;

}t;

void f0(struct bits b)

{

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

}

void f1(struct bits b)

{

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

}

void f2(struct bits b)

{

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

}

void f3(struct bits b)

{

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

}

void f4(struct bits b)

{

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

}

void f5(struct bits b)

{

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

}

void f6(struct bits b)

{

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

}

void f7(struct bits b)

{

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

}

int main(void)

{

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

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;

int h;

scanf("%c",&h);

t.i=h;

if(t.s.bit0) p\_fun[0](t.s);

if(t.s.bit1) p\_fun[1](t.s);

if(t.s.bit2) p\_fun[2](t.s);

if(t.s.bit3) p\_fun[3](t.s);

if(t.s.bit4) p\_fun[4](t.s);

if(t.s.bit5) p\_fun[5](t.s);

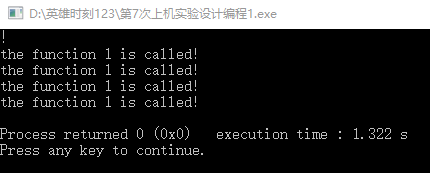
if(t.s.bit6) p\_fun[6](t.s);

if(t.s.bit7) p\_fun[7](t.s);

return 0;

}

1. 测试：
2. 测试数据： ！(是字符输入)
3. 测试结果：



（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。

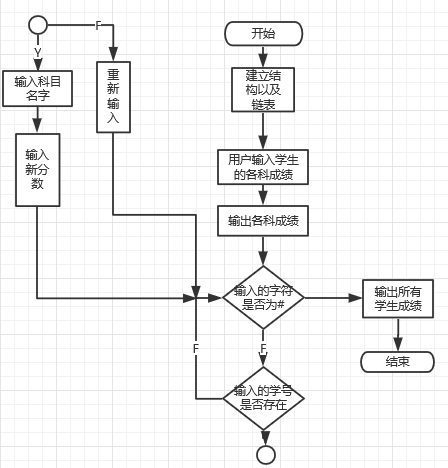
(3) 修改指定学生的指定数据项的内容。

(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. 算法流程图如下：



1. 程序清单

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct s\_list{

char number[11];

char name[21];

char subject[4][20];

float goal[4];

float average;

struct s\_list \*next;

};

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

int main(void)

{

char search1[11];

char search2[20];

int i,b=0,N;

printf("Please input the number of students\n");

scanf("%d",&N);//学生的数量

float a;

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

printf("input the information of every student\n");

create\_list(&head,N);

p=head;

printf("\nif you want change someone\'s point,");

printf("please input his student number,else input # to leave\n");

scanf("%s",search1);

for(;p!=NULL,search1[0]!='#';){

if(strcmp(p->number,search1)==0){

printf("Please input which subject\'point that you want change\n");

scanf("%s",search2);

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

if(strcmp(p->subject[i],search2)==0){

printf("input new point\n");

scanf("%f",&p->goal[i]);

p->average=(p->goal[0]+p->goal[1]+p->goal[2]+p->goal[3])/4;

break;

}

}

printf("continue correct? if continue please input student number,");

printf(" else input # to leave.\n");

scanf("%s",search1);

}

else

p=p->next;

if(p==NULL){

printf("Not exist this student,please input again.");

printf("if you want to leave ,please input #.\n");

scanf("%s",search1);

p=head;

}

}

printf("\n");

p=head;

while(p){

puts(p->number);

puts(p->name);

for(i=0,a=0;i<4;i++){

printf("%s:",p->subject[i]);

printf("%-6.2f ",p->goal[i]);

}

printf("平均成绩:%-6.2f\n",p->average);

p=p->next;

}

return 0;

}

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

{

int a,i,b;

char sj[4][20]={"English","Math","Physics","C\_language"};

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

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

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

loc\_head->subject[a][i]=sj[a][i];

loc\_head->subject[a][i]='\0';

}//将科目名称赋给数组

scanf("%s",loc\_head->number);

scanf("%s",loc\_head->name);

for(a=0,i=0;a<4;a++){

scanf("%f",&(loc\_head->goal[a]));

i=loc\_head->goal[a]+i;

}

loc\_head->average=i/4;

tail=loc\_head;

for(b=1;b<n;b++){

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

tail=tail->next;

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

tail->subject[a][i]=sj[a][i];

tail->subject[a][i]='\0';

}

scanf("%s",tail->number);

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

for(a=0,i=0;a<4;a++){

scanf("%f",&(tail->goal[a]));

i=tail->goal[a]+i;

}

tail->average=i/4;

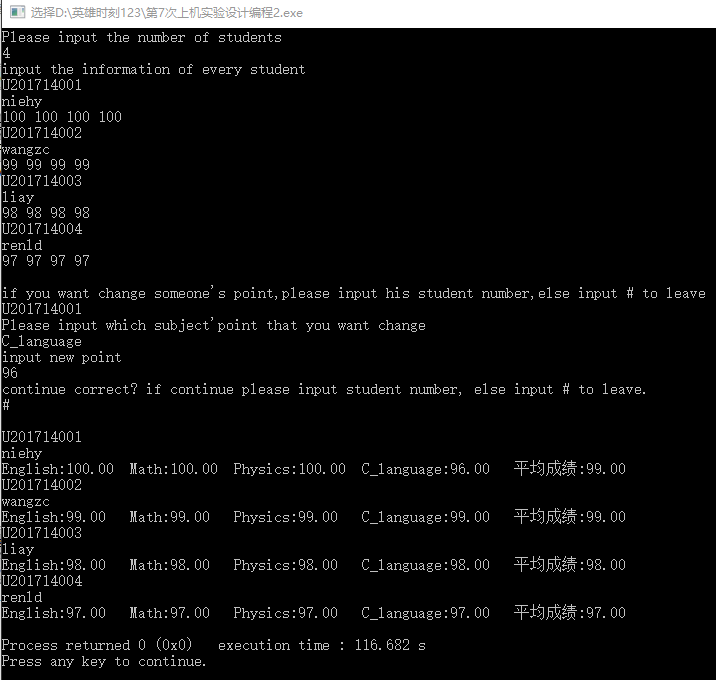
}

tail->next=NULL;

\*headp=loc\_head;

}

1. 程序运行截图如下：



**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. 解题思路:
2. 比较每个结构里面的平均成绩
3. 交换每个结构里面数据域所有的内容，直至最后一个结构
4. 若更改了学生的数据，平均值需要重新计算
5. 输出每个结构的数据域内容
6. 程序结束
7. 源程序清单：

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct s\_list{

char number[11];

char name[21];

char subject[4][20];

float goal[4];

float average;

struct s\_list \*next;

};

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

void compare(struct s\_list \*\*headp,int n);

int main(void)

{

char search1[11];

char search2[20];

int i,b=0,N;

printf("Please input the number of students.\n");

scanf("%d",&N);//学生的数量

float a;

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

printf("Input the information of every student.\n");

create\_list(&head,N);

p=head;

printf("\nIf you want change someone\'s point ,");

printf("please input his student number ,else input # to leave.\n");

scanf("%s",search1);

for(;p!=NULL,search1[0]!='#';){

if(strcmp(p->number,search1)==0){

printf("Please input which subject\'point that you want change.\n");

scanf("%s",search2);

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

if(strcmp(p->subject[i],search2)==0){

printf("Input new point.\n");

scanf("%f",&p->goal[i]);

p->average=(p->goal[0]+p->goal[1]+p->goal[2]+p->goal[3])/4;

break;

}

}

printf("Continue correct? if continue please input student number,");

printf(" else input # to leave.\n");

scanf("%s",search1);

}

else

p=p->next;

if(p==NULL){

printf("Not exist this student ,please input again.");

printf("If you want to leave ,please input #.\n");

scanf("%s",search1);

p=head;

}

}

printf("\n");

p=head;

while(p){

puts(p->number);

puts(p->name);

for(i=0,a=0;i<4;i++){

printf("%s:",p->subject[i]);

printf("%-6.2f ",p->goal[i]);

}

printf("平均成绩:%-6.2f\n",p->average);

p=p->next;

}

compare(&head,N);

return 0;

}

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

{

int a,i,b;

char sj[4][20]={"English","Math","Physics","C\_language"};

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

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

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

loc\_head->subject[a][i]=sj[a][i];

loc\_head->subject[a][i]='\0';

}//将科目名称赋给数组

scanf("%s",loc\_head->number);

scanf("%s",loc\_head->name);

for(a=0,i=0;a<4;a++){

scanf("%f",&(loc\_head->goal[a]));

i=loc\_head->goal[a]+i;

}

loc\_head->average=i/4;

tail=loc\_head;

for(b=1;b<n;b++){

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

tail=tail->next;

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

tail->subject[a][i]=sj[a][i];

tail->subject[a][i]='\0';

}

scanf("%s",tail->number);

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

for(a=0,i=0;a<4;a++){

scanf("%f",&(tail->goal[a]));

i=tail->goal[a]+i;

}

tail->average=i/4;

}

tail->next=NULL;

\*headp=loc\_head;

}

void compare(struct s\_list \*\*headp,int n)

{

int i,a,c,l;

char e;

float b;

struct s\_list \*p;

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

p=\*headp;

for(a=0;a<n-i-1;a++,p=p->next){

if(p->average>p->next->average){

for(c=0;c<11;c++){

l=p->number[c];

p->number[c]=p->next->number[c];

p->next->number[c]=l;

}

for(c=0;c<21;c++){

e=p->name[c];

p->name[c]=p->next->name[c];

p->next->name[c]=e;

}

for(c=0;c<4;c++){

b=p->goal[c];

p->goal[c]=p->next->goal[c];

p->next->goal[c]=b;

}

b=p->average;

p->average=p->next->average;

p->next->average=b;

}

}

}

p=\*headp;

printf("\nthe score in order:\n");

while(p){

puts(p->number);

puts(p->name);

for(i=0,a=0;i<4;i++){

printf("%s:",p->subject[i]);

printf("%-6.2f ",p->goal[i]);

}

printf("平均成绩:%-6.2f\n",p->average);

p=p->next;

}

}

1. 测试

a）测试数据：4

u1

nie

100 100 100 100

u2

wang

99 99 99 99

u3

li

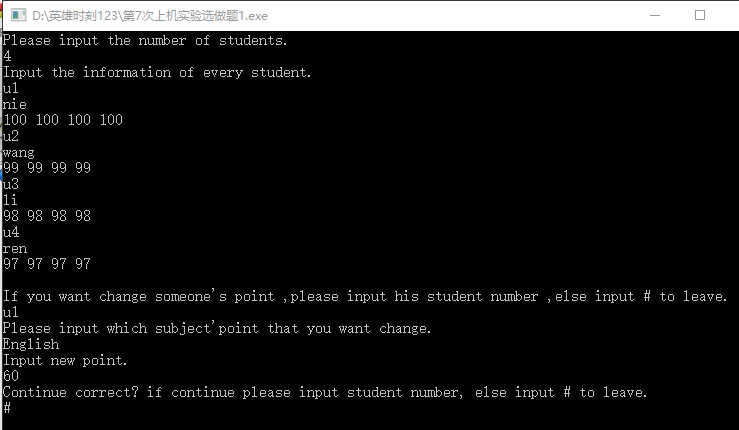
98 98 98 98

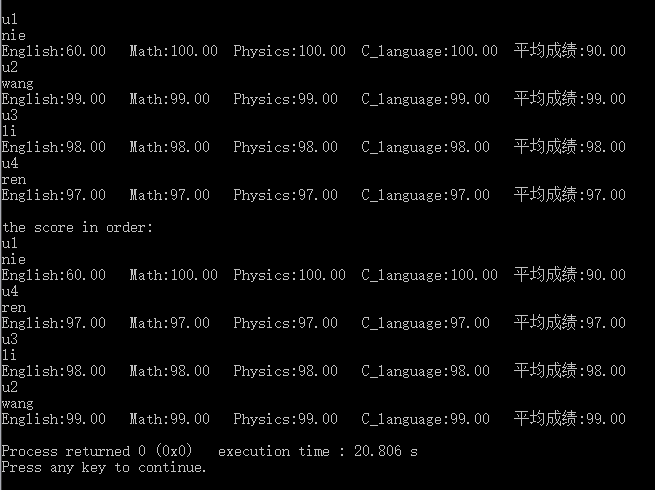
u4

ren

97 97 97 97 u1 English # 60

b）测试结果如下面的两个图：





（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. 解题思路：
2. 比较结构中的平均成绩
3. 对首节点进行单独处理，若首节点改变，令\*headp指向新的结点
4. 中部排序和尾部改变同时进行就可以

4.若更改了学生的数据，平均值需要重新计算

5.输出结构中的各个数据

6.程序结束

2）源程序清单

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct s\_list{

char number[11];

char name[21];

char subject[4][20];

float goal[4];

float average;

struct s\_list \*next;

};

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

void compare(struct s\_list \*\*headp);

int main(void)

{

char search1[11];

char search2[20];

int i,b=0,N;

printf("Please input the number of students.\n");

scanf("%d",&N);//学生的数量

float a;

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

printf("Input the information of every student.\n");

create\_list(&head,N);

p=head;

printf("\nIf you want change someone\'s point ,");

printf("Please input his student number ,else input # to leave.\n");

scanf("%s",search1);

for(;p!=NULL,search1[0]!='#';){

if(strcmp(p->number,search1)==0){

printf("Please input which subject\'point that you want change.\n");

scanf("%s",search2);

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

if(strcmp(p->subject[i],search2)==0){

printf("Input new point.\n");

scanf("%f",&p->goal[i]);

p->average=(p->goal[0]+p->goal[1]+p->goal[2]+p->goal[3])/4;

break;

}

}

printf("continue correct? if continue please input student number ,");

printf(" else input # to leave.\n");

scanf("%s",search1);

}

else

p=p->next;

if(p==NULL){

printf("Not exist this student,please input again.");

printf("if you want to leave ,please input #.\n");

scanf("%s",search1);

p=head;

}

}

printf("\n");

p=head;

while(p){

puts(p->number);

puts(p->name);

for(i=0,a=0;i<4;i++){

printf("%s:",p->subject[i]);

printf("%-6.2f ",p->goal[i]);

}

printf("平均成绩:%-6.2f\n",p->average);

p=p->next;

}

p=head;

compare(&p);

printf("the score in order\n");

while(p){

puts(p->number);

puts(p->name);

for(i=0,a=0;i<4;i++){

printf("%s:",p->subject[i]);

printf("%-6.2f ",p->goal[i]);

}

printf("平均成绩:%-6.2f\n",p->average);

p=p->next;

}

return 0;

}

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

{

int a,i,b;

char sj[4][20]={"English","Math","Physics","C\_language"};

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

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

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

loc\_head->subject[a][i]=sj[a][i];

loc\_head->subject[a][i]='\0';

}//将科目名称赋给数组

scanf("%s",loc\_head->number);

scanf("%s",loc\_head->name);

for(a=0,i=0;a<4;a++){

scanf("%f",&(loc\_head->goal[a]));

i=loc\_head->goal[a]+i;

}

loc\_head->average=i/4;

tail=loc\_head;

for(b=1;b<n;b++){

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

tail=tail->next;

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

tail->subject[a][i]=sj[a][i];

tail->subject[a][i]='\0';

}

scanf("%s",tail->number);

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

for(a=0,i=0;a<4;a++){

scanf("%f",&(tail->goal[a]));

i=tail->goal[a]+i;

}

tail->average=i/4;

}

tail->next=NULL;

\*headp=loc\_head;

}

void compare(struct s\_list \*\*headp)

{

struct s\_list \*prior1,\*prior2,\*p1,\*p2,\*t;

int i=0;

p1=\*headp;

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

if(p1->average>p2->average){

prior2->next=p1;

t=p1->next;

p1->next=p2->next;

(\*headp)=p2;

p2->next=t;

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->average>p2->average){

t=p2->next;

prior1->next=p2;

prior2->next=p1;

p2->next=p1->next;

p1->next=t;

p1=p2;

}

}

3）测试：

a）测试数据：4

u1

nie

100 100 100 100

u2

wang

99 99 99 99

u3

li

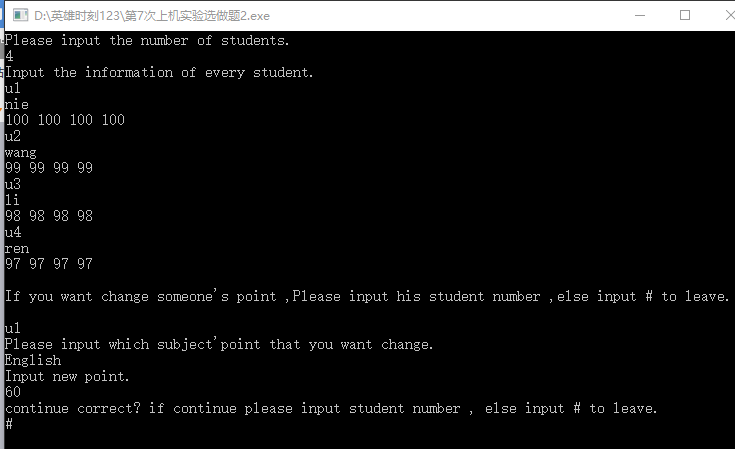
98 98 98 98

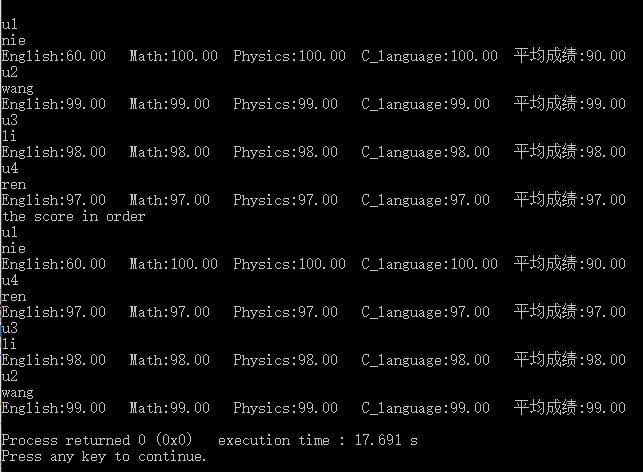
u4

ren

97 97 97 97 u1 English # 60

b）测试结果如下面的两个图：





（3）采用双向链表重做编程设计题第（2）题。

**解答：**

1. 解题思路：
2. 只需要在程序设计的第二题基础上加上一个指向前一个结点的指针就可以。
3. 第一个结点的指针指向最后一个结点
4. 最后一个结点的指针指向第一个结点
5. 遍历链表
6. 程序结束
7. 程序清单

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct s\_list{

char number[11];

char name[21];

char subject[4][20];

float goal[4];

float average;

struct s\_list \*next;

struct s\_list \*prior;

};

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

int main(void)

{

char search1[11];

char search2[20];

int i,b=0,N;

printf("Please input the number of students.\n");

scanf("%d",&N);//学生的数量

float a;

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

printf("Input the information of every student.\n");

create\_list(&head,N);

p=head;

printf("\nIf you want change someone\'s point ,");

printf("please input his student number ,else input # to leave.\n");

scanf("%s",search1);

for(;p!=head->prior,search1[0]!='#';){

if(strcmp(p->number,search1)==0){

printf("Please input which subject\'point that you want change.\n");

scanf("%s",search2);

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

if(strcmp(p->subject[i],search2)==0){

printf("Input new point.\n");

scanf("%f",&p->goal[i]);

p->average=(p->goal[0]+p->goal[1]+p->goal[2]+p->goal[3])/4;

break;

}

}

printf("Continue correct? if continue please input student number,");

printf(" else input # to leave.\n");

scanf("%s",search1);

}

else

p=p->next;

if(p==head->prior){

printf("Not exist this student ,please input again.");

printf("If you want to leave ,please input #.\n");

scanf("%s",search1);

p=head;

}

}

printf("\n");

p=head;

do{

puts(p->number);

puts(p->name);

for(i=0,a=0;i<4;i++){

printf("%s:",p->subject[i]);

printf("%-6.2f ",p->goal[i]);

}

printf("平均成绩:%-6.2f\n",p->average);

p=p->next;

}while(p!=head);

return 0;

}

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

{

int a,i,b;

char sj[4][20]={"English","Math","Physics","C\_language"};

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

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

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

loc\_head->subject[a][i]=sj[a][i];

loc\_head->subject[a][i]='\0';

}//将科目名称赋给数组

scanf("%s",loc\_head->number);

scanf("%s",loc\_head->name);

for(a=0,i=0;a<4;a++){

scanf("%f",&(loc\_head->goal[a]));

i=loc\_head->goal[a]+i;

}

loc\_head->average=i/4;

tail=loc\_head;

for(b=1;b<n;b++){

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

tail->next->prior=tail;

tail=tail->next;

for(a=0;a<4;a++){

for(i=0;sj[a][i]!='\0';i++)

tail->subject[a][i]=sj[a][i];

tail->subject[a][i]='\0';

}

scanf("%s",tail->number);

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

for(a=0,i=0;a<4;a++){

scanf("%f",&(tail->goal[a]));

i=tail->goal[a]+i;

}

tail->average=i/4;

}

tail->next=loc\_head;

loc\_head->prior=tail;

\*headp=loc\_head;

}

1. 测试

a）测试数据：4

u1

nie

100 100 100 100

u2

wang

99 99 99 99

u3

li

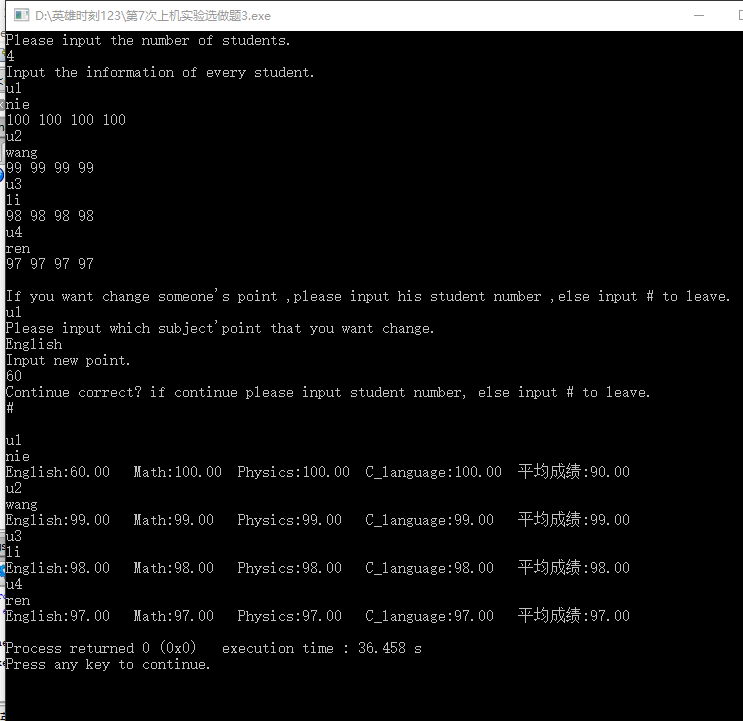
98 98 98 98

u4

ren

97 97 97 97 u1 English # 60

b）测试结果如图：



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