第3章 实 验 项 目

实验项目1　运行一个C程序

一、实验目的

（1）熟悉 Visual C++ 6.0 和 C-Free 5.0 两种 C 语言运行环境。
（2）掌握在上述两种环境下编辑、编译、连接和运行一个 C 程序的方法。
（3）通过运行简单的 C 程序，认识 C 程序的特点，掌握和理解 C 程序的结构。

二、实验要求

（1）进入 Visual C++ 6.0 或 C-Free 5.0 的工作环境。
（2）熟悉 Visual C++ 6.0 或 C-Free 5.0 集成环境，掌握系统主菜单中常用命令的使用。
（3）运行简单的 C 程序，逐步掌握编辑、编译、连接、运行和调试 C 程序的方法。

三、实验内容

1．验证性实验

（1）在 Visual C++ 6.0 或 C-Free 5.0 环境下编辑下列 C 语言程序，编译、连接并运行，观察并理解其运行结果。

```
#include<stdio.h>
int main()
{
    int a,b,c;
    printf("Enter first integer:");
    scanf("%d",&a);
    printf("Enter second integer:");
    scanf("%d",&b);
    c=a+b;
    printf("a+b=%d\n",c);
    return 0;
}
```

程序运行结果如图 3-1 所示。

图 3-1　从键盘输入两个整数并求它们的和

思考

① 去掉“#include <stdio.h>”，运行程序，观察运行结果，并分析为什么。
② 去掉“int a,b,c;”语句后的“;”，运行程序，观察运行结果，并分析为什么。
③ 将“c=a+b;”改为“C=a+b;”，运行程序，观察运行结果，并分析为什么。

（2）尝试改正下列程序中的错误，直到程序经编译后没有错误信息，并得到题目要求的运行结果。

题目要求得到的输出结果如图 3-2 所示。

图 3-2　题目要求得到的输出结果

含有错误的源程序如下：

```
#include<stdio.h>
int main()
{
    int a=1;b=2,c=3,
    printf("output:%d,%d,%d\n"a,b,c);
    return 0;
}
```

2．设计性实验

（1）编写程序，从键盘输入两个整型变量 a 和 b，求它们的差并输出，图 3-3 所示是一个示例结果。

图 3-3　从键盘输入两个整数并求它们的差

（2）编写程序，从键盘输入 x 的值，根据公式 $y=x^2+1$ 求 y 的值，输出 x 和 y 的值（假设 x 和 y 都是 int 型变量），图 3-4 所示是一个示例结果。

图 3-4　求 x^2+1 的值

实验提示：

x^2 可表示为 x*x。

（3）编写程序，从键盘输入 x 的值，求 x 的平方根并赋给变量 y，输出 x 和 y 的值（假设 x 和 y 都是 int 型变量），图 3-5 所示是一个示例结果。

图 3-5　求 x 的平方根

实验提示：

可用函数 sqrt(x)求 x 的平方根，要使用该函数，必须在程序前面加上头文件 math.h。

实验项目 2　数据类型与表达式

一、实验目的

（1）掌握 C 语言基本数据类型的常量表示、变量的定义和使用。
（2）掌握 C 语言各类运算符的优先级、结合性及相应表达式的求值方法。
（3）掌握不同类型数据之间的赋值规律，理解自动与强制数据类型转换。
（4）学会编写程序求表达式结果的方法。
（5）掌握 C 语言基本数据类型数据的输入方法。
（6）掌握常见格式控制字符对输出结果的控制作用。
（7）理解数据溢出错误和舍入误差（以整型、实型数据为例）。
（8）掌握指针的简单使用方法。
（9）进一步熟悉 C 程序的实现过程。

二、实验要求

（1）实验时仔细对比程序实际运行结果，认真分析并回答思考中的问题。

（2）如果程序有错，记录编译、连接、运行过程中遇到的提示错误。分析出错原因，记下更正的方法。

（3）可以对实验程序进行修改、补充，以便完成自己需要的程序验证和测试。在完成实验要求的工作外，要学会创造性地工作。

三、实验内容

1．验证性实验

（1）输入以下程序，运行后分析结果。

```
#include<stdio.h>
int main()
{
    int i,j,m,n;
    i=97;j=65;
    m=++i;
    n=j++;
    printf("i=%d,j=%d,m=%d,n=%d\n",i,j,m,n);
    printf("i=%c,j=%c,m=%c,n=%c\n",i,j,m,n);
    /*printf("i=%f,j=%f,m=%f,n=%f\n",i,j,m,n);*/
    return 0;
}
```

程序运行结果如图 3-6 所示。

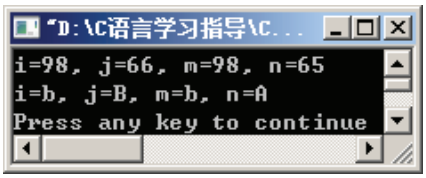


图 3-6　程序运行结果

思考

① 程序中语句“m=++i;”起什么作用？

② 程序中语句“n=j++;”起什么作用？

③ 程序中语句“printf("i=%d, j=%d, m=%d,n=%d\n",i,j,m,n);”起什么作用？

④ 程序中语句“printf("i=%c, j=%c, m=%c, n=%c\n",i,j,m,n);”起什么作用？

⑤ 程序中语句“/*printf("i=%f, j=%f, m=%f, n=%f\n",i,j,m,n);*/”执行了吗？如果把其中的/*和*/去掉，变成语句“printf("i=%f, j=%f, m=%f,n=%f\n",i,j,m,n);”，在 VC++ 6.0 中运行会出现图 3-7 所示的运行结果，这是一个什么错误？为什么会出现这个错误？

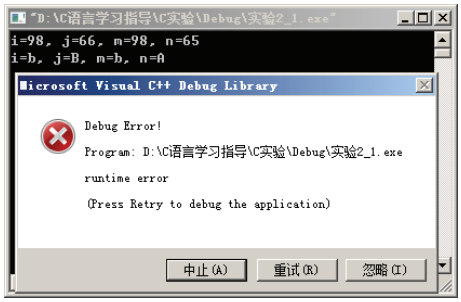


图 3-7　修改后的运行结果与错误提示

（2）输入以下程序，运行前将程序中的 k 分别赋值为 127、−128、128 和−129，分析程序运行结果。

```
#include<stdio.h>
int main()
{
    float a=3.7,b;
    int i,j=5;
    int k=127;/*分别用 127,-128,128,-129 测试*/
    unsigned u;
    long l;
    char c;
    i=a;printf("%d\n",i);          /*实型赋值给整型*/
    b=j;printf("%f\n",b);          /*整型赋值给实型*/
    u=k;printf("%d,%u\n",u,u);     /*相同长度类型之间赋值*/
    l=k;printf("%ld\n",l);         /*整型赋值给长整型，短的类型赋值给长的类型*/
    c=k;printf("%d\n",c);          /*整型赋值给字符型，长的类型赋值给短的类型*/
    return 0;
}
```

思考

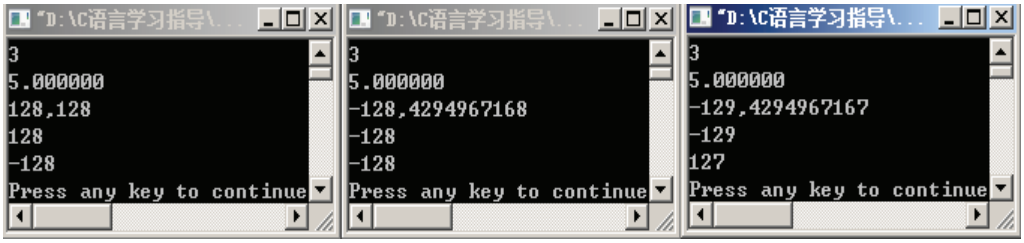
① 编译时有没有出现警告提示？连接时有没有出现错误？运行时呢？换一个编译器试试。不同的编译器对数据类型的数据范围检查的严格程度都相同吗？图 3-8 所示为 VC++ 6.0 中的一个运行结果（k=127 时）。

图 3-8　k=127 时的运行结果

② 修改 k 的值，再运行程序，分析运行结果。

（a）k=128　　（b）k=−128　　（c）k=−129

③ 图 3-9 所示为 VC++ 6.0 中的三个运行结果（k 分别为 128、−128 和−129 时）。



（a）k=128　　（b）k=−128　　（c）k=−129

图 3-9　k 分别为 128、−128 和−129 时的运行结果

④ 修改 k 的值为 127，分别把“i=a;printf("%d\n",i);”和“b=j;printf("%f\n",b);”改为“i=a;printf("%f\n",i);”和“b=j;printf("%d\n",b);”，再运行程序，分析为什么会出现图 3-10 中的运行结果。

图 3-10　修改程序后的运行结果

（3）输入以下程序，根据程序运行结果（见图 3-11）分析每个关系表达式的值。

图 3-11　关系表达式的值

```
#include<stdio.h>
int main()
{
    char c='k';
    int i=1,j=2,k=3;
    float x=3e+5,y=0.85;
    printf("%d,%d\n",'a'+5<c,-i-2*j>=k+1);
    printf("%d,%d\n",1<j<5,x-5.25<=x+y);
    printf("%d,%d\n",i+j+k==-2*j,k==j==i+5);
    return 0;
}
```

思考

① 表达式'a'+5<c 的求值过程是怎样的？
② 表达式 1<j<5 的求值过程是怎样的？
③ 表达式 k==j==i+5 的求值过程是怎样的？

（4）输入以下程序，根据程序运行结果（见图 3-12）分析每个逻辑表达式的值。

图 3-12　逻辑表达式的值

```
#include<stdio.h>
void main()
{
    char c='k';
    int i=1,j=2,k=3;
    float x=3e+5,y=0.85;
    printf("%d,%d\n",,!x, !x*!y );
    printf("%d,%d\n", x||i&&j-3, i<j&&x<y );
    printf("%d,%d\n", i==5&&c&&(j=8), x+y||i+j+k );
    return 0;
}
```

思考

① 表达式!x*!y 的求值过程是怎样的？
② 表达式 x||i&&j-3 和 i<j&&x<y 的求值过程是怎样的？
③ 表达式 i==5&&c&&(j=8)和 x+y||i+j+k 的求值过程是怎样的？

（5）分别输入以下两个程序并运行，然后分析两个程序中每条语句的作用，并分析其中的指针变量所指的对象。

```
#include<stdio.h>
int main()
{
    int a, b,*pa=&a,*pb=&b,max;
    printf("please input two intergers: a,b\n");
    scanf("%d,%d",pa,pb);
    max=*pa>*pb?*pa:*pb;
    printf("max=%d\n",max);
```

```
    return 0;
}
#include <stdio.h>
int main()
{
    int a, b,*pmax=0;
    printf("please input two intergers: a,b\n");
    scanf("%d,%d",&a,&b);
    pmax=a>b?&a:&b;
    printf("max=%d\n",*pmax);
    return 0;
}
```

如图 3-13 所示，运行两个程序，分别输入 3,4 并按回车键，运行结果都为 max=4。

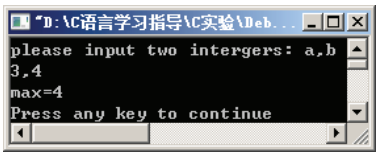


图 3-13　输入 3,4 时的运行结果

实验提示：运行本程序时，输入的两个整数之间用逗号隔开，按回车键表示输入结束。

思考

① 求两个整数中的较大数，不用指针是不是更简单？尝试写出程序。

② 在所写的程序上修改，使得任意输入两个实数，可以输出其中的较大数。

③ 在所写的程序上修改，使得任意输入两个字符，可以输出其中的 ASCII 码较大者。

④ 换成三个比较对象时，应该如何修改这些程序？

2．设计性实验

（1）已知变量 a 和 b 为 int 型，其值分别为 3 和 10；变量 x 和 y 为 double 型，其值分别为 6.8 和 1.9，计算算术表达式(double)(a+b)/2+(int)x%(int)y 的值。试编程验证自己的计算结果是否正确。

实验提示：

① 先命名一个变量用来保存最终结果，例如 result；再判断结果应该是什么类型，假设是 double 型；在程序中根据变量的类型和名字对它进行声明，例如“double result;”。

② 在程序对 a、b、x 和 y 四个变量声明的同时可以赋初值，也可以声明之后再赋值。然后用语句“result=(double)(a+b)/2+(int)x%(int)y;”将表达式的值赋给变量 result。

③ 最后在屏幕上输出变量 result 的值，这个值就是表达式的值。输出 double 型的变量 result 的值可以用语句“printf("result=%lf",result);”或“printf("result=%f",result);”。图 3-14 所示为一个示例结果。

图 3-14　(double)(a+b)/2+(int)x%(int)y 的值

（2）已知：a=8，n=3（a、n 为 int 型），分析依次计算下面表达式时 a 的值如何变化，试编程上机验证。

① a+=a

② a*=2+3

③ a%=(n%=2)

④ a=a+3

⑤ a+=a-=a*=a

实验提示：

① 程序对 a 和 n 两个变量声明的同时可以赋初值，也可以声明之后再赋值。

② 把要计算的表达式后面加上分号，变成表达式语句。这 5 个表达式语句都是复合赋值表达式或赋值表达式。

③ 在这 5 个表达式语句的每个表达式语句之后都加上一个 printf()函数调用语句，用来输出变量 a 的值。

图 3-15 所示为一个示例结果。

（3）编写程序产生一个如图 3-16 所示的输出结果。

图 3-15　（复合）赋值表达式的值

图 3-16　结果样式

实验提示：分析这个输出结果共几行，每行都可以调用 printf()函数来实现。

3．提高实验

现在是 24 小时制的 9 时 30 分，编程求出再过 1000 分钟是 24 小时制的几时几分？程序结果示例如图 3-17 所示。

图 3-17　程序结果示例

实验提示：

假设今天是周二，则 100 天后是((100+2)%7)。

实验项目 3　顺序结构程序设计方法

一、实验目的

（1）掌握 C 语言中使用最多的一种语句——赋值语句的使用方法。

（2）掌握 C 程序的基本构成，熟悉顺序结构程序设计的一般步骤。

二、实验要求

（1）理解 C 语言中程序设计的基本思路。

（2）能够编写简单的顺序结构程序。

三、实验内容

1．验证性实验

（1）对于温度的计算很多国家采用的是华氏温度标准（F），而国内采用的是摄氏温度（C）。现在，请根据温度转换公式设计一个温度转换程序，可以进行温度转换。如果输入摄氏温度，显示转换的华氏温度。运行结果如图 3-18 所示。

图 3-18　温度转换运行结果

温度转换的公式为：F＝(C×9 / 5)＋32

实验提示：

① 声明变量 f 和 c。

② 输入 c 的值。

③ 利用公式计算 f 的值。

④ 输出 f 的值。

程序代码参考：

```
#include "stdio.h"
int main(){
    float f,c;
    printf("请输入摄氏温度的值:");
    scanf("%f",&c);
    f=c*9/5+32;
    printf("华氏温度 F=%f\n",f);
```

```
    return 0;
}
```

采用指针实现的程序代码参考：

```
#include "stdio.h"
int main(){
    float f,c;
    float *pf=&f,*pc=&c;
    printf("请输入摄氏温度的值:");
    scanf("%f",pc);
    *pf=*pc*9/5+32;
    printf("华氏温度 F=%f\n",*pf);
    return 0;
}
```

思考

如果将语句“f=c*9/5+32;”换成“f=9/5*c+32;”输出结果会是多少呢？

（2）从键盘输入任意两个数，输出它们的和值与差值之积，运行结果如图 3-19 所示。

图 3-19　求两数的和值与差值之积的运行结果

实验提示：

① 定义三个变量 a,b,s。

② 输入 a,b 的值。

③ 计算它们的和值与差值之积，存放在变量 s 中。

④ 输出 s 的值。

程序参考代码如下：

```
#include "stdio.h"
int main(){
    int a,b,s;
    printf("请输入 a,b 的值:");
    scanf("%d,%d",&a,&b);
    s=(a+b)*(a-b);
    printf("s=%d\n",s);
    return 0;
}
```

用指针实现的代码如下：

```
#include "stdio.h"
int main(){
    int a,b,s;
    int *pa=&a,*pb=&b,*ps=&s;
    printf("请输入 a,b 的值:");
    scanf("%d,%d",pa,pb);
    *ps=(*pa+*pb)*(*pa-*pb);
    printf("s=%d\n",*ps);
    return 0;
}
```

思考

能否增加两个变量分别记录两数之和与之差？如果行，将程序改写，并说出哪种写法好及原因。

2．设计性实验

（1）编写程序，输入一个小写字母，改用大写字母输出。

实验提示：

① 声明两个字符型变量，一个存放小写字母，一个存放大写字母。

② 输入小写字母。

③ 将小写字母转换成大写字母。

④ 输出大写字母。

其中第③步关于大小写字母转换的问题，可以利用字符的 ASCII 码，大写字母 A 的 ASCII 码值为 65，小写字母 a 的 ASCII 码值为 97，也就是说，大小写字母的 ASCII 码的差值为 32，运行结果如图 3-20 所示。

图 3-20　小写字母转成大写字母的运行结果

（2）编写程序，输入两个整数分别赋给变量 a 和 b，然后交换两个变量的值再输出。

实验提示：

① 声明两个整型变量 a 和 b。

② 输入它们的值。

③ 交换两个变量的值，也就是让 a 获得 b 的值，b 获得 a 的值。

④ 输出变量 a 和 b。

其中第③步交换两个变量值的问题怎么解决呢？就像有两杯水，现在要把两个杯子中的水交换，那么大家很自然想到利用第三个杯子做中介。同样的道理，要交换两个变量的值，我们也需要另外一个变量作为中介。所以在第①步我们需要声明三个变量，运行结果如图 3-21 所示。

3．提高实验

（1）编写程序，输入存款金额、存款年限和存期的年利率，然后输出到期后的金额。到期金额=存款金额*（1+存期的年利率）^{存款年限}，运行结果如图 3-22 所示。

图 3-21　交换两个变量的运行结果

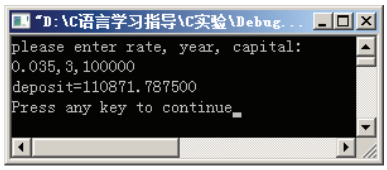


图 3-22　存款到期金额的运行结果

实验提示：

① 声明存款金额、存款年限、存期的年利率、到期后的金额 4 个变量。

② 输入需要的值。

③ 利用公式计算，其中求次方需要用到库函数 pow(x,n)。

④ 输出到期金额。

（2）编写程序，输入长方形的两边长，求它的周长和面积，运行结果如图 3-23 所示。

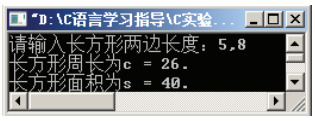


图 3-23　求长方形的周长和面积的运行结果

实验提示：

① 声明变量。

② 输入需要的值。

③ 利用公式计算周长和面积。

④ 输出周长和面积。

实验项目 4　分支结构程序设计方法

一、实验目的

（1）掌握 C 程序中 if 语句的格式及使用方法。

（2）掌握 switch 语句的格式和使用方法。

（3）学会使用分支结构编写简单的 C 语言程序。

（4）理解分支结构嵌套的格式和使用方法。

二、实验要求

（1）掌握 if 和 switch 语句的不同书写格式。

（2）熟悉利用 if 和 switch 语句编写简单的分支程序。

（3）明确分支条件的书写方法。

三、实验内容

1．验证性实验

（1）输入整数 a、b，若 a^2+b^2 大于 100，则输出 a^2+b^2 百位以上的数字，否则输出两数之和，运行结果如图 3-24 所示。

图 3-24　根据 a^2+b^2 是否大于 100 输出不同值的运行结果

实验提示：

分支选择处理的条件是 $a^2+b^2>100$。如果大于 100，就取 a^2+b^2 百位以上的数字，如果小于等于 100，就输出 a+b 的值。其中求百位以上的数字，可以根据两个整数相除仍然为整数的原则，使用$(a^2+b^2)/100$。

① 首先定义三个整型变量 a，b，y，其中 y 存储需要输出的值。

② 使用 if…else 两分支结构编写代码，其中分支条件为 $a^2+b^2>100$。

③ 输出 y 的值。

程序参考代码如下：

```
#include<stdio.h>
int main(){
    int a, b, y;
    printf("enter a, b:");
    scanf("%d, %d",&a, &b);
    if((a*a+b*b)>100)              /*第 6 行*/
        y=(a*a+b*b)/100;           /*第 7 行*/
    else
        y=a+b;                     /*第 9 行*/
    printf("y=%d\n",y);
    return 0;
}
```

思考

① 程序中第 7 行语句起什么作用？是否可以使用 y=(a^a+b^b)/100 代替？

② 能够使用 if 的单分支语句实现上面的程序吗？例如：第 6 行至第 9 行使用下面的语句代替。

```
y=a+b;
```

```
if((a*a+b*b)>100)
  y=(a*a+b*b)/100;
```

③ 分支结构可以用什么运算符实现？请改写这个程序。

（2）从键盘输入 3 个浮点数 a、b、c 的值，按从小到大的顺序依次输出，输出时要求保留小数点后两位，运行结果如图 3-25 所示。

图 3-25　三个数按顺序输出的运行结果

实验提示：

① 定义三个 double 型变量，并从键盘输入。

② 首次比较 a 和 b 的大小，如果 a 大于 b 则将 a 和 b 的值交换，使得 a 小于 b。

③ 类似地比较 a 和 c 的大小，如果 a 大于 c 则将 a 和 c 的值交换，使得 a 小于 c，这样 a 为最小的数。

④ 再比较 b 和 c 的大小，如果 b 大于 c 则将 b 和 c 的值交换，这样 c 最大。

⑤ 最后输出 a、b、c 的值。

程序参考代码如下：

```
#include<stdio.h>
int main(){
    double a, b, c, t;
    printf("输入 a, b, c: ");
    scanf("%lf, %lf, %lf", &a, &b, &c);
    if(a>b){t=a; a=b; b=t;}                          /*第 6 行*/
    if(a>c){t=a; a=c; c=t;}
    if(b>c){t=b; b=c; c=t;}                          /*第 8 行*/
    printf("%0.2lf, %0.2lf, %0.2lf\n", a, b, c);    /*第 9 行*/
    return 0;
}
```

思考

① 程序第 6 行语句起什么作用？

② 第 6 行至第 8 行的程序段实现了什么功能？举例说明其实现步骤。

③ 第 9 行的输出为什么要使用%0.2lf 的输出格式？

（3）分析题。

实验提示：

① 分析下面的程序，得出输出结果。

② 输入程序，运行验证。

③ 画出流程图。

```
#include<stdio.h>
int main(){
    int s=0,k;
    scanf("k=%d\n",k);
    switch(k){
        case 1:
        case 4:
        case 7: s++; break;
        case 2:
        case 3:
        case 6: break;
        case 0:
        case 5: s+=2; break;
    }
    printf("s=%d\n",s);
    return 0;
}
```

思考

break 语句加与不加对程序的输出结果有何影响？

2．设计性实验

（1）有下面的方程式，输入 x 的值，输出 y 的值。

$$y=\begin{cases} x+1 & x<-2 \\ x^2 & -2\leqslant x\leqslant 2 \\ \sqrt{x} & x>2 \end{cases}$$

实验提示：

① 定义变量 x，y。

② 输入 x 的值。

③ 根据 x 的值，进入相应的分支做运算。

④ 输出 y 的值。

程序运行结果如图 3-26 所示。

图 3-26　数学方程式的运行结果

（2）编写一个程序，根据用户输入的三角形的三边 a,b,c 的长度判定能否构成三角形，如果能构成就输出三角形的面积 area，如果不能构成就输出“不能构成三角形！”。

实验提示：

① 确定组成三角形的条件为：任意两边之和大于第三边。

② 三角形面积的计算公式为：area=$\sqrt{s(s-a)(s-b)(s-c)}$，其中，a、b、c 为三角形的三边长，s=(a+b+c)/2。

运行结果如图 3-27 所示。

（3）已知银行整存整取存款不同期限的月息利率分别为：月息利率＝0.33%期限为 1 年，0.36%期限为 2 年，0.39% 期限为 3 年，0.45% 期限为 5 年，0.54%期限为 8 年。输入存款的本金和年限，请使用 switch 语句编程实现求到期时能从银行得到的利息与本金的合计。

利息的计算公式为：利息＝本金×月息利率×12×存款年限，运行结果如图 3-28 所示。

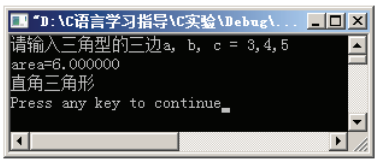


图 3-27　三角形面积的运行结果

图 3-28　银行存款的运行结果

实验提示：

① 输出结果是利息与本金之和，那么只要求出利息值就行了。

② 从利息计算公式中可以看出，存款年限和月利息率是息息相关的，其中存款年限为整数，很自然地就想到利用年限值得到相应的 case 分支语句。

③ 假设存款年限用变量 year 表示，那么就可以得到下面的结构：

```
switch(year){
    case 1:
    case 2:
    case 3:
    case 5:
    case 8:
    default:
}
```

（4）输入一个小写字母，将其转换成大写字母，再输出该字母的前序字母、该字母、该字母的后序字母，例如：输入 g，则输出 FGH；输入 a，则输出 ZAB；输入 M，则输出 LMN；输入 Z，则输出 YZA。下面给出了程序的部分代码，请把它补充完整，运行结果如图 3-29 所示。

图 3-29　字母转换的运行结果

实验提示：

① 输入字母。

② 判断输入的字母是否为小写字母。

③ 如果是小写字母，将其转换为大写字母；对于首位两个字母 A 和 Z 需要单独处理。

④ 输出。

```
#include<stdio.h>
int main(){
    char ch,c1,c2;
    printf("Enter a character:");
    ch=getchar();
    /*---请填上适当的语句------*/
    {ch-=32;
    c1=ch-1;
    c2=ch+1;
    /*---请填上适当的语句------*/
    c1=ch+25;
    /*---请填上适当的语句------*/
    c2=ch-25;
    putchar(c1);
    putchar(ch);
    putchar(c2);}
    else
    printf("您输入的不是小写字母！");
    putchar('\n');
    return 0;
}
```

3．提高实验

（1）编写一个程序实现如下功能：输入一个正整数，判断它能否被 3，5，7 整除，并输出以下信息之一：

① 能同时被 3，5，7 整除；

② 能被其中两数（要指出哪两个）整除；

③ 能被其中一个数（要指出哪一个）整除；

④ 不能被 3，5，7 任一个整除。

运行结果示例如图 3-30 所示。

（a）　　（b）　　（c）

图 3-30　判断能否整除的运行结果示例

实验提示：

① 利用取余运算可以判断一个数能否被另一个数整除。

② 此题就是要判断输入的数对 3、5、7 取余，余数是否同时为 0。

（2）编写程序输入月份和日期，给出对应的星座。下面是星座计算方法：3 月 21 日～4 月 20 日为白羊座，4 月 21 日～5 月 20 日为金牛座，5 月 21 日～6 月 20 日为双子座，6 月 21 日～7 月 22 日为巨蟹座，7 月 23 日～8 月 22 日为狮子座，8 月 23 日～9 月 22 日为处女座，9 月 23 日～10 月 22 日为天秤座，10 月 23 日～11 月 22 日为天蝎座，11 月 23 日～12 月 22 日为人马座，12 月 23 日～1 月 20 日为摩羯座，1 月 21 日～2 月 20 日为宝瓶座，2 月 21 日～3 月 20 日为双鱼座。

运行结果示例如图 3-31 所示。

（a）　　（b）　　（c）

图 3-31　计算星座的运行结果示例

实验提示：

① 此题为多分支嵌套结构，先根据月份值进行分支选择，再按照日期值进行分支选择。

② 因为月份只有 1 到 12 这 12 个正整数，因此用 case 语句实现会更简洁。

实验项目 5　循环结构程序设计方法

一、实验目的

（1）掌握三种循环结构：while、do…while、for 的区别与联系，以及它们之间相互转换的方法，并能正确使用它们。

（2）掌握与循环语句相关的 break 语句和 continue 语句的使用方法。

（3）学会使用 C 语言循环结构编写程序。

（4）理解循环结构嵌套的格式和使用方法。

二、实验要求

（1）掌握 while、do…while 和 for 语句的不同书写格式。

（2）熟悉利用 while、do…while 和 for 语句编写简单的循环程序。

（3）掌握 break 语句和 continue 语句的区别。

三、实验内容

1．验证性实验

（1）3025 这个数具有一种独特的性质：将它平分为二段 30 和 25，相加后求平方即

$(30+25)^2$，恰好等于 3025 本身。求出具有这样性质的全部四位数，运行结果如图 3-32 所示。

图 3-32　3025 性质的四位数的运行结果

实验提示：

具有这种性质的四位数没有分布规律，可采用穷举法，从中筛选出符合条件的数。

① 定义变量 n，代表这样的四位数。

② 四位数从 1000 变化到 9999，对其中的每一个数都判断下是否符合这种特性。如果符合就输出，不符合就继续判断下一个数。

程序参考代码如下：

```
#include<stdio.h>
int main(){
    int n,a,b;
    for(n=1000;n<=9999;n++){
        a=n/100;
        b=n%100;
        if((a+b)*(a+b)==n)
        printf("%d ",n);
    }printf("\n");
    return 0;
}
```

思考

① 如果求 10000 以内的水仙花数，怎么做？水仙花数满足条件是这个数等于其个位数、十位数、百位数的立方和。

② 可不可以总结出类似题目的解题规律呢？

（2）输入一行字符，分别统计出其中的英文字母、空格、数字和其他字符的个数，运行结果如图 3-33 所示。

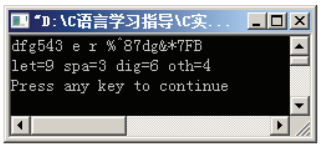


图 3-33　统计字符个数的运行结果

实验提示：

可以使用 while 循环语句编程，输入一个字符，就判定一下这个字符是英文字母、空格、数字或其他字符，直到输入“\n”时结束循环。

程序参考代码如下：

```
#include<stdio.h>
int main(){
    char c;
    int let=0,spa=0,dig=0,oth=0;
    while((c=getchar())!='\n'){
        if(c>='a'&&c<='z'||c>='A'&&c<='Z')
            let++;
        else if(c==' ')
            spa++;
        else if(c>='0'&&c<='9')
            dig++;
        else oth++;
    }
    printf("let=%d spa=%d dig=%d oth=%d\n",let,spa,dig,oth);
    return 0;
}
```

思考

这道题还可以采取其他的写法吗？尝试所有的可能。

（3）输入一组正整数，输入−1 表示输入结束，分别统计其中奇数和偶数的个数，运行结果如图 3-34 所示。

图 3-34　统计奇偶个数的运行结果

实验提示：

可设一整型变量 x，表示循环输入的整数的值，若 x%2==0 即为偶数，否则是奇数。循环条件可在 while 语句后的括号中出现，也可以用 break 语句控制。

① 声明变量 xo、xj、x，xo 表示偶数个数，xj 表示奇数个数，x 表示输入的正整数。

② 判断 x 值是否为−1，不是转到第③步，若是转到第④步。

③ 判断 x 值是奇数还是偶数，是奇数，xj++；是偶数，xo++。然后输入下一个 x。

④ 输出 xo 和 xj 的值。

程序代码参考一：

```
#include<stdio.h>
int main(){
    int x,xo=0,xj=0;     /*将偶数个数 xo 与奇数个数 xj 赋 0 值*/
    scanf("%d",&x);      /*先输入一个正整数*/
```

```
    /*输入不是-1 时循环*/
    while(x!=-1) {
        if(x%2==0)xo=xo+1 ;
        else xj=xj+1 ;
        scanf("%d", &x);
    }/*循环输入其余正整数*/
    printf("xo:%d,xj:%d\n",xo,xj);
    return 0;
}
```

程序代码参考二：

```
#include<stdio.h>
int main(){
    int x,xo=0,xj=0 ;
    /*在循环内部用 break 语句控制循环条件*/
    while(1){
        scanf("%d",&x);
        if(x==-1)break ;
        if(x%2==0)xo=xo+1 ;
        else xj=xj+1 ;
    }
    printf("xo:%d,xj:%d",xo,xj);
    return 0;
}
```

思考

可以用 do…while、for 循环来实现吗？

（4）编写一个程序，计算并输出 a+aa+aaa+…+aaa…a(n 个 a)的值。其中，n 由键盘输入，例如：a=2，n=3 时是求 2+22+222 之和，运行结果如图 3-35 所示。

（a）　　（b）

图 3-35　a+aa+aaa 的运行结果

实验提示：

① 这是个多项式求和问题，所以只要找出通项，通过循环，将其累加就可以了。但这个数列通项的变化规律不是简单的递增或递减，需要找出前后两项求和 a 之间的关系。

② 假设 t 为通项式，将 t 设定初始值 t=a，第二项为 aa=t*10+a，将该数值赋给 t 为求第三项做准备即 t=t*10+a，那么第三项 aaa=t*10+a。因此这个数列的变化规律就是 t=t*10+a，

t 初值为 a。

程序参考代码如下：

```
#include<stdio.h>
int main(){
    int n,a,t,i,s=0;
    printf("请输入 a 和 n 的值:\n");
    scanf("%d%d",&a,&n);
    t=a;
    for(i=1;i<=n;i++){
        s+=t;
        t=t*10+a;
    }printf("s=%d\n",s);
    return 0;
}
```

思考

大家可以总结出求数列之和的通用方法吗？

（5）使用迭代法求 $\sqrt[3]{a}$,求立方根的迭代公式为 $x_{i+1}=\frac{2}{3}x_i+\frac{a}{3x_i^2}$。假定 x 的初值为 a，迭代到$|x_{i+1}-x_i|<\varepsilon=10^{-5}$时为止。求当 a=27 的值，并使用系统函数 pow(a, 1.0/3)加以验证，运行结果如图 3-36 所示。

图 3-36　求立方根的运行结果

实验提示：

① 迭代法又称为递推法，其基本方法是给定一个初值，利用迭代公式反复迭代计算，直到前后两个值的绝对值小于给定的某个数值为止。

② 输入 a 的值，并让 x 的初值为 a。

③ 通过循环反复获得 x_i 的值，直到$|x_{i+1}-x_i|<\varepsilon=10^{-5}$为止。

程序代码参考：

```
#include<stdio.h>
#include<math.h>
int main(){
    float a, x, x1, e;
    printf("a =");
    scanf("%f", &a);
```

```
    x = a;
    while(1){
        x1=2*x/3.0 + a/(3.0*x*x);
        e=fabs(x1-x);
        if(e<1e-4)break;
        else x=x1;
    }
    printf("x=%2.5f\n", x1);
    printf("pow(%0.2f, 1.0/3)=%2.5f \n", a, pow(a, 1.0/3));
    return 0;
}
```

思考

这道题可以改用 for 循环和 do…while 循环来实现吗？

2．设计性实验

（1）采用 do…while 语句编程计算 π 的近似值。要求：采用公式$\pi^2/6 = 1/1^2 + 1/2^2 + 1/3^2+\cdots$实现，直到最后一项小于 10^{-12}，定义 double 变量 sum 存储累加和，term 变量存累加项，pi 记录最终结果，循环变量为 i，运行结果如图 3-37 所示。

实验提示：

这是数列求和问题，要通过循环解决，只要找出通项，随着循环的运行，将数列中的每一项都加上。

（2）编写一个程序，求出 200 到 800 之间满足下面两个条件的数：①每位上数字之积为 42；②每位上数字之和为 12，并统计输出满足条件的数的个数，运行结果如图 3-38 所示。

图 3-37　计算 π 的运行结果

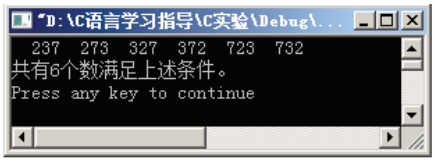


图 3-38　满足条件的数的个数的运行结果

实验提示：

用循环变量 n 记录数据从 200 变到 800，然后判断每个数 n 是否满足上面两个条件。

（3）设计一个程序，输入一个日期，要求算出这一天是本年的第几天，运行结果如图 3-39 所示。

图 3-39　某天是这年的第几天的运行结果

实验提示：

① 要算出某天是当年的第几天，应该将当年中本月之前所有月的天数相加，再加上本月至此的天数。

② 但这里有一个闰年问题，2 月是一个特殊月，闰年的 2 月有 29 天，非闰年的 2 月只有 28 天。那么可以先使用循环结构累加天数，然后使用分支结构计算三种类型的月份的天数，如 3 月为 31 天。

③ 判断某年是闰年的条件是：该年号能被 4 整除但不能被 100 整除，或者能被 400 整除。例如，1996、2000 是闰年，2014 不是闰年。

（4）编写程序，输入两个不同的正整数，求其最大公约数和最小公倍数，运行结果如图 3-40 所示。

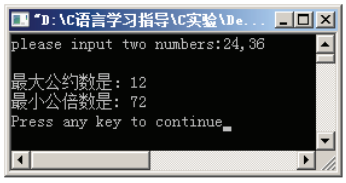


图 3-40　求最大公约数和最小公倍数的运行结果

实验提示：

利用辗转相除法求两个整数 a 和 b 的最大公约数的算法如下：

① a%b 得余数 c。

② 若 c=0，则 b 即为两数的最大公约数。

③ 若 c≠0，则 a=b，b=c，再回去执行①。

例如求 27 和 15 的最大公约数的过程为：27/15 余 12，15/12 余 3，12/3 余 0，因此，3 即为最大公约数。

求最小公倍数算法：最小公倍数=两整数的乘积/最大公约数。

3．提高实验

（1）某人摘下一些桃子，卖掉一半，又吃了一只；第二天卖掉剩下的一半，又吃了一只；第三天、第四天、第五天都如此处理，第六天一看，发现就剩下一只桃子了。编写一个程序，求此人共摘了多少只桃子，运行结果如图 3-41 所示。

实验提示：

采用迭代法。

（2）编写一个程序，功能是：输入一个五位正整数，将它反向输出。例如输入 12345，输出应为 54321，运行结果如图 3-42 所示。

图 3-41　猴子吃桃的运行结果

图 3-42　逆序输出的运行结果

实验提示：

① 此题需要逆序取得输入的正整数的每一位上的值，依次输出。

② 可以采用对 10 取余的操作求得个位上的数值。

③ 将输入的正整数缩小至原来的 1/10 获得一个新的正整数，再进行对 10 取余，直到这个正整数变为 0。

实验项目 6　分支与循环结构综合程序设计

一、实验目的

（1）掌握 C 语言中的程序控制结构。

（2）掌握常见的循环结构和分支结构混合使用的程序设计方法。

（3）学会使用综合程序设计方法实现一些典型的算法以及一些实际问题。

二、实验要求

（1）熟悉综合的程序设计方法，会编写稍微复杂的程序。

（2）能够掌握程序的运行步骤，尤其是循环结构程序。

（3）能够在程序中正确使用 break 和 continue 语句。

三、实验内容

1．验证性实验

（1）使用 C 语言循环结构打印图 3-43 所示的图形。

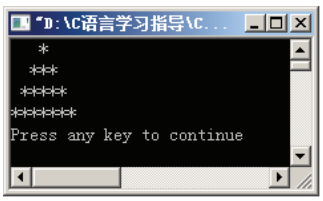


图 3-43　输出*的运行结果

实验提示：

① 分析图形，从中找出行数、空格数、星号数间的关系，如表 3-1 所示。

表 3-1　行数、空格数、星号数间的关系

行　数	空 格 数	星 号 数
1	3	1
2	2	3
3	1	5
4	0	7
i	4−i	2*i−1

② 可用双重循环控制整个图案的输出。若用循环变量 i、j 分别控制外层、内层循环，则 i 的取值从 1 到 4，表示行数，在每行中要先输出 4–i 个空格和 2*i–1 个星号，然后换行。

程序代码参考：

```
#include<stdio.h>
int main(){
    int i,j ;/*定义循环控制变量 */
    for(i=1;i<=4;i++){
        for(j=1;j<=4-i;j++)
        printf(" ");
        /* 输出 4-i 个空格 */
        for(j=1;j<=2*i-1;j++)
            printf("*");
        printf("\n");
    }
    return 0;
}
```

思考

① 如果这个图形的行数用变量 n 来表示，可以从键盘输入的话，那这样输出的图形就很灵活了，试实现它。

② 如果输出图 3-44 至图 3-47 几种图形，该怎样修改程序呢？

```
*******
 *****
  ***
   *
```

图 3-44 *图 1

```
   *
  * * *
 * * * * *
* * * * * * *
 * * * * *
  * * *
   *
```

图 3-45 *图 2

```
      1
    1 2 1
  1 2 3 2 1
1 2 3 4 3 2 1
  1 2 3 4 5 4 3 2 1
1 2 3 4 5 6 5 4 3 2 1
```

图 3-46 数字图 1

```
1 0 0 0 1
0 1 0 1 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1
```

图 3-47 数字图 2

（2）分析题。

实验提示：

① 分析下面的程序，得出输出结果。

② 输入程序，运行验证。

③ 画出流程图。

```
#include<stdio.h>
int main(){
    int i,j,x=0;
    for(i=0;i<2;i++){
        x++;
        for(j=0;j<=3;j++){
            if(j%2) continue;
            x++;
```

```
            }
            x++;
        }
        printf("x=%d\n",x);
        return 0;
}
```

思考

continue 语句的作用是什么？

（3）分析题。

实验提示：

① 分析下面的程序，得出输出结果。

② 输入程序，运行验证。

③ 画出流程图。

```
#include<stdio.h>
int main(){
    int i=5;
    do{
        switch(i%2){
          case 4: i--; break;
          case 6: i--; continue;
        }
        i--; i--;
        printf("%d ",i);
    }while(i>0);
    return 0;
}
```

思考

可以将这个程序改用 for 循环和 while 循环来实现吗？

2．设计性实验

（1）编写一个程序，求 e 的值，其中：$e \approx 1+\frac{1}{1!}+\frac{1}{2!}+\cdots+\frac{1}{n!}$，要求累加项 1/n!，精确到 10^{-12}，运行结果如图 3-48 所示。

图 3-48　求 e 的值的运行结果

实验提示：

① 求阶乘需要一个循环来实现，累加又需要一个循环，因此需要使用二重循环来实现。

② 采用 sum 存储累加值，fac 存储阶乘，外循环变量为 i，内循环变量为 j，累加项为

item，并给 sum 赋初值 1，循环的结束条件为最后一项小于 10^{-12}。

（2）设计一个程序，输出 100～400 之间的所有素数，每行输出 5 个，运行结果如图 3-49 所示。

实验提示：

① 这道题要对 100～400 之间的每一个数判断其是不是素数，如果是就输出它。很显然，这需要一个循环。判断一个数是不是素数时，又需要一个循环，所以这是一个二重循环的题目。

② 用变量 n 记录数据从 100 变到 400，这形成外循环，循环体是判断 n 是不是素数，因为需要每行输出 5 个，所以当 n 是素数时，不仅要输出 n，还要进行计数。

③ 每行输出 5 个数，也就是输出 5 个素数后换行，输出 20 个素数换行，30 个换行，那么 10，20，30 等这些数据具有什么特性呢？把它总结出来，便可实现每行输出 10 个。

（3）验证哥德巴赫猜想：任何一个大于 6 的偶数都可以表示为两个素数之和。例如：6=3+3，8=3+5，…，100=3+97。要求将 6～100 的偶数都表示成两个素数之和，每行输出 3 个等式，运行结果如图 3-50 所示。

图 3-49　输出素数的运行结果

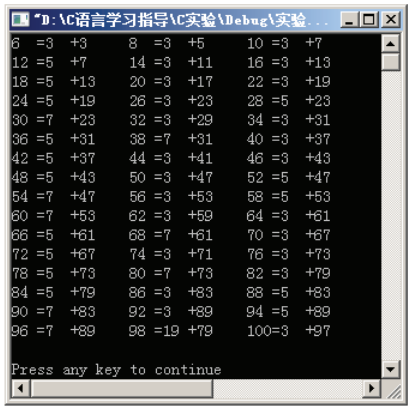


图 3-50　哥德巴赫猜想的运行结果

实验提示：

由于需要在 6～100 之间找出所有偶数满足条件的两个素数，故需要一个最外层的循环；判断素数需要一个最内层的循环；而验证猜想需要一个循环。故整个程序需要三重循环。判断 p 是否为素数可以采用 p 整除 i=2～sqrt(p)范围中的每一个数来实现。

将一个偶数分解为两个素数的算法如下。

① 假设 num 为偶数，先求出 3～num/2 中的一个素数，假定为 p1。

② 计算 p2=num−p1。

③ 判断 p2 是否为素数，如果是，则 num =p1 + p2（p1 和 p2 都是素数），完成分解。

④ 若 p2 不是素数，则继续在 3～num/2 中找下一个素数 p1。重复②～④步，直到找到满足条件的两个素数为止。

（4）将一个正整数分解质因数。例如：输入 60，打印出 60=2*2*3*5，运行结果如图 3-51 所示。

实验提示：

对 n 进行分解质因数，应先找到一个最小的质数 k，然后按下述步骤完成：

① 如果这个质数恰等于 n，则说明分解质因数的过程已经结束，打印出即可。

② 如果 n<>k，但 n 能被 k 整除，则应打印出 k 的值，并用 n 除以 k 的商,作为新的正整数 n，重复执行第①步。

③ 如果 n 不能被 k 整除，则用 k+1 作为 k 的值,重复执行第①步。

3．提高实验

（1）用 1、2、3、4 这 4 个数字，能组成多少个互不相同且无重复数字的三位数？分别是多少？运行结果如图 3-52 所示。

图 3-51　分解质因数的运行结果

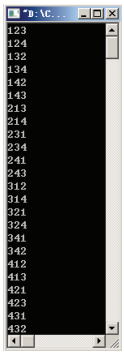


图 3-52　组成三位数的运行结果

实验提示：

① 可填在百位、十位、个位的数字都是 1、2、3、4，因此使用三重循环结构。

② 组成所有的排列后再去掉不满足条件的排列。

（2）编写一个程序，求满足如下条件的最大的 n：$1^2+2^2+3^2+\cdots+n^2 \leqslant 1000$，运行结果如图 3-53 所示。

实验提示：

① 定义 n 这个变量来计算 n^2，定义 sum 这个变量来作为 $1^2+2^2+3^2+\cdots+n^2$ 的和。

② 利用 while 循环和 n++来计算 sum 的值，执行循环的条件是 sum <= 1000。当 sum>1000 时，停止执行循环，再输出此时 n−1 的值。

（3）输出杨辉三角形。杨辉三角形如图 3-54 所示，满足如下特点：①端点的数为 1。②其余每个数字等于它上一行的左右两个数字之和。③第 n 行的第 m 个数可表示为 C(n-1,m-1)。④组合数计算方法：C(n,m)=n!/[m!(n-m)!] 。

图 3-53　求满足最大的 n 的运行结果

图 3-54　杨辉三角形的运行结果

实验提示：

① 找到图形中数值得来的规律。

② 每一行的输出都是先输出空格，再输出数字。

③ 先不考虑空格，将所有的数字靠左边输出。

④ 将空格加上输出。

实验项目 7　一维数组程序设计

一、实验目的

（1）掌握一维数组的定义、赋值和输入输出方法。

（2）理解一维数组定义时各部分所代表的意义，并能正确引用该数组元素。

（3）熟悉和掌握与一维数组有关的常用算法，如查找、排序等。

二、实验要求

（1）理解 C 语言中数组的作用及应用特点。

（2）掌握一维数组元素的引用及有序性特点。

（3）理解一维数组的实际应用。

三、实验内容

1．验证性实验

（1）定义一维整型数组 a（包含 5 个元素），通过键盘输入方式对其各元素进行赋值，并求出这些元素中的最大值。

实验提示：

① 使用语句 int a[5];定义数组 a，通过对“scanf("%d", &a[i]);”语句的循环来输入各元素的值，再通过依次比较来求取元素的极值，最后输出相应的结果信息，示例结果如图 3-55 所示。

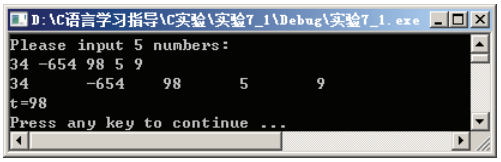


图 3-55　程序运行结果一

② 将程序中 t<a[i]改为 t>a[i]，再输入相同的数据并观察程序的执行结果。

③ 注意循环变量与下标变量的结合。

程序参考代码如下：

```
#include<stdio.h>
int main()
```

```
{
    int i, a[5], t;
    printf("Please input 5 numbers:\n" );
    for(i=0; i<5; i++ )
        scanf("%d", &a[i] );
    t=a[0];
    for(i=0; i<5; i++ )
        if(t<a[i])
            t=a[i];
    for(i=0; i<5; i++ )
        printf("%d\t", a[i] );
    printf("\nt=%d\n", t );
    return 0;
}
```

思考

① 将程序中某个循环体的 i=0 语句改为 i=1，则应该如何修改该循环体的其他部分，使得整个循环的意义不变？

② 若将循环中的 i<5 改为 i<=5，观察发生的情况，为何会发生该情况？

（2）输入 10 个学生成绩，统计并输出不及格和及格的学生人数。

实验提示：

① 先定义相关变量，同时定义两个起记数作用的变量（设为 m 和 n），并初始化为 0，分别用来统计不及格和及格的人数。

② 用循环语句输入 10 个成绩到数组中。

③ 使用循环依次比较并统计不及格和及格的人数。

④ 输出相应的统计结果。

程序参考代码如下：

```
#include<stdio.h>
int main()
{
    int score[10], m=0, n=0, i;
    printf("Input 10 scores:\n");
    for(i=0; i<10; i++ )
        scanf("%d", &score[i]);
    for(i=0; i<10; i++)
    {
        if(score[i] < 60)
            m++;
        else
            n++;
    }
    printf("Failed: %d.\nNot Failed:%d.\n", m, n);
```

```
    return 0;
}
```

程序运行结果如图 3-56 所示。

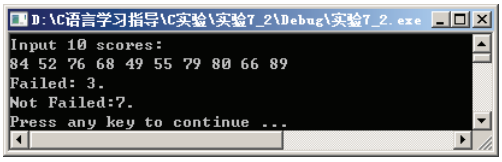


图 3-56　程序运行结果二

思考

① 将程序改成对分数的多个级别（如 60 分到 70 分为及格，70 分到 80 分为良好等）统计，该如何进行？

② 程序中的 if 语句若改为使用“>=”进行比较，该如何进行？

③ 程序中语句 m=0 只写作 m，则会有什么变化？

2．设计性实验

（1）输入 10 个整数到数组 a 中，将它们按逆序存放并输出。

实验提示：

① 用循环语句输入 10 个成绩到数组中。

② 用循环语句完成以下过程：将第 1 个元素与倒数第 1 个元素互换位置，将第 2 个元素与倒数第 2 个元素互换位置，一直到最中间的元素。

③ 使用循环依次输出互换结束后的数组元素，具体实验运行结果可参见图 3-57。

（2）编写程序，根据函数 $y=x^2-8x+\sin x$ 计算 x=1,2,3,…,10 时的函数值 y，并计算和输出这些 y 值中的极小值（保留两位小数）及对应的 x 取值。

实验提示：

① 用循环语句初始化包含 10 个元素的数组 X。

② 用循环语句根据函数表达式计算包含 10 个元素的数组 Y。

③ 使用依次比较并更新当前最小值的方法，通过循环得到数组 Y 中的最小值及对应的 X 中的元素。

④ 输出对应的结果，具体实验运行结果可参见图 3-58。

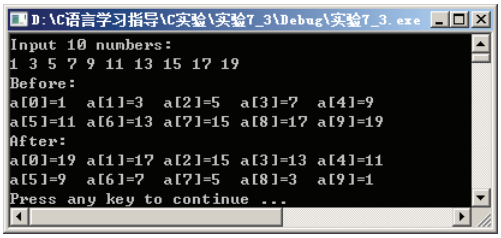


图 3-57　程序运行结果三

图 3-58　程序运行结果四

3．提高实验

（1）输入 10 个数据，删除其中的负数后输出剩下的数据。

实验提示：

先定义一个一维数组用以保存输入的 10 个数据，依次扫描每个数组的元素，并在扫描的同时检测当前元素是否为负数，若是则删除该元素，否则继续扫描并检测下一个元素，其中删除一个元素的过程为：将当前元素后面的所有元素都向前移动一个位置，此时当前元素会被其后面的第 1 个元素所覆盖，而其后面其他元素的相对位置均未发生变化，具体实验运行结果可参考图 3-59。

（2）用一维数组解决以下问题：输入两个各含有 10 个数据（整型）的数据序列，求取它们共有的数据，并对共有数据进行排序后输出。如序列 A 的元素为 2，5，4，1，6，11，10，9，3，7；序列 B 的元素为 9，10，5，15，1，11，13，7，8，3；则输出结果为 1，3，5，7，9，10，11。

实验提示：

将序列 A 中的元素逐个取出，与序列 B 中的每个元素进行比较，若遇到相等的元素，即添加到序列 C，待所有相等的元素均已取出后，再对序列 C 进行升序排序，具体实验运行结果可参见图 3-60。

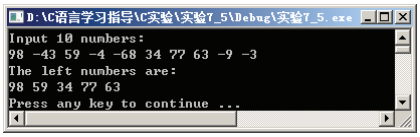


图 3-59　程序运行结果五

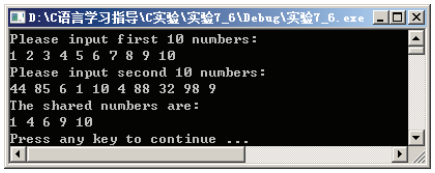


图 3-60　程序运行结果六

实验项目 8　二维数组程序设计

一、实验目的

（1）掌握二维数组的定义、赋值和输入输出的方法。

（2）掌握二维数组元素的引用及其在内存中的存储特点。

（3）掌握与矩阵相关的算法，如矩阵转置、对角线元素求和等。

二、实验要求

（1）理解 C 语言中二维数组与一维数组的异同。

（2）掌握二维数组元素的引用及其规律。

（3）理解二维数组的实际应用。

三、实验内容

1．验证性实验

（1）定义二维整型数组 a（包含 6*6 个元素）来表示二维矩阵，各元素的值按公式

a[i][j]=i*10+j;进行计算，并求出该矩阵中某对角线上的元素之和。

实验提示：

① 使用“int a[6][6];”来定义数组 a，通过对语句“a[i][j]=i*10+j;”的循环来输入各元素的值，再通过依次累加来求取各元素的和，最后输出相应的结果信息，示例结果如图 3-61 所示。

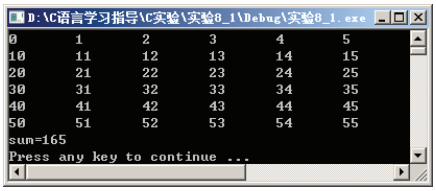


图 3-61　程序运行结果一

② 注意循环变量与下标变量的结合。

③ 将 printf("\n");语句删除再运行程序，通过对照运行结果分析程序的功能。

程序参考代码如下：

```
#include<stdio.h>
int main()
{
    int i,j,sum=0;
    int a[6][6];
    for(i=0; i<6; i++)
        for(j=0; j<6; j++)
            a[i][j] = i*10+j;
    for(i=0; i<6; i++)
    {
        for(j=0; j<6; j++)
            printf("%d\t", a[i][j]);
        printf("\n");
    }
    for(i=0; i<6; i++)
        sum += a[i][6-i-1];
    printf("sum=%d\n", sum);
    return 0;
}
```

思考

① 如果将程序中的 6-i-1 修改为 i，该程序的功能是什么？

② 语句：

```
for(i=0; i<6; i++)
{
   for(j=0; j<6; j++)
       printf("%d\t", a[i][j]);
```

```
    printf("\n");
}
```

和语句：

```
for(j=0; j<6; j++)
{
      for(i=0; i<6; i++)
            printf("%d\t", a[i][j]);
      printf("\n");
}
```

的功能有什么异同点？

（2）补充以下程序，使其可以实现：统计 3×5 的矩阵中正数、负数和零的个数。

实验提示：

① 通过观察可知程序中变量 postive、negtive、zero 代表正数、负数以及零的个数，起记数器的作用。

② 程序的主要步骤分为三步：输入、统计和输出。

③ 首先通过二维数组元素的引用来完善第（1）个填空，通过使用记数器的统计过程完善第（2）空和第（3）空，第（4）空则为选择逻辑，具体实验运行结果可参见图 3-62。

图 3-62　程序运行结果二

```
#include<stdio.h>
int main()
{
    int a[3][5], i, j, postive, negtive, zero;
    printf("Please input a 3*5 matrix:\n");
    for(i=0; i<3; i++)
        for(j=0; j<5; j++)
            scanf("%d",      (1)      );
          (2)        ;
    for(i=0; i<3; i++)
        for(j=0; j<5; j++)
            if(      (3)      )
                postive++;
            else if(a[i][j]<0)
                negtive++;
           (4)
```

```
                zero++;
        printf("正数有%d个，负数有%d个，零有%d个\n", postive, negtive, zero);
        return 0;
}
```

思考

① 能否修改整个程序，使得只用 postive 和 negtive 两个记数器变量就可以完成程序的功能。

② 如果将程序中的以下循环：

```
for(i=0; i<3; i++)
        for(j=0; j<5; j++)
```

修改为：

```
for(i=0; i<5; i++)
        for(j=0; j<3; j++)
```

程序的其他部分将会有什么变化？

2．设计性实验

（1）求 4×4 二维数组中主对角线以下（包括主对角线）的元素之和。

实验提示：

① 用双重循环语句输入 16 个数据到二维数组中。

② 将表示和的变量初值设置为 0。

③ 用循环语句计算主对角线以下（包括主对角线）的元素之和，通过观察可以发现设元素描述为 a[i][j]时，要求和的元素为满足 i>=j 的元素。

④ 使用循环累加完成计算元素之和的过程，并将累加和输出，具体实验运行结果可参见图 3-63。

（2）输入一个 3×4 的矩阵，计算并输出该矩阵的转置矩阵。

实验提示：

① 定义一个 3×4 的二维数组表示原始矩阵 a，一个 4×3 的二维数组表示转置矩阵 b。

② 用双重循环语句输入原始二维数组中的数据。

③ 通过转置矩阵与原始矩阵的关系（b[i][j]=a[j][i]）对转置矩阵进行赋值。

④ 输出得到的转置矩阵；具体实验运行结果可参见图 3-64。

图 3-63　程序运行结果三

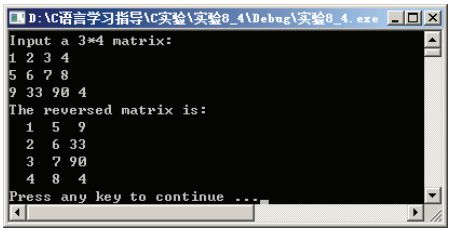


图 3-64　程序运行结果四

3．提高实验

（1）输入 3 个学生 5 门课的成绩，计算并输出每门课的最高分及取得最高分的学生编号。

实验提示：

3 个学生 5 门课程的成绩如表 3-2 所示，可以使用对应结构的 3×5 的二维数组来保存这些成绩，并对该二维数组进行操作来完成实验要求。

表 3-2　二维数组元素和 3×5 个学生成绩数据的对应关系

课程编号 / 学生编号	0	1	2	3	4
0	a[0][0]	a[0][1]	a[0][2]	a[0][3]	a[0][4]
1	a[1][0]	a[1][1]	a[1][2]	a[1][3]	a[1][4]
2	a[2][0]	a[2][1]	a[2][2]	a[2][3]	a[2][4]

可以看出，第 1 门课的最高分即第 1 列元素（即 a[0][0]、a[1][0]和 a[2][0]）的最大值，第 2 门课的最高分即第 2 列元素（即 a[0][1]、a[1][1]和 a[2][1]）的最大值，以此类推，可以发现求第 j（设 j 从 0 开始）门课最高分的过程为计算 a[i][j]的最大值，其中 i 的取值范围为从 0 到 2。最后通过整理可知，要求得所有 5 门课的最大值只需将 j 从 0 取到 4，其中 j=0 时计算的是第 1 门课的最高分，j=1 时计算的是第 2 门课的最高分，以此类推，一直到 j=4 时为第 5 门课的最高分。具体实验运行结果可参见图 3-65。

（2）任意输入两个 4×4 的矩阵 A 和 B，计算并输出 A 与 B 的乘积矩阵。

实验提示：

4×4 矩阵的乘积公式如下：

$$C = A \times B = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{pmatrix} \times \begin{pmatrix} b_{00} & b_{01} & b_{02} & b_{03} \\ b_{10} & b_{11} & b_{12} & b_{13} \\ b_{20} & b_{21} & b_{22} & b_{23} \\ b_{30} & b_{31} & b_{32} & b_{33} \end{pmatrix} = \begin{pmatrix} c_{00} & c_{01} & c_{02} & c_{03} \\ c_{10} & c_{11} & c_{12} & c_{13} \\ c_{20} & c_{21} & c_{22} & c_{23} \\ c_{30} & c_{31} & c_{32} & c_{33} \end{pmatrix}$$

其中，矩阵 C 中的任意一个元素可由以下公式计算得到：

$$c_{ij} = a_{i0} \times b_{0j} + a_{i1} \times b_{1j} + a_{i2} \times b_{2j} + a_{i3} \times b_{3j}$$

具体实验运行结果可参见图 3-66。

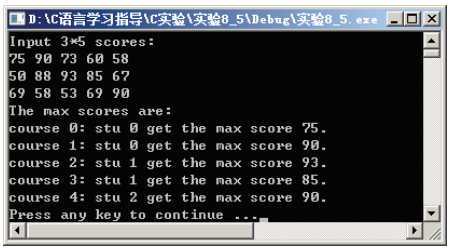


图 3-65　程序运行结果五

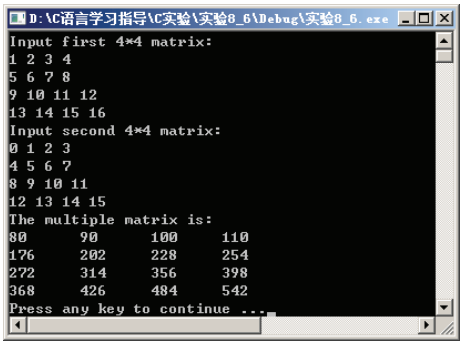


图 3-66　程序运行结果六

实验项目 9　字符数组程序设计

一、实验目的

（1）掌握字符数组的定义和初始化的方法。
（2）掌握字符数组的输入输出方法。
（3）掌握字符串的存取方法和字符串函数的使用方法。

二、实验要求

（1）理解 C 语言中字符数组与一般数组的异同。
（2）掌握字符串在计算机中的表示方式及引用时的注意事项。
（3）理解字符数组的实际应用。

三、实验内容

1．验证性实验

（1）定义字符数组 str，使用语句 getchar()和直接赋值'\0'的方式将字符串输入字符数组 str 中，验证输入字符串的正确性（参考程序一）或统计其中特征字符的个数（参考程序二用来统计元音字母的个数）。

实验提示：

① 通过程序一中的“char str[80];”来定义字符数组 str（程序二中为 char str1[20]），使用循环语句“while((str[i++] = getchar()) != '?');”完成字符串中各字符的输入，再使用“str[i-1]='\0'”语句表示字符串结束。最后使用“puts(str);”函数或 while 循环来完成字符串验证或特征字符个数的统计，对应的运行结果如图 3-67 或图 3-68 所示。

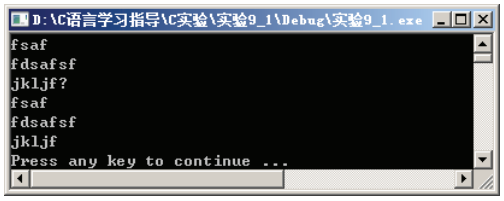


图 3-67　程序一的运行结果

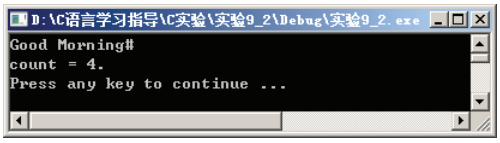


图 3-68　程序二的运行结果

② 注意循环变量与下标变量的结合，以及存储字符串时，字符串结束标识'\0'不能忘记，在此使用“str[i-1]='\0'”语句来添加字符串结束符。

③ 分别将“str[i-1]='\0';”语句删除再运行程序，通过对照运行结果分析程序的功能。

程序一：

```
#include<stdio.h>
int main()
{
    int i=0;
```

```
    char str[80];
    while((str[i++]=getchar())!='?')    /*第 6 行*/
        ;
    str[i-1]='\0';
    puts(str);
    return 0;
}
```

程序二：

```
#include <stdio.h>
int main()
{
    int i=0, j, count=0;
    char str1[20], str2[ ]={ 'a', 'e', 'i', 'o', 'u' };
    while((str1[i++]=getchar())!='#')
        ;
    str1[i-1]='\0';
    i=0;
    while(str1[i])                      /*第 10 行*/
    {
        for(j=0; j<5; j++)
            if(str1[i] == str2[j])
                count++;                /*count 用于计数*/
        i++;
    }
    printf("count = %d.\n", count);
    return 0;
}
```

思 考

① 对照删除“str[i-1]='\0';”后出现的结果想想该语句的功能是什么？

② 程序中以下语句

```
while((str[i++]=getchar())!='?')
        ;
str[i-1]='\0';
```

的功能是什么，'\0'在此处的作用是什么？

（2）输入一个字符串（少于 20 个字符），统计并输出其中数字字符的个数。

实验提示：

① 先输入一个字符到字符数组中。

② 使用循环语句依次扫描每个字符串中的字符，并使用计算器对其中的数字字符进行计数。

③ 循环结束后，输出相应的统计结果。

程序代码如下：

```
#include<stdio.h>
int main()
{
    int i, count=0;
    char str[20];
    printf("Input a string:\n");
    gets( str );
    /*从第 1 个字符开始扫描直到'\0'前的字符*/
    for(i=0; str[i]!='\0';i++)
    {
        if(str[i]>='0'&&str[i]<='9')    /*判定是否为数字字符*/
            count++;
    }
    printf("%s has %d digits.\n", str, count);
    return 0;
}
```

程序运行结果如图 3-69 所示。

图 3-69　程序运行结果一

思考

① 语句“gets(str);”的作用是什么？能否用 scanf 函数来表达该语句？它们有什么异同？

② 循环语句

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

的作用是什么，能否用其他语句来进行替换？

2．设计性实验

（1）输入一个字符串（少于 20 个字符），求取并输出此字符串的长度（要求不用 strlen 函数）。

实验提示：

① 先输入一个字符串到字符数组中。

② 设置记数器初值为零，并使用循环语句依次扫描字符串中的所有字符，每扫描一个计数器加 1，直到遇到字符'\0'时循环结束。

③ 输出相应的计算器的值。具体实验运行结果可参见图 3-70。

（2）输入三句话（每句不超过 80 个字符），要求分别统计其中英文大写字母、小写字母、数字，以及其他字符的个数。

实验提示：

① 先定义一个 3×80 的二维字符数组 a 用于存放要操作的字符串，二维数组中的每一行依次存储输入的一句话。

② 对于其中的每一行数组，可以当作一个字符（一维字符数组）来进行处理，通过循环和计数器结合统计其中的特征字符个数。

③ 在对每一句话处理时，注意循环的次数不是 1～80，而是根据字符串的长度来决定循环次数。

④ 输出相应的计数器统计结果，具体实验运行结果可参见图 3-71。

图 3-70　程序运行结果二

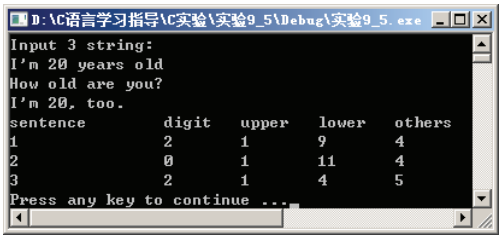


图 3-71　程序运行结果三

3．提高实验

（1）输入一个以回车键结束的字符串（少于 20 个字符），删除其中所有的数字字符，同时将所有的小写字符变为大写，然后再输出该字符串。

实验提示：

先输入一个字符串到字符数组 str 中。首先从原字符串的起始位置开始检测，逐个判断当前字符是否是数字字符，若是数字字符，则删除该字符得到新的字符串，并从新字符串的起始位置重新开始检测；若不是数字字符，则继续检测下一个字符。其中删除字符串中某个字符的过程为：将与该字符相邻的下一个字符开始一直到字符串结束，所有字符依次往前移动一个位置。重复以上过程，直到字符串结束，最后将得到的新字符串用 strupr 函数处理即可得到所要求的字符串。具体实验运行结果可参见图 3-72。

（2）输入一个由数字字符组成的字符串（少于 6 个字符），将其转换为十进制的整型数据并输出该数据的值。

实验提示：

要将数字字符转换为数字，需要利用这些字符所对应的 ASCII 码以及这些 ASCII 码之间的关系。主要通过它们的 ASCII 码与字符'0'的 ASCII 码的差值来计算，如字符'0'要转化为数字 0，可通过'0'-'0'表达式得到，同理，'1'-'0'可得到 1，以此类推。在得到这些数字序列后，再给每个数字赋以权值就能得到最后的数据，如 123 中 1 的权值为 100。赋权值的过程可描述如下（以 123 为例）：s=0,s=s*10+1,s=s*10+2,s=s*10+3。通过整理可结合循环进行程序实现。具体实验运行结果可参见图 3-73。

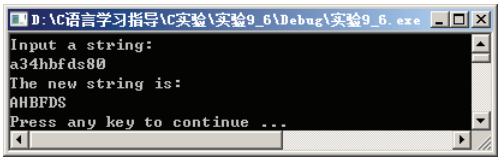


图 3-72　程序运行结果四

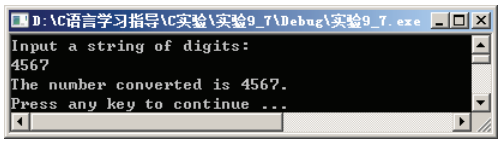


图 3-73　程序运行结果五

实验项目 10　数组与指针程序设计

一、实验目的

（1）了解指针的概念并掌握指针变量的定义。
（2）掌握指向数组的指针及通过指针来访问数组元素的方法。
（3）掌握字符串指针及指向字符串的指针变量。
（4）理解指针数组的概念并掌握其简单的使用方法。
（5）熟悉和掌握指针的概念及其使用方法。

二、实验要求

（1）了解指针及其使用特点。
（2）熟悉用指针对数组元素进行引用。
（3）熟悉用指针对字符串进行简单操作。
（4）了解指针数组及其使用方法。

三、实验内容

1．验证性实验

（1）使用指针变量及其操作来实现两个整数间值的互换，了解指针及与所指变量之间的相互表示、引用等简单操作。

实验提示：

① 通过语句“int *pA, *pB;”定义两个指针变量，使用语句“pA = &a; pB = &b;”使得指针 pA 指向整数 a，pB 指向整数 b。再结合指针运算符“*”来访问整数 a、b，并完成 a、b 间值的互换，参考程序运行结果如图 3-74 所示。

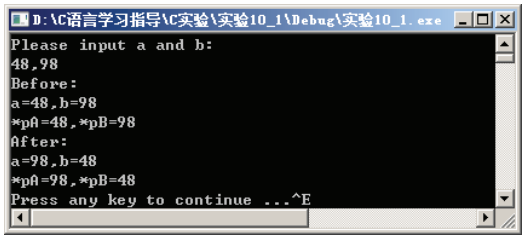


图 3-74　程序运行结果一

② 注意在使用指针变量后，数据输入语句“scanf("%d,%d", pA, pB);”的描述方式。

③ 观察指针相关的操作符*和&的使用方式。

程序参考代码如下：

```
#include<stdio.h>
int main()
{
    int a, b, t;
    int *pA, *pB;
    pA=&a;
    pB=&b;
    printf("Please input a and b:\n");
    scanf("%d,%d", pA, pB);
    printf("Before:\n");
    printf("a=%d,b=%d\n", a, b);
    printf("*pA=%d,*pB=%d\n", *pA, *pB);
    t=*pA;
    *pA=*pB;
    *pB=t;
    printf("After:\n");
    printf("a=%d,b=%d\n", a, b);
    printf("*pA=%d,*pB=%d\n", *pA, *pB);
    return 0;
}
```

思 考

① 语句“scanf("%d,%d", pA, pB);”在未使用指针时通常是什么形式？在此的 pA 相当于以前形式中的什么？

② 语句“pA = &a;”和“pB = &b;”的实际意义是什么？本程序中删除该语句后的结果是什么？为何会出现如此结果？

（2）任意输入 5 个数字数组中，使用指针方式计算这些数字的乘积。

实验提示：

① 定义 5 个整数的数组 a 和一个同类型指针，并将 a 赋值给该指针，再定义一个表示累积的变量，且初始化为 1。

② 使用指针和循环语句相结合输入 5 个数字到数组 a 中。

③ 使用指针和循环语句相结合实现 5 个数字的累积过程。

④ 输出最后的累积结果。

程序参考代码如下：

```
#include<stdio.h>
int main()
{
    int a[ 5 ], *p=a;
```

```
    int i, m=1;
    printf("Please input 5 digits:\n");
    for(i=0; i<5; i++)
        scanf("%d", p + i);
    for(i=0; i<5; i++)
        m *= *(p+i);
    printf("m=%d\n", m);
    return 0;
}
```

程序运行结果如图 3-75 所示。

图 3-75　程序运行结果二

思考

① 第一个 for 语句可否修改为如下形式？

```
for(i=0; i<5; i++, p++)
        scanf("%d", p);
```

如果可以为什么？是否还有其他改写方式？

② 程序中的指针 p 和数组名称 a 之间有什么异同？

③ 语句“m *= *(p + i);”的作用是什么，其中的*(p + i)与 a[i]、p[i]以及*(a + i)是什么关系？

2．设计性实验

（1）任意输入 3×5 的矩阵数据到二维数组中，使用指针方式计算其中每一行的和。

实验提示：

① 先定义相应的二维数组 a 以及 1 个指针，以及表示和的数组 s，并初始化 s 的所有元素为 0。

② 通过二重循环（设循环变量依次为 i, j）结合指针输入数组的所有元素，其中第一层循环体中添加语句“p = a[i];”，第二层循环体为“scanf("%d", p + j);”。

③ 再通过一个二重循环分别计算每一行的和，其中第一层循环体中添加语句“p = a[i];”，第二层循环体为“s[i] += *(p + j);”。

④ 使用一重循环输出所有的和，具体实验运行结果可参见图 3-76。

（2）使用指针方式实现两个字符串的连接，并将连接后的字符串输出（要求不使用 strcat 函数）。

实验提示：

① 先定义两个字符数组，其中第一个字符数组的大小必须能够容纳连接后的字符串，再定义两个字符类型的指针，分别初始化为两个数组的首地址。

② 通过指针与循环结合找到第一个字符串中'\0'的位置。

③ 通过指针与循环结合将第二个字符串中的内容依次复制到第一个字符数组中，直到第二个字符串为'\0'时结束。

④ 将整个字符串的末尾添加'\0'标识后输出整个新字符串，具体实验运行结果可参见图 3-77。

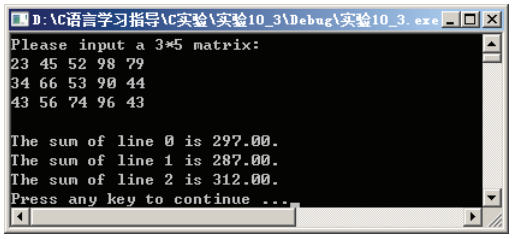


图 3-76　程序运行结果三

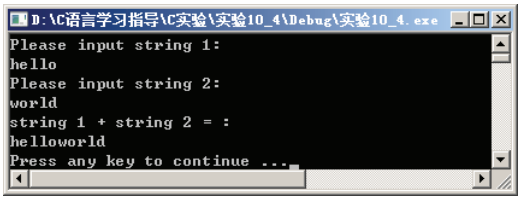


图 3-77　程序运行结果四

3．提高实验

（1）输入 10 个数据，使用指针引用的方式将它们按从小到大的顺序排列。

实验提示：

在进行定义时，可直接定义指针并初始化为数组名称，在输入数据后，可通过简单的指针替换数组名称的方式实现相应的排序算法。同时需要注意数组名称表示地址时为常量，虽然在循环中也不能出现自增和自减操作，具体实验运行结果可参见图 3-78。

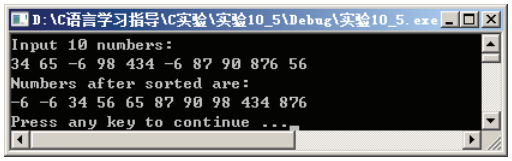


图 3-78　程序运行结果五

（2）使用指针数组完成：输入 3 个单词到数组中，再输入另外一个单词，查找前面的数组中是否出现过该单词，并输出相应的提示信息。

实验提示：

在进行定义时，可直接定义一个二维字符数组和一维的指针数组，并将该指针数组初始化为二维字符数组中每一行的首地址，这样一来，指针数组的每一个元素即代表题目中的一个单词，可通过字符串比较函数 strcmp 进行较简单的单词比较并得到结果，同时输出相应的提示信息，具体实验运行结果可参见图 3-79 和图 3-80。

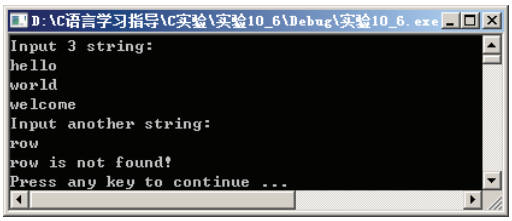


图 3-79　程序运行情况一

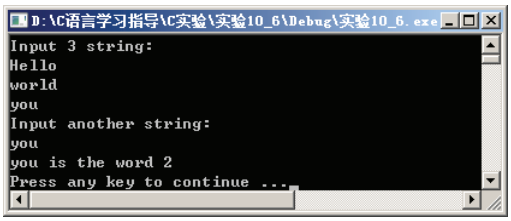


图 3-80　程序运行情况二

实验项目 11　函数的定义和调用

一、实验目的

（1）理解函数的作用及程序中使用函数的意义。
（2）掌握 C 程序中函数定义的一般格式。
（3）区别有返回值函数与无返回值函数调用的不同方式。
（4）掌握函数实参与形参的对应关系以及“值传递”的方式。
（5）区分函数原型声明与函数定义的区别。

二、实验要求

（1）了解标准库函数与用户自定义函数及其使用方法。
（2）熟悉函数的定义和调用方法。
（3）明确函数的实参与形参的对应关系。

三、实验内容

1．验证性实验

（1）定义一个函数 int prime(int n),其功能是判断一个整数是不是素数。当 n 为素数时,函数返回值为 1，否则返回值为 0。在 main 函数中输入一个整数，调用 prime 函数，输出该整数是否为素数的信息。

实验提示：

① 定义函数 int prime(int n)，判断 n 是否为素数，若是，函数返回值为 1，否则返回 0。

② 编写 main 函数，输入一个整数，调用①中的函数 prime，判断此整数是否为素数，并输出结果。

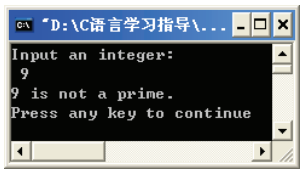
③ 对于多函数程序，可以每个函数独立进行编辑、编译，如果编译有错，可分别修改，最后再合并，这样便于调试。

程序代码参考：

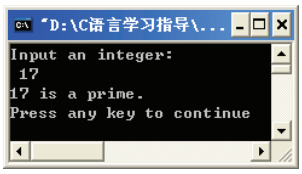
```
#include<stdio.h>
int main()
{
    int prime(int n);                          /*第 5 行*/
    int n;
    printf("Input an integer:\n ");
    scanf("%d",&n);
    if(prime(n))                               /*第 9 行*/
        printf("\n %d is a prime.\n",n);
    else
        printf("%d is not a prime.\n",n);
    return 0;
```

```
}
int prime(int n)
{
    int flag=1,i;
    for(i=2;i<=n/2 && flag;i++)
        if(n%i==0)  flag=0;
    return (flag);
}
```

程序运行结果如图 3-81 所示。



（a）不是素数



（b）是素数

图 3-81　判断某个数是否为素数

思考

① 程序中第 5 行语句“int prime(int n);”起什么作用？是否可以省略？

② 程序第 9 行 if 语句的表达式为什么是 prime(n)？还有其他表示方法吗？

③ 函数 int prime(int n)的定义中，变量 flag 起什么作用？若删除变量 flag，如何修改程序，使其得到相同的结果？

（2）统计一个整数中某数字出现的次数。输入一个正整数 repeat (0<repeat<10)，做 repeat 次如下运算：输入 1 个整数，统计并输出该数中数字 2 的个数。

要求定义并调用函数 int countdigit(long number,int digit)，它的功能是统计整数 number 中数字 digit 的个数。例如：countdigit(10090,0)的函数值是 3；countdigit(34567,2)的函数值是 0。

实验提示：

① 定义函数 int countdigit(long number,int digit)。利用单循环结构，将整数 number 中的数字逐个求出，并与数字 digit 作相等比较，若相等，则统计计数。

② 编写 main 函数，输入 repeat 的值（明确统计操作的次数），做 repeat 次如下操作：

输入 1 个整数，调用函数 int countdigit(long number,int digit)，统计该整数中数字 2 出现的次数，并输出结果。

③ 根据题意，main 函数调用 countdigit 函数时，形式参数 digit 对应的实参固定为 2。

程序代码参考：

```
#include<stdio.h>
int countdigit(long number, int digit)
{
    int num,count=0;
```

```
    number=number<0?-number:number;      /*第 6 行*/
    while(number){                       /*第 7 行*/
        num=number%10;
        if(num==digit) count++;
        number/=10;
    }                                    /* 第 11 行 */
    return count;
}
int main()
{
    int i,repeat;
    int count;
    long in;
    printf("Input repeat:\n");
    scanf("%d",&repeat);
    for(i=1;i<=repeat;i++){
        printf("Input a number:\n");
        scanf("%ld",&in);
        count=countdigit(in,2);
        printf("count=%d\n",count);
    }
    return 0;
}
```

程序运行结果如图 3-82 所示。

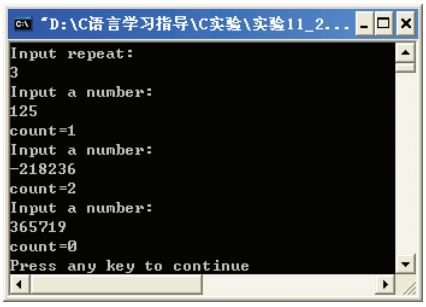


图 3-82　统计一个整数中数字 2 的个数

思考

① 程序第 6 行语句起什么作用？

② 程序第 7 行中 while(number)表示什么意义？还可以写成怎样的形式？

③ 第 7 行至第 11 行的程序段实现了什么功能？举例说明其实现步骤。

2．设计性实验

（1）编写程序，输入三角形的三边长 a、b、c，求三角形面积 area。要求：

① 定义函数 area（a,b,c），给定三角形的三条边长，求三角形的面积。

② 定义 main 函数，输入三角形的边 a、b、c，判定能否构成三角形，若能构成三角形，调用函数 area 求得面积并输出；反之，输出“不能构成三角形!”的提示语句。

例如：

输入：3.1,4.2,5.3

输出：area=6.51

输入：1.1,2.2,3.3

输出：不能构成三角形!

试验提示：

① 三角形面积的计算公式为 area=$\sqrt{s(s-a)(s-b)(s-c)}$，其中，a、b、c 为三角形的三边长，s=$\frac{a+b+c}{2}$。程序中需要使用求平方根的函数 sqrt()。

② 根据题意，函数 area()的返回值类型和其三个形式参数的数据类型定义为实型。

③ main 函数中，根据 area 函数的定义，明确其调用的形式，及其所提供的实际参数。

（2）对实数 x 和正整数 n，编写函数 expon 求 x^n,函数的返回值类型为 double。在 main 函数中输入实数 x 和正整数 n 的值，调用函数 expon 求实数 x 的 n 次方的值，并输出计算结果（保留 3 位小数）（要求不能使用 Pow 函数求 x^n）。

例如：

输入：2.1,3

输出：9.261

输入：2,−3

输出：0.125

实验提示：

① 根据题意，函数 expon 的原型为：double expon(double x,int n)。

② 实数 x 的 n(n>0)次方，即 x*x*…*x(n 个 x 相乘)，利用循环语句即可求得该值。

③ 当 n<0 时，x^n 为 1/(x*x*…*x)(n 个 x 相乘)。

④ 定义函数 main()，输入实数 x 和整数 n 的值,调用函数 expon 求得 x^n，输出结果。

（3）程序填空。计算代数多项式 $1.1+2.2x+3.3x^2+4.4x^3+5.5x^4$ 的值，要求采用循环语句求和（不使用 Pow 函数求 x^n）。

例如：

输入：1.5

输出：54.52

```
#include<stdio.h>
double poly(double);
int main()
{
    double x,y;
    scanf("%lf",&x);
    /*---请填上适当的语句------*/
```

```
    printf("%.2f\n",y);
    return 0;
}
/*---请填上适当的语句------*/
```

实验提示：

① 定义 poly 函数，其功能是求多项式的值。

② 分析多项式的特点，采用循环语句求和。

③ main 函数调用 poly 函数得到求和结果。

3．提高实验

编写程序，N 名裁判给某歌手打分（假定分数都为整数）。评分原则是去掉一个最高分，去掉一个最低分，剩下的分数取平均值为歌手的最终得分。裁判给分的范围是：60≤分数≤100，裁判人数 N=10。要求：每个裁判的分数由键盘输入。

例如：

输入：89 90 92 95 90 93 88 92 93 91

输出：91.25

实验提示：

（1）根据题意，需要求最高分和最低分，所以可以定义两个函数：

① max()：返回两个数中较大的值。

② min()：返回两个数中较小的值。

（2）在 main 函数中：

① 定义两个整型变量并赋初值 maxscore=0，minscore=100，分别用来存放 N 个数中的最大值和最小值。

② 定义整型变量并赋初值 sum=0，用于求累加和。

③ 采用循环结构，逐个输入每位裁判的分数，并随时记录输入过程中的最大值 maxscore 和最小值 minscore 以及累加值 sum。

④ 最后输出（sum-maxscore-minscore）/（N−2）的值。

实验项目 12　函数的嵌套调用与递归函数

一、实验目的

（1）进一步掌握函数的定义和调用。

（2）深入理解函数的形参和实参的概念。

（3）掌握函数嵌套调用的方法。

（4）掌握函数的递归调用方法。

二、实验要求

（1）熟悉函数调用的方法。

（2）熟悉函数调用时，形参和实参的一致性对应关系。

（3）理解递归的概念。

（4）掌握用递归方法解决问题。

三、实验内容

1．验证性实验

（1）按下面要求编写程序：

① 定义函数 int total(m)计算 1+2+3+…+m 的值。

② 定义函数 main()，输入正整数 n，计算并输出下列算式的值。要求调用函数 total 计算 1+2+3+…+n。

$$s=1+\frac{1}{1+2}+\frac{1}{1+2+3}+\cdots+\frac{1}{1+2+3\cdots+n}$$

实验提示：

① 根据题意，total 函数的原型为 int total（int m）。

② 在 total 函数体内，使用循环语句计算表达式 1+2+3+…+m 的值，并将求和得到的值返回给函数 total。

③ main 函数中输入正整数 n 的值，使用循环语句计算算式的值。算式累加和的通项式表示为 $\frac{1}{1+2+3+\cdots+k}$，其中 k 的取值范围为 1～n。

程序参考代码如下：

```
#include<stdio.h>
int total(int m);                /* 第 3 行 */
int main()
{
     int k,n;
     double s=0;                 /* 第 7 行 */
     printf("Input n:\n");
     scanf("%d",&n);
     for(k=1;k<=n;k++)
          s+=1.0/total(k);       /* 第 11 行 */
     printf("s=%.2lf\n",s);
     return 0;
}
int total(int m)
{
    int i,sum=0;
    for(i=1;i<=m;i++)
        sum+=i;
    return sum;
}
```

程序运行结果如图 3-83 所示。

图 3-83　求算式的值

思考

① 程序第 3 行语句起什么作用？还可以写在什么位置？什么情况下可以省略？
② 程序第 7 行语句改为“ int s=0;”可以吗？为什么？
③ 第 11 行语句改为“s=s+1/total(k);”可以吗？为什么？

（2）求 Fibonacci（斐波那契）数列前 20 项的值。要求：用递归法，定义函数 int fib(int k)，求 Fibonacci 数列第 k 项的值。Fibonacci 数列又称黄金分割数列，指的是这样一个数列：1、1、2、3、5、8、13、21…即第一项和第二项的值为 1，从第三项开始，以后每一项的值为其前两项的值之和。

实验提示：

① 在数学上，斐波那契数列以如下递归的方法定义：F0=0，F1=1，F(n)=F(n−1)+F(n−2)（n≥2）。利用递归式即可定义递归函数 int fib(int k)。

② 定义 main 函数，使用循环语句调用函数 int fib(int k)逐项求得 Fibonacci 数列的前 20 项的值，并输出。

程序参考代码如下：

```
#include<stdio.h>
int fib(int k);
int main()
{
    int k,count=0;
    for(k=0;k<20;k++)     /*  第 7 行  */
    {
        printf("%d\t",fib(k));
        count++;
        if(count%5==0)printf("\n");
    }
    return 0;
}
int fib(int k)
{
    if(0==k||1==k)
        return 1;
    else
```

```
        return fib(k-2)+fib(k-1);
}
```


程序运行结果如图 3-84 所示。

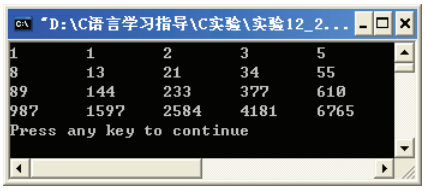


图 3-84　Fibonacci 数列前 20 项的值

思考

① 程序第 7 行语句改为 for(k=1;k<=20;k++)可以吗？为什么？

② main 函数中的变量 count 起什么作用？

③ 若不用递归，函数 int fib(int k)如何实现求 Fibonacci 数列第 k 项的值？

2．设计性实验

（1）计算并输出下列算式的值。

$$s=1+\frac{1+2}{2!}+\frac{1+2+3}{3!}+\cdots+\frac{1+2+\cdots+n}{n!}$$

要求：

① 定义函数 fact(n)，计算 n 的阶乘 n!=1*2*3*…*n，函数返回值类型是 double。

② 定义函数 cal(m,n)，计算任意区间(m～n)内整数的累加和 s=m+(m+1)+…+n，函数返回值类型是 double。

③ 定义函数 main()，输入正整数 n，计算并输出算式的值。算式中每一项的分子是累加和，要求调用函数 cal(m,n)，计算分子 1+2+…+n；每一项的分母是阶乘，要求调用函数 fact(n)计算 n!。

例如：

输入：3

输出：3.50

输入：10

输出：4.08

实验提示：

① 函数 cal(m,n)的功能是计算累加和 m+(m+1)+…+n，其两个参数分别为求和数的下限和上限。

② 算式的求和通式为 $\frac{1+2+3+\cdots+k}{k!}$，其中 k 的取值范围是 1～n。求和时，每加一项，k 的值需要递增 1 。

③ main 函数中调用函数 cal(m,n)计算分子的值，注意分子的算式求和总是从 1 开始的。

（2）编写程序，输入 n 的值，求 s=1！+2！+3！+…+n！。要求定义递归函数 fact(n)

求 n!。

例如：

输入：6

输出：873.00

实验提示：

① 求 n！的递归公式为：

$$fact(n)=\begin{cases}1 & (n=0 \text{ 或 } 1)\\ fact(n)=n*fact(n-1) & (n>1)\end{cases}$$

② 根据递归公式，定义递归函数 fact(n)，用于计算 n!。

③ 定义 main 函数，输入 n 的值，用循环语句实现 i= 1～n 的循环，核心计算公式：

s=s+fact(i)

④ 注意程序中求和变量 s 的数据类型为 double 型。

（3）编写程序，定义函数 DtoB(n)，其功能是将一个十进制整数转换成二进制数。

例如：

输入：25

输出：11001

实验提示：

① 将一个十进制整数 n 转换为二进制数可以采用“除 2 取余”法，即 n 除以 2 取余数，然后再用商除以 2 取余数，反复计算，直到被除数为零为止。

② 函数 DtoB(n)的原型为 void DtoB(int n)，函数中可以定义一个 int 型数组，将 n 除以 2 的余数逐个保存到数组中，然后逆序将数组元素的值逐个输出，也可以将其定义成递归函数。

③ 在 main 函数中，输入 n 的值，然后调用函数 DtoB(n)输出 n 的二进制形式。

3．提高实验

（1）王小二自夸刀功不错。有人放一张煎饼在砧板上，问他：“饼不许离开砧板，切 100 刀最多能分成多少块？”

例如：

输入：100

输出：切 100 刀最多可以分为 5051 块。

实验提示：

① 令 q(n)为切 n 刀能将饼分成最多的块数，可以得到以下结果：

q(1)=1+1=2

q(2)=1+1+2=4

q(3)=1+1+2+3=7

q(4)=1+1+2+3+4=11

…

② 在切法上是让每两刀都有交点。不难得出以下公式：

q(n)=q(n−1)+n （n≥1）

q(0)=1　　（一刀都不切当然只有一块）

③ 采用循环结构，用递推法求解问题（或者定义递归函数）。

（2）这是一个古典的数学问题：相传在古代印度的 Bramah 庙中，有位僧人整天把三根柱子上的金盘倒来倒去，原来他是想把 64 个一个比一个小的金盘从一根柱子上移到另一根柱子上去。移动过程中恪守下述规则：每次只允许移动一只盘，且大盘不得落在小盘上面。假设三根柱子的编号为 A、B、C。利用递归算法，编写程序，描述将 A 上三个盘子移到 C 上的操作步骤。

程序运行结果如图 3-85 所示。

图 3-85　将 A 上的三个盘子移到 C 上的程序运行结果

实验提示：

有人会觉得这很简单，事实却恰恰相反。n 个盘子由 A 移到 C，需移动的次数是 2^n-1，64 个盘子移动的次数为：

$$2^{64}-1=18\ 446\ 744\ 073\ 709\ 552\ 000$$

一年的秒数是：365×24×60×60=31 536 000，若 1 秒移 1 个盘子，则

18 446 744 073 709 552 000÷315 360 00 =584 942 417 355（年）

即约为 5849 亿年，从能源角度推算，太阳系寿命只有 150 亿年。

问题似乎一下变得很复杂，但换个角度考虑：如果有办法先将 63 个盘子按要求移到别的柱子上，那问题就容易解决了。只需做：

① 将 63 个盘子从 A 座移动到 B 座。

② 将最底下的盘子从 A 座移到 C 座。

③ 再将 63 个盘子从 B 座移到 C 座。

这样全部任务完成了。但是，有一个问题实际上未解决：如何将 63 个盘子从 A 座移到 B 座呢？用类似的方法：

① 将 62 个盘子从 A 座移动到 C 座。

② 将最底下的盘子从 A 座移到 B 座。

③ 再将 62 个盘子从 C 座移到 B 座。

这就是递归方法。如此层层递归，直到最后完成将 1 个盘子从一个座移到另一个座，问题就解决了。

综合以上分析，得到递归算法如下：

① 将 A 座上 n−1 个盘子移动到 B 座上（借助 C）。

② 将 A 座上 1 个盘子移动到 C 座上。

③ 将 B 座上 n−1 个盘子移动到 C 座上（借助 A）。

实验项目 13　变量的作用域与存储属性

一、实验目的

（1）理解变量的作用域概念并掌握全局变量和局部变量的作用域特点。

（2）掌握局部变量之间或全局变量和局部变量同名时的系统处理原则。

（3）理解变量的生存期的概念。

（4）掌握 4 种不同性质变量的存储特点及其使用特点。

二、实验要求

（1）熟悉全局变量和局部变量的概念。

（2）熟悉不同函数内部变量同名时的屏蔽原则。

（3）了解变量的静态存储属性和动态存储属性。

（4）掌握变量的生存期的概念。

三、实验内容

1．验证性实验

（1）完成主教材实例 6-7 和实例 6-9，体会变量的作用域特点。

（2）用一维数组存放 10 个学生的成绩，编写函数 float average(float array[],int n)求最高分、最低分和平均分。

实验提示：

① 因为一个函数只能返回一个函数值。题中要求三个值（最高分、最低分和平均分），所以建议定义两个全局变量 max 和 min，分别用于保存最高分和最低分。平均分由函数 average 返回。

② 在 main 函数中，用循环语句实现将 10 个数保存到一维数组中。

③ 调用函数 average。

④ 输出最高分、最低分和平均分。

程序参考代码如下：

```
#include<stdio.h>
#define N 10
float max, min;                 /*第 4 行*/
float average(float array[],int n)
{
    int i; float aver,sum=array[0];
    max=min=array[0];
    for(i=1; i<n; i++)
```

```
    {
        if(array[i]>max) max=array[i];
        else if(array[i]<min)min=array[i];
        sum=sum+array[i] ;
    }
    aver=sum/n;
    return(aver);
}
int main()
{
    float ave, score[N];                    /*第 20 行*/
    int i;
    printf("Input 10 scores:\n");
    for(i=0; i<N; i++)
       scanf("%f",&score[i]);
    ave=average(score, N);                  /*第 25 行*/
    printf("max=%6.2f\nmin=%6.2f\n", max, min);
    printf("ave= %6.2f\n",ave);
    return O;
}
```

程序运行结果如图 3-86 所示。

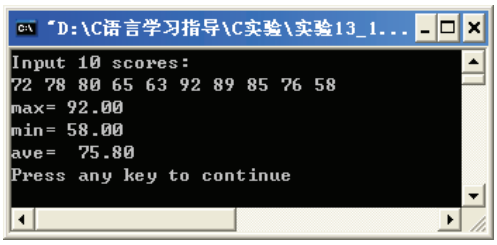


图 3-86　求最高分、最低分和平均分的程序运行结果

思考

① 将程序第 20 行中 score[N]改为 score[n]可以吗？为什么？

② 将第 25 行中的语句“ave=average(score, N);”改为“ave=average(score[N]);”可以吗？为什么？

③ 将第 4 行定义变量 max 和 min 的语句移到 average 函数体内，可以吗？为什么？

（3）阅读以下程序，并回答问题。

```
#include<stdio.h>
int k=1;
void fun();
int main()
{
```

```
    int j;
    for(j = 0; j < 2; j++)
       fun();
    printf("k=%d", k);
    return 0;
}
void fun()
{
    int k=1;          /* 第 14 行 */
    printf("k=%d,", k);
    k++;
}
```

问题：

① 程序的输出结果是什么？说明理由。

② 将第 14 行改为“static int k=1;”后，程序的输出是什么？说明理由。

③ 将第 14 行改为“k=1;”后，程序的输出是什么？说明理由。

④ 将第 14 行改为“;”后，程序的输出是什么？说明理由。

2．设计性实验

（1）程序填空。已知函数 void add()，在 main 函数中输入变量 n 的值，调用 add 函数，求 1+2+3+…+n 的值，并输出结果。

例如：

输入：50

输出：Sum is 1275.

```
#include<stdio.h>
void add();
int result;
int main()
{
    /*--请填上适当的语句--*/
    return 0;
}
void add()
{
    static int num=0;
    num++;
    result+=num;
}
```

实验提示：

① add 函数被调用一次，全局变量 result 就累加一次变量 num 的值。因为 num 是静态局部变量，所以调用一次 add 函数，其值就增加 1。

② 利用这个特点，main 函数中只要循环调用 add 函数 n 次就可以计算得到结果。

（2）程序填空。打印 2!，4!，6!，…，20! 的值。要求在 fact 函数中不使用循环语句，利用静态局部变量的特点求得阶乘的值。

程序运行结果如图 3-87 所示。

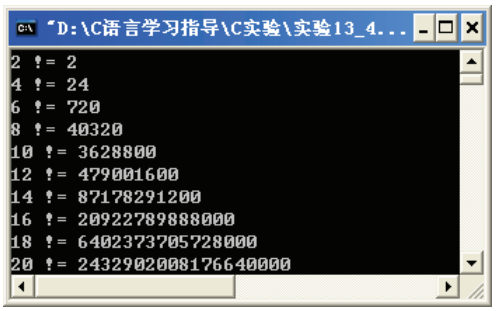


图 3-87　打印 2!，4!，…，20! 的值的程序运行结果

```
#include<stdio.h>
double fact(int  n);
int main()
{
    int i;
    for(i=2; i<=20; i+=2)
        printf("%d != %.0f\n", i, fact(i));
    return 0;
}
double fact(int n)
{
        /*-----请填上适当的语句-----*/
}
```

实验提示：

① 根据题意，相邻的两个阶乘值并非连续，如 2! 与 4!。

② 若已知 2! 的值，要求出 4!，需要在 2! 的基础上再乘以 3 和 4，即计算阶乘值的表达式中需要乘以两个数。

③ 参考主教材实例 6-12 的源程序。

3．提高实验

从键盘输入 n（0<n<11）个整数的值，先将这 n 个数原样输出，然后对其按从大到小的顺序进行排序后输出。要求：

① 定义函数 void input()，输入 n 个整数到一维数组。

② 定义函数 void sort()，对 n 个数从大到小排序并输出。

③ 定义函数 void output()，将 n 个整数输出。

④ 定义 main 函数，输入 n 的值，根据题意分别调用 input 函数、sort 函数和 output 函数进行必要的数据处理。

程序运行结果如图 3-88 所示。

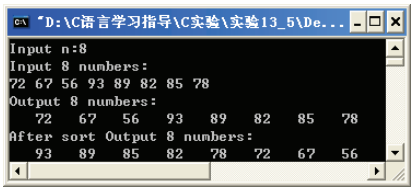


图 3-88　对 n 个数进行排序的程序运行结果

实验提示：

① 因为要求定义的函数都是无参的，其共同的操作目标是 n 个整数，所以要定义全局变量 n 和全局数组用于保存 n 个整数。

② 用冒泡或者选择排序法对数组中的 n 个数进行排序。

③ main 函数中调用 input 函数输入 n 个整数，调用 output 函数输出其原始值，然后调用 sort 函数对 n 个整数进行排序并输出（sort 函数调用 output 函数将排序后的数输出）。

实验项目 14　指针与函数

一、实验目的

（1）掌握以指针变量作参数的函数的定义和调用。
（2）掌握数组作参数时函数的多种定义形式及其调用。
（3）掌握返回指针的函数的定义和调用。
（4）理解函数指针的概念。
（5）掌握使用函数指针调用函数的方法。
（6）了解 main 函数参数的使用。

二、实验要求

（1）复习指针及其使用特点。
（2）巩固指针和数组相关的知识。
（3）复习指针数组的使用。
（4）熟悉用指针对函数的引用。
（5）理解 main 函数参数的意义。

三、实验内容

1．验证性实验

（1）完成教材实例 6-15 和实例 6-16，比较两个程序的差异。体会局部变量的特点，掌握用指针变量访问其所指对象的方法。

（2）编写程序，定义一个函数，其功能是求一个字符串的长度。

实验提示：

① 定义函数 int strlength(char *)，其功能是求字符串的长度。

② 在 main 函数中输入一个字符串，调用函数 int strlength(char *) 求该字符串的长度。

③ 输出计算结果。

程序代码参考：

```
#include<stdio.h>
int main()
{
    int strlength(char *);
    int len;
    char str[30];
    printf("Input a string: \n");
    gets(str);                                          /*第 9 行*/
    len=strlength(str);                                 /*第 10 行*/
    printf("The length of string is %d \n",len);
    return 0;
}
int strlength(char *p)
{
    int n;
    n=0;
    while(*p!='\0'){
    n++;
    p++;
    }
    return n;
}
```

程序运行结果如图 3-89 所示。

图 3-89　求字符串的长度的程序运行结果

思 考

① 将程序第 9 行改为“scanf("%s",str);”，运行程序，分析结果。

② 将程序第 10 行改为“len=strlength(str[30]);”可以吗？为什么？

③ 程序执行过程中，形参变量 p 与实参数组 str 之间有什么关系？

（3）编写程序，定义函数 double poly(double *,double)，计算多项式 $a_0+a_1\sin x+a_2\sin x^2+a_3\sin x^3+\cdots+a_9\sin x^9$ 的值。

实验提示：

① 定义函数 double poly(double *,double)，其功能是求多项式的值。

② 在 main 函数中：

（i）将多项式各项系数保存到一维数组 a 中；
（ii）输入 x 的值；
（iii）调用函数 double poly(double *,double) 求多项式的值；
（iv）输出计算结果。
程序代码参考：

```
#include<stdio.h>
#include<math.h>
double poly(double *p,double x)
{
    double sum,t=1;
    int i;
    sum=*p;                          /*第 8 行*/
    p++;
    for(i=1;i<10;i++,p++){
        t*=x;
        sum+=(*p)*sin(t);
    }
    return sum;
}
int main()
{
    double x,y;
    double a[10]={1.2,-1.4,-4.0,1.1,2.1,-1.1,3.0,-5.3,6.5,-0.9};
    printf("Input x:\n");
    scanf("%lf",&x);
    y=poly(a,x);                      /*第 23 行*/
    printf("%.2f\n",y);
    return  0;
}
```

程序运行结果如图 3-90 所示。

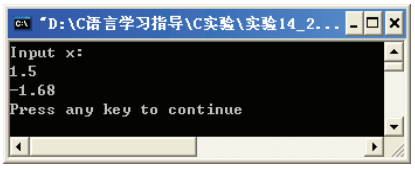


图 3-90　求多项式的值的程序运行结果

思考

① 程序第 8 行中*p 的值是什么？
② 将程序第 23 行改为“y=poly(&a,x);”可以吗？为什么？
③ 将程序第 23 行改为“y=poly(a[10],x);”可以吗？为什么？

2. 设计性实验

（1）编写程序，输入 n（0<n<11）个学生的英语成绩，将大于平均分的成绩输出。要求：

① 定义函数 void aboveAve(int score[],int n)，求 n 个整数的平均值，并将大于平均值的数输出。

② 定义 main 函数，输入 n 的值，将 n 个整数输入一个一维数组中，调用 aboveAve 函数将大于平均分的成绩输出。

例如：

输入：6（n 的值）

67 78 56 89 92 75

输出：78 89 92

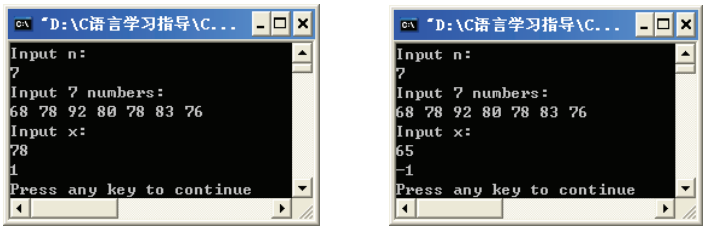
实验提示：

① 在 main 函数中将 n 个整数输入一个一维数组中，要将该数组中的所有元素传递给 aboveAve 函数，所以要以数组名作实参。

② aboveAve 函数中首先要求出 n 个数的平均值，然后再利用循环结构，将数组中的元素逐个与平均值比较大小，若大于平均值，则输出，否则不作处理。

（2）程序填空。定义函数 int search(int list[], int n, int x)，在有 n 个元素的数组 list 中查找元素 x，若找到则返回数组元素的下标，否则返回−1。在 main 函数中输入 n 的值，将 n 个数输入数组 a 中，然后输入 x 的值，调用 search 函数并输出函数值。

程序运行结果如图 3-91 所示。



（a）查找成功　　（b）查找失败

图 3-91　在 n 个数中查找 x 的程序运行结果

```
#include<stdio.h>
int search(int list[],int n,int x);
int main()
{
    int i, n,x, a[10], res;
    printf("Input n:\n");
    scanf("%d", &n);
    printf("Input %d numbers:\n",n);
    for(i=0; i<n; i++)
        scanf("%d", &a[i]);
    printf("Input x:\n");
    scanf("%d", &x);
    res=search(a, n, x);
```

```
    printf("%d\n", res);
    return  0;
}
/*-----请填上适当的语句-----*/
```

实验提示：

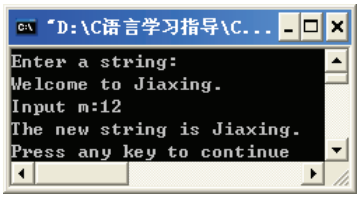
① 在数组 list 中查找元素 x 可以用单循环，采用顺序查找的方法实现。

② 注意控制循环语句的执行。

3．提高实验

（1）编写程序，定义函数 char *substrcpy(char *str1, int m, char *str2)，其功能是将字符串 str1 中第 m 个字符开始的所有字符复制成一个新的字符串 str2，函数 substrcpy 返回新字符串的首地址。若 m 大于字符串的长度，则函数返回 NULL。

程序运行结果如图 3-92 所示。



（a）输入有效 m 值

（b）输入无效 m 值

图 3-92　求字符串的子串的程序运行结果

实验提示：

① 在函数 char *substrcpy(char *str1,int m,char *str2)中先求字符串 str1 的长度 len，然后与变量 m 的值比较，若 len<m，函数返回值为 NULL，反之，求新串，并将新字符串的首地址返回给函数。

② 在主函数中：

（i）将字符串输入一维字符数组中，再输入 m 的值；

（ii）调用 substrcpy 函数得到新的字符串；

（iii）输出新字符串。

（2）编写程序，求两个整数的较大值，要求用命令方式运行该程序，两个整数在命令行中输入，运行结果如图 3-93 所示。

图 3-93　用命令行方式求两个整数的较大值的运行结果

实验提示：

① 根据题意，程序需要带参数的 main 函数。

② 命令行由 3 个字符串（可执行文件名和两个整数的值）组成，执行 main 时，参数 argc 的初值为 3，指针数组 argv 的 3 个元素分别保存 3 个字符串的首地址。

③ 因为命令行中输入的两个整数是以字符串的形式保存的，要比较整数的大小还需要将字符串转换成整数。可以考虑使用将字符串转换为整数的系统函数 atoi，该函数原型为 int atoi(char *str)，程序中需要包含头文件 stdlib.h。

实验项目 15　结构体应用

一、实验目的

（1）掌握结构体类型的定义，结构体类型变量的说明，以及成员的引用方法。
（2）掌握结构体类型数组的概念和应用。
（3）掌握结构体指针变量的概念和应用。
（4）掌握单链表的概念和基本操作。

二、实验要求

（1）熟悉结构体的概念和应用。
（2）了解结构体变量的定义与引用。
（3）了解结构体数组的定义与使用。
（4）领会结构体指针变量的定义与使用。
（5）熟悉单链表的基本操作。

三、实验内容

1．验证性实验

输入一位同学某门课程考试的相关信息，包括学号、姓名、性别和这门课程的成绩。

实验提示：

① 声明一个结构体类型，包含学号、姓名、性别和成绩四个成员项。

② 编写 main 函数，用步骤①中声明的结构体类型定义一个结构体变量和一个指向这种结构体类型的指针变量，给 4 个成员项赋值并输出。

程序代码参考：

```
#include<stdio.h>
#include<string.h>
struct exam        /*第 3 行*/
{   long num;
    char name[10];
    char sex;
    float score;
};
int main()
```

```
{   struct exam stud1,*p ;
    char ch;
    p=&stud1;                /*第 12 行*/
    stud1.num=200701;
    strcpy(stud1.name,"wang");
    ch=getchar();
    stud1.sex=ch;
    (*p).score=543;          /*第 17 行*/
    printf("%ld,%c,%f,%s\n",p->num,p->sex,p->score,p->name);
    return 0;
}
```

实验运行结果如图 3-94 所示。

图 3-94　结构体变量的定义和表示的实验运行结果

思考

（1）程序第 3 行定义的结构体类型的名字是什么？

（2）程序第 12 行语句“p=&stud1;”起什么作用？是否可以省略？

（3）程序第 17 行语句“(*p).score=543;”中*p 两侧的括号能不能省略？访问结构中的成员有哪几种方法？具体在本程序中的哪些语句中体现？

2．程序填空题

完善下列程序段，以 sum 成员项为准实现结构数组的排序。

```
#include<stdio.h>
int main()
{
    struct student
    {
        char name[20];
        int num;
        float math;
        float eng;
        float cuit;
        float sum;
    }stu[4];
    struct student temp;
    int i,j,k;
    for(i=0;i<4;i++)
```

```
    {
        printf("input %d name=?",i);
        gets(stu[i].name);
        printf("input %d num,math,eng,cuit=?",i);
        scanf("%d%f%f%f",&stu[i].num,&stu[i].math,
            &stu[i].eng,&stu[i].cuit);
        getchar();
        stu[i].sum=stu[i].math+stu[i].eng+stu[i].cuit;
    }
    for(i=0;i<3;i++)
    {
       k=i;
       for(j=i+1;j<4;j++)
           if( ____①____ )k=j;
       if(k!=i)

           {
               temp=____②____;
               ____③____;
               ____④____=temp;
           }
   }
    for(i=0;i<4;i++)
    {
        printf("%-10s%d%8.2f%8.2f%8.2f%8.2f\n",
            stu[i].name,stu[i].num,stu[i].math,
            stu[i].eng,stu[i].cuit,stu[i].sum);
    }
    return 0;
}
```

实验提示：

（1）目前已学到的排序算法有冒泡法和选择法，其中冒泡法的思路是：将相邻两个数比较，将小的调到前头；而选择法的思路是：每一趟选出一个最小的数，并和当前第一个位置的数交换。此题中用的是选择法，对 sum 成员项进行比较。

（2）ANSI C 标准允许相同类型的结构体变量相互赋值，题中要实现两个结构体变量的整体交换，采用三个赋值语句。

实验运行结果如图 3-95 所示。

3．设计性实验

（1）定义一个包括 5 个学生信息的结构数组，每个学生包括学号、姓名和总分。输入一个学生的学号，在该结构体数组中查找该学号，如果找到，则输出该学生的姓名和总分；如果找不到，则输出显示“Not Found!”

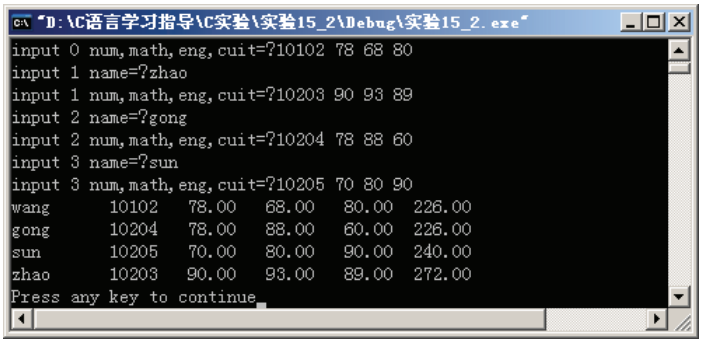


图 3-95　结构体数组排序的实验运行结果

实验提示：

查找的过程是将用户输入的学号和结构体数组中的学号成员项中的内容逐个比对，如果找到循环提前结束；若找不到，循环正常结束（也就是说一直找到数组中的最后一个元素都没有符合的信息）。为了区分循环究竟是提前结束还是正常结束，引入标识量 flag，flag=1 表示找到，循环提前结束；flag=0 表示没有找到。

实验运行结果分别如图 3-96 和图 3-97 所示。

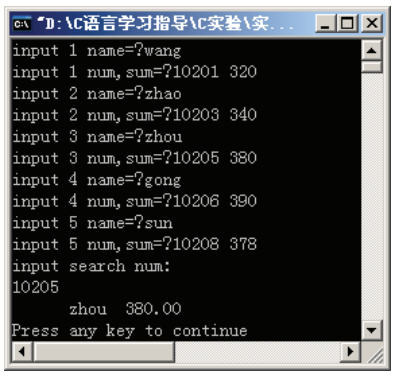


图 3-96　找到时的实验运行结果

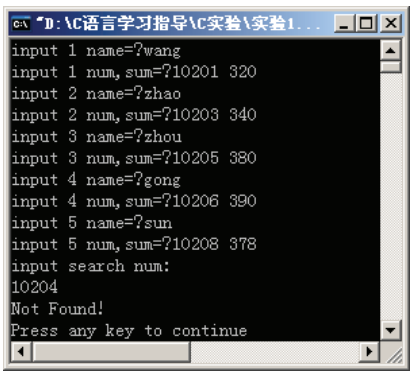


图 3-97　未找到时的实验运行结果

（2）有 10 个学生，每个学生的数据包括学号、姓名、数学、物理和化学 3 门课程的成绩。从键盘输入 10 个学生的数据，要求：

① 输出数学最低分的学生的数据（包括学号、姓名、3 门课程成绩）。

② 求出每个学生的平均成绩，并按平均成绩由高到低排序。

实验提示：

① 要求输出数学最低分的学生的数据，归结为求极值问题。解决此类问题的方法是：假定第一个数组元素的值是最小的，用 min 记录其下标，即 min=0；然后利用循环在数组中进行查找，若当前元素的成绩项的值大于假定的成绩项的值（即 stu[i].math<stu[min].math），则当前最大的值的下标应修改为 min=i。当循环结束后，min 记录的即为数组中最小值的下标。找最大值的方法类似。

② 要求按平均成绩由高到低排序，可采用冒泡或选择法。

实验运行结果如图 3-98 所示。

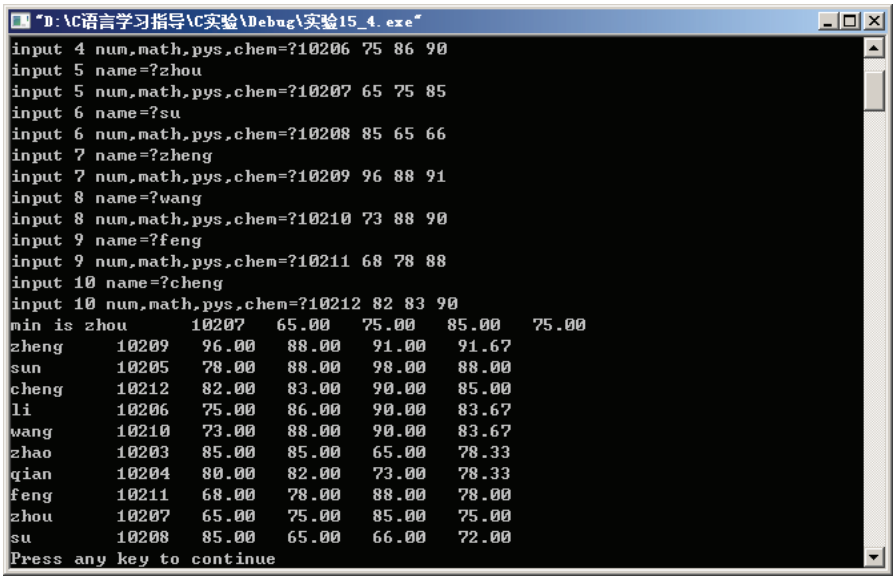


图 3-98　排序和求最低分的实验运行结果

（3）用头插法创建一个单链表并顺序输出，参考结果如图 3-99 所示。

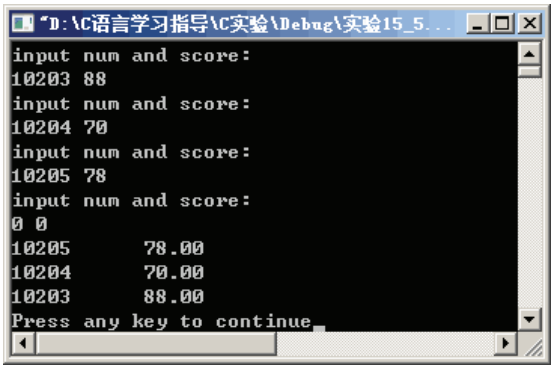


图 3-99　用头插法创建一个单链表并输出的参考结果

实验提示：

头插法的具体生成过程如下：

① 建立一个“空”链表。

② 输入数据元素 a_n，建立结点并插入表头。

③ 输入数据元素 a_{n-1}，建立结点并插入表头。

④ 以此类推，直至输入 a_1 为止。

所谓头插法指的是建立的新结点插入在表头位置，也就是插入点在头结点的后面。因此如果要建立一个顺序为 a_1、a_2、…、a_{n-1}、a_n 的序列，需要逆序输入 n 个数据元素的值。如果不能确定 n 值的大小，则以某个结束标识（比如当输入值为 0 时）为止。

4．提高实验

（1）用单链表实现设计性实验（1）中的查找操作，并要求将查找到的学生记录删除。

实验提示：

① 用头插法或尾插法建立一个单链表。

② 采用顺序查找。

③ 删除结点时注意语句顺序，如图 3-100 所示。

④ 最后输出单链表中的信息。

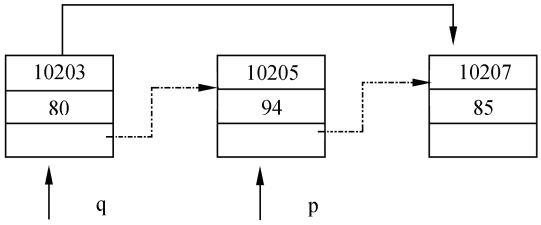


图 3-100　单链表删除过程示意图

实验运行结果与图 3-96 一样。

（2）11 个学生围成一圈，从第一个人开始顺序报号 1、2、3。凡报到 3 的学生退出圈子。再依次循环，找出最后留在圈子中的人的原来的序号。

实验提示：

① 建立一个单循环链表，每个结点的数据域依次存放的是学生的编号（模拟 11 个学生围成一圈）。

② 从链表的第一个结点开始依次遍历到第 3 个结点的前驱，将第 3 个结点从链表中删除（模拟学生报数和学生退出）。

③ 再从被删结点的后续结点开始，重复②中的学生报数和学生退出过程，直到链表中只剩下一个结点为止。

实验运行结果如图 3-101 所示。

图 3-101　退出顺序和最后留在圈中人的序号的实验运行结果

实验项目 16　文件及应用

一、实验目的

（1）理解文件和文件指针的概念。

（2）掌握文件的打开、关闭以及文件的读写操作。

（3）了解数据文件在程序中的应用。

二、实验要求

（1）预习数据流文件与程序文件的区别。

（2）熟悉各种文件读写函数的使用。

（3）预习数据流文件操作的特点。

三、实验内容

1．验证性实验

新建一个磁盘文件 ex16_1.dat，从键盘输入一个字符串，将其中的小写字母全部转换成大写字母写入该文件中（要求写入文件的同时把结果显示在屏幕上）。

实验提示：

① 以写的方式打开文件 ex16_1.dat。

② 从键盘输入一个字符串。

③ 用单循环结构，将字符串中所有的小写字母转换成大写字母写入该文件中。

④ 关闭文件。

程序代码参考：

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    FILE *fp;                                        /*第 6 行*/
    char str[80];
    int i=0;
    if((fp=fopen("ex16_1.dat", "w"))==NULL){    /*打开文件*/
        printf("Can not open the file\n");
        exit(1);
    }
    printf("Input  a string: \n");
    gets(str);
    printf("The string in file is: \n");
    while(str[i]!='\0'){
        if(str[i]>='a'&&str[i]<='z')
          str[i]=str[i]-32;
        putchar(str[i]);
        fputc(str[i],fp);                            /*第 20 行*/
        i++;
    }
    putchar('\n');
    fclose(fp);
    return  0;
}
```

程序运行结果如图 3-102 所示。

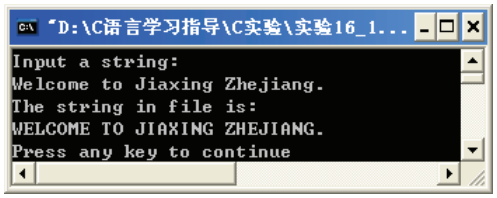


图 3-102　字符串写入文件的程序运行结果

思考

① 将程序第 6 行改为“file *fp;”可以吗？为什么？

② 第 20 行语句的功能是用 fputc 函数将一个字符写入文件中。若改用 fprintf 函数，如何修改语句？

2．设计性实验

（1）完成以下功能：f(x,y)=(3.14*x−y)/(x+y)，若 x、y 取值为区间[1,6]的整数，找出使 f(x,y)取最小值的 x1、y1，并将 x1、y1 以格式"%d,%d"写入文件 design.dat 中（要求写入文件的同时把结果显示在屏幕上）。

程序执行结果：

文件 design.dat 中的内容为：1,6

显示屏幕上输出：1,6

实验提示：

① 定义一个函数求表达式(3.14*x−y)/(x+y)的值，函数原型为 double f(int x,int y)。

② 利用双重循环结构求 f(x,y) 取最小值的 x1、y1。

③ 对文件 design.dat 的操作过程参见验证性实验内容。

（2）新建一个含有字符'$'的 data.txt 文本文件，统计文本文件 data.txt 中字符'$'出现的次数，并将统计结果写入文件 result.txt 中。

例如：

文件 data.txt 中的内容为：books 15$,pens 20$,pencils 10$,bags 35$。

文件 result.txt 中的内容为：字符'$'出现 4 次。

实验提示：

① 先用记事本创建文本文件 data.txt。

② 利用单循环结构，使用 fgetc 函数将文件 data.txt 中的字符逐个读出，并统计字符'$'的个数；使用 fprintf 函数将结果写入文件 result.txt 中。

③ 注意：无论文件 data.txt 还是文件 result.txt，对文件进行读写操作前都需要先打开文件，操作结束后关闭文件。

（3）从文件 string.txt 中输入一个字符串，将组成字符串的所有非英文字母的字符删除，将该字符串保存到文件 letter.txt 中（要求写入文件的同时把结果显示在屏幕上）。

例如：

文件 string.txt 中的内容为：a+5x−siny+b

文件 letter.txt 中的内容为：axsinyb

实验提示：

① 先用记事本创建文本文件 string.txt。

② 使用 fgets 函数将文件 string.txt 中的字符串读出，保存到一个一维字符数组中。用单循环结构，将字符串中的非英文字母删除。

③ 用 fputs 函数将字符串写入文件 letter.txt 中。

④ 需要注意，文件 string.txt 需要以读的方式打开，文件 letter.txt 需要以写的方式打开，操作结束后要关闭文件。

3．提高实验

输入 5 位学生的信息，包括学号、姓名和三门课程成绩。计算每个学生的平均成绩。将每个学生的信息（学号、姓名、三门课程成绩和平均分）保存到磁盘文件 stud_info.dat 中（要求写入文件的同时把结果显示在屏幕上）。

程序运行结果如图 3-103 所示。

图 3-103　学生成绩的管理的程序运行结果

实验提示：

① 根据题意，考虑构造结构体类型 struct student，有 4 个成员项，分别为学号、姓名、三门课程成绩（一维数组）和平均分，并定义结构体数组。

② 利用单循环结构输入学生的信息。

③ 再利用单循环结构计算学生的平均分（也可以在②中输入成绩后就计算平均分）。

④ 建议使用 fwrite 函数将数据保存到文件 stud_info.dat 中，注意打开文件时文件操作

方式的选择。

⑤ 建议分别定义多个函数完成程序。

实验项目 17　C 语言程序综合应用

一、实验目的

（1）加深对 C 语言程序设计基本理论和基本知识的理解。

（2）深入理解结构化和模块化程序设计思想。

（3）把所学的理论知识与实践结合起来，提高用计算机解决实际问题的能力。

（4）培养学生的逻辑思维能力、团队合作精神、实际动手能力和创新能力。

二、实验要求

（1）具备一定的 C 语言程序设计基础。

（2）确定所选课题的功能模块，详细描述各模块的具体内容。

（3）认真分析设计过程中涉及的算法，用流程图描述算法。

（4）熟练掌握程序调试的方法和步骤。

三、实验内容

1．学生档案管理

（1）题目描述

编写一个程序，管理学生档案，系统能实现以下功能：

① 输入信息：首先输入记录数，然后逐条输入学生的基本信息。

② 修改信息：根据学号对学生的基本信息进行修改。

③ 增加信息：添加新学生的基本信息。

④ 插入信息：在指定位置插入一个学生的信息。

⑤ 删除信息：根据学号或姓名删除指定学生的信息。

⑥ 查询：根据学号或姓名查询某个学生的信息，根据学号区间段查询某些学生的信息。

⑦ 排序：选择根据学号进行升序或者按性别进行降序排序，并显示排序后的结果。

⑧ 统计：根据性别统计男、女生比例，根据家庭地址统计各省份生源数。

⑨ 输出：输出所有学生信息或查询学生信息的结果。

⑩ 保存：将所有学生信息保存到一个文件 student.dat 中。

（2）设计提示

① 先确定学生档案管理的数据结构。如每个学生的信息包括：学号、姓名、性别、出生日期、家庭地址等，每个数据项各用什么数据类型？

② 划分实现学生档案管理的功能模块。如主菜单、输入数据、修改、增加、插入、删除、查询、排序和输出等功能，并确定各功能模块的实现算法。

2．选票统计

（1）题目描述

从 100 名优秀运动员中评选出 10 名最佳运动员，具体规则如下：

① 运动员号按 1、2、3、…顺序编号。

② 由键盘接收所收到的选票，每张选票至多可写 10 个不同的编号。

③ 对应名次的运动员编号可以有空缺，但必须用 0 表示。

④ 若选票中编号超出规定的范围，或编号出现重复，作废选票。

⑤ 按选票中所列最佳运动员顺序给他们计分，计分标准如下：从第 1 名至第 10 名所得分数依次为 15、12、9、7、6、5、4、3、2、1。

⑥ 按各运动员所得分数高低进行排序，列出前十名最佳运动员排名，格式为：

名次　　运动员编号　　合计得分　　合计得票数

如果得分相同，则得票多者在前；如果得分与票数都相同，则编号小的在前。

（2）设计提示

① 根据题意，明确选票及其运动员的完整数据信息。

② 选票统计过程分为三个步骤：投票、选票检查（即看是否为有效票）和有效票统计。

③ 根据要求，可将问题分解为如下多个功能模块：

（i）输入选票；

（ii）选票检查；

（iii）统计每个运动员的总得票数和总分；

（iv）运动员得分高低排序；

（v）输出运动员排名。

④ 可以考虑通过无限循环接收所有选票，用输入运动员编号为−1 来终止循环。

3．多项式求和

（1）题目描述

编写一个程序，实现多项式的求和。例如多项式 $2X^5+3X^2-5.1X+6$ 与多项式 X^4-3X^2+X-2 求和的结果为：$2X^5+X^4-4.1X+4$。系统能实现以下功能：

① 输入：输入一个多项式。

② 插入：在一个多项式中插入一项。

③ 删除：在一个多项式中删除一项。

④ 查找：在一个多项式中查找某一项。

⑤ 合并：多项式求和。

⑥ 输出：输出多项式。

（2）设计提示

① 先确定多项式的数据结构。如确定每一项由哪些数据组成，每个数据项用什么数据类型比较合适。

② 明确功能模块：如主菜单、输入数据、删除数据、查询、合并和输出等功能，并确定各功能模块的实现算法。