《程序设计基础》实验指导书

（第二版）

信息学院

《程序设计基础》教研组

2019 年 9 月 1 日

目　录

前 言

C 语言是现代最流行的通用程序设计语言之一，它既具有高级程序设计语言的优点，又具有低级程序设计语言的特点，既可以用来编写系统程序，又可以用来编写应用程序。因此，C 语言正在被迅速地推广和普及。上机实验是该课程教学的一个重要环节，因此要求学生做一定数量的上机实验。本指导书可增强同学上机实验的针对性。整个教学和实验中，采用 DEV C++ 作为实验环境，强调学生切实培养动手实践能力，掌握调试程序的方法，通过调试理解 C 语言程序运行的过程以及 C 语言的语法规则，为后续的课程设计，计算机等级考试及其他应用做好充分的准备。

本实验指导书通过大量的实例，循序渐进地引导学生做好各章的实验。根据实验教学大纲，共选择编写了 13 个实验，其中必做 11 个，选做 2 个 。每个实验内容结构如下：

（1）实验学时

（2）实验目的

（2）预习要求

（4）实验内容

（5）实验注意事项

（6）思考题

其中思考题属于扩展应用部分，学生可以根据自己的学习情况选择完成。

在实验之前，要求学生对实验作好预习工作。在实验中，学生根据实验指导中的内容进行验证与总结，然后再去完成实验内容中安排的任务。一般要求准备好相关代码，实验课中最好以调试和讨论为主。

实验结束后，应及时提交实验报告，报告具体内容可根据实验内容和实验要求进行增删。实验报告一般要求包含：

（1）实验题目

（2）设计思路或算法分析

（3）流程图

（4）程序源代码

（5）程序运行结果及分析

（6）存在的问题。

实验一 编程环境介绍

Dev-C++是一个 Windows 环境下 C/C++的集成开发环境(IDE)，它是一款自由软件，遵守 GPL 许可协议分发源代码。它集合了 MinGW 等众多自由软件，并且可以取得最新版本的各种工具支持，而这一切工作都是来自全球的狂热者所做的工作。Dev-C++是 NOI、NOIP 等比赛的指定工具，缺点是 Debug 功能弱。

Dev-C++使用 MingW32/GCC 编译器，遵循 C++ 11 标准，同时兼容 C++98 标准。开发环境包括多页面窗口、工程编辑器以及调试器等，在工程编辑器中集合了编辑器、编译器、连接程序和执行程序，提供高亮度语法显示的，以减少编辑错误，还有完善的调试功能，适合初学者与编程高手的不同需求，是学习 C 或 C++的首选开发工具！多国语言版中包含简繁体中文语言界面及技巧提示，还有英语、俄语、法语、德语、意大利语等二十多个国家和地区语言提供选择。

一、下载并安装 Dev-C++

Dev-C++ 目前最新版本：5.4.2（由 Orwell 更新），可以通过合法的渠道下载。Dev-C++有 32 位版本与 64 位版本两种，用户可以根据自己计算机的硬件、操作系统的实际情况选择合适的版本。具体安装过程不再赘述。

二、启动 Dev-C++

1. 鼠标点击任务栏中的“开始”按钮，选“程序”菜单项，然后选“程序”下的子菜单项“Bloodshed Dev-C++”项，显示该项下的子菜单。

2. 单击”Dev-C ++”菜单项，即可启动 Dev-C ++集成开发工具，如图 1- 1 所示。

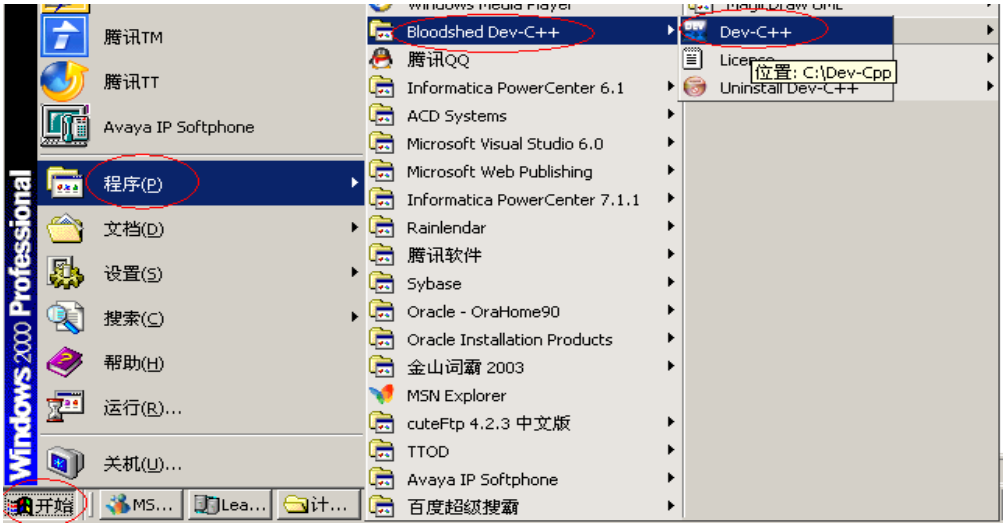

图 1- 1　启动 Dev-C++

3. 启动成功后，界面如图 1- 2 所示。这与其它类似软件的人机界面相同。

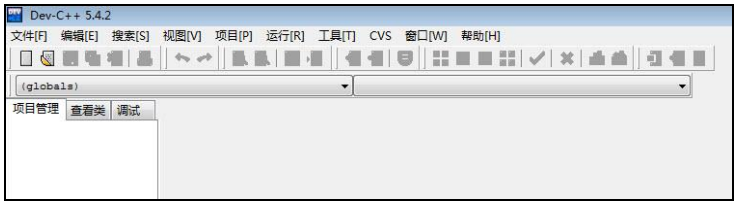

图 1- 2　启动 Dev-C++成功后界面

三、新建源程序

1.从主菜单选择“文件”→“新建”→ “源代码”，如图 1- 3 所示。

图 1- 3　新建源代码

如果第一次显示界面上的文字是英文的，则可以根据以下操作将界面改为中文。点击主菜单“工具”→“环境选项”，在弹出的对话框中选择“界面”页，在 Language（语言）下拉列表中选择 Chinese 即可，如图 1- 4 所示。

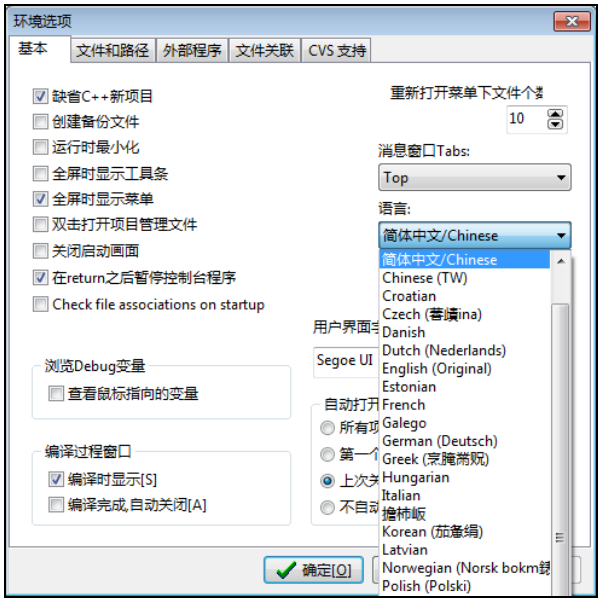

图 1- 4　修改环境显示语言

2.此时屏幕右下侧出现一片白色区域，称为“源程序编辑区域”，用记可以在此输入自己的程序代码。如图 1- 5 所示。

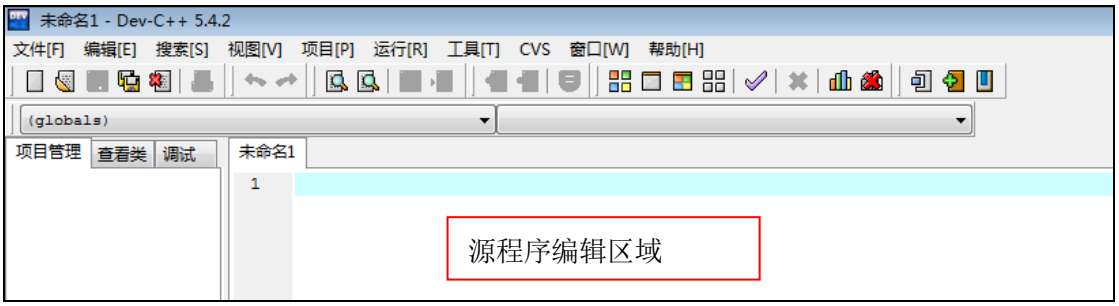

图 1- 5　修改环境显示语言

3.保存源程序到硬盘

一个好的习惯是创建了一个新程序后，在还未输入代码之前先将该程序保存到硬盘某个目录下，然后在程序的编辑过程中经常性地保存程序，以防止机器突然断电或者死机。从主菜单选择“文件”→“保存”就可以将文件保存到指定的硬盘目录。

4.在程序编辑区域编辑程序

如图 1- 6 所示。在输入程序的过程中可以随时对程序进行保存。

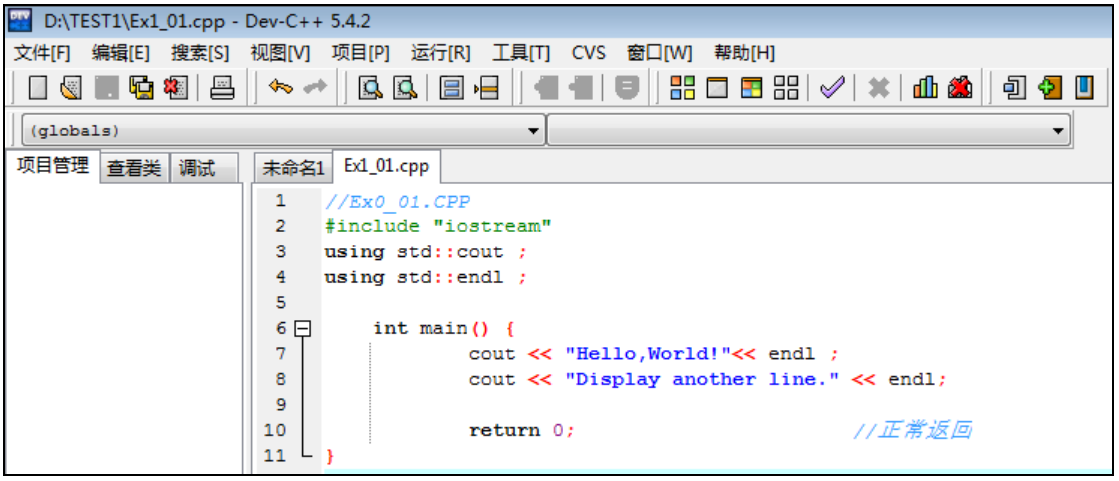

图 1- 6　输入用户源程序代码

四、预处理、编译、链接程序

用户输入完源程序代码后，可以从主菜单选“运行”→ “编译”或快捷键 F9，编译刚才的代码，如果用户的源程序不止一个，也可以使用快捷键 F12 重新编译全部程序，具体菜单选项与快捷键列表如图 1- 7 所示。

如果程序中存在词法、语法等错误，则编译过程失败，编译器将会在屏幕右下方的“Compile Log（编译日志）”标签页中显示错误信息，如下图 8 所示，并且将源程序相应的错误行标成红色底色，用户可以根据相关提示进行修改。

“Compile Log（编译日志）”标签页中显示的错误信息是寻找错误原因的重要信息来源，**每一位同学都要学会看这些错误信息**，并且每一次你碰到错误并且最终解决了错误时，要**记录错误信息以及相应的解决方法**。这样以后看到类似的错误提示信息，能熟练反应出是源程序哪里有问题，从而提高程序调试效率。

排除了程序中存在的词法、语法等错误后，编译成功，如图 1- 8 所示。此时在源文件所在目录下将会出现一个同名的.exe 可执行文件（如 Ex1_01.exe），既可以使用菜单项运行该文

件，也可以双击这个 EXE 可执行文件来运行该程序。

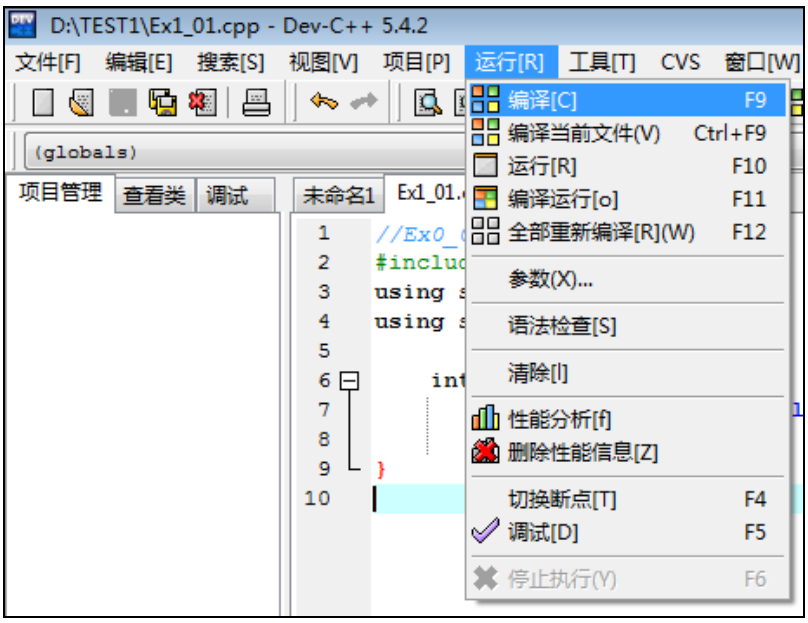

图 1- 7　编译、运行菜单选项

图 1- 8　编译成功提示

五、调试程序

通过预处理、编译和链接的程序仅仅是该程序中没有词法和语法等错误，而无法发现程序深层次逻辑问题（譬如算法不对导致结果不正确）。当程序运行出错时，需要找出错误原因。仔细读程序来寻找错误固然是一种方法，但是有时光靠读程序已经解决不了问题，**此时需要借助于程序调试（Debug）手段**。这是一种有效的排错手段，每一位同学都需要掌握。

(1) 设置程序断点

调试的基本思想是让程序运行到你认为可能有错误的代码前，然后停下来，在人的控制下逐条语句的运行，通过在运行过程中查看相关变量的值，来判断错误产生原因。如果想让程序运行到某一行前能暂停下来，就需要将该行设成断点。**具体方法是在代码所在行行首单击**，该行将被加亮。默认的加亮颜色是红色。如图 1- 9 所示，将 system(“pause”)语句设成断

点，则程序运行完 printf 语句后，将会暂停。需要说明的是，你可以在程序中根据需要设置多个断点。如果想取消不让某行代码成为断点，则在代码行首再此点击即可。

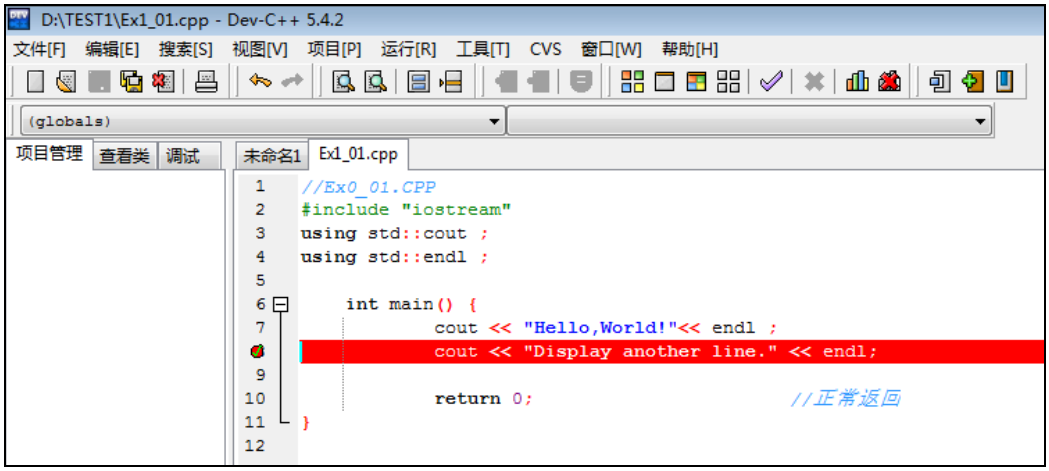

图 1- 9　设置一个断点

(2) 开始调试程序

设置断点后，此时程序运行进入 debug（调试）状态。要想运行程序，就不能使用主菜单“运行”→ “执行”，而是需要用主菜单“运行”→“调试”（或者按快捷键 F5），如图 1- 10 所示。

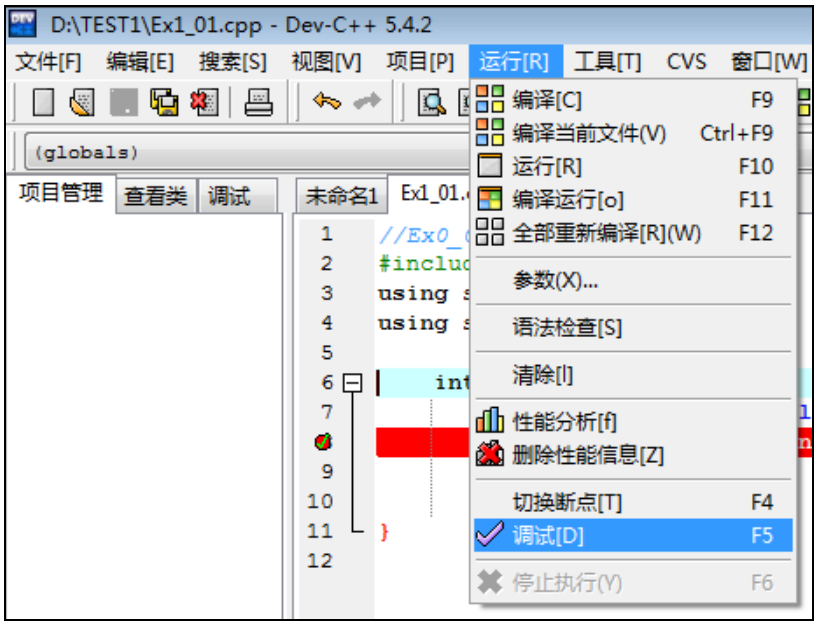

图 1- 10　调试运行程序

程序运行到第一个断点处将自动停住，此时断点处加亮色由红色变成蓝色，表示接下去将运行蓝色底色的代码，在此行前面的所有代码已经运行完毕，在本例中，已经在屏幕上显示一行信息，如图 1- 11、图 1- 12 所示。

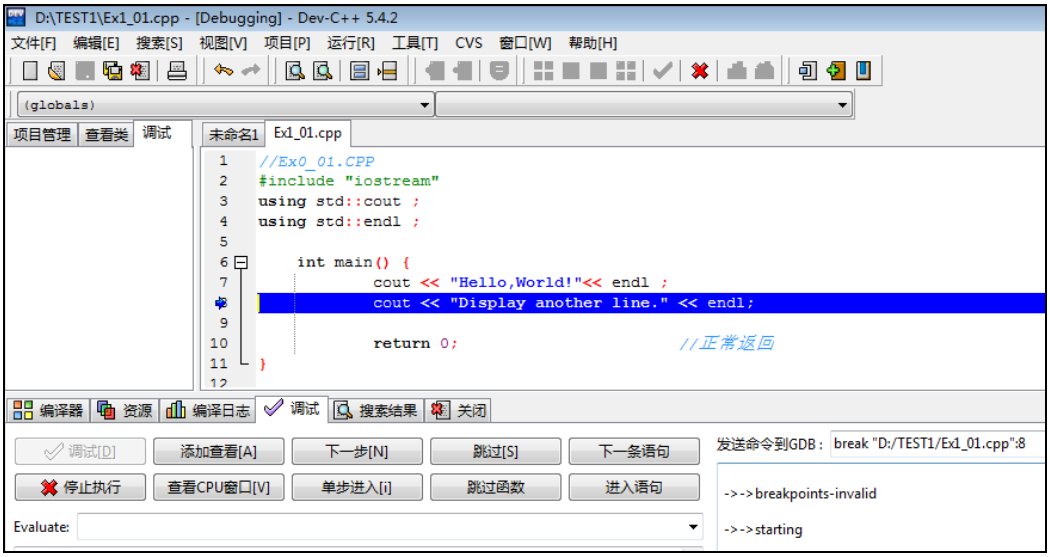

图 1- 11　调试运行程序

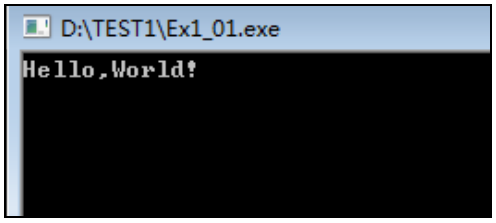

图 1- 12　调试中屏幕显示部分结果

注意：如果有时发现即使设置了断点，点击了主菜单“运行”→ “调试”，程序还是不在断点处停留。解决方法：取消断点，重新编译程序，然后再设置断点，点击主菜单“运行”→ “调试”即可。

(3) 单步执行程序

当程序运行到断点处时，系统提示用户可以有多种选择，如：运行下一行代码？进入到函数内部？运行到下一个断点处？等，如图 1- 13 所示。

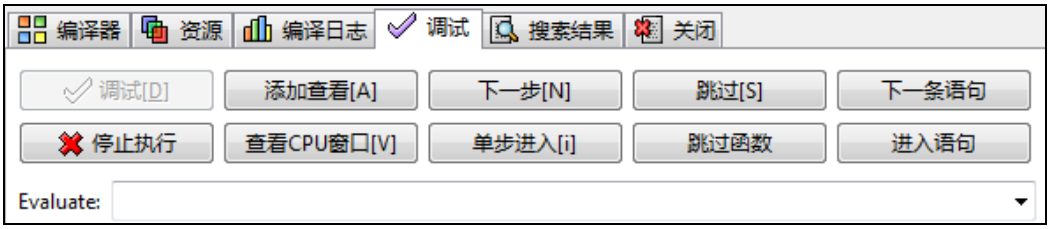

图 1- 13　断点后运行选择

图中各选择的含义通过实际的操作加以体会。

(4) 设置 watch 窗口

在调试程序时，可能要看程序运行过程中变量的值，以检测程序对变量的处理是否正确，可以在调试时 通过调试菜单下的添加变量（Add Watch）窗口来增加变量 watch，新增的变量将会显示在最左边 Explore 的 Debug 页中。如果左边 Explore 中的当前页不是 Debug 页，则可以点击 Debug 标签使之成为当前页。

实验二　数据类型、运算符和表达式

一、实验目的

（1）了解 C 语言数据类型的意义，掌握基本数据类型变量的特点和定义方法。

（2）学会使用 C 的算术运算符，以及包含这些运算符的算术表达式。

（3）掌握自加（++）和自减（--）运算符的使用。

（4）进一步熟悉 C 程序的编辑、编译、连接和运行的过程。

二、实验要点

基本数据类型包括整型、字符型、实型。

三、实验要求

（1）上机前先阅读和编写以下要调试的程序。

（2）上机输入和调试程序并存在磁盘上。

（3）检查实验结果是否正确。

四、实验内容

（以下内容在实验报告中的实验预习报告内容中完成）

1. 调试程序,分析输出结果

（1）输入并运行以下程序。

```
void main()
{    float a,b;
     a=123456.789e5;
     b=a+20;
     printf("a=%f,b=%f\n",a,b);
}
```

将第二行改为：

```
double a,b;
```

重新运行该程序，分析运行结果。

说明：由于实型变量的值是用有限的存储单元存储的，因此其有效数字的位数是有限的。float 型变量最多只能保证 7 位有效数字，后面的数字是无意义的，不能准确表示该数。

（2）输入并运行以下程序。

```
void main()
{   char c1,c2;
    c1=97;c2=98;
    printf("%c %c\n",c1,c2);
    printf("%d %d\n",c1,c2);
}
```

① 将第二行改为：int c1,c2;再运行。

② 再将第三行改为：c1=300;c2=400;再运行，分析运行结果。

说明：字符型数据可作为整型数据处理，整型数据也可以作为字符型数据处理，但应注意字符数据只占一个字节，它只能存放 0-255 范围的整数。

2．完成以下填空，并把程序调通，写出运行结果。

下面的程序计算由键盘输入的任意两个整数的平均值：

```
void main()
{   int x,y ;
    _____;
    scanf("%d,%d",&x,&y);
    _____;
    printf("The average is :%f ",a);
}
```

3. 指出以下程序的错误并改正，上机把程序调通

```
void main();
{    int a;
     a=5;
     printf("a=%d, a)
}
```

4. 编写程序并上机运行

要将“China”译成密码，译码规律是：用原来字母后面的第 3 个字母代替原来的字母。例如，字母“A”后面第 4 个字母是“E”，用“E”代替“A”。因此，“China”应译为“Fklqd”。请编一程序，用赋初值的方法使 c1、c2、c3、c4、c5 五个变量的值分别为‘C’、‘h’、‘i’、‘n’、‘a’，经过运算，使 c1、c2、c3、c4、c5 分别变为‘F’、‘k’、‘l’、‘q’、‘d’、，并输出。输入程序，并运行该程序。分析是否符合要求。

五、思考与总结

（1）总结各种整型变量的取值范围。

（2）总结各种实型变量的有效数字位数和取值范围。

（3）总结算术运算符和自加、自减运算符的优先级与结合性。

实验三　选择结构程序设计

一、实验学时

2 学时

二、实验目的

（一）掌握 C 语言关系表达式和逻辑表达式的运算和使用；

（二）正确使用条件控制语句（if 语句、switch 语句）进行选择结构程序设计。

三、预习要求

（一）关系运算符和关系表达式、逻辑运算符和逻辑表达式；

（二）if 语句的三种形式（单分支、双分支、多分支），以及 if 语句的嵌套；

（三）switch 语句的形式。

四、实验内容

（一）分析下面程序，掌握关系及逻辑表达式的运算规则。

```
/*  c3-1.c    关系及逻辑表达式运算规则   */
#include "stdio.h"
void main( )
{   int a=3,b=5,c=8;
    if(a++<3 && c--!=0)   b=b+1;
    printf("a=%d\tb=%d\tc=%d\n",a,b,c);
}
```

注意该程序中的条件判断表达式 a++<3 && c--!=0 是一个逻辑表达式，关系表达式 a++<3 的值为假，因此后一部分 c--!=0 就不再计算。试比较下列各部分运行结果。

```
#include "stdio.h"
void main( )
{   int a=3,b=5,c=8;
    if(a++<3 && c--!=0)   b=b+1;
    printf("a=%d\tb=%d\tc=%d\n",a,b,c);

    int a=3,b=5,c=8;
    if(c--!=0 && a++<3)   b=b+1;
    printf("a=%d\tb=%d\tc=%d\n",a,b,c);

    int a=3,b=5,c=8;
    if(a++<3 || c--!=0)   b=b+1;
    printf("a=%d\tb=%d\tc=%d\n",a,b,c);

    int a=3,b=5,c=8;
    if(c--!=0 || a++<3)   b=b+1;
    printf("a=%d\tb=%d\tc=%d\n",a,b,c);   }
```

（二）输入下面两段程序并运行，掌握 case 语句中 break 语句的作用。

1．/*　c3-2.c　　不含 break 的 switch */	/*　c3-3.c　　含 break 的 switch */
#include "stdio.h"	#include "stdio.h"
void main( )	void main( )
{ int a,m=0,n=0,k=0;	{ int a,m=0,n=0,k=0;
scanf("%d",&a);	scanf("%d",&a);
switch(a)	switch(a)
{ case　1: m++;	{ case 1: m++; break;
case　2:	case 2:
case　3: n++;	case 3: n++; break;
case　4:	case 4:
case　5: k++;	case 5: k++;
}	}
printf("%d,%d,%d\n",m,n,k);	printf("%d,%d,%d\n",m,n,k);
}	}

分别从键盘上输入 1、3、5，写出程序运行的结果。

（三）完善程序，从键盘上输入 x 的值，按下式计算 y 的值。

$$y=\begin{cases} x & x<1 \\ 2x-1 & 1\le x<10 \\ 3x-11 & x\ge 10 \end{cases}$$

*编程提示：*注意逻辑表达式的正确表达方法，数学中的 1≤x<10 应使用 C 语言的逻辑表达式（x>=1 && x<10）来表示。

下面是用多分支选择结构实现本题的程序结构：

```
/* c3-4.c　if 语句实现的多分支结构 */
#include "stdio.h"
void main( )
{ 定义变量;
  输入 x;
  if ( _____ )                  //按 y=x 为变量 y 赋值;
    else if( _____ )            //按 y=2x-1 为变量 y 赋值;
          else
                _____           //按 y=3x-11 为变量 y 赋值;
  printf("y=%f\n",y);
}
```

*注意：在赋值语句中 2x 应该写成 2*x。*

（四）编写程序，给出一个百分制成绩，要求输出相应的等级 A、B、C、D、E。90 分以上为'A'，80～89 分为'B'，70～79 分为'C'，60～69 分为'D'，60 分以下为'E'。

编程提示：

1．先定义一个整型变量存放百分制成绩、定义一个字符型变量存放相应的等级成绩；

2．输入百分制成绩；

3．将百分制成绩按 10 分，分档作为 switch 语句中括号内的表达式；

4．按　case 10:

case　9:

case　8:

case　7:

case　6:

default:

这六种匹配情况分别选择不同的入口；

5．输出转换后的等级成绩。

（五）下面程序运行时从键盘上分别输入(20,15)，(15,20)，写出运行结果。

```
/* c3-5.c　分支结构中的复合语句*/
#include "stdio.h"
void main( )
{    int a,b,t;
     t = 0;
     scanf("%d,%d",&a,&b);
     if (a>b)
       { t = a ;
         a = b ;
         b = t ;
        }
  printf("a=%d,b=%d\n",a,b) ;
}
```

（六）编写程序，给出一个不多于 3 位的正整数 n，要求：（1）求出它是几位数；（2）分别打印出每一位数字（数字之间加一个空格）；（3）按逆序打印出各位数字（数字之间加一个空格）。

编程提示：

1．定义变量（考虑需要几个变量）并输入一个 3 位以下的正整数 n

2．将 n 拆分成三个一位数:

表达式：n%10　可将一个三位数 n 拆分出三位数中的个位数；

表达式：n/100　可将一个三位数 n 拆分出三位数中的百位数；

表达式：((n%100))/10　或(n-(n/100)*100)/10　可将一个三位数 n 拆分出三位数中的十位数。

3．用一个嵌套的选择结构，按照百位数、十位数是否为 0 决定 n 为几位数。

4．按相反的顺序输出 n。

思考：如果是对一个 5 位的正整数进行上述处理，程序应如何改动？

（七）写出与表达式 z =（ x>=y ? x : y ）等价的 if 语句，并上机验证。

五、实验注意事项

（一）C 程序中表示比较运算的等号用“==”表示，赋值运算符用“=”表示，不能将

赋值号“=”用于比较运算。

（二）控制表达式是指任何合法的 C 语言表达式（不只限于关系或逻辑表达式），只要表达式的值为“非零”，则为“真”，“零”则为“假”

（三）在 if 语句的嵌套结构中，else 与 if 的配对原则是：每个 else 总是与同一个程序中、在前面出现的、而且距它最近的一个尚未配对的 if 构成配对关系。

（四）case 及后面的常量表达式，实际仅是起标号作用。控制表达式的值与某个情况常量一旦匹配，那么，在执行下面语句的过程中，只要不遇到 break 语句，就一直执行下去，而不再判别是否匹配。允许出现多个“case”与一组语句相对应的情况。

六、思考题

（一）下面程序的功能是实现表达式 z=（ x>=y ? x : y），请将程序填写完整。

```
/*  分支结构的程序  */
#include "stdio.h"
void main( )
{    int x, y, z;
     printf("Please input x,y:");
     scanf("%d%d",&x,&y);
     if (__________) z=x;
       else z=y;
     printf("z=%d ",z);
}
```

（二）下面程序的运行结果为_____。

```
/*  分支结构的程序  */
#include "stdio.h"
void main( )
{    int a=1,b=5,c=8;
     if(a++<3 && c--!=0)   b=b+1;
     printf("a=%d,b=%d,c=%d \n",a,b,c);
}
```

（三）程序填空，从键盘上输入 x 的值，按下式计算 y 的值。

$$y=\begin{cases} x & x<1 \\ 2x-1 & 1\le x<10 \\ 3x-11 & x\ge 10 \end{cases}$$

```
/*  多分支结构的程序  */
#include "stdio.h"
void main( )
{    float x,y;
     printf("x=");
     scanf("%f",&x);
     if ( __________ )   y=x;
     else if( x>=1 && x<10 )
             y=2*x-1;
          else
            y=3*x-11;
```

```
    printf("y=%f\n",y);
  }
```

（四）下面程序运行时从键盘上输入 15,20，运行结果为_____。

```
/*  分支结构的程序  */
#include "stdio.h"
void main( )
{   int a,b,t;
    t = 0;
    scanf("%d,%d",&a,&b);
    if (a>b)
      { t = a ;
        a = b ;
        b = t ;
      }
    printf("b=%d\n",b) ;
}
```

实验四　循环结构程序设计（一）

一、实验学时

2 学时

二、实验目的

（一）掌握用 while,do-while，for 语句实现循环的方法；

（二）掌握在设计条件型循环结构的程序时，如何正确地设定循环条件，以及如何控制循环的次数。

（三）掌握与循环有关的算法。

三、预习要求

预习教材有关 while,do-while,for 语句的语法格式，并能通过这三种语句编写、调试单层循环结构的程序。

四、实验内容

（一）分析并运行下面程序段，循环体的执行次数是_____。

```
int a=10,b=0;
do { b+=2;a-=2+b;}  while(a>=0);
```

（二）当执行以下程序段时，循环体执行的次数是_____。

```
x = -1;
do { x=x*x; } while( !x);
```

（三）编程求 1！+2！+3！+……+20！的值。

*注意：*根据题目，考虑所定义的各个变量应该为何种类型。程序结构如下：

```
/*  c4-1.c  求 1！+2！+3！+……+20！  */
#include "stdio.h"
void main( )
{ 定义变量 i 作为循环控制变量;
  定义变量 p 和 sum 分别存放各个整数的阶乘和阶乘之和;
  变量 p 和 sum 赋初值;
  for( i=1; i<=20; i++ )
  { 变量 p 连乘 ;
    变量 sum 累加; }
  输出 sum 的值 ;
}
```

（四）编写一个程序，求出两个数 m 和 n 的最大公约数和最小公倍数。

*编程提示：*求最大公约数的方法有三种：

1．从两个数中较小数的开始向下判断，如果找到一个整数能同时被 m 和 n 整除，则终止循环。设 n 为 m 和 n 中较小的数，则如下程序段可实现：

```
for(k=n; k>=1; k--)  if(m%k==0 && n%k ==0) break;
```

k 即为最大公约数。

```
/*  c4-2.c  求最大公约数算法 1  */
  #include "stdio.h"
  void main( )
  {


  }
```

2．从整数 1 开始向上找，直至 m 和 n 中较小的数，每找到一个能同时被 m 和 n 整除的整数，将其存入一个变量中，当循环结束时，变量中存放的即为最大公约数。设 n 为 m 和 n 中较小的数，则如下程序段可实现：

```
for(k=1; k<=n; k++) if(m%k==0 && n%k==0) x=k;
```

变量 x 的值即为最大公约数。

```
/*  c4-3.c  求最大公约数算法 2  */
  #include "stdio.h"
  void main( )
  {


  }
```

3．用辗转相除法，即将求 m 和 n 的最大公约数问题转化为求其中的除数和两个数相除所得余数的公约数。每次循环中，先求两个数的余数，然后以除数作为被除数，以余数作为除数，当余数为 0 时结束循环，此时除数即为最大公约数。设 m 和 n 中 n 为较小的数，则可用如下程序段实现：

```
  b=m%n;
  while(b!=0)
  { m=n; n=b; b=m%n;}
  printf("%d\n",n);
```

```
/*  c4-4.c  求最大公约数算法 3  */
  #include "stdio.h"
  void main( )
  {


  }
```

类似地，求最小公倍数的方法也可以从 m 和 n 中较大的数开始向上找，或者从 m*n 向下找，请自己考虑程序的设计方法。

另外，两个数的最大公约数和最小公倍数的关系为：最小公倍数=m*n/最大公约数

可利用此关系进行程序设计。

（五）编程实现，从键盘上输入一行字符，统计其中英文字母、数字、空格和其它字符

的个数。

*编程提示：*先定义一个字符型的变量（如 c），再定义 4 个整型变量作为计数器，作为计数器的变量要先赋初值 0。在循环中每次从键盘上读入一个字符，在循环体中对读入的字符进行判断，相应的计数器加 1，当读入的字符为’\n’时结束。

编程中可使用如下的循环结构：

```
while((c=getchar())!=’\n’))
{ if(……)  ……;
  else if(……) ……;
    ……;
  else  ……;
}
```

```
/*  c4-5.c    统计字符串中指定字符的个数  */
#include "stdio.h"
void main( )
{


}
```

注意：

1. while((c=getchar( ))!=’\n’))中括号的使用，第二层的小括号不能省略，想一想为什么？

2. 字符常量’0’与数值常量 0 是不同的。

（六）下面程序的功能是：计算 1 到 100 之间的奇数之和及偶数之和，并输出。请在程序中的横线上填入适当的内容，将程序补充完整并运行。

```
/*  c4-6.c    计算 1 到 100 之间的奇数之和及偶数之和  */
#include "stdio.h"
void main( )
{ int a,b,c,i;
  _____;                    /*变量初始化*/
  for(i=0; i<=100; i+=2)
  { a+=i;                   /*变量 a 存放偶数的和*/
   _____;
    c+=b;                   /*变量 c 存放奇数的和*/
  }
  printf("sum of evens is %d\n",a);
  printf("sum of odds is %d\n",_____);
}
```

（七）编程打印出所有的“水仙花数”，所谓水仙花数是指一个 3 位数，其各位数字的立方和等于该数本身。如 $153=1^3+3^3+5^3$。

*编程提示：*定义一个变量作为循环变量，再定义 3 个变量分别存放三位数的每位数字，在循环体中将一个三位数拆分成个位、十位、百位后判断循环变量的值是否为水仙花数，如果是则输出，否则不输出。程序的基本结构为：

```
/*  c4-7.c    打印出所有的“水仙花数”  */
#include "stdio.h"
```

```
  void main( )
{ 定义 4 个整型变量;
    for( j=100; j<=999; j++ )
    {  a=j/100;                    /*分离出百位数*/
       b=j/10-a*10;                /*分离出十位数*/
       c=_____________;            /*分离出个位数*/
      if(j==a*a*a+b*b*b+c*c*c)
        输出 j ;
    }
    printf("\n");
  }
```

（八）以下程序的功能是:从键盘上输入若干个学生的成绩，统计并输出最高成绩和最低成绩,当输入负数时结束输入。请将程序补充完整。

```
/*   c4-8.c   求最大值最小值程序   */
  #include "stdio.h"
  void main( )
  { float x,amax,amin;
    scanf("%f",&x);
    amax=x;
    amin=x;
    while ( ___________________ )
    { if (x>amax) amax=x;
      if ( ___________________ ) amin=x;
      scanf("%f",&x);
    }
    printf("\namax=%f\namin=%f\n",amax,amin);
  }
```

（九）求两个正整数[m,n]之间所有既不能被 3 整除也不能被 7 整除的整数之和。

编程提示：定义两个变量 m,n 和 t，再定义一个循环变量和结果变量 s,从键盘输入 m 和 n 的值，判断两个变量的值，如果 m>n，则交换两个变量。然后用循环依次判断 m 和 n 之间的每一个数，在循环体中通过条件语句来判断这个数是否既不能被 3 整除也不能被 7 整除，如果满足条件，累加求和，如果不满足，则继续循环。

程序的基本结构如下：

```
/*   c4-9.c    按条件求数列和   */
  #include "stdio.h"
  void main( )
  { 定义变量;
    变量赋初值;
    输入 m,n 的值;
    if( m>n )
      m 和 n 交换;
    for ( ___________________ )
      if ( i%3 != 0 && i%7 != 0 )
        变量 s 累加求和 ;
    printf("Sum is : %ld \n",s );
  }
```

（十）下面程序的功能是：计算正整数 num 的各位上的数字之和。例如，若输入：252，

则输出应该是：9；若输入：202，则输出应该是：4。请将程序补充完整。

```
/*   c4-10.c   求整数各位数字和   */
  #include "stdio.h"
  void main( )
  { int num,k;
       ____________________;                    /* k 赋初值 */
    printf("\Please enter a number:") ;
    scanf("%d",&num) ;
    do
    { k=____________________ ;                  /* 取最低位并累加 */
     num/=10;                                   /* 去掉最低位 */
     } while(num) ;
    printf("\n%d\n",k) ;
  }
```

五、实验注意事项

（一）while,do-while，for 语句中应有使循环趋向于结束的语句,否则就可能构成死循环。

（二）while,do-while 语句什么情况下的运行结果是相同的，什么情况下不同。

（三）注意在循环结构程序设计中，正确使用{　}构成复合语句。

六、思考题

（一）求两个正整数 x 和 y 的最大公约数，请填空。

```
/*   求最大公约数程序   */
  #include "stdio.h"
  void main( )
  { int x,y,t,i;
    printf("请输入两个数: ");
    scanf("%d,%d",&x,&y);            /*注意 scanf 语句中间用“，”隔开*/
    if (x > y) {t = x; x = y; y = t;}
    for( ____________________ )
    { if( x%i==0 && y%i==0 )
        break;
    }
    printf("最大公约数是 : %d\n",i);
  }
```

（二）计算 1 到 100 之间的奇数之和及偶数之和。请填空。

```
/*   计算 1 到 100 之间的奇数之和及偶数之和   */
  #include "stdio.h"
  void main( )
  { int a,b,c,i;
    a=0, c=0;                        /* 变量赋初值 */
    for(i=0;i<=100;i+=2)
    { a+=i;                          /* 变量 a 存放偶数的和 */
      ____________________ ;
      c+=b;                          /* 变量 c 存放奇数的和 */
    }
    printf("Sum of Evens is %d\n",a);
```

```
    printf("Sum of Odds is %d\n",c-101);
  }
```

（三）下面程序的功能是：计算正整数 num 的各位上的数字之和。例如，若输入：252，则输出应该是：9；若输入：202，则输出应该是：4。请将程序补充完整。

```
/*  计算整数各位数字和  */
  #include "stdio.h"
  void main( )
  { int num,k=0;
    printf("请输入一个整数: ") ;
    scanf("%d",&num) ;
    do
    { k=_____ ;
      num/=10;
    } while(num) ;
    printf("\n%d\n",k) ;
  }
```

（四）求两个正整数[m,n]之间所有既不能被 3 整除也不能被 7 整除的整数之和。请填空。

```
/*  按条件求数列和  */
#include "stdio.h"
  void main( )
  { int m,n,i,t;
    long int s=0;
    scanf("%d,%d",&m,&n);
    if( m>n )
    { t=m; m=n; n=t;}
    for ( _____ )
      if ( i%3 != 0 && i%7 != 0 )
         s += i;
    printf("Sum is : %ld\n",s );
  }
```

实验五　循环结构程序设计（二）

一、实验学时

2 学时

二、实验目的

（一）掌握使用 for,while,do-while 语句实现循环嵌套的方法；
（二）巩固 break 和 continue 语句的使用。

三、预习内容

预习教材中有关用 for,while,do-while 语句实现循环嵌套的方法以及循环嵌套的执行过程。

四、实验内容

（一）根据公式：sum=$1+\frac{1}{2!}+\frac{1}{3!}+\ldots+\frac{1}{n!}$，计算 sum 的值。

注意：根据题目，考虑所定义的各个变量应该为何种类型。
编程提示：定义一个变量存放最后的求和结果（假设为 sum），sum 的数据类型应为实型，定义变量 t 计算整数的阶乘。使用双重循环，程序的基本结构为：

```
for(i=1,sum=0; i<=20; i++)
{ t 赋初值 1 ;
  for( j=1; j<=i; j++ )
    变量 t 连乘求积;
  变量 sum 累加 t 的倒数;
}
```

注意上述程序结构和内循环变量的终值。想一想是否可以将 t=1 放在外循环之前？

```
/*  c5-1.c   求数列和  */
#include "stdio.h"
void main( )
{


}
```

（二）编程输出九九乘法表。
编程提示：我们日常看到的乘法表是：

```
1×1=1
1×2=2   2×2=4
1×3=3   2×3=6   3×3=9
…
1×9=9   …       …       …      9×9=81
```

每个算式可以归为：i×j=? 的形式，而且每行中的算式数量随着行数变化。考虑外层循环变量和内层循环变量应当取何值呢？

程序的基本结构为：

```
for( i=1; i<=9; i++ )
{ for(j=1; __________; j++ )
    输出乘法算式;
  输出回车换行符;
}
```

/*　c5-2.c　输出九九乘法表　*/

```
#include "stdio.h"
void main( )
{


}
```

（三）编程求 100～300 之间的素数和。

编程提示：首先，弄清素数的概念是本题的关键，素数：只能被 1 和它本身整除的数为素数。判断一个数是否为素数需要使用循环结构才能实现，求出 100～300 之间的全部素数要使用循环的嵌套结构。程序结构提示如下：

/*　c5-3.c　求 100～300 之间的素数和　*/

```
#include "stdio.h"
void main( )
{ 定义变量;
  外层循环变量 i 从 100 递增到 300
  {  标志变量赋 0;
     内层循环变量从 2 递增到 i-1
        如果不是素数（能整除），则标志变量赋 1,跳出循环;
     如果标志变量为 0（是素数），进行求和; }
  输出求和结果;
}
```

（四）编程输出以下图形。

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

编程提示：输出图形的这一类问题，首先要看一看图形的特点，找到规律：一共有几行，每行先输出什么字符，输出几个；后输出什么字符，输出几个。一般外循环变量控制行数，内循环变量控制各种字符的数量。

程序的基本结构为：

```
for(i=0; i<=2; i++)
{ 连续输出若干空格;
  连续输出若干个“*”;
  输出一个换行;  }
```

/*　c5-4.c　输出字符图形　*/

```
#include "stdio.h"
void main( )
{

}
```

想一想，输出下面的三种图形分别应当怎样实现：

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

（五）运行以下程序，分析程序的运行结果并上机验证。

```
/*  c5-5.c   循环结构程序  */
  #include "stdio.h"
  void main( )
  { int i=0, a=0;
    while( i<20 )
    { for( ; ; )
      { if((i%10)==0) break;
        else i--;}
        i+=11;a+=i;
    }
    printf("%d\n",a);
  }
```

五、实验注意事项

（一）对于双重循环来说，外层循环往往是控制变化较慢的参数（例如所求结果的数据项的个数、图形的行数等），而内层循环变化快，一般控制数据项的计算、图形中各种字符的数量等。

（二）注意在循环结构程序设计中，正确使用{ }构成复合语句。

（三）外层循环变量增值一次，内层循环变量从初值到终值执行一遍。

（四）程序书写时，最好使用缩进结构以使程序结构清晰。

六、思考题

（一）根据公式：sum=$1+\frac{1}{2!}+\frac{1}{3!}+\ldots+\frac{1}{n!}$，计算 sum 的值，请把程序补充完整。

```
/*  求数列和  */
  #include "stdio.h"
  void main( )
  { long int n,j;
    float sum=0,t;
    for(n=1; n<=20; n++)
    { t=1.0;
      for(j=1; j<=n; j++)
        t=t*j;
      ____________________;
```

```
    }
    printf("%10.2f\n",sum);
  }
```

（二）下面函数的功能是输出九九乘法表，请把程序补充完整。

```
/*  输出九九乘法表  */
#include "stdio.h"
  void main( )
{ int i,j;
   for( i=1; i<=9; i++ )
    { for( j=1; _____________; j++ )
        printf("%d*%d=%-4d", i,j,i*j);
      printf("\n"); }
}
```

（三）下面函数的功能是求出 100～300 间的素数和，请把程序补充完整。

```
/*  求 100～300 间的素数和  */
#include "stdio.h"
void main( )
{   int i, j, flag, sum=0;
    for( i=100; i<=300; i++ )
{ flag=0;
      for( j=2; j<=i-1; j++ )
        if( i%j==0 ) { flag=1;break; }
      if( __________ ) sum+=i;
}
    printf("The sum is %d\n",sum);
  }
```

（四）下面程序的功能是：输出以下图形：

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

请把程序补充完整。

```
/*  输出字符图形  */
#include "stdio.h"
  void main( )
{ int i,j,k;
    for(i=0;i<=2;i++)
    { for( j=0; j<=10-i ;j++ )  printf(" ");
      for( k=0; ____________________ ;k++ )  printf("*");
      printf("\n"); }
  }
```

实验六　一维数组程序设计

一、实验学时

2 学时

二、实验目的

（一）掌握一维数组的定义、初始化方法；
（二）掌握一维数组中数据的输入和输出方法；
（三）掌握与一维数组有关的程序和算法；
（四）了解用数组处理大量数据时的优越性。

三、预习要求

（一）理解数组的概念、利用数组存放数据有何特点；
（二）一维数组的定义、初始化方法；
（三）一维数组中数据的输入和输出方法。

四、实验内容

（一）下面的几个程序都能为数组元素赋值，请输入程序并运行。比较一下这些赋值方法的异同。

1．在定义数组的同时对数组初始化。

```
/*    c6-1.c      在定义数组的同时对数组初始化*/
#include "stdio.h"
void main( )
{ int a[4]={0,1,2,3};
  printf("\n%d   %d   %d   %d\n",a[0],a[1],a[2],a[3]);
}
```

2．不使用循环对单个数组元素赋值。

```
/*    c6-2.c      不使用循环对单个数组元素赋值*/
#include "stdio.h"
void main( )
{ int a[4];   a[0]=2;a[1]=4;a[2]=6;a[3]=8;
  printf("\n%d %d %d %d\n",a[0],a[1],a[2],a[3]);
}
```

3．用循环结构，从键盘输入为每个数组元素赋值，输出各数组元素。

```
/*    c6-3.c      利用循环通过键盘对数组元素赋值*/
  #include "stdio.h"
  void main( )
{ int i,a[4];
  for(i=0; i<4; i++)
    scanf("%d",&a[i]);
  printf("\n");
  for(i=0; i<4; i++)
    printf("%d   ",a[i]);
  printf("\n");
```

```
}
```

（二）编写一程序，为一维数组 a 中的元素赋值，并按照逆序输出。

*编程提示：*通过对一维数组的输入输出来实现。

1. 首先复习教材上知识点，充分理解例题。

2. 对一维数组的输入可以参照实验内容（一）中的三种方法，选择其一输出，用循环结构来实现。

3. 注意是逆序输出，可以通过输出时，在 for 语句中利用循环变量递减的方法来实现。

```
/*    c6-4.c      利用循环实现一维数组的输入输出*/
#include "stdio.h"
void main( )
{ int i,a[10];                          /* 定义循环变量 i 和一维数组 a */
  for (i=0;i<=9;i++)
    scanf("%d",&a[i]);
  for (                 _____    _)       /* 请补充完整循环语句 */
    printf("%d   ",a[i]);                 /* 按照逆序输出 */
  printf("\n");
}
```

（三）编写程序，输出一维数组 a 中的元素最小值及其下标。

编程提示：

1. 定义一个整型变量存放最小值下标，将其初始化为 0 ,例如：int p=0;即从数组第零个元素开始判断。

2. 通过循环，依次判断数组中的每一个元素 a[i]是否小于 a[p],如果是，则将 p 和 a[p]的值作相应的改变。

```
/*    c6-5.c      输出一维数组中元素的最小值及其下标*/
#include "stdio.h"
void main( )
{ int i,m,p,a[10]={9,8,7,6,1,3,5,18,2,4}; /* m 为最小值，p 为其下标 */
     m=a[0];
     p=0;
     for(i=1;i<10;i++)
            if (a[i]<m)
{                    ; p=i;}   /* 请补充完整此语句 */
     printf("%d,%d\n",a[p],p);      /* 输出一维数组 a 中的最小值及其下标 */
 }
```

（四）编写一程序，求一维数组中下标为偶数的元素之和。

编程提示：

1. 定义一个数组 a 并初始化。

2. 定义一个整型变量 sum，存放下标为偶数的元素和，并初始化为 0。

3. 从数组的第 0 个元素开始，每次循环变量递增 2，一直到数组的最后一个元素，将其累加到 sum 变量。

4. 输出 sum 变量即为下标为偶数的元素之和。

```
/*    c6-6.c      求一维数组中下标为偶数的元素之和*/
   #include "stdio.h"
void main( )
{ int   i,sum=0;                              /* 初始化 sum 为 0 */
int   a[]={2,3,4,5,6,7,8,9};
```

```
for(i=0; i<8;                )          /* 请补充完整循环语句 */
sum+=a[i];
printf("sum=%d\n",sum);
}
```

（五）编写一程序，将 100 以内的素数存放到一个数组中。

*编程提示：*这是一个双层循环嵌套的程序。

1．首先复习教材上的内容，掌握判断素数的方法。

2．定义一个数组存放 100 以内的素数，想一想该数组的大小应该为多少？

3．定义一个整型变量作循环变量。

4．定义一个整型变量作为数组元素下标的计数器，想一想该变量应赋什么样的初值？

5．在外层循环中，对 1～100 之间的所有整数进行判断；内层循环则判断每个整数是否为素数。如果是素数，存放到数组中，并使数组下标变量加 1；否则继续判断下一个整数。

6．用循环语句输出数组中的所有素数，注意循环变量的初值和终值如何确定。

（六）将一个数组中的值按逆序重新存放。例如，原来的顺序为 8,6,5,4,1,2,要求改为按 2,1,4,5,6,8 的顺序存放（注意是逆序存放而不是逆序输出）。

编程提示：

1．定义一个数组，为该数组赋值（可以在定义时初始化，也可以用循环语句）。

2．在循环中，使第 0 个元素与第 5 个元素交换，第 1 个元素与第 4 个元素交换，第 2 个元素与第个 3 元素交换（注意循环次数按 n/2 确定，n 为数据个数）。

3．输出逆序存放后的各数组元素（使用循环语句）。

五、实验注意事项

（一）C 规定，数组的下标下界为 0，因此数组元素下标的上界是该数组元素的个数减 1。

例如，有定义：int a[10]; 则数组元素的下标上界为 9。

（二）由于数组的下标下界为 0，所以数组中下标和元素位置的对应关系是：第一个元素下标为 0，第二个元素下标为 1，第三个元素下标为 2，依次类推，第 n 个元素下标为 n-1。

（三）数值型数组要对多个数组元素赋值时，使用循环语句，使数组元素的下标依次变化，从而为每个数组元素赋值。

例如：int a[10],i;

```
for(i=0;i<10;i++)  scanf("%d",&a[i]);
```

不能通过如下的方法对数组中的全部元素赋值。

```
int a[10],i;
scanf("%d",&a[i]);
```

六、思考题

（一）定义一个数组名为 ftop 且有 5 个 int 类型元素的一维数组，同时给每个元素赋初值为 0，请写出数组的定义语句_____。

（二）下面程序的功能是：为一维数组 a 中的元素赋值，并按照逆序输出。请在程序中的横线上填入正确的内容。

```
#include "stdio.h"
void main( )
```

```
{ int i,a[10];                          /* 定义循环变量 i 和一维数组 a */
        for(i=0;i<=9;i++)
          scanf("%d",&a[i ]);
    for(                ;i>=0;i--)
          printf("%d ",a[i]);              /* 按照逆序输出 */
printf("\n");
}
```

（三）下面程序的功能是：输出一维数组 a 中的最小值及其下标。请在程序中的横线上填入正确的内容。

```
#include "stdio.h"
void main( )
{ int i,p=0,a[10];                    /* 定义 a 为数组名，p 为下标名 */
        for(i=0; i<10; i++)
          scanf("%d",&a[i]);
        for(i=1; i<10; i++)
          if (a[i]<a[p])  {                  ; }
        printf("%d,%d",a[p],p);   /* 输出一维数组 a 中的最小值及其下标 */
    }
```

（四）下面程序的功能是：求一维数组中下标为偶数的元素之和并输出。请在程序中的横线上填入正确的内容。

```
   #include "stdio.h"
void main( )
 { int   i,sum=0;
int   a[ ]={2,3,4,5,6,7,8,9};
for(i=0; i<8;                  )
sum+=a[i];
printf("sum=%d\n",sum);
 }
```

实验七　二维数组程序设计

一、实验学时

2 学时

二、实验目的

（一）掌握二维数组的定义、赋值及输入输出的方法；

（二）掌握与二维数组有关的算法如查找、矩阵转置等；

（三）掌握在程序设计中使用数组的方法。数组是非常重要的数据类型，循环中使用数组能更好地发挥循环的作用，有些问题不使用数组难以实现。

（四）掌握在 VC++环境下上机调试二维数组程序的方法，并对结果进行分析。

三、预习要求

熟悉二维数组的定义、引用和相关算法（求最大值、最小值）的程序设计，同时要掌握在程序设计中利用双重循环来实现二维数组的输入和输出。

四、实验内容

（一）二维数组的初始化，即给二维数组的各个元素赋初值。下面的几个程序都能为数组元素赋值，请输入程序并运行，比较这些赋值方法有何异同。

1．在定义数组的同时对数组元素分行初始化。

```
/*    c7-1.c      二维数组的初始化(分行)*/
#include "stdio.h"
void main( )
{ int i,j,a[2][3]={{1,2,3},{4,5,6}};
for(i=0; i<2; i++)
{ for(j=0; j<3; j++)
printf("%d ",a[i][j]);
printf("\n");
}
}
```

2．不分行的初始化。把{ }中的数据依次赋值给数组的各个元素。

```
/*    c7-2.c      二维数组的初始化(不分行)*/
#include "stdio.h"
void main( )
{ int i,j,a[2][3]={1,2,3,4,5,6};
     for(i=0;i<2;i++)
      { for(j=0;j<3;j++)
           printf("%d ",a[i][j]);
           printf("\n");
       }
}
```

3．为部分数组元素初始化。

如：数组定义语句为：int i,j,a[2][3]={{1,2},{4}};

4．可以省略第一维的定义，但不能省略第二维的定义。

如：int a[ ][3]={1,2,3,4,5,6};

依次运行以上程序，比较这四种定义方式的不同之处。

（二）求一个 4×4 矩阵的主对角线元素之和，填空并运行程序。

编程提示：

1．定义一个 4 行 4 列的二维数组 a。

2．可利用双重循环的嵌套为该二维数组的各个数组元素赋值，一般格式为：

```
for(i=0; i<4; i++)
    for(j=0; j<4; j++)
     scanf("%d",&a[i][j]);
```

3．用循环求和，并注意矩阵对角上线元素的特征是：行下标和列下标相同。

4．输出对角线元素之和。

```
/*    c7-3.c      求一个 4×4 矩阵的主对角线元素之和*/
#include "stdio.h"
void main( )
{ int a[4][4]={{1,2,3,4},{5,6,7,8},{3,9,10,2},{4,2,9,6}};
int i,sum=0;
for(i=0; i<4; i++)
        _____;                    /*把对角线元素的和放在变量 sum 中 */
printf("sum=%d\n",sum);          /*输出对角线元素的和*/
}
```

（三）打印出以下的杨辉三角（要求打印出 10 行）

```
1
1    1
1    2    1
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
```

编程提示：

1．杨辉三角的特点是：第 1 列和对角线上的元素为 1，其它各元素的值都是上一行上一列元素和上一行前一列元素之和。

2．定义一个 10×10 的二维数组 a。

3．先用一个单层循环为第一列和对角线上的元素赋值。如：

```
for(i=0; i<10; i++)  { a[i][i]=1;a[i][0]=1; }
```

4．再用一个双重循环嵌套为其它元素赋值。

```
for(_____; i<10; i++)
  for(_____; _____; j++)
```

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

想一想，划横线的地方应当填入什么内容？

5．计算之后再用一个双重循环嵌套输出杨辉三角。注意各层循环变量的初值和终值怎样确定。

（四）统计 3 个学生，每个学生 4 门课程的考试成绩，要求输出每个学生的总成绩，每个学生的平均成绩，3 个学生的总平均成绩。填空并运行程序。

/*　c7-4.c　学生成绩处理 */

```
#include "stdio.h"
void main( )
{ int  stu[3][4],i,j,t[3];
  float  sum=0,a[3];
  for(i=0; i<3; i++)            /* 输入三个学生的 4 门课程考试成绩 */
    for(j=0; j<4; j++)
    scanf("%d",&stu[i][j]);
  for(i=0; i<3; i++)
  { t[i]=0;
    for(j=0; j<4; j++)
    { sum+=stu[i][j];           /* sum 存放三个学生的 4 门课程总成绩 */
      t[i]+=stu[i][j];          /* t[i]存放第 i 个学生的 4 门课程成绩 */
    }
    printf("%-6d",t[i]);        /* 输出第 i 个学生的总成绩 */
    _____________________;
    printf("%-6.2f\n",a[i]);  /*a[i]存放第 i 个学生的 4 门课程平均成绩*/
  }
  printf("average = %.2f\n",sum/12.0);
}
```

（五）已知二维数组 a 中的元素为：

```
 4    4   34
37    3   12
 5    6    5
```

求二维数组 a 中的最大值和最小值。程序的输出应为：The max is: 37
The min is: 3

填空并运行程序。

/*　c7-5.c　求二维数组中元素的最大值与最小值 */

```
#include "stdio.h"
void main( )
{ int a[3][3]={4,4,34,37,3,12,5,6,5},i,j,max,min;
  max=a[0][0];
  _____________________;
  for( i=0; i<3; i++)
    for( j=0; j<3; j++)
    { if ( max < a[i][j] )   max = a[i][j];
```

```
        if ( min > a[i][j] )   min = a[i][j];
      }
    printf("The max is: %d\n", max);
    printf("The min is: %d\n", min);
  }
```

（六）以下程序是查找二维数组 a 的最大元素及其下标，填空并运行程序。

```
/*   c7-6.c     求二维数组中元素的最大值及其下标 */
  #include "stdio.h"
  void main()
  { int a[4][4]={{1,2,3,4},{3,4,5,6},{5,6,7,8},{7,8,9,10}};
    int i,j,max,l,c;          /* max 存放最大值，l、c 分别存放行和列的下标 */
    max=a[0][0];
    for(i=0; i<4; i++)
      for(j=0; j<4; j++)
        if(max<a[i][j]) {_____________;l=i; c=j;}
    printf("max=%d,l=%d,c=%d%\n",max,l,c);
  }
```

（七）下面的程序是自动形成并输出如下矩阵，填空并运行程序。

```
    1    2    3    4    5
    1    1    6    7    8
    1    1    1    9    10
    1    1    1    1    11
    1    1    1    1    1
```

```
/*   c7-7.c     生成指定的矩阵*/
  #include "stdio.h"
  void main( )
   { int i,j,k,a[5][5];
     k=2;
     for(i=0; i<5; i++)                  /* 行循环 */
       for(j=0; j<5; j++)                /* 列循环 */
         if(_____________)  a[i][j]=1;   /* 产生矩阵的下三角元素 */
         else a[i][j]=k++;               /* 产生矩阵的上三角元素 */
     for(i=0; i<5; i++)
     { for(j=0; j<5; j++)
         printf(“%4d”,a[i][j]);
         printf(“\n”);                   /* 每输出一行后换行 */
     }
   }
```

（八）下面程序的功能是实现方阵（如：3 行 3 列）的转置（即行列互换）

如：如果原来的矩阵：100　　200　　300
　　400　　500　　600
　　700　　800　　900

程序输出应为：
　　100　　400　　700
　　200　　500　　800
　　300　　600　　900

填空并运行程序。

```
/*    c7-7.c      矩阵转置*/
   #include "stdio.h"
   void main( )
  { int i,j,temp;
    int array[3][3]={{100,200,300},{400,500,600},{700,800,900}};
    printf("\n 原来的矩阵为：\n");
    for(i=0; i<3; i++)
      { for(j=0; j<__________; j++)  printf("%7d",array[i][j]);
        printf("\n");}
    for(i=0; i<3; i++)
      { for(j=0;__________;j++)
        {temp=array[i][j];array[i][j]=array[j][i];array[j][i]=temp;} }
    printf("\n 转置后的矩阵为：\n");
    for(i=0; i<3; i++)
      { for(j=0; j<3; j++)  printf("%7d",array[i][j]);
        ____________________}              /*输出一行后要换行*/
  }
```

请将此矩阵转置程序与教材上的矩阵转置程序进行比较，各用于何种类型矩阵比较合适？

五、实验注意事项

（一）C 规定，二维数组的行和列的下标都是从 0 开始的。

例如，有定义：int b[3][[5]; 则数组 b 的第一维下标的上界为 2，第二维下标的上界为 4。说明定义了一个整型二维数组 b,它有 3 行 5 列共 3*5=15 个数组元素,行下标为 0,1,2,列下标为 0,1,2,3,4,则数组 b 的各个数组元素是：

```
b[0][0],b[0][1],b[0][2],b[0][3],b[0][4]
b[1][0],b[1][1],b[1][2],b[1][3],b[1][4]
b[2][0],b[2][1],b[2][2],b[2][3],b[2][4]
```

（二）要对二维数组的多个数组元素赋值，应当使用循环语句并在循环过程中，使数组元素的下标变化。可用下面的方法为所有数组元素赋值：

```
int i,j,a[3][3];
for(i=0; i<3; i++)
  for(j=0; j<3; j++)
```

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

六、思考题

（一）定义一个 5 行 5 列的二维数组 a，使下三角的所有元素初始化为 1，在划线处填空。

```
int i,j,a[5][5];
for(i=0; i<5; i++)
  for(j=0; j<5; j++)
    if(_____)  a[i][j]=1;
```

（二）求一个 4×4 矩阵的主对角线元素之和，填空并运行程序。

```
#include "stdio.h"
void main( )
{ int a[4][4]={{1,2,3,4},{5,6,7,8},{3,9,10,2},{4,2,9,6}};
  int i,sum=0;
  for(i=0; i<4; i++)
    _____;
  printf("sum=%d\n",sum);
}
```

（三）求二维数组 a 中的最大元素及其下标，填空并运行程序。

```
#include "stdio.h"
void main( )
{int a[4][4]={{1,2,3,4},{3,4,5,6},{5,6,7,8},{7,8,9,10}};
 int i,j,max,l,c;
 max=a[0][0];
 for(i=0; i<4; i++)
   for(j=0; j<4; j++)
     if( max<a[i][j] )  {_____;l=i;c=j;}
 printf("max=%d,l=%d,c=%d%\n",max,l,c);
}
```

（四）统计 3 个学生，每个学生 4 门课程的考试成绩，要求输出每个学生的总成绩，每个学生的平均成绩，3 个学生的总平均成绩，填空并运行程序。

```
#include "stdio.h"
void main( )
{ int  stu[3][4],i,j,t[3];
  float a[3],sum=0;
  for(i=0; i<3; i++)
    for(j=0; j<4; j++)
    scanf("%d",&stu[i][j]);
  for(i=0; i<3; i++)
  { t[i]=0;
    for(j=0; j<4; j++)
```

```
        { sum+=stu[i][j];
          t[i]+=stu[i][j];
        }
        printf("%-6d",t[i]);
        _____ ;
        printf("%-6.2f\n",a[i]);
    }
    printf("average = %.2f\n",sum/12.0);
}
```

实验八　字符数组程序设计

一、实验学时

2 学时

二、实验目的

（一）掌握字符数组的定义、初始化和应用；
（二）掌握字符串处理函数的使用。

三、预习要求

重点预习的内容：C 语言中字符串的存储表示；字符数组输入输出的方法；常用的字符串处理函数的使用。

四、实验内容

（一）输入下面的程序并运行，观察程序运行的结果，并分析原因（注意程序第 2 行中有些单引号之间是空格）。

```
/*    c8-1.c    字符数组的输出*/
 #include "stdio.h"
 void main( )
{ char a[10]={'I', ' ', 'a', 'm', ' ','a',' ', 'b', 'o', 'y'};
  printf("%s\n",a);
}
```

将字符数组 a 的大小改为 11，再运行程序，并将结果与修改前的结果进行比较，分析原因。

（二）按照要求编写程序：有一行文字，不超过 80 个字符，分别统计出其中英文大写字母、小写字母、数字、空格、及其它字符的个数。

编程提示：

1. 定义一个一维字符数组。
2. 定义 5 个整型变量分别统计大写字母、小写字母、数字、空格和其它字符的个数（即作为 5 个计数器使用），并为这 5 个变量赋初值。
3. 用 scanf 函数或 gets 函数为字符数组赋一个字符串。
4. 在循环中对字符数组的每个元素进行判断，相应的计数器加 1。注意循环控制的条件和进行判断的条件怎样设置。
5. 循环结束后输出各计数器的值。

思考：如果是对一篇英文文章进行统计,又该怎么编程呢？文章的行数和每行字数可以自己来设。提示：对文章的内容要用二维字符数组来存储。

```
/*   c8-2.c      统计字符个数   */
  #include "stdio.h"
  void main( )
```

```
{

}
```

（三）下面程序的功能是实现将两个字符串连接起来并输出结果，注意不使用 strcat 函数。请填空并运行程序。

编程提示：

1．定义两个一维字符型数组 str1、str2 和两个循环变量。

2．为两个字符数组输入两个字符串（可使用 scanf 函数或 gets 函数整体赋值，要注意 scanf 和 gets 函数的区别，在对字符串赋值时，scanf 函数不能出现空格）。

3．确定字符数组 str1 结束的位置。

4．再将字符数组 str2 中的内容连接到字符数组 str1 的后面。

5．为字符数组 str1 赋字符串结束的标志'\0'。

6．输出连接后的字符数组 str1。

```
/*    c8-3.c      字符串连接*/
  #include "stdio.h"
  void main( )
 { char str1[100],str2[100];
   int i=0,j=0;
   printf("please input the string1:");
   scanf("%s",str1);
   printf("please input the string2:");
   gets(str2);
   for(i=0; str1[i]!='\0'; i++) ;          /*注意,此处空语句不可少*/
    for(j=0;str2[j]!='\0';j++)
     { str1[i]=str2[j];
        i++;
     }
   _____________ ;                        /*给出新的字符串的结束符*/
   printf("the catenated string is %s",str1);
 }
```

（四）下面程序的功能是用 strcat 函数实现将字符串 2 连接到字符串 1 的后面并输出，请补充完整。

```
/*    c8-4.c      字符串连接*/
  #include "stdio.h"
  void main( )
   { char str1[80]="This Is a ",str2[80]="c Program";
     printf("String1 is: %s\n",str1);
     printf("String2 is: %s\n",str2);
     _____________;          /*使用 strcat 函数实现，注意其格式*/
     printf("Result is: %s\n",str1);
   }
```

（五）下面程序的功能是实现将一个字符串中的所有大写字母转换为小写字母并输出，请补充完整。

例如：当字符串为"This Is a c Program"

输出："this is a c program"

```
/*    c8-5.c      字符串中的大写字母转为小写字母*/
#include "stdio.h"
void main( )
 { char str[80]="This Is a c Program";
   int i;
   printf("String is: %s\n", str);
   for (i=0; str[i]!='\0'; i++)
     if (str[i]>='A' && str[i]<='Z')
       _____________;            /*将大写字母转换成小写字母*/
   printf("Result is: %s\n",str);
 }
```

思考：如果将字符串中的所有小写字母转换为大写字母，又将如何修改程序？

（六）编写程序实现在一个字符串中查找指定的字符，并输出指定的字符在字符串中出现的次数及位置，如果该字符串中不包含指定的字符，请输出提示信息。

编程提示：

1．定义两个一维数组，a 字符数组用来存放字符串，b 整数数组用来存放指定的字符在字符串中出现的位置(即对应的下标)。

2．定义 i,j,m 三个循环控制变量和一个标志变量 flag，并初始化 flag 的值为 0。

3．用 scanf 或者 gets 函数为字符数组赋一个字符串。

4．在循环中对字符数组的每个元素和指定字符 ch 进行匹配判断，如果相同，就把其下标依次存放在数组 b 中，并置 flag 的值为 1。

5．循环退出后判断标志变量 flag 的值，如果仍为 0，说明字符串中没出现指定的字符，否则，就输出该字符在字符串中出现的次数和位置。

五、实验注意事项

（一）注意 C 语言中字符串是作为一维数组存放在内存中的，并且系统对字符串常量自动加上一个‘\0’作为结束符，所以在定义一个字符数组并初始化时要注意数组的长度。

（二）注意用 scanf 函数对字符数组整体赋值的形式。

六、思考题

（一）下面程序运行的结果是：_____。

```
#include "stdio.h"
void main( )
 { char a[11]={'I', ' ', 'a', 'm', ' ','a', ' ', 'b', 'o', 'y'};
   printf("%s\n",a);
 }
```

（二）下面的程序用来实现将两个字符串连接起来，请将源程序补充完整。

```
#include "stdio.h"
void main( )
 { char str1[100],str2[100];
   int i=0,j=0;
   printf("please input the string1:");
   scanf("%s",str1);
   printf("please input the string2:");
   gets(str2);
   for(i=0; str1[i]!='\0'; i++);
     for(j=0; str2[j]!='\0'; j++)
        { str1[i]=str2[j];
          i++; }
    _____________;
    printf("the catenated string is %s",str1);
 }
```

（三）下面程序的功能是用 strcat 函数实现将字符串 2 连接到字符串 1 的后面并输出，请补充完整。

```
#include "stdio.h"
void main( )
 { char str1[80]="This Is a ",str2[80]="c Program";
   printf("String1 is: %s\n",str1);
   printf("String2 is: %s\n",str2);
   _________________;
   printf("Result is: %s\n",str1);
 }
```

（四）下面的程序用来实现将一个字符串中的所有大写字母转换为小写字母并输出。请将源程序补充完整。

例如，当字符串为"This Is a c Program"
　　输出："this is a c program"

```
#include "stdio.h"
void main( )
 { char str[80]="This Is a c Program";
   int i;
   printf ("String is: %s\n",str);
   for (i=0; str[i]!='\0'; i++)
     if (str[i]>='A' && str[i]<='Z')
     _________________;
   printf("Result is: %s\n",str);
 }
```

实验九　函数

一、实验学时

2 学时

二、实验目的

（一）掌握函数的定义、函数类型、函数参数、函数调用的基本概念；
（二）掌握变量名作函数参数的程序设计方法；
（三）掌握函数的嵌套调用的方法；
（四）掌握数组元素作函数参数；
（五）掌握数组名作函数参数的程序设计方法；
（六）掌握字符数组作函数参数的程序设计方法；
（七）了解全局变量、局部变量的概念和使用方法；
（八）使用功能键 F7 单步执行，使用 Ctrl+F7 观察变量的值，学会程序调试基本方法。

三、预习要求

（一）函数的定义、函数类型、函数参数、函数调用的基本概念；
（二）函数实参与形参的对应关系以及参数的传递；
（三）以变量名和数组名作函数参数时的使用方法；
（四）全局变量、局部变量的概念和使用方法。

四、实验内容

（一）下面程序的功能是：根据输入的整数 x 和 n，利用函数 fact 实现求 x^n。
例如：输入：2，3　　输出 2^3=8
请在程序中的横线上填入正确的内容，将程序补充完整。

```
/*    c9-1.c      利用函数 fact 实现求 x 的 n 次方*/
#include "stdio.h"
void main( )
{ long int  fact(long x,long n) ;                /*声明 fact 函数*/
  long int  x ;
  long int  n;
  printf("please enter X and  N(>=0): ");
  scanf("%ld,%ld", &x, &n );
  printf("%ld,%ld=%ld",x,n,__________ );  /*调用 fact 函数 */
}

long int  fact(long int x, long int n)          /*定义 fact 函数求 xn */
{ long int  i,s;
  _____________ ;                              /*求累积变量的初始化*/
```

```
    if (n= =0)  return 0;
    for(i=1; i<=n; i++)                    /*用循环实现 x^n*/
     s=s*x;
     _____ ;                               /*返回结果 x^n*/
}
```

（二）下面程序的功能是：计算$C_m^n=\frac{m!}{n!*(m-n)!}$的值。请在程序中的横线上填入适当的内容，将程序补充完整。

例如：输入：5，3　　输出：$C_m^n=10$

编程提示：（1）定义求阶乘函数，在此基础上定义求组合数函数。

（2）主函数调用求组合数函数，求组合数函数再三次调用求阶乘函数。

/*　c9-2.c　　利用函数组合数*/

```
#include "stdio.h"
long int  jf (int  n)                      /*定义求阶乘函数 jf*/
{ int  i;
  long int  t=1;
  for(i=1; i<=n; i++)
    t*=i;
  _____ ;
}

long int  cmn(int  m, int  n)              /*定义求组合数函数 cmn*/
{ return(jf(m)/(jf(n)*jf(m-n))) ; }        /*用 return 语句返回结果*/

 void main( )
{ int m,n;
  printf("please enter m and n: ");
  scanf("%d, %d", &m, &n );
  _____ ;
}
```

（三）下面程序的功能是:读入一个整数 m,计算如下公式的值:

$$t=1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{m}$$

例如:若输入 5,则应输出：“The result is 2.28333”。

请在程序中的横线上填入适当的内容。

/*　c9-3.c　　利用函数实现级数求和*/

```
#include "stdio.h"
double fun(int m)
{ float t=1.0;  int i;
  for(i=2; i<=m; i++)
    _____;
```

```
    return  ____________ ; }

  void main( )
{ int m;
   printf("\nPlease enter 1 integer number:");
   scanf("%d",&m);
   ____________ ;          /*按照例子中的输出形式输出结果*/
}
```

（四）下面程序的功能是：输入一个十进制整数，输出其对应的二进制数。

编程提示：

在 main 函数中定义一个变量并为其赋值，然后调用函数 fun 将该十进制数转换为二进制。

函数 fun 的形参即为被转换的整数，在 for 循环中每次求出 m%k 存放到数组 aa 中，同时将 m/k 的整数商赋给 m 继续判断，直止 m 的值为 0。最后按反序输出数组 aa 的元素。

请在程序中的三处横线上填入适当的内容，将程序补充完整：

```
/*   c9-4.c     通过函数调用实现数制转换*/
#include "stdio.h"
void  fun(int m)
{ int aa[20],i, k=2;
  for(i=0; m; i++) {aa[i]=m%k; ____________ ;}
  printf("\n");
  for(; i; i--)  printf("%d  ",aa[_________]);}
void main( )
{ int n;
  printf("\n 请输入一个十进制整数：\n");  scanf("%d",&n);
  fun( ____________ ); }
```

如果将十进制数转换为八进制数，应对程序的哪个语句进行修改？怎样修改？

（五）写一个程序，判别一个整数数组中各元素的值，若大于 0 则输出该值，若小于或等于 0 则输出 0 值。

编程提示：

判断函数：1. 函数的类型为 void，函数中不使用 return 语句；
2. 函数的形参为一整型变量；
3. 函数体中使用选择结构，根据对变量值的判断输出相应结果。

main 函数：1. 定义一个一维整型数组；
2. 为整型数组赋若干个数值；
3. 调用判断函数对数组元素逐一进行判断，以数组元素作实参。

```
/*   c9-5.c   判别一个整数数组中各元素的值   */
#include "stdio.h"
void main( )
{
```

```
}
```

（六）一维数组 a 中的元素为：1,4,2,7,3,12,5,34,5,9。求一维数组 a 中的最大元素及其下标。程序的输出应为：The max is: 34,position is: 7 。要求:求最大元素位置用函数实现，在 main 函数中调用该函数。

编程提示：

定义一个全局变量 max，用来存放最大元素。求最大元素位置的子函数：

1. 函数的类型为整型；
2. 函数的形参应为整型一维数组和一整型变量（存放数组元素的个数）；
3. 函数体中，定义一个整型 pos，用来存放当前最大元素在数组中的下标，初值为 0；将全局变量 max 的初值设置为数组中的第一个元素；
4. 函数体中使用循环结构，将数组元素依次和 max 中的值进行比较，将两者中的最大元素存入 max 中，并将最大元素的下标存入 pos 中；
5. 循环结束后，用 return 语句，将 pos 的值返回到主函数。

main 函数：

1. 定义一个一维整型数组并为该数组赋若干个数；
2. 以赋值语句的形式，将求最大元素位置函数的返回值赋给一个变量，以数组名和数组的元素个数作实参。

注意：以数组名作实参时，实参与形参之间是地址的传递，实参数组和形参数组是公用一段内存单元。

/*　c9-6.c　　通过函数调用求一维数组中的最大元素及其下标*/

```
#include "conio.h"
#include "stdio.h"
int max;
int fun( int arr[ ],int n )
{ int pos,i;
  max = arr[0];
  pos = 0;
  for( i=1; i<n; i++)
    if (max < arr[i])
    { max = arr[i];
      pos=i; }
  return (pos);
}
void main( )
{ int a[10]={1,4,2,7,3,12,5,34,5,9},n;
  _____;
 printf("The max is: %d ,pos is: %d\n", max , n);
}
```

（七）下面程序的功能是：求二维数组 a 中的上三角元素之和。

例如：a 中的元素为：

4	4	34	37
7	3	12	8

```
        5    6     5    52
        24   23    2    10
```

程序的输出应为：The sum is:147。

请在程序中的横线上填入适当的内容，将程序补充完整。

```
/*    c9-7.c      通过函数调用求二维数组中的上三角元素之和*/
#include "conio.h"
#include "stdio.h"
int arrsum( int arr[4][4])
{ int i,j,sum;
  sum=0;
  for( i=0; i<4; i++)
     for  ( __________;j<4; j++)
     sum+=arr[i][j];
   return (sum);
}
void  main( )
{ int a[4][4]={4,4,34,37,7,3,12,8,5,6,5,52,24,23,2,10},i,j;
  printf("The max is: %d\n",__________);}
```

（八）将字符串 1 的第 1,3,5,7,9,.....位置的字符复制到字符串 2 并输出。

例如：当字符串 1 为"This Is a c Program"，则字符串 2 为"Ti sacPorm"。

编程提示：

子函数：1. 函数的类型为 void，函数中不使用 return 语句；

2. 函数的形参应为两个字符型一维数组；

3. 函数体中使用循环结构，将字符串 1 中相应位置上的字符逐一复制到字符串 2 中，注意循环变量每次递增的数目。

main 函数：1. 定义一个一维字符型数组；

2. 为字符数组赋一个字符串；

3. 调用转换函数，以两个数组名作实参；

4. 输出转换后的字符数组的内容。

```
/*    c9-8.c      通过函数调用实现对字符串的处理*/
#include "conio.h"
#include "stdio.h"
#include "string.h"
void fun(char str1[ ],char str2[ ])
{ int i,j;
  j=0;
  for(i=0;i<strlen(str1);i+=2)
  { str2[j]=str[i];
    j++; }
  str2[j]=' \0' ; }

void main()
```

```
{ char str1[80]="This Is a c Program",str2[80];
   printf("String is: %s\n",str1);
  _____________;
  printf("Result is: %s\n",str2);
}
```

（九）输入下面的程序并分析运行结果。用 F7 单步执行，注意程序的执行过程，观察变量 d 的值，理解全局变量和局部变量的区别，理解各种局部变量的作用范围。

```
/*   c9-9.c     全局变量与局部变量的作用范围 */
int d=1;
fun(int p)
{ int d=5; d+=p++; printf("%d",d); }
main( )
{ int a=3;  fun(a);
  { int d=16;  d+=a++;
    printf("%d  ",d); }
  printf("%d  ",d);
}
```

五、实验注意事项

（一）定义函数时，函数名后的圆括号后面不能加“;”。

（二）在函数体内，不能再对形参进行定义和说明。

（三）变量作实参时，只使用变量名，实参变量对形参变量的数据传递是“值传递”。

（四）一维数组作函数的实参时，只使用数组名如：fun(a); 。

下面对函数的调用都是不正确的：

```
fun(int a[4]);
fun(int a[ ]);
fun(int a);
```

六、思考题

（一）下面程序的功能是：判别一个整数数组中各元素的值，若大于 0 则输出该值，若小于或等于 0 则输出 0 值。请在程序中的横线上填入适当的内容，将程序补充完整。

```
#include "stdio.h"
void nzp(int v)
{ int i=0;
  if(v>0)  printf("%d ",v);
  else   printf("%d ",i);
}
void main( )
{ int a[5],i;
  printf("input 5 numbers\n");
  for(i=0;i<5;i++)
```

```
    { scanf("%d",&a[i]);
      _____________ ; }
}
```

（二）一维数组 a 中的元素为：1,4,2,7,3,12,5,34,5,9。下面程序的功能是：求一维数组 a 中的最大元素及其下标。程序的输出应为：The max is: 34,position is: 7 。

请在程序中的横线上填入适当的内容，将程序补充完整。

```
#include <conio.h>
#include <stdio.h>
int max;
int fun( int arr[ ],int n )
{ int pos,i;
  max = arr[0];
  pos = 0;
  for ( i=1; i<n; i++)
    if (max < arr[i])
    { max = arr[i];
      pos=i; }
  return(pos);
}
void main()
{ int a[10]={1,4,2,7,3,12,5,34,5,9},n;
 _____________;
 printf("The max is: %d ,pos is: %d\n", max , n);
}
```

（三）下面程序的功能是：将字符串 1 的第 1,3,5,7,9,.....位置的字符复制到字符串 2 并输出。

例如，当字符串 1 为"This Is a c Program"，则字符串 2 为"Ti sacPorm"

请在程序中的横线上填入适当的内容，将程序补充完整。

```
#include "conio.h"
#include "stdio.h"
#include "string.h"
void fun(char str1[ ],char str2[ ])
{ int i,j;
  j=0;
  for(i=0;i<strlen(str1);i+=2)
 { str2[j]=str[i];
   j++;}
 str2[j]=' \0' ;  }

void main( )
{ char str1[80]="This Is a c Program",str2[80];
```

```
  clrscr( );
  printf("String is: %s\n",str1);
  _____ ;
  printf("Result is: %s\n",str2);
}
```

（四）请写出下面的程序的运行结果。

```
#include "stdio.h"
int d=1;
void fun(int p)
{ int d=5; d+=p++; printf("%d",d);}

void main( )
{ int a=3;  fun(a);
  { int d=16;  d+=a++;
    printf("%d  ",d); }
  printf("%d  ",d);
}
```

程序的运行结果是：_____ 。

实验十　指针（一）

一、实验学时

2 学时

二、实验目的

（一）掌握指针的概念，会定义和使用指针变量；
（二）了解或掌握指针与数组的关系，指针与数组有关的算术运算、比较运算。
（三）学会用指针作为函数参数的方法。

三、预习要求

（一）宏定义，带参数的宏定义，不带参数的宏定义；
（二）地址和指针的概念；
（三）数组和指针的关系；
（四）字符串和指针的关系；
（五）函数的定义、函数类型、函数参数、函数调用的基本概念。

四、实验内容

（一）编辑源程序，回答后面的问题。

```
/*   c10-1.c   */
#define S(a,b) a*b
#include <stdio.h>
void main( )
{   int c,d,t ;
    scanf("%d, %d",&c,&d);
    t=S(c+d,c-d);   /*宏展开后代入变量 c,d 的值，从而求得变量 t 的值*/
    printf("%d",t);
}
```

（2）输入数据：
（3）结果输出：
（4）分析本题重点：
①宏展开后的表达式应该为： c+d*c-d
②代入 c， d 的值（例如输入 2， 3）， 则 t=2+3*2-3=5
③易错的地方是结果为： （2+3） * （2-3） = -6 （无形中加了括号） 假设此题变化为： #indefine S(a, b) (a)*(b)，则正确结果是-6。通过此题应该掌握带参的宏展开的实质：不仅仅是作简单的置换，还要进行参数的替换。

```
/*   c10-2.c   输入两个整数，并使其从大到小输出，用指针变量实现数的比较 */
#include <stdio.h>
void main()
```

```
{   int *p1,*p2,*p,a,b;
    scanf("%d,%d",&a,&b);
    p1=&a; p2=&b;
    if(a<b)
    {   p=p1; p1=p2; p2=p;}
    printf("a=%d,b=%d\n",a,b);
    printf("max=%d,min=%d\n", *p1 ,*p2);
}
```

（2）输入数据：

（3）结果输出：

/*　c10-3.c　输入两个整数，并使其从大到小输出，用函数实现数的交换*/

```
#include <stdio.h>
void swap(int *p1, int *p2)
{   int p;
    p=*p1;
    *p1=*p2;
    *p2=p;
}
void main()
{   int a,b;
    int *p,*q;
    scanf("%d,%d",&a,&b);
    p=&a; q=&b;
    if(a<b) swap(p,q);
    printf("\n%d,%d\n",a,b);
}
```

（2）输入数据：

（3）结果输出：

（4） 如果将 swap 函数修改为如下形式，分析如何调试和修改？

```
void swap(int *p1, int *p2)
{   int *p;
    *p=*p1;
    *p1=*p2;
    *p2=*p;
}
```

/*　c10-4.c　用指针法输入 12 个数,然后按每行 4 个数输出　*/

（1）编程提示：定义一个整型数组和一个整型指针，这样通过数组就可以静态分配 内存空间，存储数据；然后将指针与数组相关，使指针指向与数组相同的首地址处，这样就可以通过指针或者数组都可以对存储空间加以操作。

（2）源程序：

```
#include <stdio.h>
```

```
void main()
{   int j,k,a[12],*p ;
    p=a;    //使指针 p 指向与数组 a 相同的首地址处
    for(j=0;j<12;j++)
        scanf("%d",p++);    //移动 P 的位置，输入数据
    p=a;    //指针重定位
    for(j=0;j<12;j++)
    {
        if(j%4==0)
        printf("\n");   //按每行 4 个数输出
        printf("%4d", *p++);
    }
printf("\n");
}
```

（3）输入数据：

（4）结果输出：

五、实验注意事项

（一）注意变量、变量的指针、变量的地址间的相互关系；

（二）注意传地址与传值的区别。

六、思考题

（一）从键盘输入十个整数，要求用冒泡法（或选择法）实现从大到小的排列输出。

（二）将指针作为函数参数：一个数组有 10 个元素{1，8， 10， 2，-5， 0， 7， 15，4， -5}， 利用指针作为函数参数编程，输出数组中最大和最小的元素值。

（三）练习数组指针作为函数参数：求 3×4 的二维数组{1， 3， 5， 7， 9， 11， 13，17， 19， 21， 23， 25}中的所有元素之和。

实验十一　指针（二）

一、实验学时

2 学时

二、实验目的

（一）进一步理解指针的概念，掌握其在数组和字符串中的应用。
（二）学会使用函数的指针和指向函数的指针变量。
（三）了解指向指针的指针的概念及其使用方法。

三、预习要求

（一）字符、字符串和字符数组的关系与表示方法；
（二）函数指针；
（三）指向指针的指针。

四、实验内容

/*　c11-1.c　输入三个字符串，按由小到大的顺序输出。用指针方法处理 */

（1）算法分析：字符串的比较要用到函数 strcmp，比较后值的保存及交换要用到 函数 strcpy 及一个中间变量。 参数的传送方式是“传址”方式。

（2） 程序如下：

```
#include <stdio.h>
#include<string.h>
void main( )
{
  char str1[20], str2[20], str3[20]; /*定义三个字符数组*/
  char swap();  /*声明函数*/
  printf("input three string: \n");
  gets(str1); gets(str2); gets(str3);
  if(strcmp(str1 ,str2)>0)   swap(str1 ,str2); /*调用函数 swap */
  if(strcmp(str1 ,str3)>0)   swap(str1 ,str3);
  if(strcmp(str2 ,str3)>0)    swap(str2,str3);
  printf("the order is: \n");
  printf(%s \ n%s \ n%s \ n ", str1, str2, str3);
}
    char swap(char*p1,char*p2)  /*定义交换两个字符串的函数 swap */
{
    char *p[20];
    strcpy(p,p1); strcpy(p1 ,p2); strcpy(p2,p); /* 完成比较交换功能 */
}
```

（3）运行程序。Input three lines:

I study very hard. ↙ He is a teacher. ↙ Today is fine. ↙

（4）查看运行结果，想想为什么是这个结果？

/*　c11-2.c　*/

调试并修改下列程序,使之具有如下功能：任意输入 2 个数，调用两个函数分别求：①2 个数的和；②2 个数交换值。要求用“函数指针”调用这两个函数，结果在主函数中输出。

（1） 源程序：

```
#include <stdio.h>
void main( )
{
  int a,b,c,(*p)();
  scanf("%d,%d",&a,&b);
  p=sum ;
  *p(a,b,c);
  p=swap ;
  *p(a,b);
  printf("sum=%d\n",c);
  printf("a=%d,b=%d\n",a,b);
}
sum(int a,int b,int c)
{ c=a+b ; }
swap(int a,int b)
{ int t ; t=a ; a=b ; b=t ; }
```

（2）调试程序时注意参数传递的是数值还是地址。

五、实验注意事项

注意数组的指针与数据元素值之间的关系。

六、思考题

/*　c11-3.c　练习指针数组*/

有三个字符串“Data structure”、“Coputer design”、“C Progrom”，请按字符顺序输出这三个字符串。 （要求用指针数组指向这三个字符串。）

/*　c11-4.c　*/

输入 10 个整数，找出其中最大的数并与最后一个数对换。写三个函数：

（1）输入 10 个数；

（2） 进行处理；

（3）输出 10 个数。

要求：在主函数中，用一个函数指针来访问这三个函数。

/*　c11-5.c　练习指针与字符串*/

在一行字符串中删去指定的字符。例如，要求在一行文 字：　“I have 150 Yuan!!”中删去字符“0”，使其变为“I have 15 Yuan!!”。

/*　c11-6.c　*/

输入一个 3 位数，计算该数各位上的数字之和，如果在[1,12]之 内，则输出与和数相对应的月份的英文名称，否则输出***。

例如：输入： 123 输出： 1+2+3=6→ June 输入： 139 输出： 1+3+9=13→ ***

要求：用指针数组记录各月份英文单词的首地址。

实验十二　结构体、共用体和位运算

一、实验学时

2 学时

二、实验目的

1、掌握结构体类型变量的定义和使用；
2、掌握结构体类型数组的概念和使用；
3、掌握链表的概念，初步学会对链表进行操作；
4、掌握共用体的概念与使用；
5、掌握按位运算的概念和方法，学会使用位运算符；
6、学会通过位运算实现对某些位的操作。

三、预习要求

（1）结构体

C 语言提供了一种如果用简单变量来分别代表属性，难以反映出他们之间的内在联系数据类型称为结构体。如学生姓名、编号、性别、年龄、各科成绩、 地址等。 他们是同一个处理对象，学生的属性，在这之间，即有字符型、也有长整、短整型、实型等各 种数据类型。

例：

Num	name	sex	age	score	addr
10010	Li fum	m	18	88.5	beijin
整型	字符型	字符	整型	实型	字符型

```
struct student
{   int num;
    char name[20];
    char sex;
    short int age; float score; char addr[30];
}
```

上面就定义了一个结构体类型， struct 是关键字，结构体类型是 student 。其中有 6 个不同的数据项。

结构体类型不同于基本数据类型的特点： （1）由若干个数据项组成，每个数据项称为一个结构体的成员，也可称为“域”。 （2）结构体类型并非只能有一种，而可以有千千万万。

```
struct 结构体名
{
成员项表列
};
```

定义一个结构体类型，并不意味着系统将分配一段内存单元来存放各数据项成员。 因为这仅仅只定义了类型。结构体类型需用户自己定义。

（2）位运算符和位运算

四、实验内容

/*　c12-1.c　*/

编写函数 print,打印一个学生的成绩数组,该数组中有 5 个学生的数据记录,每个记录包括 num、name、score[3]，用主函数输入这些记录，用 print 函数输出这些记录。

（1）算法分析：依据题意，先定义一个包含有三个成员项的结构体数组，在主函数 中利用循环依次输入数据，并调用函数 print，完成输出数据的功能。

（2） 程序如下：

```
#define N 5
#include <stdio.h> struct student
{   char num[6];
    char name[8];
    int score[4]; /*注意，考虑到下标从 1 开始，有三门课程成绩 score[3]所以设定 分数的数组长度加 1, score[4]，防止数据溢出*/
} stu[N];
void main( )
{
    int i,j ;
    void print(struct student stu[N]);
    for(i=0;i<N;i++)
    {
        printf("\nInput score of student %d:\n",i+1);
        printf("NO.:"); scanf("%s",stu[i] .num);
        printf("name:");
        scanf("%s",stu[i] .name);
        for(j=1;j<=3;j++)
        {
            printf("score %d:",j);
            scanf("%d",&stu[i].score[j]);
        }
    printf("\n");
    }
    print(stu);
}
void print(struct student stu[N])
{
    int i,j ;
    printf("\n NO.  name    score1  score2  score3\n");
    for(i=0;i<N;i++)
     {
        printf("%5s%1 Os",stu[i] .num,stu[i] .name);
        for(j=1;j<=3;j++)
            printf("%9d",stu[i] .score[j]);
        printf("\n");
    }
}
```

(3)在编辑环境下输入源程序
(4)编译直到程序没有错误
(5) 输入:

```
Input score of student 1:
NO. :101/
name: Li /
score1 :90/ score2:79/ score3:89/
Input score of student 2:
NO. :102/
name: Ma/
score1 :97/ score2:90/ score3:58/
Input score of student 3:
NO. :103/
name :Wang/ score1 :77/ score2:70/ score3:78/
Input score of student 4:
NO. : 104/
name: Fun/ score1 :56/
score2:89↙ score3:65↙
Input score of student 5:
NO. :105↙
name :Xue↙ score1 :87↙ score2:65↙ score3:63↙
```

（6）查看运行结果

/*　c12-2.c　*/

设计一个函数，使给出一个数的原码，能得到该数的补码。
（1）算法分析：一个正数的补码等于该数原码,一个负数的补码等于该数的反码加 1。
（2）编写源程序

```
#include <stdio.h>
void main( )
{
  unsigned int a ;
  unsigned int getbits(unsigned);
  printf("\nInput an octal number:");
  scanf("%o",&a);
  printf("result:%o\n",getbits(a));
}
unsigned int getbits(unsigned value) /*求一个二进制的补码 函数*/
{
  unsigned int z ;
  z=value&0 100000 ;
  if(z==0 100000)
    z=～ value+1 ;
  else
    z=value ; return(z);
}
```

（3）运行情况如下：

①Input an octal number:2345↙
result :2345
②Input an octal number: 152525↙
result :25253
实验报告要求对照各程序运行结果分析以上各程序的算法（框图表示），并对其结果进行分析。

/*　c12-3.c　*/

下面的程序是将指针用于结构数据动态链表操作的示例，请初步熟悉它们的使用方法。

```
#define NULL 0
#include <stdio.h>
#include <stdlib.h>
#define LEN sizeof (struct student)
struct student
{
  float score;
  struct student *next;
};/* 定义结构*/
int n = 3;  /* 共 3 个数据元素*/
float x;    /* 全局量 struct student 和 x */
struct student *creat() /* 建立一个链表,返回指向链表首结点的指针(地址) */
{
    struct student *head, *p, *rear ;
    for (int i = 0; i < n; i++)
    {
        p = (struct student *)malloc(LEN);/* 新创建的结点*/
        scanf("%f", &x);
        p->score = x; /* 新结点赋值*/
        if (i == 0)
        {
            head = p; /* head 为首指针，rear 为尾指针*/
            rear = p;
        }
        Else
        {
            rear->next = p;
            rear = p;
        }
    }
    rear-> next = NULL;
    return (head);
}

void print(struct student *head) /*遍历一个 head 为指向的链表*/
{
    struct student *p; p = head;
    while (p != NULL) {
```

```
        printf("%6.2f", p->score); p = p->next;
        }
        printf("\n\n");
    }

    void main()
    {
        struct student *head;
        printf("input score:\n");
        head = creat();/*建立单链表*/
        printf("Display the linklist:"); print(head);/*遍历单链表*/
    }
```

五、实验注意事项

注意结构体与普通数据结构的关系。

六、思考题

/*　c12-4.c　*/

有 10 个学生，每个学生的数据包括学号、姓名、3 门课的成绩， 从键盘输入 10 个学生的数据，要求打印出 3 门课的总平均成绩。

/*　c12-5.c　*/

编写一个函数 getbits， 从一个 16 位的单元中取出某几位（即该几位保留原值，其余位为 0）。函数调用形式为：

getbits(value, n1, n2)

valuen 为该 16 位 （两个字节）单元中的数据值，n1 为欲取出的起始位，n2 为欲取出的 结束位。如：

getbits(0101675,5,8)

表示对八进制 101675 这个数，取出它从左面起的第 5 位到第 8 位。

实验十三　文件

一、实验学时

2 学时

二、实验目的

（一）掌握文件以及缓冲文件系统、文件指针的概念；
（二）学会使用文件打开、关闭、读、写等文件操作函数；
（三）学会用缓冲文件系统对文件进行简单的操作。

三、预习要求

文件类型指针
文件的打开与关闭 文件的读写操作函数

四、实验内容

/*　c13-1.c　*/

将一个磁盘文件中的信息复制到另一个磁盘文件中。

```
#include <stdio.h> #include “ stdlib.h” void main( )
{
  FILE*in,*out ; // 定义文件指针
  char ch,infile[1 0],outfile[1 0];
  printf("Enter the infile name:\n");
  scanf("%s",infile);
  printf("Enter the outfile name:\n");
  scanf("%s",outfile);
  if((in=fopen(infile," r"))==NULL)
  { // 判断文件是否正确读操作
    printf("cannot open infile\n"); exit(0);
  }
  if((out=fopen(outfile,"w"))==NULL)
  { // 判断文件是否正确写操作
    printf("cannot open outfile\n"); exit(0);
  }
//判文件是否结束，如果不结束，则读文件 in 的内容写入到文件 out 之中
while(!feof(in)) fputc(fgetc(in),out);
fclose(in); //关闭文件
fclose(out);
}
```

运行情况如下：

Enter the infile name:
file1.c↙　（输入原有磁盘文件名）
Enter the outfile name:
file2.c↙　（输入新复制的磁盘文件名）

程序运行结果是将 fule1. c 文件中的内容复制到 fule2. c 中去。可以用下面命令验证：

c:\>type file1.c
computer and c　（file1.c 中的信息）
c:\>type file2.c
computer and c　（file2.c 中的信息）

以上程序是按文本文件方式处理的。也可以用此程序来复制一个二进制文件，只需 将两个 fopen 函数中的“r” 和 “w”分别改为“rb” 和 “wb” 即可。

分析以上各程序的算法(用框图表示)，解释产生该结果现象的相关知识点及实现语句。

/*　c13-2.c　*/

阅读以下程序，回答后面的问题。

```
#include <stdio.h>
void main(void)
{
short a=0x253f,b=0x7b7d; char ch;
FILE *fp1,*fp2;
fp1=fopen("c:\\abc1.bin","wb+");
fp2=fopen("c:\\abc2.txt","w+");
fwrite(&a,sizeof(short), 1 ,fp1);
fwrite(&b,sizeof(short), 1 ,fp 1);
fprintf(fp2,"%hx %hx",a,b); rewind(fp 1); rewind(fp2);
while((ch = fgetc(fp1)) != EOF) putchar(ch);
putchar('\n');
while((ch = fgetc(fp2)) != EOF) putchar(ch);
putchar('\n');
fclose(fp 1);
fclose(fp2);
}
```

（1）请思考程序的输出结果， 然后通过上机运行来加以验证。

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

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

/*　c13-3.c　*/

修改以下程序，实现将指定的文本文件内容在屏幕上显示出来，命令行的格式为：display filename.txt (注：display 是源程序文件的文件名)

```
#include<stdio.h> #include<stdlib.h>
void main(int argc, char* argv[])
{
```

```
char ch;
FILE *fp;
if(argc!=2) {
printf("Arguments error!\n");
exit(-1);
}
if((fp=fopen(argv[1 ],"r"))==NULL) {    /* fp 指向 filename */
printf("Can't open %s file!\n",argv[1]);
exit(-1);
}
while(ch=fgetc(fp)!=EOF)    /* 从 filename 中读字符 */
putchar(ch);    /* 向显示器中写字符 */
fclose(fp); /* 关闭 filename */
}
```

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

（2）将程序保存为 display.c ，在“ 项目设置－调试－程序参数”中设置好参数（填入 要显示的文件名，如 filename.txt），然后进行编译和调试。

/*　c13-4.c　*/

有 10 个学生，每个学生的数据包括学号、姓名、3 门课的成绩， 从键盘输入 10 个学生的数据，要求打印出 3 门课的总、平均成绩，并计算出每人的平均成绩,将原有 数据和计算出的平均分数存放在磁盘文件 stu.txt 中。

五、实验注意事项

注意文件打开的不同方式。

六、思考题

/*　c13-5.c　*/

通讯录管理系统：该系统通过文本菜单进行操作，功能包括：创建通讯录、显 示记录、查询记录、修改记录、添加记录、删除记录和记录排序等，各功能模块均采用 独立的函数来表示，通过主函数直接或是间接调用，特别注意的是，通讯录数据采用结 构体定义和管理，并可以直接从文件中读入数据或是将数据写入文件中，体会这样做的 的优越性。

附录：常见错误提示信息的英汉对照

Ambiguous operators need parentheses : 不明确的运算需要用括号括起
Ambiguous symbol ’ xxx’ : 不明确的符号
Argument list syntax error : 参数表语法错误
Array bounds missing : 丢失数组界限符
Array size toolarge : 数组尺寸太大
Bad character in paramenters :参数中有不适当的字符
Bad file name format in include directive :包含命令中文件名格式不正确
Bad ifdef directive synatax :编译预处理 ifdef 有语法错
Bad undef directive syntax :编译预处理 undef 有语法错
Bit field too large :位字段太长
Call of non-function :调用未定义的函数
Call to function with no prototype :调用函数时没有函数的说明
Cannot modify a const object :不允许修改常量对象
Case outside of switch: 漏掉了 case 语句
Case syntax error :Case 语法错误
Code has no effect 代码不可述不可能执行到
Compound statement missing{ 分程序漏掉 fl{fl
Conflicting type modifiers 不明确的类型说明符
Constant expression required 要求 常量表达式
Constant out of range in comparison 在比较中常量超出范围
Conversion may lose significant digits 转换时会丢失意义的数字
Conversion of near pointer not allowed 不允许转换近指针
Could not find file ’ xxx’ 找不到 XXX 文件
Declaration missing ; 说明缺少″；″
Declaration syntax error 说明中出现语法错误
Default outside of switch Default 出现在 switch 语句之外
Define directive needs an identifier 定义编译预处理需要标识符
Division by zero 用零作除数
Do statement must have while Do-while 语句中缺少 while 部分
Enum syntax error 枚举类型语法错误
Enumeration constant syntax error 枚举常数语法错误
Error directive :xxx 错误的编译预处理命令
Error writing output file 写输出文件错误
Expression syntax error 表达式语法错误
Extra parameter in call 调用时出现多余错误
File name too long 文件名太长
Function call missing) 函数调用缺少右括号
Fuction definition out of place 函数定义位置错误
Fuction should return a value 函数必需返回一个值

Goto statement missing label Goto 语句没有标号
Hexadecimal or octal constant too large 16 进制或 8 进制常数太大
Illegal character ’ x’ 非法字符 x
Illegal initialization 非法的初始化
Illegal octal digit 非法的 8 进制数字
Illegal pointer subtraction 非法的指针相减
Illegal structure operation 非法的结构体操作
Illegal use of floating point 非法的浮点运算
Illegal use of pointer 指针使用非法
Improper use of a typedefsymbol 类型定义符号使用不恰当
In-line assembly not allowed 不允许使用行间汇编
Incompatible storage class 存储类别不相容
Incompatible type conversion 不相容的类型转换
Incorrect number format 错误的数据格式
Incorrect use of default Default 使用不当
Invalid indirection 无效的间接运算
Invalid pointer addition 指针相加无效
Irreducible expression tree 无法执行的表达式运算
Lvalue required 需要逻辑值 0 或非 0 值
Macro argument syntax error 宏参数语法错误
Macro expansion too long 宏的扩展以后太长
Mismatched number of parameters in definition 定义中参数个数不匹配
Misplaced break 此处不应出现 break 语句
Misplaced continue 此处不应出现 continue 语句
Misplaced decimal point 此处不应出现小数点
Misplaced elif directive 不应编译预处理
elif Misplaced else 此处不应出现 else
Misplaced else directive 此处不应出现编译预处理 else
Misplaced endif directive 此处不应出现编译预处理 endif
Must be addressable 必须是可以编址的
Must take address of memory location 必须存储定位的地址
No declaration for function ’ xxx’ 没有函数 xxx 的说明 No stack 缺少堆栈
No type information 没有类型信息
Non-portable pointer assignment 不可移动的指针 （地址常数） 赋值
Non-portable pointer comparison 不可移动的指针 （地址常数） 比较
Non-portable pointer conversion 不可移动的指针 （地址常数 ） 转换
Not a valid expression format type 不合法的表达式格式
Not an allowed type 不允许使用的类型
Numeric constant too large 数值常太大
Out of memory 内存不够用
Parameter ’ xxx’ is never used 能数 xxx 没有用到
Pointer required on left side of -> 符号->的左边必须是指针
Possible use of ’ xxx’ before definition 在定义之前就使用了 xxx （警告）

Possibly incorrect assignment 赋值可能不正确
Redeclaration of ’ xxx’ 重复定义了 xxx
Redefinition of ’ xxx’ is not identical xxx 的两次定义不一致
Register allocation failure 寄存器定址失败
Repeat count needs an lvalue 重复计数需要逻辑值
Size of structure or array not known 结构体或数给大小不确定
Statement missing ; 语句后缺少 II ； II
Structure or union syntax error 结构体或联合体语法错误
Structure size too large 结构体尺寸太大
Sub scripting missing] 下标缺少右方括号
Superfluous & with function or array 函数或数组中有多余的 II&II
Suspicious pointer conversion 可疑的指针转换
Symbol limit exceeded 符号超限
Too few parameters in call 函数调用时的实参少于函数的参数
Too many default cases Default 太多(switch 语句中一个)
Too many error or warning messages 错误或警告信息太多
Too many type in declaration 说明中类型太多
Too much auto memory in function 函数用到的局部存储太多
Too much global data defined in file 文件中全局数据太多
Two consecutive dots 两个连续的句点
Type mismatch in parameter xxx 参数 xxx 类型不匹配
Type mismatch in redeclaration of ’ xxx’ xxx 重定义的类型不匹配
Unable to create output file ’ xxx’ 无法建立输出文件 xxx
Unable to open include file ’ xxx’ 无法打开被包含的文件 xxx
Unable to open input file ’ xxx’ 无法打开输入文件 xxx
Undefined label ’ xxx’ 没有定义的标号 xxx
Undefined structure ’ xxx’ 没有定义的结构 xxx
Undefined symbol ’ xxx’ 没有定义的符号 xxx
Unexpected end of file in comment started on line xxx 从 xxx 行开始的注解尚未结束文件不能结束
Unexpected end of file in conditional started on line xxx 从 xxx 开始的条件语句尚未结束文件不能结束
Unknown assemble instruction 未知的汇编结构
Unknown option 未知的操作
Unknown preprocessor directive: ’ xxx’ 不认识的预处理命令 xxx
Unreachable code 无路可达的代码
Unterminated string or character constant 字符串缺少引号
User break 用户强行中断了程序
Void functions may not return a value Void 类型的函数不应有返回值
Wrong number of arguments ： 调用函数的参数数目错
‘xxx’ not an argument :xxx 不是参数
‘xxx’ not part of structure xxx 不是结构体的 一部分
xxx statement missing (xxx 语句缺少左括号

xxx statement missing) xxx 语句缺少右括号
xxx statement missing ; xxx 缺少分号
xxx’ declared but never used 说明了 xxx 但没有使用
xxx’ is assigned a value which is never used 给 xxx 赋了值但未用过