第1篇　实验指导

本篇设置实验1至实验9共9个实验，分别与主教材的第1章至第9章相对应。实验内容整体上分为基础实验和综合实验两类。基础实验紧密结合主教材中的例题程序设置，以“理解—验证—完善—扩充—提高”为实验教学线索，重在巩固课堂教学知识，提高读者调试程序和编写、完善程序的能力；综合实验以算法设计和程序实现为重点，训练读者运用所学知识解决实际问题的能力。

上机实验是学习C语言的基础性实践环节，要学好、用好C语言就要重视实验课，认真做好每个实验。做好每个实验有3个必要的环节，一是上机实验前预习好实验内容，知道要“做什么”和“怎么做”；二是上机时勤思考，敢动手，学会“试一试，看这样行不行”的上机实验方法，对于实验中遇到的一些问题，“想一想，试一试”就会找到答案；三是实验结束后对实验过程、实验方法、实验结论等进行整理和总结。

实验 1　编辑运行 C 语言程序

一、实验目的

(1) 熟悉 C 语言的编辑运行环境,学会编辑运行 C 语言程序的基本方法。

(2) 进一步了解标识符的特点。

(3) 认识 C 语言程序的结构特点,学习程序的基本编写方法。

二、实验内容

1. 使用 Microsoft Visual C++ 6.0(后文简称 VC++ 6.0)编辑运行"自然数累加"程序

附:"自然数累加"程序(主教材 1.1 节中的 C 语言示例程序)

```
#include <stdio.h>
int main()
{
    int i = 1,s = 0;
    while(i <= 100)                    /* 循环控制 */
    {
        s = s + i;                     /* 数据累加 */
        i = i + 1;                     /* 生成下一个要累加的数 */
    }
    printf("sum = %d\n",s);            /* 输出结果 */
    return 0;
}
```

2. 验证标识符大小写的不等价性

在上面的程序运行完成之后,将"i=i+1"中的 i 改为 I,然后编译程序,查看编译结果,并做好记录。

3. 实验内容拓展

模仿"自然数累加"程序,编写、调试计算 10!的程序。

三、实验指导

1. 使用 VC++ 6.0 编辑运行"自然数累加"程序

编辑运行 C 语言程序共有 4 个步骤,即建立源程序文件、编译源程序、构建可执行文件以及运行可执行文件。

(1) 建立源程序文件。

① 启动 VC++ 6.0,初始界面如图 1-1 所示。

② 打开源程序编辑窗口。选择 File→New 命令,打开 New 对话框,切换到 Files 选项

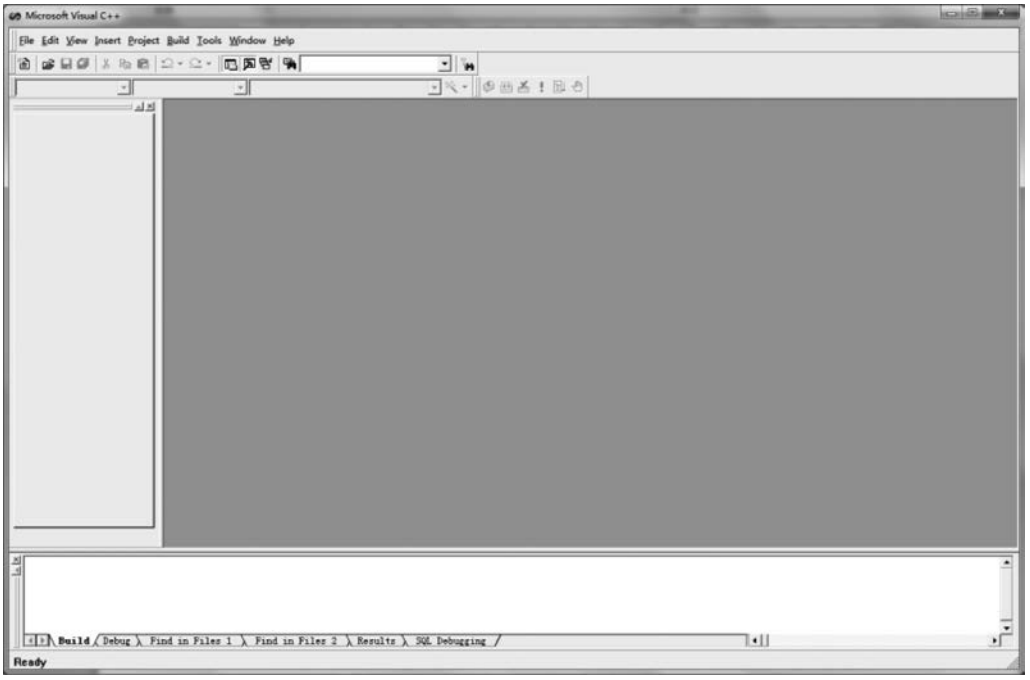

图 1-1　VC++ 6.0 初始界面

卡,在列表中选择 C++ Source File 项,在 File 文本框中输入文件名,在 Location 文本框中指定文件的存储位置,如图 1-2 所示。单击 OK 按钮,打开源程序编辑窗口,如图 1-3 所示。

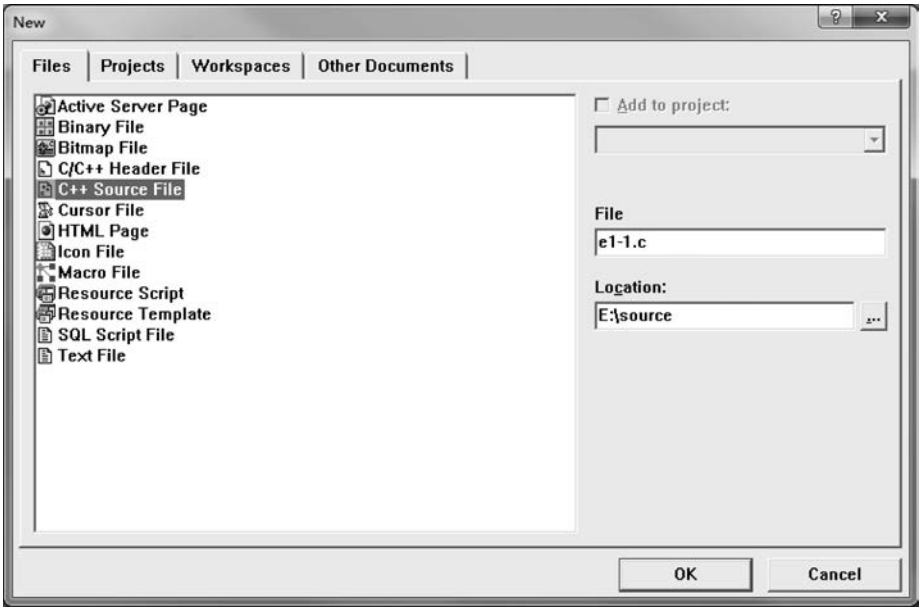

图 1-2　New 对话框

③ 在源程序编辑窗口中输入和编辑源程序,并及时保存源程序文件。图 1-4 是输入源程序后的程序编辑窗口。

说明:在 VC++ 6.0 环境中,源文件的扩展名默认为 C++程序的扩展名.cpp。

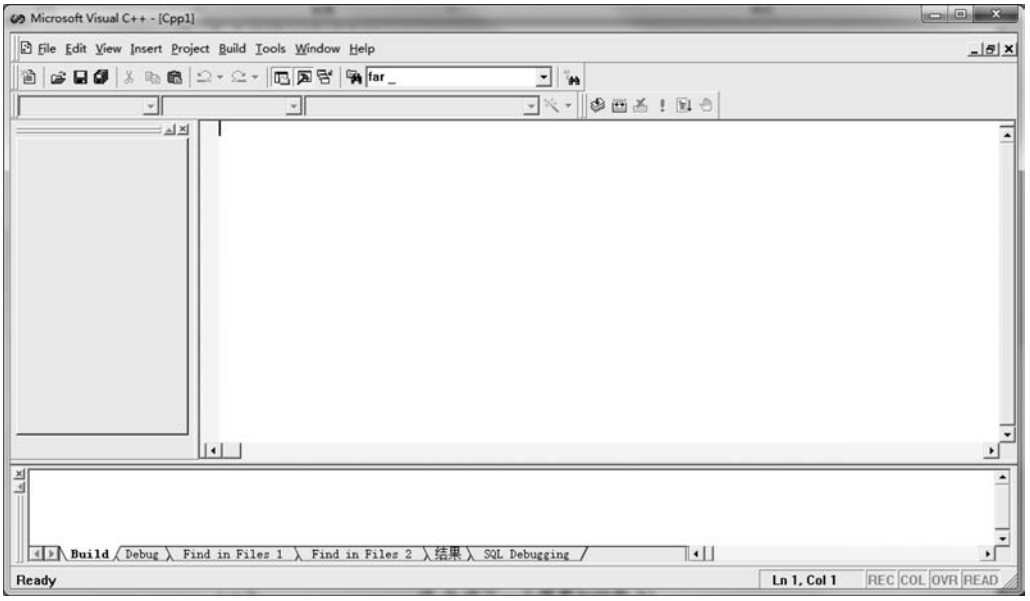

图 1-3　VC++ 6.0 源程序编辑窗口

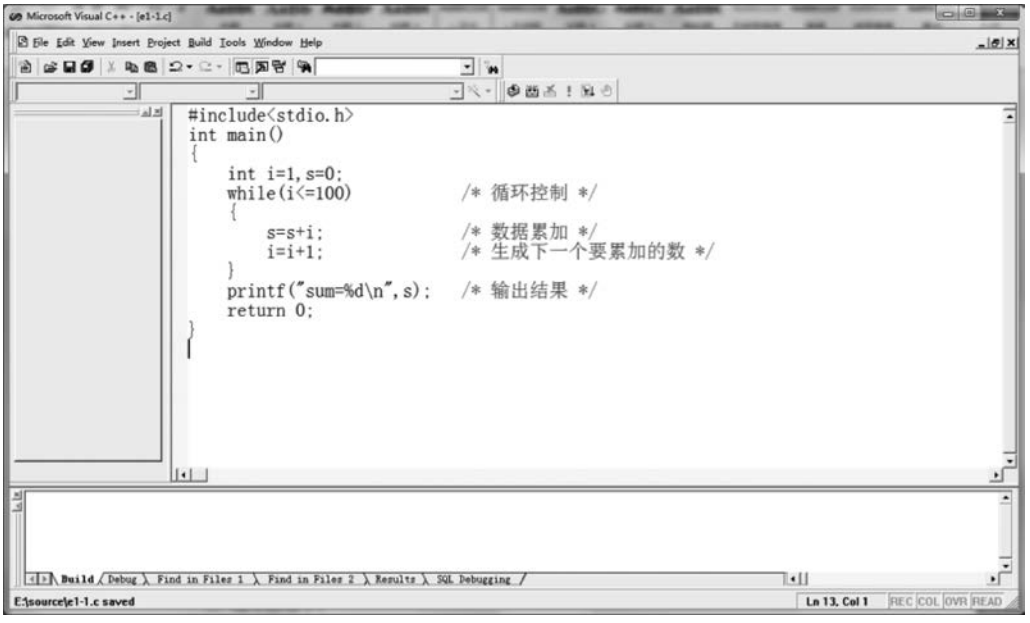

图 1-4　输入源程序后的程序编辑窗口

（2）编译源程序。

编辑完 e1-1.c 程序后，选择 Build→Compile e1-1.c 命令或单击编译按钮，系统显示如图 1-5 所示的提示信息，要求开辟一个供编译使用的工作空间（workspace），单击“是（Y）”按钮，系统开始编译程序，并在下方的窗口中显示编译报告。图 1-6 是完成编译后的窗口。

编译时发现的错误显示在编译报告中，此时可根据编译报告立即编辑和修改源程序，无编译错误时显示“0 error(s)”。

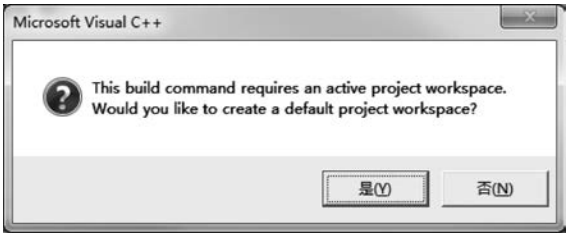

图 1-5　执行 Build→Compile e1-1.c 命令后的提示信息

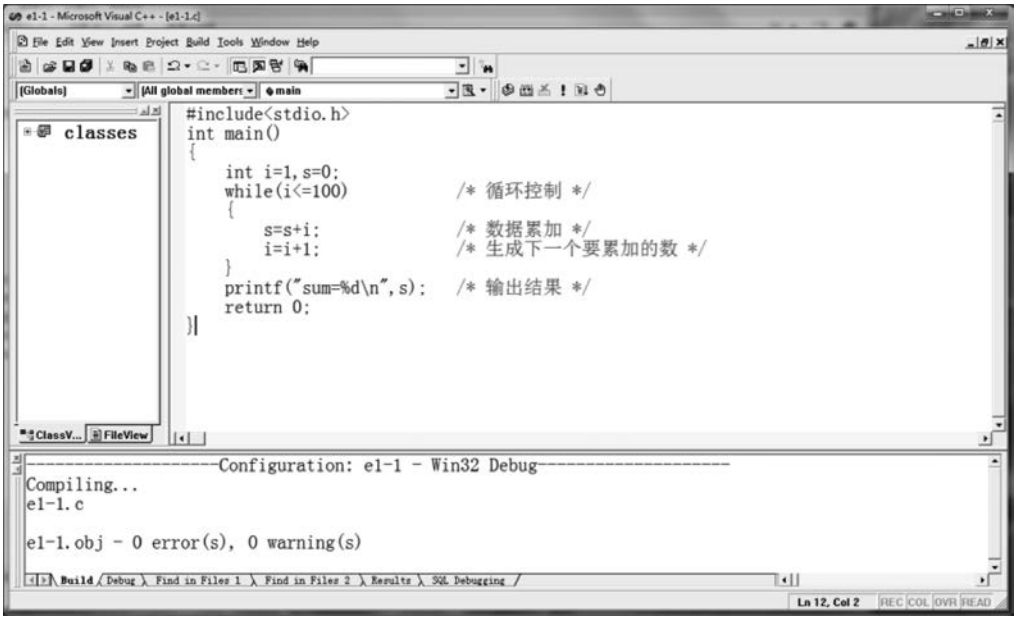

图 1-6　完成编译后的窗口

提示：在编译报告中，双击错误提示信息所在行，系统会自动在编辑窗口中定位存在错误的语句。

(3) 构建可执行文件。

编译完成后，选择 Build→Build e1-1.exe 命令或单击构建按钮，即可构建可执行文件 e1-1.exe。构建可执行文件时，系统自动生成构建结果报告，无构建错误时显示“0 error(s)”。图 1-7 是完成构建后的窗口。

说明：Build 命令中可执行文件的主文件名与源程序文件的主文件名相同，扩展名为 .exe。例如，本例中源程序文件名为 e1-1.c，可执行文件名为 e1-1.exe。

(4) 运行可执行文件。

成功构建 e1-1.exe 文件之后，选择 Build→!Execute e1-1.exe 命令或单击执行按钮 !，即可运行程序，结果如图 1-8 所示。

2. 验证标识符大小写的不等价性

(1) 在上面的程序中将“i=i+1”中的 i 改为 I，然后编译程序，查看编译结果，分析错误原因。

(2) 根据编译报告修改程序中的错误，例如可将程序中所有的标识符 i 均修改为 I。修

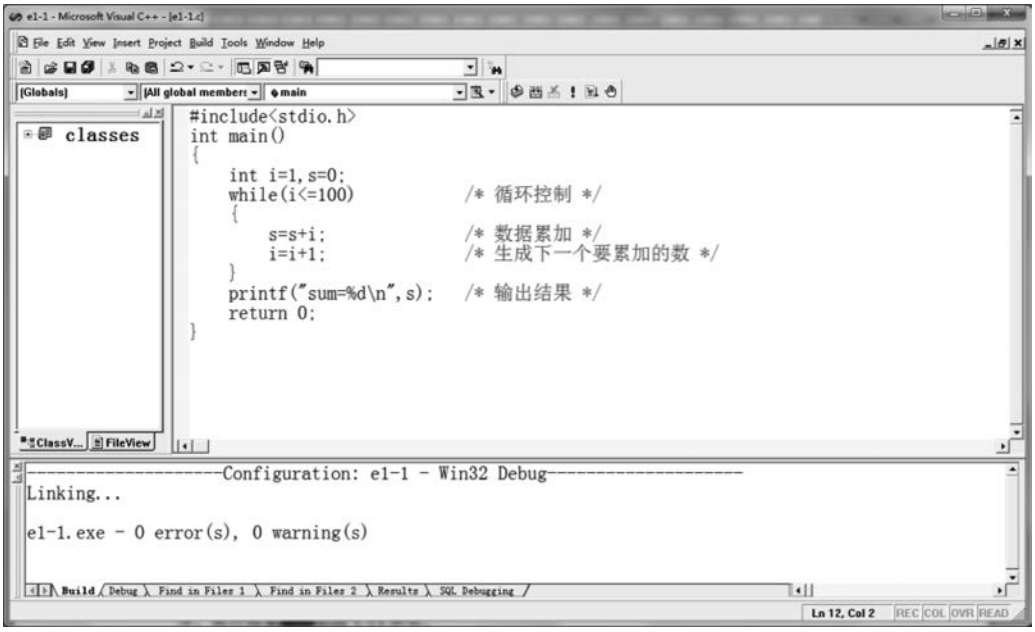

图 1-7　完成构建后的窗口

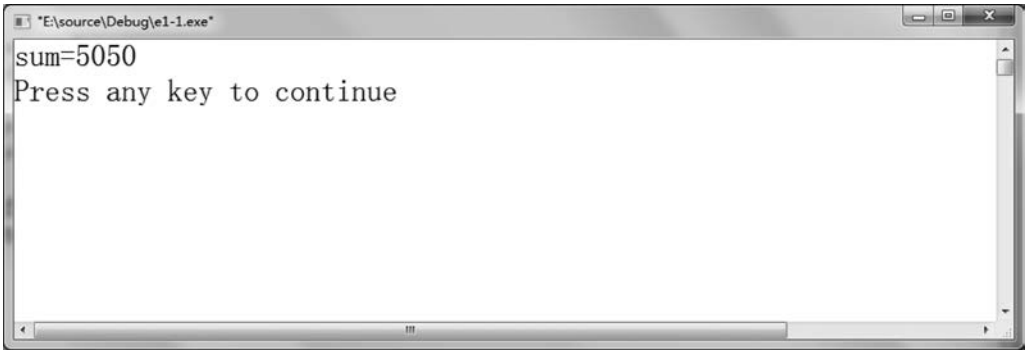

图 1-8　程序执行结果

改后再次编译,编译通过后构建可执行文件并运行程序。

该项内容说明标识符 i 与 I 是不相同的,验证了 C 语言中标识符的性质,即标识符中字母的大写和小写形式是不等价的。

希望读者自己进行类似的替换,然后查看程序的编译结果。例如,将程序中的 printf 改为 Printf,然后编译程序,构建可执行文件,查看结果。

该实验的目的是提醒读者在编辑程序时注意标识符大小写的区别。

3. 实验内容拓展

1) 问题分析

以下是“自然数累加”程序的算法。

步骤① 为 i 和 s 赋初值,使 i=1,s=0;

步骤② 判断 i 的值,若 i≤100,执行步骤③,否则转步骤⑤;

步骤③ s 加上 i;

步骤④ i加上1,转步骤②;

步骤⑤ 输出s的值,结束。

只需要对“自然数累加”算法稍加修改,即可获得计算10!的算法,如下所述。其中,与“自然数累加”算法的不同之处用阴影进行了标注。

步骤① 为i和s赋初值,使i=1,s=1;

步骤② 判断i的值,若i≤10,执行步骤③,否则转步骤⑤;

步骤③ s乘以i;

步骤④ i加上1,转步骤②;

步骤⑤ 输出s的值,结束。

2) 编辑运行程序

打开之前已建立的“自然数累加”程序,根据上述算法中加阴影的部分对程序进行修改后保存文件,然后编译运行程序。

说明:C语言中的乘法运算用*表示,例如a乘以b表示为a*b。

四、实验注意事项

(1) 在编辑源程序之前首先在磁盘上建立一个专用文件夹(如D盘中的myjob),以存储源程序文件。建立个人专用的程序文件夹,便于进行程序文件的管理。建议在进行后续实验时也将程序文件存储在该文件夹中。

(2) 在VC++ 6.0环境中保存文件时默认的扩展名为.cpp,这是C++语言程序的扩展名。若要存储为扩展名是.c的C语言程序,需要对扩展名进行指定。

(3) 在输入、编辑源程序时要注意保持良好的程序风格,以便于阅读、调试程序。但为了提高实验效率,程序中的注释信息以适量为宜。

实验 2　简单程序设计

一、实验目的

(1) 掌握基本输入与输出函数的格式及主要用法。
(2) 掌握数据类型的概念及简单变量的定义和使用方法。
(3) 掌握简单运算符的运算特点。
(4) 熟悉宏定义命令和文件包含命令的用法。
(5) 学习顺序结构程序设计的基本方法。

二、实验内容

1. 数据类型及输出格式练习

按以下要求修改主教材中例 2-2 的程序(本书中的示例序号均指主教材中的示例,此后不再说明)。

(1) 将变量 a、b 的数据类型改为 float,然后运行程序,记录运行结果。
(2) 将每个实数的域宽设置为 10,并保留两位小数,然后运行程序,观察结果变化情况。
(3) 修改(2)中的程序,将输出结果设置为左对齐。

附:例 2-2 源程序

```
#include <stdio.h>
int main()
{
    int a,b;                              /* 将变量 a、b 定义为整数型变量 */
    a = 8;                                /* 使 a 的值为 8 */
    b = 2000;                             /* 使 b 的值为 2000 */
    printf("%d + %d = %d\n",a,b,a + b);   /* 输出结果 */
    return 0;
}
```

2. 数据格式化输入练习

(1) 编辑运行例 2-6 的程序。

附:例 2-6 源程序

```
#include <stdio.h>
int main()
{
    int a,b;
    printf("Input a,b:");
    scanf("%d,%d",&a,&b);                 /*  %d 之间用逗号分隔 */
    printf("%d + %d = %d\n",a,b,a + b);
```

```
    return 0;
}
```

(2) 将程序中两个%d之间的逗号“,”去掉,然后编译运行程序。

3. 输入与输出字符数据练习

修改例2-7的程序,从键盘输入一个字符后将该字符及其ASCII码值都显示出来。以下是执行结果示例:

```
Input: A
Output: A,65
```

附:例2-7源程序

```
#include <stdio.h>
int main()
{
    char ch;                              /* 定义字符型变量ch */
    ch = getchar();                       /* 使用getchar()函数输入字符数据 */
    putchar(ch);                          /* 使用putchar()函数输出字符数据 */
    return 0;
}
```

4. 整型变量及算术运算练习

下面的程序使用了除法运算(/)和求余运算(%),编辑运行该程序,观察并分析程序的运行结果。

```
#include <stdio.h>
int main()
{
    int m,a,b;                            /* 定义整型变量 */
    scanf("%d",&m);
    a = m/5;                              /* 除法运算 */
    b = m%5;                              /* 求余运算 */
    printf("a = %d,b = %d\n",a,b);
    return 0;
}
```

5. 宏命令及数据的输入与输出综合练习

按以下要求修改例2-3的程序。

(1) 用符号常量PI表示圆周率。

(2) 用pow()函数实现r*r运算。

(3) 圆半径r的值通过键盘输入。

(4) 输出结果保留两位小数,并左对齐显示。

附:例2-3源程序

```
#include <stdio.h>
int main()
{
    float r = 5.7693,s;
    s = 3.1416 * r * r;
    printf("R = %10.3f,S = %10.3f\n",r,s) ;
    return 0;
}
```

6. 顺序结构程序设计

某学生有两门考试课程，实行百分制考核。编写程序，输入这两门课程的成绩，计算其平均成绩(主教材第 2 章习题的编程题第 1 题)。

三、实验指导

1. 数据类型及输出格式练习

(1) 要将变量 a、b 的数据类型改为 float，则需要修改程序中的代码“int a，b”和“printf ("%d＋%d＝%d\n"，a，b，a＋b)”。

(2) float 型实数输出域宽的设置由格式“%m.nf”确定，其中 m、n 均为具体的整数，m 指定域宽，n 指定小数位数。格式“%10.2f”将每个实数的域宽设置为 10，并保留两位小数。

(3) “%-m.nf”格式将输出结果设置为左对齐，在本程序中可使用格式符“%-10.2f”。

2. 数据格式化输入练习

(1) 执行程序后分别按以下格式输入数据，观察并分析程序的执行结果。

格式一：

```
8,2000 ↵(数据之间用“,”分隔)
```

格式二：

```
8 2000 ↵(数据之间用空格分隔)
```

在按格式二输入数据时，运行的结果显然是不正确的，这说明既没有语法错误也没有逻辑错误的程序运行后也可能得不到正确的结果。当运行程序后得到错误结果时，既要对源程序进行检查，也要对输入数据的格式、合法性等进行检查。

(2) 修改程序中的数据输入语句，将两个%d 之间的逗号“,”去掉，改为%d%d 格式，然后编译运行程序，仍分别按以上两种格式输入数据，观察并分析程序的执行结果。

(3) 将程序中变量 a、b 的数据类型修改为 float，然后编辑运行程序，观察并分析程序的执行结果。

3. 输入与输出字符数据练习

1) 程序分析

例 2-7 程序的运行结果如下：

```
A ↵
APress any key to continue
```

显然，根据要求需在输入数据及输出结果之前增加提示信息，并增加 ASCII 码值的显示。

对于任何一个可显示的 ASCII 码字符，既可使用 printf()函数的%c 格式或者 putchar()函数输出该字符，也可使用 printf()函数的%d 格式输出它的 ASCII 码值。

用户可使用 printf("%c，%d"，ch，ch)函数输出存储在 ch 变量中的字符及其 ASCII 码值。

2) 参考程序

程序如下：

```
#include <stdio.h>
int main()
{
    char ch;                          /* 定义字符型变量 ch */
    printf("Input: ");
    ch = getchar();                   /* 使用 getchar()函数输入字符数据 */
```

```
    printf("Output: %c,%d\n",ch,ch);    /* 使用printf()函数输出字符及其ASCII码值 */
    return 0;
}
```

4. 整型变量及算术运算练习

(1) 运行程序,输入一个是5的倍数的整数,观察并分析运行结果。

(2) 运行程序,输入一个不是5的倍数的整数,观察并分析运行结果。

(3) 运行程序,输入一个比5小的整数,观察并分析运行结果。

(4) 问题思考:两个整数进行除法运算(运算符为\)的结果是什么?

5. 宏命令及数据输入与输出综合练习

(1) 用宏命令define定义符号常量PI,如下所示。

```
#define PI 3.1416
```

(2) 用pow()函数实现r*r运算,并将程序中的圆周率3.1416用PI表示,如下所示:

```
s = PI * pow(r, 2);
```

由于使用了数学函数pow(),因此要在程序开始处使用下面的宏命令:

```
#include<math.h>
```

(3) 输入r可使用以下语句实现:

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

(4) 输出结果格式可通过格式符"%-.2f"实现。

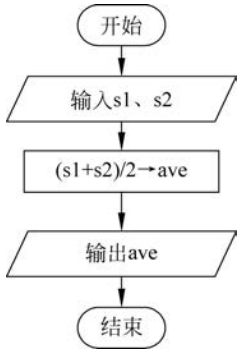

图1-9 计算平均成绩的算法流程图

6. 顺序结构程序设计

1) 程序分析

该问题的算法在主教材中已作介绍,算法流程图如图1-9所示,其中s1、s2分别表示两门课程的成绩,ave表示平均成绩。

2) 参考程序

程序如下:

```
#include<stdio.h>
int main()
{
    int s1,s2,ave;
    printf("s1,s2=");
    scanf("%d,%d",&s1,&s2);              /* 输入成绩1、成绩2 */
    ave=(s1+s2)/2;                       /* 计算平均成绩 */
    printf("Average: %d\n",ave);         /* 输出平均成绩 */
    return 0;
}
```

3) 程序调试

调试程序的注意事项如下。

(1) 输入数据的格式要与程序中要求的格式一致。例如,本参考程序要求用逗号",”分隔数据。

(2) 根据程序的运行情况调整数据输入与输出的格式,使数据的输入与输出格式更符合使用习惯。

(3) 运行程序,输入负数或超过100的较大数据,观察并分析程序的执行结果。希望读者进一步完善算法,将无效数据区分出来,并进行相应处理。

实验 3　选择结构程序设计

一、实验目的

(1) 学会使用关系表达式、逻辑表达式表示条件。

(2) 掌握 if 命令的 3 种形式的用法。

(3) 掌握 switch 命令的用法。

(4) 学会选择结构程序设计方法。

二、实验内容

1. if 命令及简单条件(单一关系表达式表示的条件)练习

修改例 3-1 的程序,使得平均成绩不低于 90 分时显示平均成绩及“优等生”信息,否则显示平均成绩及“加油!”信息,以下为程序执行结果示例。

第 1 次执行程序:

```
输入两门课程的成绩: 85,81 ↵
83,加油!
```

第 2 次执行程序:

```
输入两门课程的成绩: 95,89 ↵
92,优等生
```

附:例 3-1 源程序

```
#include <stdio.h>
int main()
{
    int s1,s2,ave;                          /* s1、s2 为课程成绩,ave 为平均成绩 */
    printf("输入两门课程的成绩: ");
    scanf("%d,%d",&s1,&s2);                 /* 输入课程成绩 s1、s2 */
    ave = (s1 + s2)/2;                      /* 计算平均成绩 ave */
    if(ave >= 90)                           /* 选择控制 */
        printf("优等生\n");                 /* ave 不低于 90 时执行该语句 */
    else
        printf("加油!\n");                  /* ave 低于 90 时执行该语句 */
    return 0;
}
```

2. if 命令及复杂条件(逻辑表达式表示的条件)练习

修改上面的程序,将判定优等生的条件改为:若任意一门课程的成绩不低于 90,即判定为“优等生”。

3. 单分支 if 命令练习

修改上面的程序,使其只有在优等生情况下才显示相关信息。

4. if-else if 命令练习

按以下要求修改例 3-8 的程序。

(1) 课程考核实行百分制,任何百分制之外的成绩数据均被视为错误数据。

(2) 当成绩有效时,任何一门课程成绩均不低于 90 时显示“优等生”,否则显示“加油!”。

(3) 在显示“优等生”或“加油!”之前首先显示平均成绩值。

附:例 3-8 源程序

```
#include<stdio.h>
int main()
{
    int s1,s2,ave;
    printf("输入两门课程的成绩: ");
    scanf("%d,%d",&s1,&s2);
    ave=(s1+s2)/2;
    if(ave<0)
        printf("数据错误!\n");          /* ave<0 成立时 */
    else if(ave>=90)
        printf("优等生\n");             /* ave<0 不成立,而 ave>=90 成立时 */
    else
        printf("加油!\n");              /* ave<0 不成立,ave>=90 也不成立时 */
    return 0;
}
```

5. 多分支选择结构练习

例 3-14 使用 if-else if 结构实现了“学生成绩分等级显示”的程序,事实上使用 switch 结构也能方便地解决这一问题,试用 switch 分支结构改写例 3-14 的程序。

6. 综合实验

编写学生成绩分等级程序。某学期有两门课程,每门课程都按百分制实行理论考核,按 A、B 两级实行实验考核。要求按照以下标准对学生的学习情况进行综合评价。

(1) 综合评价结论由理论考核成绩和实验考核成绩共同确定。

(2) 理论考核按照平均成绩分为以下 5 个等级。

优秀(excellence):平均成绩≥90;

良好(all right):80≤平均成绩<90;

中等(middling):70≤平均成绩<80;

及格(pass):60≤平均成绩<70;

不及格(fail):平均成绩<60。

(3) 当两门课程的实验考核成绩均达到 A 级时,综合评定等级即为理论考核的等级,否则按照理论考核的等级降一级认定。若理论考核等级为“不及格”,那么不管实验考核成绩如何,综合评定等级都为“不及格”。

三、实验指导

1. if 命令及简单条件(单一关系表达式表示的条件)练习

1) 程序分析

本实验没有改变原有程序的选择结构条件,只是分支处理有所改变。修改后的 if 命令

如下。

```
if(ave>=90)
    printf("%d,优等生\n",ave);
else
    printf("%d,加油!\n",ave);
```

2）程序调试

本程序调试应注意以下两点。

（1）数据输入格式要符合程序要求，即需要与 scanf()函数要求的数据格式一致。

（2）数据组合要全面，既要有各种等级搭配，又要能测试到程序的每个分支。

2. if 命令及复杂条件（逻辑表达式表示的条件）练习

（1）判定优等生的条件可用以下逻辑表达式表示。

```
s1>=90||s2>=90
```

（2）调试程序时至少要测试以下 3 组数据。

① s1≥90 的一组数据。

② s2≥90 的一组数据。

③ s1＜90 并且 s2＜90 的一组数据。

（3）问题思考：

为什么至少要测试以上 3 组数据？

3. 单分支 if 命令练习

（1）修改实验内容 1 中的程序，使其只有在优等生情况时才显示相关信息。以下为参考代码。

```
if(ave>=90)
    printf("%d,优等生\n",ave);
```

（2）修改实验内容 2 中的程序，使其只有在优等生情况时才显示相关信息。以下为参考代码。

```
if(s1>=90||s2>=90)
    printf("优等生\n");
```

（3）调试程序时所用的测试数据与上文相同。

4. if-else if 命令练习

1）程序分析

（1）任何百分制之外的成绩数据均被视为错误数据，可使用以下逻辑表达式表示错误数据的条件。

```
s1>100||s2>100||s1<0||s2<0
```

（2）优等生条件可使用下面的逻辑表达式表示。

```
s1>=90&&s2>=90
```

（3）显示“优等生”信息时可使用下面的语句。

```
printf("%d,优等生\n",ave);
```

(4) 显示“加油!”信息时可使用下面的语句。

```
printf("%d,加油!\n",ave);
```

2) 程序调试

本程序共有3个分支,有多种数据组合,在设计测试数据时要有尽量大的覆盖度。表1-1是一个测试数据表,供用户调试程序时参考。

表1-1 测试数据表

s1(成绩1)	s2(成绩2)	测试目标	预期显示结果
70	800	错误数据分支	数据错误!
79	−89	错误数据分支	数据错误!
88	90	优等生分支、加油分支	89,加油!
99	92	优等生分支、加油分支	95,优等生
0	0	优等生分支、加油分支	0,加油!
90	89	优等生分支、加油分支	89,加油!

5. 多分支选择结构练习

1) 编程分析

为减少case的数量,可先将分布在[0,100]范围的平均成绩ave(保留整数)通过表达式ave/10映射到[0,10]的较小范围内,如表1-2所示。在switch结构中使用switch(ave/10)形式进行选择控制,算法流程图如图1-10所示。

表1-2 ave/10与成绩等级对应表

ave(int型)的值	ave/10的值	对应等级
100	10	优秀(excellence)
小于100、不小于90	9	优秀(excellence)
小于90、不小于80	8	良好(all right)
小于80、不小于70	7	中等(middling)
小于70、不小于60	6	及格(pass)
小于60	5、4、3、2、1、0	不及格(fail)

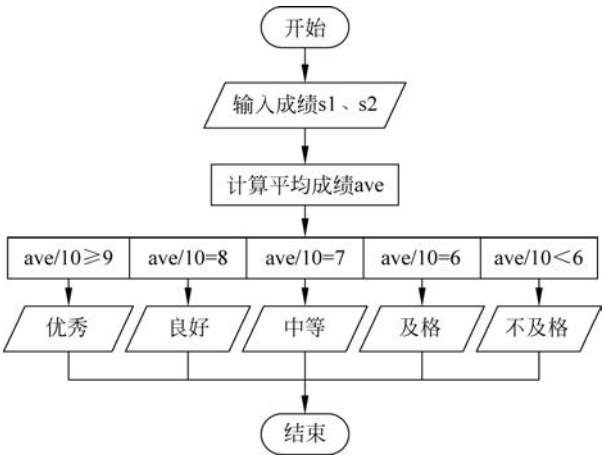

图1-10 使用switch结构的算法流程图

2）参考程序

程序如下：

```
#include<stdio.h>
int main()
{
    int s1,s2,ave;                          /* s1、s2 存储课程成绩,ave 存储平均成绩 */
    printf("Score: ");
    scanf("%d,%d",&s1,&s2);                 /* 输入课程成绩 */
    ave=(s1+s2)/2;                          /* 计算平均成绩 */
    switch(ave/10)
    {
        case 10:                            /* ave 为 100 */
        case 9:                             /* ave 小于 100、不小于 90 */
            printf("Result: excellence\n"); /* 优秀时输出等级结果 */
            break;
        case 8:                             /* ave 小于 90 不小于 80 */
            printf("Result: all right\n");  /* 良好时输出等级结果 */
            break;
        case 7:                             /* ave 小于 80、不小于 70 */
            printf("Result: middling\n");   /* 中等时输出等级结果 */
            break;
        case 6:                             /* ave 小于 70、不小于 60 */
            printf("Result: pass\n");       /* 及格时输出等级结果 */
            break;
        default:                            /* ave 小于 60 */
            printf("Result: fail\n");       /* 不及格时输出等级结果 */
    }
    return 0;
}
```

3）程序调试

（1）本程序有多个分支，在调试程序时应仔细设计多组数据，以保证每个分支都能得到测试。

（2）以下是两组测试样例，供读者参考。

```
Score: 77,98↵(此时 ave 为 87,ave/10 为 8,与"case 8: "匹配 */
Result: all right
Score: 89,92↵(此时 ave 为 90,ave/10 为 9,与"case 9: "匹配 */
Result: excellence
```

6. 综合实验

1）编程分析

（1）本实验题目是前述学生成绩分等级显示问题的进一步扩展。理论成绩分等级的程序分析和设计请参考前述教学内容。本实验需要特别考虑的是按照理论成绩分等级后要进一步对实验考核成绩进行判断，然后才能确定综合结论，可以考虑在 switch 语句中使用 if 语句处理。

（2）输入数据的方式也应是进行程序设计时认真考虑的问题，合理的数据输入方式会使输入的数据清晰，而且不容易出错。例如，可以将每门课程的理论考核成绩和实验考核成绩合为一体输入。假设有以下成绩：

```
a 课程的理论考核为 90,实验考核为 A
b 课程的理论考核为 80,实验考核为 B
```

则可以考虑使用以下方式输入数据：

90A,80B↵

2) 参考程序

程序如下：

```
#include <stdio.h>
int main()
{
    int a1,b1;                          /* a1、b1 分别存储 a、b 两门课程的理论考核成绩 */
    char a2,b2;                         /* a2、b2 分别存储 a、b 两门课程的实验考核成绩 */
    int ave;
    repeat:
    printf("请输入考核成绩: ");
    scanf("%d%c,%d%c",&a1,&a2,&b1,&b2);
    if(a1<0||b1<0||a1>100||b1>100)
        goto repeat;                                  /* 输入非法数据时要求重新输入 */
    ave=(a1+b1)/2;
    switch(ave/10)
    {
        case 10:
        case 9:
            if(a2=='A'&&b2=='A')
                printf("Result: excellence\n");    /* 理论考核优秀,实验均为A级 */
            else
                printf("Result: all right\n");     /* 理论考核优秀,实验不都为A级 */
            break;
        case 8:
            if(a2=='A'&&b2=='A')
                printf("Result: all right\n");     /* 理论考核良好,实验均为A级 */
            else
                printf("Result: middling\n");      /* 理论考核良好,实验不都为A级 */
            break;
        case 7:
            if(a2=='A'&&b2=='A')
                printf("Result: middling\n");      /* 理论考核中等,实验均为A级 */
            else
                printf("Result: pass\n");          /* 理论考核中等,实验不都为A级 */
            break;
        case 6:
            if(a2=='A'&&b2=='A')
                printf("Result: pass\n");          /* 理论考核及格,实验均为A级 */
            else
                printf("Result: fail\n");          /* 理论考核及格,实验不都为A级 */
            break;
        default:
            printf("Result: fail\n");              /* 理论考核不及格 */
    }
    return 0;
}
```

关于该参考程序的说明如下。

(1) 该程序对数据输入的合法性进行了一定的检验,能够确保理论考核的输入数据在0～100范围内,但对实验考核输入数据的合法性未作检验。程序中体现的实验考核成绩只有A和非A两种情况。

(2) 若实验考核成绩为A级,必须输入大写的“A”,否则将处理为非A级。当然,也可以修改程序,使得无论输入A或a都认定为A级。

实验 4　循环结构程序设计

一、实验目的

（1）通过本实验加深对循环控制结构相关概念的理解。

（2）熟练掌握 while、do-while 和 for 几种循环控制命令的特点及用法，掌握循环结构程序的设计和调试方法。

（3）掌握循环嵌套结构程序的设计方法。

二、实验内容

1. while 循环控制命令练习

在例 4-1 的程序中，while 命令的循环控制变量 i 的初始值为 1，其循环体被执行 N 次。要求将 i 的初值修改为 0，while 命令的循环体仍被执行 N 次，试编辑运行程序。

附：例 4-1 源程序

```
#include<stdio.h>
#define N 5
int main()
{
    int score,i=1,sum=0;              /* score存储成绩,i用于计数,sum用于成绩累加 */
    while(i<=N)                       /* 当i<=N时执行循环体 */
    {
        printf("Data:");
        scanf("%d",&score);           /* 输入课程成绩 */
        sum=sum+score;                /* 累加课程成绩 */
        i++;
    }
    printf("Average: %d\n",sum/N);    /* 输出平均成绩 */
    return 0;
}
```

2. do-while 循环控制命令练习

在例 4-2 的程序中，do-while 命令的循环控制变量 i 的初始值为 1，其循环体被执行 N 次。要求将 i 的初值修改为 0，do-while 命令的循环体仍被执行 N 次，试编辑运行程序。

附：例 4-2 源程序

```
#include<stdio.h>
#define N 5
int main()
{
```

```
    int score,i = 1,sum = 0;              /* score存储成绩,i用于计数,sum用于成绩累加 */
    do{
        printf("Data:");
        scanf("%d",&score);               /* 输入课程成绩 */
        sum = sum + score;                /* 累加课程成绩 */
        i++;
    }while(i <= N);                       /* i<=N为循环条件 */
    printf("Average: %d\n",sum/N);        /* 输出平均成绩 */
    return 0;
}
```

3. for循环控制命令练习

在例4-3的程序中,for命令的循环控制变量i的初始值为1,其循环体被执行N次。要求将i的初值修改为0,for命令的循环体仍被执行N次,试编辑运行程序。

附:例4-3源程序

```
#include <stdio.h>
#define N 5
int main()
{
    int score,i,sum = 0;
    for(i = 1;i <= N;i++)
    {
        printf("Data:");
        scanf("%d",&score);               /* 输入课程成绩 */
        sum = sum + score;                /* 累加课程成绩 */
    }
    printf("Average: %d\n",sum/N);        /* 输出平均成绩 */
    return 0;
}
```

4. 循环控制命令基本编程练习

(1) 编写程序计算n!,n的值从键盘输入。

(2) 编写程序求s不超过1000时n的最大值,s=1+2+3+…+n。

5. 循环嵌套结构练习

输出右三角乘法表。乘法表的形式如下:

```
                                                                1*1=1
                                                        1*2=2   2*2=4
                                                1*3=3   2*3=6   3*3=9
                                        1*4=4   2*4=8   3*4=12  4*4=16
                                          …       …
                1*8=8   …   4*8=32  5*8=40  6*8=48  7*8=56  8*8=64
1*9=9   2*9=18  …   5*9=45  6*9=54  7*9=63  8*9=72  9*9=81
```

6. 综合实验

取彩球问题。有12个彩球,包括3个白色、3个红色、6个黄色,从中任意取n个球(2≤n≤12),求出所有不同的取法。