第 1 章　绪论

1.1　什么是计算机程序

初学者往往有一种误解，认为计算机是“万能的”，会自动进行所有的工作。其实，计算机的每一步操作都是根据人们事先设定的指令进行的。例如，设定一条指令告诉计算机进行一次加运算，再设定一条指令命令计算机把加运算的结果输出到显示器上等。如果要让计算机执行一系列的操作，就必须事先编好一个指令序列，并将该指令序列输入到计算机中。这一条条的指令序列，就是我们常说的程序。

所谓程序，是指一组计算机能够识别和执行的指令序列（指令集合）。每一条指令使计算机执行指定的操作。只要让计算机执行这个程序，计算机就会“自动地”执行其中的每一条指令，有条不紊地进行工作。一个特定的指令序列，用来完成一定的功能。为了使计算机系统能实现各种功能，需要成千上万个程序。这些程序大多是由计算机软件设计人员根据需要设计好的，作为计算机软件系统的一部分提供给用户使用。此外，用户还可以根据自己的实际需求设计一些应用程序，例如学生学籍管理系统、财务管理程序、工程实践中的计算机程序等。

总之，计算机的一切操作都是由程序控制的，计算机的本质是程序的机器，程序和指令是计算机系统中最基本的概念。只有懂得程序设计，才能了解计算机的工作原理，才能更好地利用计算机来为人类文明和社会发展服务。

1.2　程序设计语言的发展

一个完整的计算机系统由硬件系统和软件系统组成。硬件系统是计算机实现自动控制与运算的物质基础，软件系统加载在硬件系统之上控制硬件完成各种功能。无论是系统软件还是应用软件，都是用程序设计语言编写的。

人们要利用计算机解决实际问题，一般都要预先编制解决该问题的程序。如前所述，程序是以某种程序设计语言为工具编制出来的一个指令序列，它表达了人们解决问题的思路，也规定了计算机所要执行的一系列的操作。因此，程序设计语言实际上就是用户用来编写程序的语言，它是人与计算机之间交互（交换信息）的工具。

程序设计语言的发展经历了从机器语言到汇编语言再到各种高级程序设计语言的过程。机器语言是这个发展过程的始点，它包含着程序设计语言未来发展的一切细胞；而 C 语言是这个过程的重要阶段，它不仅是计算机软件设计与开发的主流语言之一，也是认识和深入掌握其他程序设计语言的基础。

程序设计语言根据其对问题的处理方式，一般分为机器语言、汇编语言和高级语言三大类。

1. 机器语言

从根本上说，计算机只能识别和接受由 0 和 1 组成的指令，即计算机的工作是基于二进

制的。对计算机而言，一组机器指令就是程序，称为机器语言程序。

机器语言是最底层的计算机语言。用机器语言编写的程序，计算机硬件可以直接识别。在用机器语言编写的程序中，每一条机器指令都是二进制形式的指令代码。指令代码一般包括操作码和地址码，其中操作码告诉计算机要进行什么样的操作，而地址码则告诉计算机被操作的对象的存放位置。对于不同的计算机硬件（主要是 CPU），其指令系统是不同的，因此，针对一种计算机所编写的机器语言程序不能在另一种计算机上运行。由于机器语言程序是直接针对计算机硬件的，因此它的执行效率比较高，能充分发挥计算机的速度性能。但是，用机器语言编写程序的难度较大，容易出错，而且程序的可读性比较差，也不易移植。

2. 汇编语言

为了克服用机器语言编写程序的缺点，人们采用能帮助记忆的英文缩写符号（称为指令助记符）来代替机器语言指令代码中的操作码，用地址符号来代替地址码，以便于理解与记忆。用指令助记符及地址符号书写的指令称为汇编指令（也称为符号指令），而用汇编指令编写的程序称为汇编语言源程序。汇编语言也称为符号语言。

汇编语言与机器语言一般是一一对应的，因此，汇编语言也是与具体使用的计算机有关。由于汇编语言采用了助记符，因此，它比机器语言直观，容易理解和记忆，用汇编语言编写的程序也比机器语言编写的程序易读、易检查、易修改。

例如，为了计算表达式“3+2”的值，用汇编语言编写的程序与用机器语言（8086CPU 的指令系统）编写的程序如下：

```
PUSH    BP                  01010101
MOVE    BP, SP              10001011  11101100
DEC     SP                  01001100
DEC     SP                  01001100
PUSH    SI                  01010110
PUSH    DI                  01010111
MOVE    DI   0003           10111111  00000011  00000000
MOVE    SI   0002           10111110  00000010  00000000
MOVE    AX   DI             10001011  11000111
MOVE    AX   SI             00000011  11000110
MOVE    [BP-02], AX         10001001  01000110  11111110
POP     DI                  01011111
POP     SI                  01011110
MOVE    SP, BP              10001011  11100101
POP     BP                  01011110
RET                         11000011
```

其中，每一行的前半部分为汇编语言指令，后半部分（二进制形式的指令代码）为对应的机器语言指令。

这里需要说明的是，计算机不能直接识别汇编语言指令，用汇编语言编写的程序必须经过一种我们称为汇编程序的翻译程序（编译程序）将其翻译成机器语言编写的程序后，计算机才能识别并执行。这种翻译的过程称为“汇编”，负责翻译的程序称为汇编程序。

3. 高级语言

机器语言和汇编语言都是面向机器的语言，称为低级语言。低级语言对机器硬件的依赖

性很大，用它们开发的程序易读性、可移植性和通用性均很差，一般的计算机用户较难掌握。

程序设计的目标是：在保证程序正确的前提下，程序的可读性、易维护性及可移植性尽量好。为使程序的可读性好，不仅要求编程者具有良好的程序设计风格，还要求所使用的编程语言语句易于理解与掌握。所谓程序易维护，是指当程序的功能需要修改和增强时，所需的开销尽可能小。一个可移植性好的程序，应该在各种计算机和操作环境中都可以正常运行，并得到相同的结果。为此，高级程序设计语言应运而生。

高级程序设计语言是面向问题的程序设计语言，与计算机硬件结构无关，其表达方式接近于被描述的问题，易于理解与掌握。高级程序设计语言充分利用了一些数学符号和有关规则，比较接近数学语言，又被称为算法语言。

高级程序设计语言中的语句一般采用自然词汇，并采用与自然语言语法相近的自封闭语法体系，这使得所编写的程序代码更容易阅读和理解。高级语言提供了丰富的数据类型，并且允许用户自定义数据类型。此外，为了方便进行数值计算，高级语言还提供了丰富的运算符。与低级语言相比，高级语言最显著的特点是程序语句面向问题而不是面向机器，即独立于具体的硬件系统，大大地简化了程序的编制和调试，使得程序的通用性、可移植性和编制程序的效率得以大幅度提高，从而使不熟悉具体硬件系统的普通用户也能方便地使用某一种高级语言编制计算机程序。

1954 年出现的 Fortran 语言是第一种高级程序设计语言。在随后的 50 多年间，出现了数百种高级语言，而在国内先后流行且用得较多的高级语言有：适用于科学计算的 Fortran 语言、适用于程序设计入门的 Basic/QB/VB 语言、适用于程序设计教学的 Pascal 语言、适用于系统软件和应用软件开发的 C 语言、适用于数据库管理的 FoxBASE/FoxPro/SQL 语言、适用于商业与管理应用程序开发的 COBOL 语言、适用于逻辑推理的 Lisp 语言和 Prolog 语言、面向对象的程序设计语言 C++及 Java 等十几种。

高级语言的发展也经历了从早期语言到结构化设计语言、从面向过程到非过程化程序语言的过程。为解决 20 世纪 60 年代中后期出现的“软件危机”，1969 年提出了结构化程序设计的方法。1970 年，第一个结构化程序设计语言——Pascal 语言出现，标志着结构化程序设计时期的开始。80 年代，随着 Smalltalk 和 C++语言的出现，程序设计方法从面向过程这一传统方法转向面向对象的程序设计方法，目前面向对象的程序设计方法（OOP）已成为主流的程序设计方法。

高级语言的下一个发展目标是面向应用，即只需要告诉程序你要干什么，程序就能自动生成算法和代码，自动进行处理，这就是非过程化的程序语言。

这里需要注意程序设计语言与程序两者之间的关系：程序设计语言是进行程序设计的工具，是指计算机全部指令的集合；而任何计算机程序都需要用程序设计语言来编写，是为实现某个算法从某种高级语言中选择所需要的指令组成的集合。也就是说，程序是完成某个特定任务的一组指令序列。

例如，用高级语言求解表达式“3+2”的值，就比用机器语言和汇编语言求解要简单得多了。

用 Basic 语言编程如下：

```
10 A=3
20 B=2
```

```
    30 C=A+B
```

用 Pascal 语言编程如下：

```
Program Add
Var a,b,c: interger;
Begin
    a=3;
    b=2;
    c=a+b;
End
```

用 C 语言编程如下：

```
void main()
{   int a,b,c;
    a=3;
    b=2;
    c=a+b;
}
```

与前面的机器语言程序和汇编语言程序相比，可以看出：程序设计语言越低级，就越靠近机器硬件，其描述的程序就越复杂，其中的每一条指令（语句）就越难懂。反之，程序设计语言越高级，就越靠近人的表达与思维，其描述的程序就越简单，其中的每一条指令（语句）也就越容易理解。

这里需要注意的是：用任何一种高级语言编写的程序（称为源程序）都要经过编译程序（翻译程序）翻译成机器语言程序（称为目标程序）后机器才能执行，或者通过解释程序边解释边执行。不同的高级语言所需要的编译程序也是不同的。

1.3 C 语言的发展及其特点

C 语言是国际上广泛流行的计算机程序设计语言，其最早的原型是 ALGOL 60。1963 年，剑桥大学将其发展成为 CPL（Combined Programming Language）；1967 年，剑桥大学的 Martin Richards 对 CPL 进行了简化，产生了 BCPL（Basic Combined Programming Language）；1970 年，美国贝尔实验室（Bell Labs）的 Ken Thompson 对 BCPL 进行了修改，并取名叫 B 语言，意思是提取 CPL 的精华（Boiling CPL down to its basic good feature），并用 B 语言写了第一个 UNIX 系统；1973 年，AT&T 贝尔实验室的 Dennis Richie（D. M. Richie）在 BCPL 和 B 语言的基础上设计出了一种新语言，取 BCPL 中的第二个字母为名，这就是大名鼎鼎的 C 语言，Dennis Richie（D. M. Richie）也被称为“C 语言之父”；同年（1973 年），Ken Thompson 和 D. M. Richie 合作把 UNIX 90%以上的内核（Kernel）和应用程序用 C 语言改写，C 语言成为 UNIX 环境下使用最为广泛的主流程序语言。随着 UNIX 的日益广泛使用，C 语言很快便风靡全世界，成为当今世界上应用最为广泛的高级程序设计语言之一。

C 语言是一种用途广泛、功能强大、使用灵活的过程性（procedural）编程语言，既可用于编写应用软件，又能用于编写系统软件。C 语言既保持了 BCPL 和 B 语言的优点（精炼，接近硬件），又克服了它们的缺点（过于简单，无数据类型等）。C 语言的新特点主要表现在具有多种数据类型（如字符、数值、数组、结构体和指针等），C 语言的主要优点如下：

（1）简洁紧凑、灵活方便。

C 语言一共有 37 个关键字，9 种控制语句，程序书写自由，主要用小写字母表示。它把高级语言的基本结构和语句与低级语言的实用性结合起来。实际上，C 语言是一种很小的内核语言，只包括极少的与硬件有关的成分，C 语言不直接提供输入和输出语句、有关文件操作的语句和动态内存管理的语句等（这些操作由 C 语言的编译系统所提供的库函数来实现），C 语言的编译系统相当简洁。

（2）运算符丰富。

C 语言的运算符共有 34 个，包含的范围很广泛。C 语言把括号、逗号、赋值、强制类型转换等都作为运算符处理，从而使 C 语言的运算类型极其丰富，表达式类型多样化。灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

（3）数据类型丰富。

C 语言提供的数据类型包括：整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等，能用来实现各种复杂的数据类型的运算。C 语言还引入了指针的概念，使程序效率更高。另外，C 语言具有强大的图形功能，支持多种显示器和驱动器，并且计算功能和逻辑判断功能强大。

（4）C 语言是结构式语言。

结构式语言的显著特点是代码及数据的分隔化，即程序的各个部分除了必要的信息交流外彼此独立。这种结构化方式使程序层次清晰，便于使用、维护及调试。C 语言将系统提供的各种功能以函数形式提供给用户，这些函数可以方便地被调用。任何 C 语言程序最多包含顺序、选择及循环三种基本结构，程序结构清晰，且三种基本结构可以进行复合得到更为复杂的结构。

（5）语法限制不严格，程序设计自由度大。

虽然 C 语言也属于强类型语言，但它的语法比较灵活，给编程人员提供了较大的自由度。一般的高级语言语法检查比较严格，能检查出几乎所有的语法错误，而 C 语言放宽了语法检查，由编程人员自己保证程序的正确。例如，C 语言编译程序对数组下标是否越界不进行检查，而要由编程者自己检查和保证所编写代码的正确性。

（6）允许直接访问物理地址，可以对硬件直接操作。

C 语言既具有高级语言的功能，又具有低级语言的许多功能，它能够像汇编语言一样对位、字节和地址进行操作，可以用来写系统软件。因此，曾经有人称 C 语言是一种中级语言，集高级语言和低级语言的优点于一身。C 语言的这种双重性，使它既是成功的系统描述语言，也是通用的程序设计语言。

（7）生成的目标代码质量高，程序执行效率高。

C 语言原先是专门为编写系统软件而设计的，许多大的软件都用 C 语言编写，这是因为 C 语言的可移植性好，硬件控制能力高，表达和运算能力强。许多以前只能用汇编语言处理的问题，后来可以改用 C 语言来处理了。C 语言程序与汇编语言程序相比，一般只比汇编语言程序生成目标代码的效率低 10%～20%。

（8）适用范围广，可移植性好。

C 语言的一个突出优点就是适用于多种操作系统（如 DOS、UNIX），也适用于多种机型。由于 C 语言的编译系统相当简洁，因此很容易移植到新系统。而且，C 语言的编译系统在新

系统上运行时，可以直接编译“标准链接库”中的大部分功能，不需要修改源代码。因此，几乎在所有的计算机系统中都可以使用 C 语言。

C 语言不仅在速度和结构上有它的优势，而且每个 C 语言系统都提供了专门的函数库，程序设计人员可以根据不同需要对其进行剪裁，以适应各种程序的设计。由于 C 语言允许和鼓励分别编译，所以 C 语言可以使程序员方便地管理大型项目，最大限度地减少重复劳动。

当然，C 语言也存在一些不足，主要表现在以下几个方面：

（1）C 语言的缺点主要表现在数据的封装性上，这一点使得 C 语言在数据的安全性上存在很大的缺陷，而 C++语言提供的封装和信息隐藏机制则很好地克服了这一缺点。

（2）C 语言的语法限制不太严格。如对变量的类型约束不严格，影响了程序的安全性，对数组下标不作检查等。从应用的角度来说，C 语言比其他高级语言较难掌握。

（3）C 语言的指针是 C 语言的一大特色，可以说 C 语言优于其他高级语言的一个重要原因就是因为它有指针操作，使其可以直接进行靠近硬件的操作，但是 C 语言的指针操作也给它带来很多不安全的因素。C++语言在这方面做了很好的改进，在保留了指针操作的同时又增强了安全性。

C 语言不仅是面向过程的程序设计语言中功能最强、效率最高的语言，更是面向对象设计语言 C++、Java 和 C#的基础。

1.4 最简单的 C 语言程序

本节介绍几个简单的 C 语言程序，使读者对 C 语言程序有一个大致的了解，下面的几个例子虽然简单，但反映了一般 C 语言程序的特点及基本的组成成分。

1.4.1 最简单的 C 语言程序举例

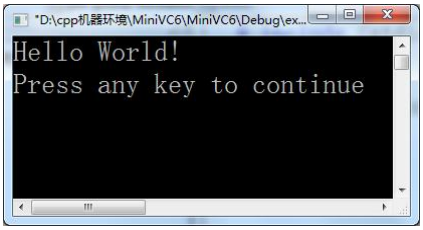
【例 1.1】要求在屏幕上输出以下一行信息：Hello World!

解题思路：在主函数中用 printf 函数原样输出以上文字。

编写程序：

```
# include <stdio.h>                  //编译预处理指令
int main(){                          //定义主函数
                                     //函数开始标志
    printf("Hello World!\n");        //输出所指定的一行信息
    return 0;                        //函数执行完毕时返回函数值 0
}                                    //函数结束标志
```

运行结果如下：



以上运行结果是在 Visual C++ 6.0 环境下运行程序时屏幕上得到的显示结果。其中，第一行是程序运行后输出的结果，第二行是 Visual C++ 6.0 在输出完运行结果后自动输出的一行信息，告诉用户：“如果想继续进行下一步，请按任意键”。当用户按任意键后，屏幕上不再显示运行结果，而返回程序窗口，以便进行下一步工作，如修改程序等。为节省篇幅，本书在以后显示运行结果时，不再包括内容为“Press any key to continue”的行。

程序分析：

程序第二行中的 main 是函数的名字，表示这个函数是“主函数”，main 前面的 int 表示此函数的类型是 int 类型（整型）。在执行主函数后会得到一个值（即函数值），其值为整型。程序第五行 return 0;的作用是：在 main 函数执行结束前将整数 0 作为函数值，返回到调用函数处。每一个 C 语言程序都必须有一个 main 函数，函数体由花括号{}括起来。程序第四行是一个输出语句，printf 是 C 语言编译系统提供的函数库中的输出函数（详见第 4 章的相关内容）。Printf 函数中双撇号内的字符串“Hello World!”按原样输出，\n 是换行符，即在输出“Hello World!”后，屏幕上的光标位置移到下一行的开头。这个光标位置称为输出的当前位置，即下一个输出的字符将出现在此位置上。每个语句最后都有一个分号;，表示该语句结束。

在使用函数库中的输入输出函数时，编译系统要求程序提供有关此函数的信息（例如对这些输入输出函数的声明和宏的定义、全局变量的定义等），程序第一行#include <stdio.h>的作用就是用来提供这些信息的。stdio.h 是系统提供的一个文件名，stdio 是“standard input & output”的缩写，文件后缀.h 的意思是头文件（header file），因为这些文件都是放在程序各文件模块的开头的，输入输出函数的相关信息已事先放在 stdio.h 文件中。现在，用#include 指令把这些信息调入供使用。对编译预处理指令#include 的使用，读者在此只需要记住：在程序中如果要用到标准库函数中的输入输出函数，应该在本文件模块的开头写上下面一行：

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

在以上程序各行的右侧，如果有//，则表示从此到本行结束是“注释”，用来对程序中有关部分进行必要的说明。在写 C 语言程序时应当多用注释，以便自己和别人理解程序各部分的作用。在程序进行预编译处理时，将每个注释替换为一个空格，因此在编译时注释部分不产生目标代码，注释对于运行不起作用。简单地说，注释只是给人看的，而不是让计算机执行的。

说明：C 语言允许的两种注释方式：

（1）以//开始的单行注释。如上例中的注释，这种注释可以单独占有一行，也可以出现在一行中其他内容的右侧。这种注释范围从//开始，以换行符结束，也就是说，这种注释不能跨行。如果要注释的内容一行写不下，可用多个单行注释，在下一行重新用//，然后继续写注释。

（2）以/*开始，以*/结束的块注释。这种注释可以包含多行内容，它可以单独占有一行（在行开头以/*开始，行末以*/结束），也可以包含多行。编译系统在发现一个/*后，会开始找注释结束符*/，把二者间的内容作为注释。

需要注意的是，在字符串中的//和/*都不作为注释的开始，而是作为字符串的一部分。如：

```
printf("//How are you!\n");
```

或

```
printf("/*How are you!*/\n");
```

输出分别是：

```
//How do you do!
```

和

```
/*How do you do!*/
```

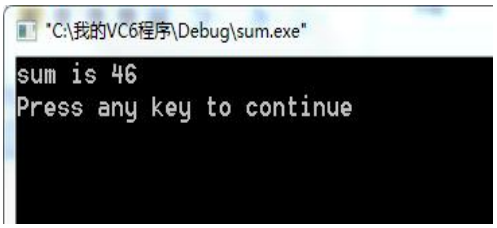
注释可以用汉字或英文字母表示。

【例 1.2】设置三个变量，变量 a 和变量 b 用来存放两个整数，sum 用来存放 a+b 的和，用赋值运算符“=”把相加的结果传送给 sum。

编写程序：

```
# include <stdio.h>                    //编译预处理指令
int main()                             //定义主函数
{                                      //函数开始
    int a,b,sum;                       //变量类型声明，定义 a、b、sum 为整型变量
    a=12;                              //对变量 a 赋值（12）
    b=34;                              //对变量 b 赋值（34）
    sum=a+b;                           //进行 a+b 的运算，并将结果存放到 sum 中
    printf("sum is %d\n", sum);        //输出结果
    return 0;                          //使函数返回值为 0
}                                      //函数结束
```

运行结果如下：



然后换行，程序执行结束。

程序分析：

本程序用来求两个整数 a 与 b 的和。第 4 行是声明部分，定义 a、b 和 sum 为整型（int）变量。第 5、6 行是两个赋值语句，给变量 a 和 b 分别赋以 12 和 34。第 7 行使 sum 的值为 a 与 b 之和。第 8 行输出结果，printf 函数圆括号内有两个参数，一个是双撇号中的内容 sum is %d\n，它是输出格式字符串，作用是输出用户希望输出的字符和输出的格式。其中，sum is 是用户希望输出的字符，%d 是指定的输出格式，d 表示用“十进制整数”输出。圆括号内的第二个参数 sum，表示要输出变量 sum 的值。在执行 printf 函数时，将 sum 变量的值（以十进制整数表示）取代双撇号中的%d。现在 sum 的值是 46，所以在输出时十进制整数 46 取代了%d，\n 是换行符。最后，输出双撇号中的字符 sum is 46，然后换行，程序执行结束。

由于本程序正常运行和结束，因此 main 函数的返回值应为 0。

【例 1.3】求两个整数中的较大者。

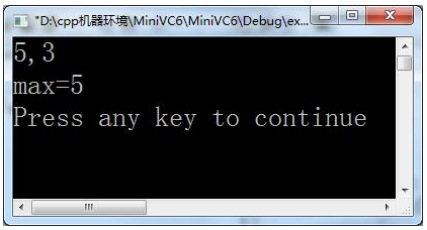
解题思路：用一个函数来实现求两个整数中的较大者。

编写程序：

```
# include <stdio.h>                    //编译预处理指令
```

```
//主函数
int main()                          //定义主函数
{                                   //主函数体开始
    int max(int x, int y);          //对被调用函数 max 的声明
    int a,b,c;                      //定义变量 a、b、c 为整型变量
    scanf("%d,%d", &a, &b);         //输入变量 a 和 b 的值
    c=max(a, b);                    //调用 max 函数，将得到的值赋给 c
    printf("max=%d\n", c);          //输出 c 的值
    return 0;                       //函数返回值为 0
}                                   //主函数体结束
//求两个整数中较大者的 max 函数
int max(int x, int y)           //定义 max 函数，函数值为整型，形式参数 x 和 y 为整型
{
    int z;                      //max 函数中的声明部分，定义本函数中用到的变量 z 为整型
    if (x>y) z=x;               //若 x>y 成立，将 x 的值赋给变量 z
    else z=y;                   //否则（即 x>y 不成立），将 y 的值赋给变量 z
    return(z);                  //将 z 的值作为 max 的函数值，返回到调用函数 max 的位置
}
```

运行结果如下：



第 1 行输入 5 和 3，赋给变量 a 和 b，第 2 行输出 max=5。

程序分析：

本程序包含两个函数：主函数 main 和被调用函数 max。

max 函数的作用是将 x 和 y 中较大者的值赋给变量 z。第 18 行 return 语句将 z 的值作为函数 max 的函数值，返回给调用的 max 函数（即主函数 main）。返回值是通过函数名 max 带回到 main 函数中去的。

程序第 5 行是对被调用函数 max 的声明（declaration）。之所以要做这个函数声明，是因为在主函数 main 中要调用 max 函数（程序第 8 行 c=max(a, b);），而 max 函数的定义却在 main 函数之后。C 语言编译程序对源程序的编译是自上而下进行的，在对程序第 8 行进行编译时，编译系统无法知道 max 是什么，因而无法把它作为函数调用处理。为了使编译系统能识别 max 函数，就要在调用函数 max 之前用 int max(int x, int y);对 max 函数进行“声明”。所谓声明，就是告诉编译系统 max 是什么，以及它的有关信息。有关函数声明详见第 7 章的相关内容。

程序第 7 行 scanf 是输入函数的名字（scanf 和 printf 都是 C 语言的标准输入输出函数），该 scanf 函数的作用是输入变量 a 和 b 的值。scanf 后面的圆括号中包含两部分内容：一是双撇

号中的内容，它指定输入的数据按什么格式输入，%d 要求输入的值为十进制整数。二是输入的数据准备放到哪里，即赋给哪个变量。现在，scanf 函数中指定的变量是 a 和 b，在 a 和 b 的前面各有一个&，在 C 语言中&是地址符，&a 的含义是“变量 a 的地址”，&b 的含义是“变量 b 的地址”。执行 scanf 函数，从键盘输入两个整数，送到变量 a 和 b 的地址处，然后把这两个整数分别赋给变量 a 和 b。

程序第 8 行用 max(a, b)调用 max 函数。在调用时，将 a 和 b 作为 max 函数的参数（称为实际参数，简称实参）的值分别传送给 max 函数中的参数 x 和 y（称为形式参数，简称形参），然后执行 max 函数的函数体部分（程序第 14～19 行），使 max 函数中的变量 z 得到一个值（即 x 和 y 中较大者的值）。return(z)的作用是把 z 的值作为函数 max 的值带回到程序第 8 行“=”的右侧（主函数调用 max 函数的位置），取代 max(a, b)，然后把这个值赋给变量 c，同时在第 8 行输出结果。在执行 printf 函数时，对双撇号括起来的 max=%d\n 的处理：将 max=原样输出，%d 由变量 c 的值取而代之，\n 执行换行。

注意：本例程序中的两个函数都有 return 语句，注意它们的异同。两个函数都定义为整型，都有函数值，都需要用 return 语句为函数指定返回值。但是，main 函数中的 return 语句指定的返回值一般为 0，而 max 函数的返回值是 max 函数中求出的两个整数中的较大值 z，只有通过 return 语句才能把求出的 z 值作为函数的值并返回到调用它的位置上（即 main 函数，程序第 7 行）。不要误以为在 max 函数中求出最大值 z 后就会自动地作为函数值返回调用处，必须用 return 语句指定将哪个值作为函数值，也不要不加分析地在所有函数的最后都写上 return 0;。

本例用到了函数调用、实际参数和形式参数等概念，读者可能觉得较难理解，我们将在第 7 章中予以详细介绍。在此引入此例，主要是使读者对 C 语言程序的组成和形式有一个初步的了解。

1.4.2 C 语言程序的结构

通过以上几个例子，可以看出一个 C 语言程序的结构有以下特点：

（1）一个程序由一个或多个源程序文件组成。

一个规模较小的程序，往往只包含一个源程序文件，如例 1.1 和例 1.2 所示。其源程序文件中只有一个函数（main 函数），例 1.3 的源程序文件中有两个函数 main 和 max，它们同属于一个源程序文件。在一个源程序文件中包括三个部分：

1）预处理指令。

C 语言编译系统在对源程序进行“翻译”之前，先由一个“预处理程序”（也称为“预处理器”“预编译器”）对预处理指令进行预处理。预处理指令#include <stdio.h>的作用就是将 stdio.h 头文件的内容读进来，放在#include 指令行，取代了#include <stdio.h>。由预处理得到的结果与程序其他部分一起组成一个完整的、可以进行编译的最后的源程序，然后由编译程序对该源程序正式进行编译，才得到目标程序。类似的预处理指令还有#define 等。

2）全局声明。

全局声明是指在函数之外进行的数据声明。例如，将例 1.2 程序中的 int a,b,sum;放到 main 函数的前面，这就是全局声明。在函数之外（通常在 main 函数之前）声明的变量称为全局变量，如果是在程序开头（定义函数之前）声明的变量，则在整个源程序文件范围内有效。在函数之中（函数体内）声明的变量称为局部变量，局部变量只在定义它的函数范围内有效。关于

全局变量和局部变量的概念及用法在本书第 7 章中将详细介绍。

3）函数定义。

如前述例 1.1～例 1.3 中的主函数和例 1.3 中的 max 函数，每个函数用来实现一定的功能。在调用这些函数时，会完成函数定义中指定的功能。

（2）函数是 C 语言程序的主要组成部分。

程序几乎全部的工作都是由各个函数分别完成的，函数是构成 C 语言程序的基本单位。在设计良好的程序中，每个函数都用来实现一个或几个特定的功能，编写 C 语言程序就是编写一个个函数。

一个 C 语言程序是由一个或多个函数组成的，其中必须包含一个 main 函数（且只能有一个 main 函数）。如例 1.1 和例 1.2 的程序只有一个主函数 main，而例 1.3 的源程序则由两个函数（主函数 main 及函数 max）组成，编译器在编译时对整个源程序文件统一进行编译。

一个小程序只包含一个源程序文件，在一个源程序文件中包含若干个函数（其中有且仅有一个主函数 main）。当程序规模较大时，所包含的函数比较多，如果把所有的函数都放在一个源程序文件中，则该源程序文件显得太大，不便于编译和调试。为了便于调试和管理，可以让一个程序包含若干个源程序文件，每个源程序文件又包含若干个函数。一个源程序文件就是一个程序块，即将一个程序分成若干个程序块。

编译器在进行编译时是以源程序文件为对象进行编译的。在分别对各源程序文件进行编译并得到相应的目标程序后，再将这些目标程序连接成为一个统一的二进制的可执行程序。

在程序中被调用的函数，可以是系统提供的库函数（如 printf 和 scanf 函数），也可以是用户根据需要自己编制和设计的函数（如例 1.3 中的 max 函数）。C 语言的函数库十分丰富，不同的 C 语言编译系统除了提供标准库函数外，还增加了其他一些专门的函数，如 Turbo C 提供了三百多个库函数。不同的编译系统所提供的库函数的个数和功能也不尽相同。

（3）一个 C 语言函数的构成。

一个 C 语言函数包含两个部分：函数首部和函数体。

1）函数首部。

函数首部通常是指函数的第一行，包括：函数名、函数类型、函数属性、函数参数（形式参数）名、参数类型等。

例如，例 1.3 中的 max 函数的首部为：int max (int x, int y)，其中，函数名 max 左侧的类型 int 指定函数 max 的函数类型为整型，max 为该用户自定义的函数的函数名，函数名 max 右侧圆括号内的变量 x 和 y 为函数的形式参数，它们的类型被定义（或声明）为整型。如果函数名右侧的圆括号内所列出的形式参数（形参列表）不止一个而是多个时，每两个形式参数之间必须以逗号隔开。

注意：一个函数名后面必须跟一对圆括号，括号内写函数的参数名，并声明（指定）参数的类型。如果该函数没有参数，可以在括号中写 void，也可以是空括号（括号内什么也不写），如：

```
int main(void)
```

或

```
int main()
```

2）函数体。

函数体即函数首部下面的花括号内的部分。如果在一个函数中包含有多层花括号，则最外层的一对花括号是函数体的范围。

函数体一般包括以下两部分：

a）声明部分

声明部分包括：定义在本函数中所用到的变量，对本函数所调用的函数进行声明等。

b）执行部分

执行部分由若干个语句组成，指定在函数中所进行的操作。

在某些情况下也可以没有声明部分，甚至可以既没有声明部分，也没有执行部分。如：

```
void temp()
{
}
```

这里 temp 是一个空函数，表示什么也不做，但这是合法的。

（4）程序的执行总是从 main 函数开始。

程序总是从 main 函数开始执行的，不论 main 函数在整个程序中的位置如何。main 函数可以位于程序的最前面，也可以放在程序最后面，或在一些函数之前、另一些函数之后。

（5）相关操作是由函数中的 C 语句完成的。

程序对计算机的操作是由 C 语句完成的，如赋值、输入与输出数据的操作都是由相应的 C 语句实现的。C 语言程序书写格式较为自由，一行内可以写几个语句，一个语句也可以分写在多行上，但习惯上一行只写一个语句。

（6）在每个数据声明和语句后面必须有一个分号。

分号是 C 语句的必要组成部分，如：

```
c=a+b;
```

其中分号是不能缺少的。

（7）C 语言本身不提供输入、输出语句。

C 语言中，输入和输出的操作是由库函数 scanf 和 printf 等函数来完成的。C 语言对输入输出实行“函数化”，这是由于输入输出通常涉及具体的计算机设备，把输入输出的操作用库函数来实现，就可以使 C 语言本身的规模较小，编译程序简单，从而很容易在各种机型上实现，程序的可移植性好。

（8）程序应当包含注释。

一个好的、有使用价值的源程序都应当加上必要的注释，以增加程序的可读性。

1.5 C 语言程序的运行

一个 C 语言程序编写完成后，就可以进入输入、编译与运行的过程。C 语言程序的上机过程一般分为以下几个步骤：

（1）输入 C 语言源程序，建立 C 语言源程序文件，其扩展名为.c（.C）。

（2）对 C 语言源程序文件进行编译与链接，生成目标文件与可执行文件（扩展名为.exe 或.EXE）。如果在这一步骤中发现有错误，则要对源程序文件进行编辑和修改，再进行编译与链接，直到在编译链接过程中没有错误警告为止。

（3）运行可执行文件得到结果。如果在运行过程中发现有错误，则要对源程序文件进行修改，再进行编译、链接与运行，直到没有错误为止。

下面以 Visual C++ 6.0 为环境介绍源程序的输入、编译、链接与运行的各个操作步骤。

1．源程序的输入

用户编写好的 C 语言程序，只有输入到计算机系统中，经过处理之后才能运行。因此，上机过程的第一步是要输入源程序，建立源程序文件。

首先打开 Visual C++的主窗口。具体操作如下：

从 Windows 桌面上顺序单击“开始”按钮的“程序”选项，再单击 Microsoft Visual Studio 6.0 选择 Microsoft Visual C++ 6.0 选项；如果在 Windows 桌面上已经建立了该快捷方式图标，则可以直接双击 Microsoft Visual C++ 6.0 图标。此时将显示 Visual C++ 6.0 的主窗口，如图 1-1 所示。主窗口的左侧是项目工作区，右侧是编辑窗口。

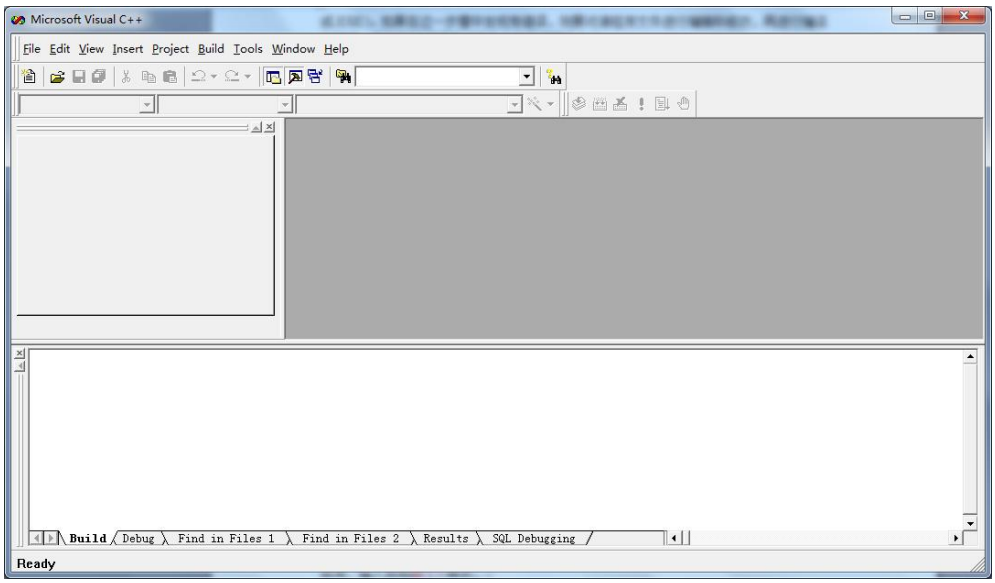


图 1-1　Visual C++ 6.0 主窗口

然后在主窗口中单击 File（文件）菜单项，在下拉的文件菜单中单击 New（新建）命令，此时显示一个 New（新建）对话框。单击该对话框上方的 Files 选项，在其下拉菜单中选择 C++ Source File（表示要建立新的源程序文件），如图 1-2 所示的新建对话框（1）。在该对话框的 File（文件名）输入框中输入文件名（假设文件名为 EX1_1），在 Location:（路径）输入框中输入源程序文件的存储路径（假设为 D:\program），如图 1-3 所示的新建对话框（2）。

在图 1-3 所示的新建对话框（2）中单击 OK 按钮，将回到 Visual C++的主窗口，由于已经指定了文件名，因此在主窗口的标题栏中显示出了文件名，如图 1-4 所示。

接下来就可以在图 1-4 所示的编辑窗口中输入和编辑源程序了。现在输入例 1.3 中的 C 语言程序，输入后如图 1-5 所示。

新的源程序输入完成并检查无误后，应将它保存在文件中，其操作过程为：在主菜单中单击 File（文件）选项，再在下拉菜单中单击 Save（保存）命令。

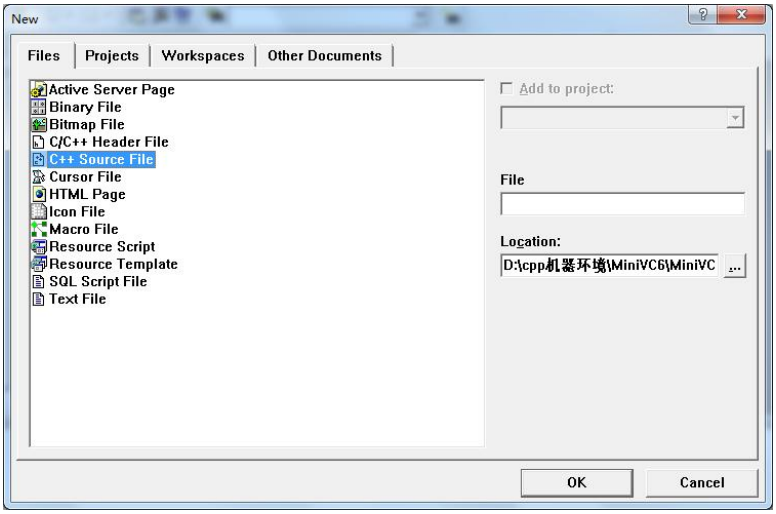


图 1-2　新建对话框（1）

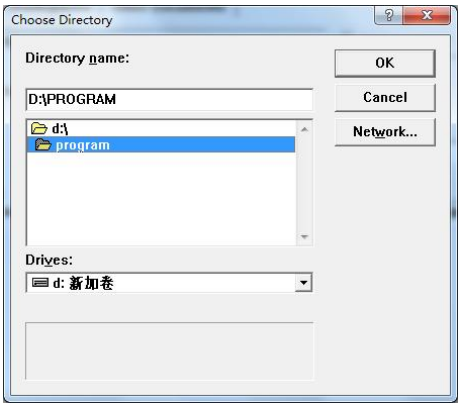


图 1-3　新建对话框（2）

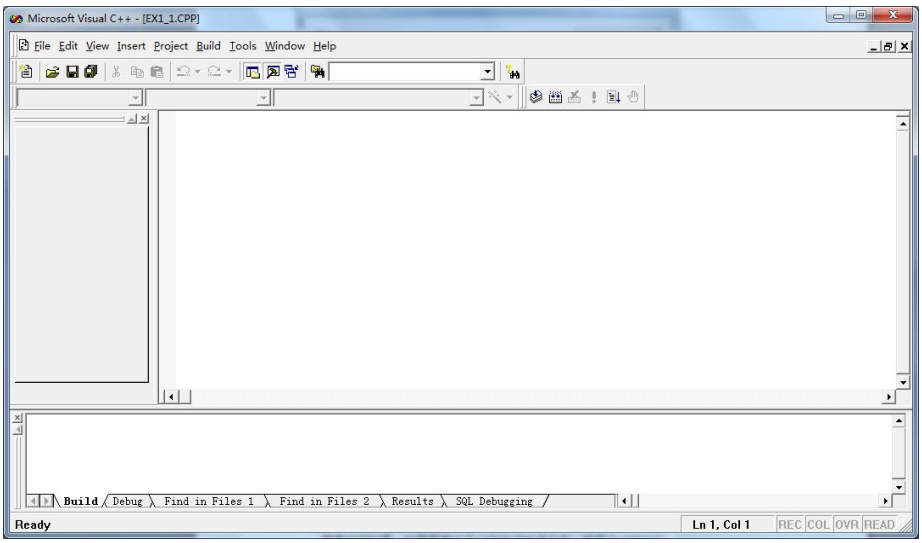


图 1-4　标题栏中显示文件名的 Visual C++主窗口

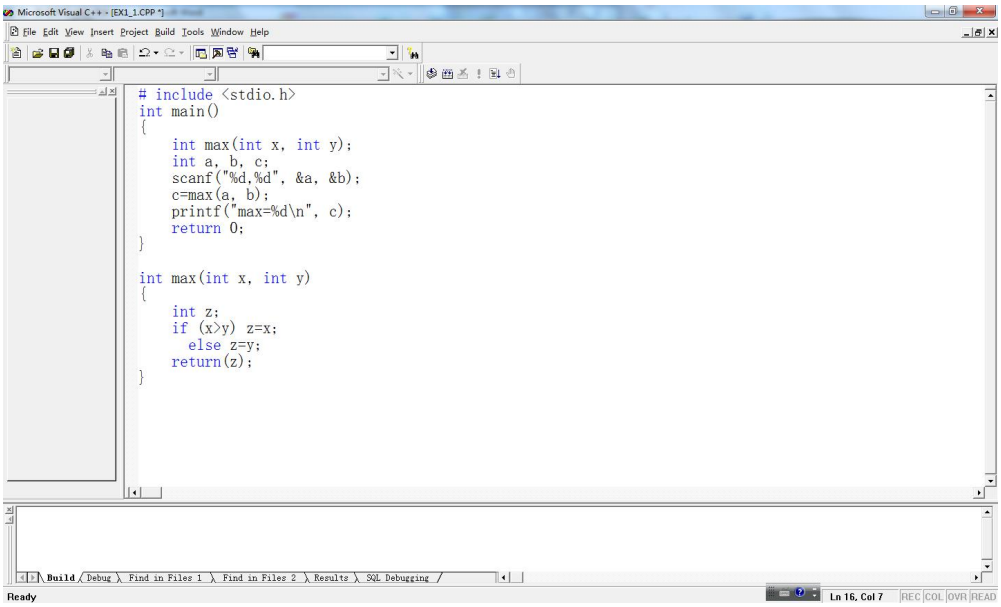


图 1-5　输入例 1.3 的源程序后的 Visual C++主窗口

需要说明的是，如果需要打开已有的源程序文件，其操作过程为：在“资源管理器”或“我的电脑”中找到已有的 C 语言源程序文件名（如 EX1_1.C），然后双击该文件名，此时即显示如图 1-5 所示的 Visual C++主窗口。

2. 编译

当用户将源程序输入计算机后，计算机还不能立即执行，还必须对源程序进行编译。这是因为计算机不能识别高级语言程序，只能直接识别用机器语言编写的程序（目标程序）。因此，对于用高级语言编写的源程序，必须将它翻译成机器语言程序后，计算机才能执行，这种将高级语言程序翻译成目标程序的过程称为编译过程，这种翻译的程序称为编译程序。

编译的过程是很复杂的。编译程序一方面要对源程序中的每一条语句进行识别和分析，最终翻译成与之对应的机器语言指令；另一方面还要找出源程序中的错误。如果在编译过程中发现源程序中有语法错误，则要显示相应的错误信息。在这种情况下，必须重新调用编辑程序对源程序进行编辑修改，而修改后的源程序还需要重新进行编译。这个过程可能要重复多次，直到编译过程中不再有错误发生为止。编译通过后，即生成相应的目标程序，它是由计算机能识别的机器代码组成。

对 C 语言源程序进行编译的操作如下：

单击主菜单中的 Build（构建）选项，在下拉菜单中选择 Compile（编译）<文件名>命令（如 compile EX1_1.C），如图 1-6 所示。

在选择编译命令后，屏幕显示如图 1-7 所示的提示信息，意思是：此编译命令要求一个有效的项目工作区，你是否同意建立一个默认的项目工作区？一般单击“是”这个选项，表示同意建立一个默认的项目工作区。此时就开始编译。

在进行编译的过程中，编译系统将检查源程序中有没有语法错误。如果有语法错误，还会指出错误的位置和性质，并且在主窗口的下方给出编译信息，如图 1-8 所示。

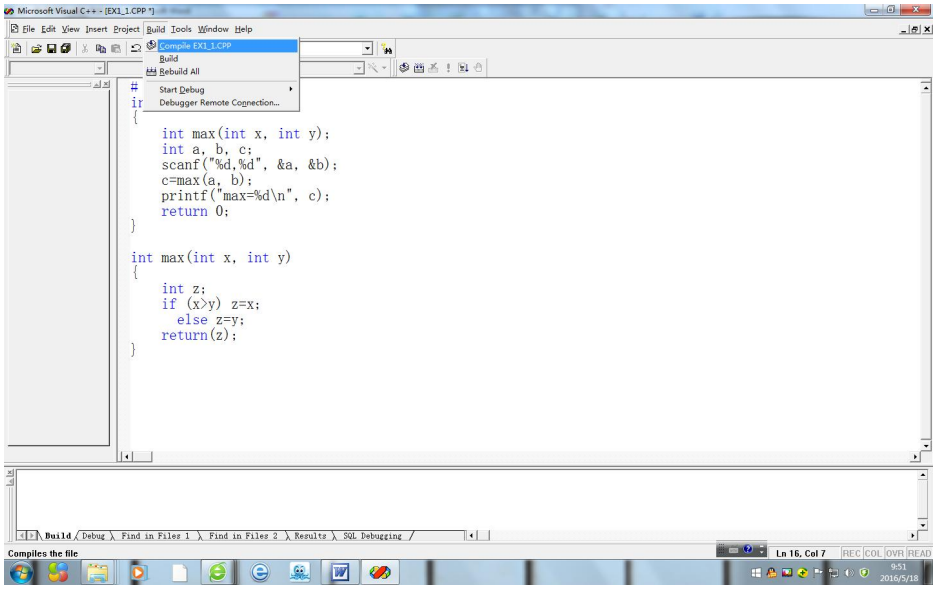


图 1-6　选择编译命令

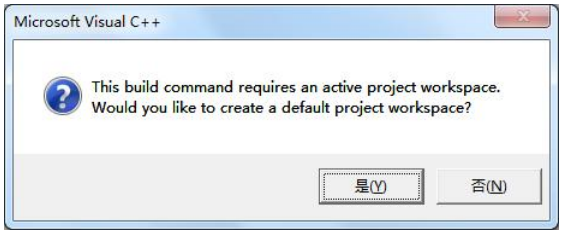


图 1-7　开始编译前的提示信息

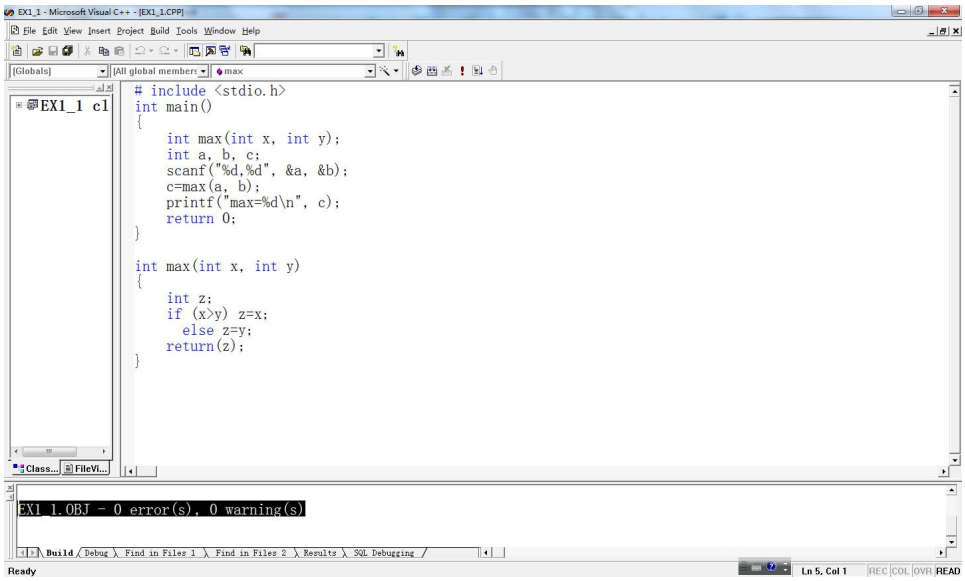


图 1-8　编译结束

需要特别注意的是，在每次编译一个新文件之前，应先单击 File（文件）选项，在其下拉菜单中单击 Close Workspace（关闭工作区）命令将原有的工作区关闭，以免新文件在原有工作区中编译。

3. 链接

在用户编写的程序中，一般都要调用一些库函数（如指数函数、对数函数、三角函数、输入输出函数等），有时还要调用一些用户自编或由专门人员编写的函数。但在调用这些函数时只给出了函数名以及有关的参数，在编译过程中不可能生成这些被调用函数的代码。因此，源程序经编译生成的目标程序文件还不能真正执行，还需要将被调用函数的代码链接进来。

所谓链接，是指将编译生成的目标文件与被调用函数的目标模块进行链接，最后生成一个计算机能真正执行的可执行文件。

在链接的过程中，也要进行查错，主要是检查调用各模块之间的联系以及存储空间分配等方面的错误。如果发现有链接错误，则要对有关源程序进行编辑修改，然后重新进行编译和链接。

对经过编译后的 C 语言程序进行链接的操作如下：

单击主菜单中的 Build（构建）选项，在下拉菜单中选择 Build（构建）<文件名>命令（如 Build EX1_1.exe），如图 1-9 所示。

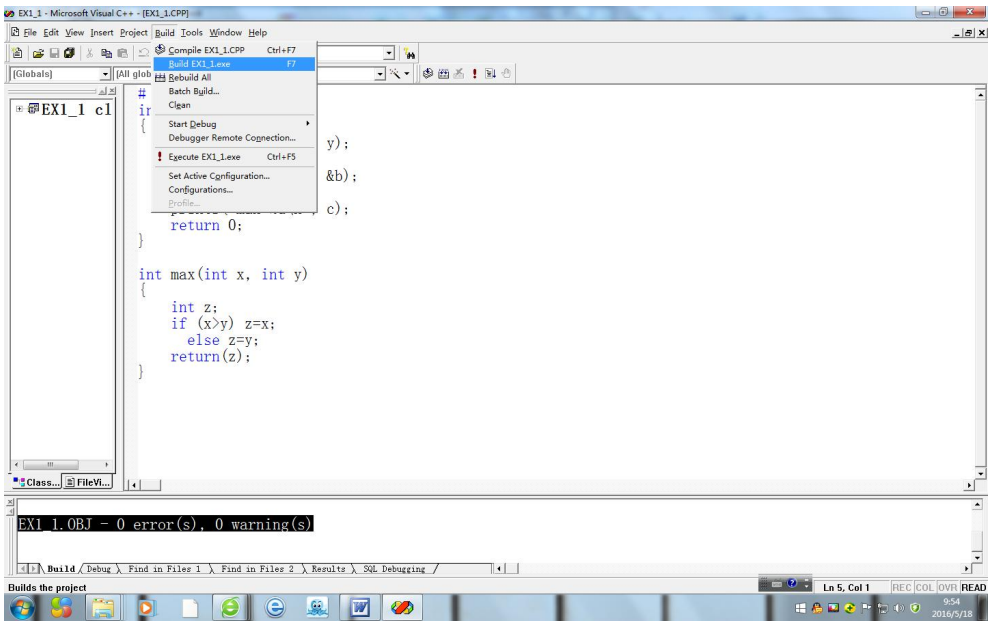


图 1-9　选择构建命令

在链接过程中如果没有错误产生，即生成一个扩展名为.exe 的可执行文件。链接完成后，在主窗口的下方将给出调试信息，如图 1-10 所示。

编译与链接是两个独立的过程，在有些编译系统中用两个独立的命令来实现，也有的编译系统用一个命令就完成编译和链接这两个过程。C 语言编译系统也可以用一个命令完成编译与链接这两个过程，其操作过程为：单击主菜单中的 Build（构建）选项，在下拉菜单中选择 Build（构建）命令。

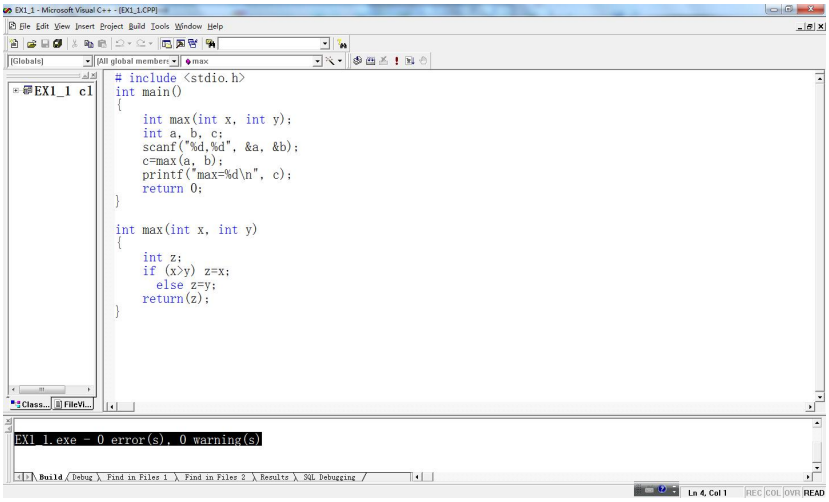


图 1-10　链接过程结束

4. 运行

源程序经过编辑、编译和链接过程，并且没有错误产生，在生成可执行文件后，就可以运行该可执行文件，得到所需要的结果。

必须指出，在编译过程中，虽然可以发现源程序中的大部分语法等错误，但不能发现程序中的所有错误，特别是不能发现程序中的逻辑错误（即程序应该实现的功能没有实现或结果错误）。因此，程序在运行过程中还有可能出现错误，系统会显示错误信息。有时虽然没有显示错误信息，但运行结果不正确，或运行过程中出现异常情况（如出现了死循环、死机等）。在这种情况下，还需要对源程序进行编辑修改，然后再进行编译、链接、再运行，直到运行结果正确为止。

在 Visual C++ 6.0 环境下，要运行一个刚通过编译、链接生成的可执行文件的过程如下：

单击主菜单中的 Build（构建）选项，在下拉菜单中选择! Execute（执行）<可执行文件名>命令（如! Execute EX1_1.exe），如图 1-11 所示。

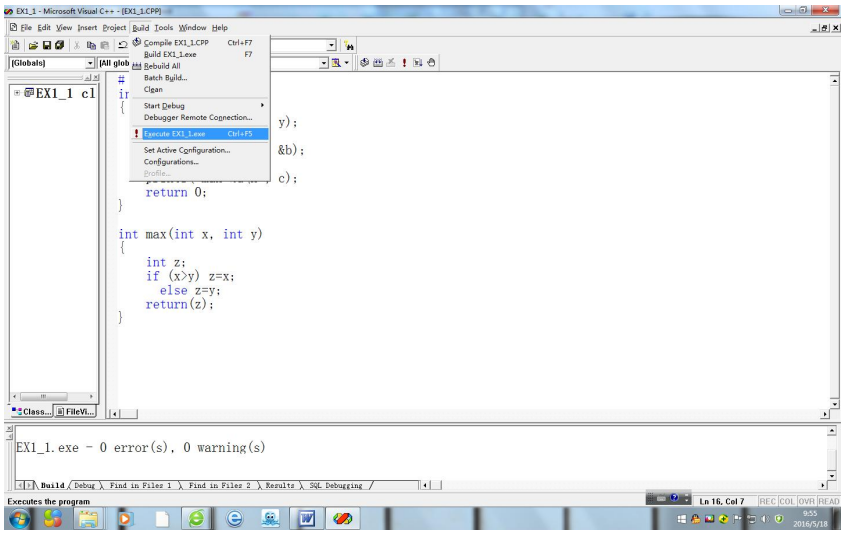
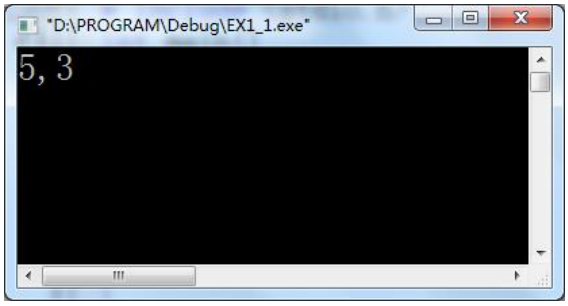
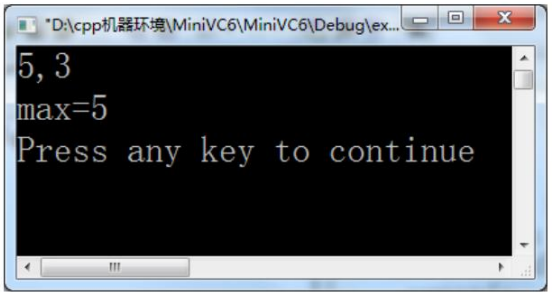


图 1-11　选择执行命令

在选择! Execute EX1_1.exe 命令后，即开始执行该可执行文件，并切换到输出结果的窗口，如图 1-12 所示。



（a）提示输入



（b）程序最终运行结果

图 1-12

本节简单介绍了 C 语言程序在 Visual C++ 6.0 集成环境下的上机操作过程，对于 Visual C++ 6.0 集成环境更为详细的操作方法的介绍，读者可参阅其他相关资料。

1.6　本章小结

本章首先介绍了什么是计算机程序与程序设计语言，接着介绍了计算机程序设计语言的发展阶段及分类，紧接着介绍了 C 语言的产生、发展及其主要特点，并通过几个简单的 C 语言程序示例，分析了 C 语言程序的基本构成及程序结构，最后介绍了在 Visual C++ 6.0 这个集成环境下一个 C 语言源程序的输入、编译、链接及运行的全过程。

本章主要内容如下：

（1）所谓程序，是指一组计算机能够识别和执行的指令序列（指令集合）。每一条指令使计算机执行指定的操作。只要让计算机执行这个程序，计算机就会“自动地”执行其中的每一条指令，有条不紊地进行工作。

（2）一个完整的计算机系统由硬件系统和软件系统组成。硬件系统是计算机实现自动控制与运算的物质基础，软件系统加载在硬件系统之上控制硬件完成各种功能。无论是系统软件还是应用软件，都是用程序设计语言编写的。

（3）程序设计语言根据其对问题的处理方式，一般分为机器语言、汇编语言和高级语言

三大类。计算机可以直接识别机器语言程序，在机器语言程序中，每一条机器指令都是二进制形式的指令代码。汇编语言也称为符号语言，它用指令助记符及地址符号书写指令，称为汇编指令或符号指令，而用汇编指令编写的程序称为汇编语言源程序。高级程序设计语言是面向问题的程序设计语言，与计算机硬件结构无关，其表达方式接近于被描述的问题，易于理解与掌握。高级程序设计语言充分利用了一些数学符号和有关规则，比较接近数学语言，又被称为算法语言。

（4）C 语言是一种用途广泛、功能强大、使用灵活的过程性编程语言，既可用于编写应用软件，又能用于编写系统软件。其主要特点有：简洁紧凑，灵活方便；数据类型及运算符丰富，语法限制不太严格，程序设计自由度大；允许直接访问物理地址，可以对硬件直接操作；生成的目标代码质量高，程序执行效率高；可移植性好，适用范围广等。

（5）C 语言程序的结构特点表现在：一个程序由一个或多个源程序文件组成；函数是 C 语言程序的主要组成部分；一个 C 语言函数由函数首部和函数体构成。一个 C 语言程序是由一个或多个函数组成的，其中必须包含一个 main 函数（且只能有一个 main 函数）。编译器在进行编译时是以源程序文件为对象进行编译的，在分别对各源程序文件进行编译并得到相应的目标程序后，再将这些目标程序链接成为一个统一的二进制的可执行程序。

（6）C 语言程序的上机过程一般分为以下几个步骤：输入 C 语言源程序，建立 C 语言源程序文件，其扩展名为.c（.C）；对 C 语言源程序文件进行编译与链接，生成目标文件与可执行文件（扩展名为.exe 或.EXE）；运行可执行文件得到结果。

习题 1

一、简述以下术语的含义

1．源程序、目标程序、可执行程序
2．程序编辑、程序编译、程序链接
3．程序、程序模块、程序文件
4．函数、主函数、被调用函数、库函数
5．程序调式、程序测试

二、简答题

1．什么是程序？什么是程序设计？
2．什么是计算机程序设计语言？程序设计语言可划分为几类？简述高级语言的特点。
3．简述 C 语言的优缺点及 C 语言程序的基本构成。

三、选择题

1．第一个体现结构化程序设计思想的高级程序设计语言是（　　）。
A．Fortran 语言　　B．C 语言　　C．Pascal 语言　　D．Java 语言
2．以下关于源程序与目标程序的关系，不正确的是（　　）。
A．用机器语言编写的源程序就是目标程序

B．用汇编语言编写的源程序需要经过汇编程序汇编为目标程序

C．用 C 语言编写的源程序需要经过编译程序编译为目标程序

D．不同的高级语言的编译器应该是一样的，都负责将源程序编译为目标程序

四、编程题

1．编写一个 C 语言程序，输出以下信息：

```
***************
How do you do!
***************
```

2．编写一个 C 语言程序，输入 a、b、c 三个值，输出其中的最小者。

五、上机运行以下程序，分析运行结果，掌握注释的用法。

1．

```
# include<stdio.h>
int main()
{
    printf("How do you do!\n");   //这是行注释，注释范围从//起至换行符止
    return 0;
}
```

2．

```
# include<stdio.h>
int main()
{
    printf("How do you do!\n");   /*这是块注释*/
    return 0;
}
```

3．

```
# include<stdio.h>
int main()
{
    printf("How do you do!\n");   /*这是块注释，如果在本行内写不完，可以在下一行继续写，这部
分内容均不产生代码*/
    return 0;
}
```

4．

```
# include<stdio.h>
int main()
{
    //printf("How do you do!\n");
    return 0;
}
```

5．

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

```
{
    printf("//How do you do!\n");   //自输出的字符串中加入//
    return 0;
}
```

6.

```
# include<stdio.h>
int main()
{
    /* printf("How do you do!\n");
    return 0; */
}
```