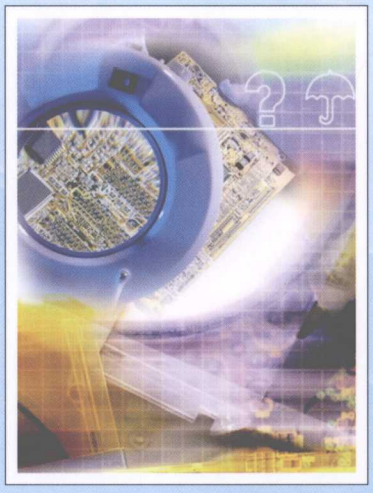
21世纪高等学校系列教材
21 Shiji Gaodeng Xuexiao Xilie Jiaocai

C语言程序设计

蒋清明 主编　黄晓宇 副主编　向德生 何宏 编著

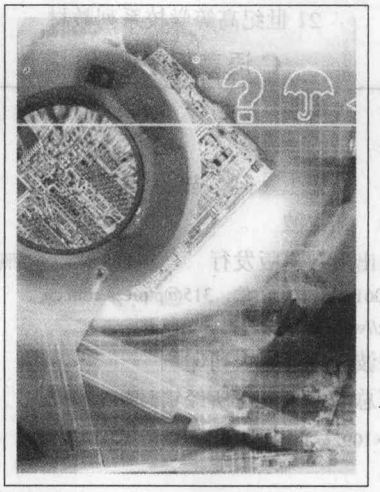


人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校系列教材
21 Shiji Gaodeng Xuexiao Xilie Jiaocai

C语言程序设计

蒋清明 主编　黄晓宇 副主编　向德生 何宏 编著



人民邮电出版社
北京

图书在版编目（CIP）数据

C 语言程序设计 / 蒋清明主编；向德生，何宏编著. —北京：人民邮电出版社，2008.4
（21 世纪高等学校系列教材）
ISBN 978-7-115-17502-1

Ⅰ. C… Ⅱ. ①蒋…②向…③何… Ⅲ. C 语言—程序设计—高等学校—教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2008）第 005273 号

内 容 提 要

本书较全面地讲述了 C 语言程序设计的基础知识，主要内容包括基本数据类型和运算符、控制结构、函数、数组、指针、结构与共用、文件以及文本与图形处理。每一章都附有精选的、多种类型的练习题，有助于读者复习、巩固所学知识，培养读者的实际编程能力。本书结构严谨，重点突出，由浅入深，举例经典。

本书可以作为各类高等院校、高职院校计算机专业及理工科非计算机专业学生学习“计算机程序设计”课程的教材，也可作为广大计算机爱好者学习 C 程序设计语言的参考书。

21 世纪高等学校系列教材

C 语言程序设计

◆ 主　　编　蒋清明
副 主 编　黄晓宇
编　　著　向德生　何　宏
责任编辑　邹文波

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：17.5
字数：424 千字　　2008 年 4 月第 1 版
印数：1 – 6 500 册　　2008 年 4 月河北第 1 次印刷

ISBN 978-7-115-17502-1/TP

定价：29.00 元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

随着经济全球化、社会信息化时代的到来，当代大学生不但要会使用计算机获取专业领域知识，还要会使用计算机进行编程，解决专业领域中的具体问题。C 语言是当前流行的操作系统 Windows、Linux、UNIX 上的一门系统开发语言，同时又是进行各专业问题计算的有效语言，因此，C 语言已成为各高校计算机专业和非计算机专业学生必学的一门语言。在非计算机专业等级考试中，C 语言已替代了 Pascal 和 Fortran 语言，因此，学好 C 语言的重要性已不言而喻。

然而，在 C 语言的教学过程中，预期教学目标与最终效果有着明显的差距，教师感觉难教，学生感觉难学、难理解，学会了也不会编程。针对这种情况，我们在编写本书的过程中，主要采取如下措施，以求收到更好的效果。

措施 1：体系合理。本书首先讲述 C 语言的输入/输出函数、运算符和数据类型，让学生尽快入门，学会简单的编程；然后讲述结构化、模块化编程；最后讲述数组、指针、结构、共用等构造类型。

措施 2：举例经典。为了配合非计算机专业的等级考试和提高计算机专业编程能力，本书的例题基本上采用经典算法讲解，在编写教材前，我们已将等级考试一些常考的算法进行分类，然后分解到各章之中。

措施 3：问题突破。为了帮助大家提高解决问题的能力，我们还编写了一本《C 语言程序设计上机实验指导与习题解答》，该书分为实验部分、问题解答和等级考试模拟试题三个方面。在学习过程中参考该书，有助于提高解决问题的能力。

我们在内容体系上做了精心的考虑，希望这些措施在教学过程中得到体现与落实，我们更希望学生在学习语言的过程中做到以下两点。

1. 学好语法知识。任何一门计算语言都有其相应的语法知识，它们是编写计算机程序的基础。

2. 多上机、多思考、多模仿。只有这样，才能提高自己的编程能力，为将来的学习和工作打下坚实的基础。

本书编写分工如下。蒋清明编写第 1 章、第 2 章，黄晓宇编写了第 7 章、第 8 章，向德生编写了第 5 章、第 6 章和第 9 章，何宏编写了第 3 章、第 4 章，全书由蒋清明、黄晓宇统稿。徐建波教授、肖建华教授在百忙中抽出时间对本书进行了审阅，在此表示感谢。

由于作者水平有限，加上时间仓促，错误之处在所难免，恳请广大读者批评指正。

编　者

2008 年 1 月

目 录

第1章 绪　论

1.1 C语言的发展过程

1.1.1 计算机语言的发展过程

计算机语言是人与计算机进行交互的工具，是用户进行计算机软件开发、编写计算机程序的工具。计算机语言的发展过程大致可以分为如下3个阶段。

1. 机器语言

计算机指令采用二进制（0、1）表示，也就是说，计算机能识别的指令代码只能是二进制形式。采用这种二进制形式表示的语言称为机器语言，或称为低级语言。如PC中两个数进行加法的指令为：0000010　11111001。由于机器语言采用的是二进制序列表示指令，十分难记；另外，采用机器语言编写的计算机程序具有不可移植性，即对某一种体系结构的计算机编写的计算机程序，在另一种体系结构的计算机上不能运行。

2. 汇编语言

由于机器语言难学，难记，为解决这一问题，计算机科学家将机器语言的每一条指令采用助记符表示，即机器语言的符号法，称为汇编语言。如上面PC的加法指令用符号表示为：ADD　AH，BL 。采用汇编语言编写的计算机程序必须经过翻译过程，变为机器语言后，计算机才能识别运行，这种翻译程序称为汇编程序，对应的过程称为汇编过程。用汇编语言编写的计算机程序仍与体系结构有关，具有不可移植性。但采用机器语言和汇编语言编写的计算机程序具有运算效率高的特点。

3. 高级语言

高级语言是一种更接近于自然的数学形式语言，如两个数的加法可写为z=x+y。采用高级语言编写的计算机程序与机器类型无关，具有可移植性、易学易记等特点，采用高级语言编写的程序称为源程序。但高级语言编写的计算机程序，计算机不能直接识别，必须经过翻译过程将其译为机器语言后，计算机才能识别运算。其翻译过程分为两种：一种是边翻译，边运行，翻译一句，执行一句，这种过程称为解释过程，对应的语言称为解释语言。每次执行程序时，都必须经过相同的翻译过程，如早期的BASIC语言和FoxBase等。采用解释语言编写的计算机程序不能离开其

解释环境；另一种是编译语言，它是将整个源程序全部翻译成机器语言指令后，计算机才能运行，这样的翻译过程称为编译过程，对应的翻译程序称为编译程序。源程序经编译后生成的机器语言程序称为目标程序，计算机不能直接运行目标程序，还必须经过连接过程，才能变为可执行文件，对应连接过程的程序称为连接程序，这样生成的可执行文件具有永逸性，即经过一次编译、连接后，生成的可执行文件以后不需要再进行编译连接过程，可以脱离语言环境，在同类型的计算机上仍可运行，如 ForTran 语言、Pascal 语言、Lisp 语言、Ada 语言和 C 语言等。

1.1.2 C 语言的发展过程

20 世纪 70 年代初，编写计算机系统软件时使用了一种符号法的自展组合语言 BCPL，BCPL 进一步发展为一种系统软件描述语言 B 语言。20 世纪 80 年代初，美国贝尔实验室软件开发人员 Dennis M. Ritchie 将 B 语言发展成为 C 语言。C 语言继承了 B 语言的特点，成为编写系统软件的重要工具语言。最初 C 语言有各种不同的标准，1983 年美国标准协会制定了 C 语言标准草案，称为 83ANSI C，1989 年正式修订后成为大家公认的标准，称为 89ANSI C。该标准中规定了 C 语言的关键字为 28 个，1999 年在原 89ANSI C 基础上增加了新的面向对象特性，并增加了 4 个关键字，该标准即为现在的 99ANSI C。

不同的编译器开发商在遵照 C 语言标准的基础上，对标准 C 新增了一些特性，如增加了图形图像处理能力，或在标准 C 的基础上，增加了特定的库函数，编译器的实现方式不同，这样市面上出现了 Borland 公司的 Turbo C，Microsoft 公司的 Microsoft C 等不同的编译器，都可实现对 C 语言程序的编辑、编译、连接和运行。Microsoft C 增加面向对象特性后，发展为 Microsoft C++ 和可视化编程的 Microsoft Visual C++。

1.2 C 语言的特点

C 语言作为一种系统开发语言，与其他高级语言或中级语言相比，具有如下特点。

（1）C 语言有丰富的运算符。C 语言除提供了其他高级语言提供的算术运算、关系运算、逻辑运算、下标运算和赋值运算等运算符外，还提供了位运算、地址运算、成员运算等运算符，这些运算符有助于程序员编写出高效的系统软件。

（2）C 语言有丰富的数据类型。C 语言包括整数型、字符型、实数型、空类型等基本数据类型和数组、指针、结构、共用、枚举、位结构等构造数据类型，还允许用户自定义新的数据类型。

（3）C 语言是结构化程序设计语言。C 语言提供了结构化程序设计的 3 种基本结构，即顺序结构、选择结构和循环结构。采用 3 种基本结构反复嵌套可实现任何复杂的运算。

（4）C 语言是模块化语言。C 程序由函数组成，这些函数可以是系统提供的库函数，也可以是用户自定义的函数，程序员可以利用函数构造计算机程序。

（5）任何一个 C 程序有且仅有一个称之为主函数的 main 函数。程序执行从主函数开始，其他函数通过主函数直接或间接调用才能执行，主函数执行结束时，标志程序执行结束。

（6）C 语言有丰富的预处理功能。预处理有利于提高程序的可读性、可移植性、正确性和书

写程序的高效性。

（7）C 语言是面向过程的语言，其函数采用面向过程的思想进行设计。

（8）C 程序具有可移植性。不同的程序员可以在不同的平台上设计实现某一大型软件中的子功能，然后在另一平台上进行组装，构成大型软件。

1.3 C 程序的结构和书写格式

1.3.1 C 程序的结构

在介绍 C 程序的基本结构与特征前，我们先看如下两个 C 程序的例子。

例 1.1 向控制台输出信息"Hello,World."。

```
#include <stdio.h>/*预处理命令：包含有标准输入输出库函数的头文件 stdio.h*/
void main()/*主函数*/
{
   printf("Hello,World.\n");
}
```

例 1.2 输入两个数 a、b，并输出其最大值。

解题思路 输入两个数 a、b，调用自定义函数 max(a,b)，求出其最大值并赋给 c，然后输出最大值 c。

```
#include <stdio.h>
int max(int a,int b)
{
   return a>b?a:b;
}
void main()
{
   int a,b,c;
   scanf("%d%d",&a,&b);
   c=max(a,b);
   printf("max=%d\n",c);
}
```

通过以上两例可以看出，C 函数的基本结构为：

```
[返回值类型] 函数名([形参说明表])
{
   变量定义部分;
   语句执行部分;
}
```

其中，C 函数中要用到的变量必须先定义，然后才能使用，因此变量定义在执行语句前。

而 C 程序结构如下：

```
[预处理语句]
[外部变量定义]
```

[用户自定义函数]
主函数定义

其中，[ ]中的内容为可省略部分。

1.3.2 C 程序的书写格式

在编辑 C 语言源程序时，我们应注意如下几点。

（1）C 程序采用块注释方法，块注释书写方法为：

```
/* 注释部分 */
```

注释部分只为了提高程序的可读性，不参与程序的编译和运行。但在书写格式上要注意："/"与"*"之间或"*"与"/"之间不能有空格。C 程序在 Visual C++ 6.0 编程环境下，也可采用 C++的注释方法，即若要对某行进行注释，只需在该行前面加上两个正斜杠符"//"即可。

（2）C 语言一般采用小写字母作为标识符。而 BASIC 语言中，一般采用大写字母作为标识符。

（3）C 语言是区分大小的。如：MAX、max 和 Max 表示的是 3 个不同的标识符。

（4）C 程序书写格式灵活，一个语句可连续写在多行上，一行也可以写多个语句。如例 1.2 中的 max 函数可以写成如下形式：

```
int max(int a,int b){    return a>b?a:b;}
```

（5）为了使书写的程序结构清晰、层次分明，建议采用"缩进对齐"的格式编辑 C 语言源程序，即同一结构层次的语句应左对齐，而结构下的语句相对于结构本身而言向右缩进。

C 程序书写格式灵活，这对程序员书写程序没有什么约束，如标识符可以采用小写字母，也可以采用大写字母表示，程序可以采用缩进对齐的格式书写，也可以不采用缩进对齐的格式书写，但我们建议初学者养成良好的程序书写规范，以便于交流和调试。

1.4 Visual C++ 6.0 上机操作

1.4.1 C 程序可执行文件的生成过程

C 语言程序可执行文件的生成过程如下。

（1）利用编辑器生成文本文件，该文本文件又称为 C/C++源程序，其扩展名为.C++或.C。编辑器可以是 C 系统提供的，也可以是其他文本编辑器，如：Notepad、Edit、Edlin 等。

（2）采用 C 编译器将源程序编译为二进制的机器目标文件，生成的目标文件扩展名为.obj。

（3）采用 C 连接程序将目标文件与库文件连接，生成可执行文件，可执行文件扩展名为.exe。

上述三步过程如图 1.1 所示。

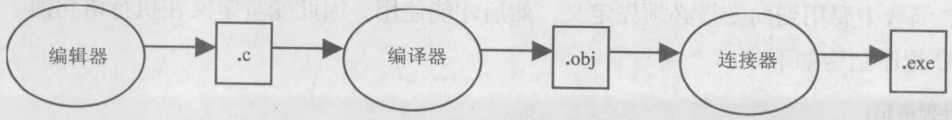


图 1.1 C 程序可执行文件的生成过程

1.4.2 Visual C++ 6.0 上机操作过程

Visual C++ 6.0 开发环境是一个基于 Windows 操作系统、并包含 C 语言子集的可视化集成开发环境（Integrated Development Environment，IDE），它集编辑、编译、连接、运行和调试等操作于一体，这些操作都可以通过单击菜单选项或工具栏按钮来完成，使用方便、快捷。在 Visual C++ 6.0 开发环境下，C 程序按工程（project）进行组织，每个工程可包括一个或多个 C/CPP 源文件，但只能有一个 main 函数。下面以例 1.1 为示例（例 1.1 源文件命名为 LT1_1.c）介绍在 Visual C++ 6.0 IDE 中建立工程并进行 C 程序调试的主要操作步骤。

由于 Visual C++ 6.0 的汉化版本很多，菜单项的汉化名称不尽相同（如主菜单项“Build”，有的版本翻译成“组建”，有的版本则翻译成“编译”，而其下拉菜单项中第二个子菜单项名也叫“Build”，有的版本翻译成“生成”，有的翻译成“构件”），所以下面在介绍相应菜单项名称时，用圆括号附上其英文菜单项名。

1. 启动 Visual C++ 6.0 IDE

如图 1.2 所示，可以从桌面上或“开始”按钮中的“程序”项中启动 Visual C++ 6.0 IDE，也可以从“开始”按钮中的“运行”项输入命令“MSDev”启动 Visual C++ 6.0 IDE。集成开发环境分为标题区、菜单区、工具栏区、工作区、程序编辑区、调试信息区等。

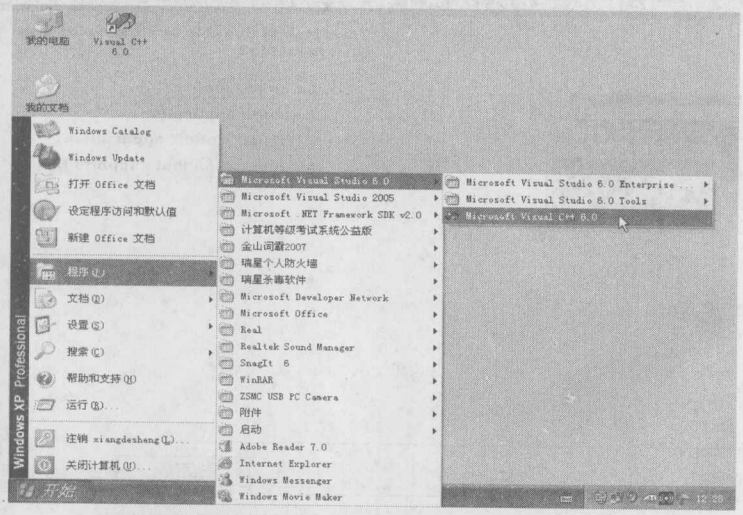


图 1.2 从开始按钮启动 Visual C++ 6.0 IDE

2. 工程（Project）的创建

从 Visual C++ 6.0 IDE“文件（File）”菜单上选择“新建（new）”菜单项，此时将弹出新建对话框，如图 1.3 所示。该对话框有分别用于创建新的“文件”、“工程”、“工作区”和“其它文档”等 4 个选项标签。选中“Win32 Console Application”项，在“工程”文本框中输入欲建工程名称，如 LT1_1（Visual C++ 6.0 IDE 自动将用户输入的工程名作为文件夹名）；然后在“位置”文本框中输入欲保存该工程的路径，或是通过单击其右边的 ... 按钮，在弹出的“选择目录”对话框中选择保存路径（图 1.3 中选择的位置为 D 盘根目录）。单击“确定”弹出如图 1.4 所示

的界面，在图 1.4 中选择“一个空工程（An empty project）”后单击“完成”按钮。然后在“新建工程信息（New Project Information）”对话框中单击“确定”按钮即可。

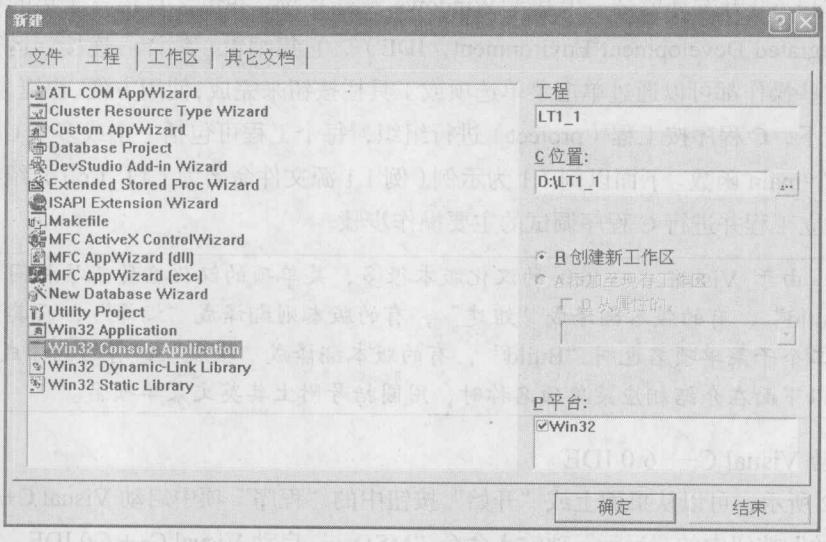


图 1.3　Visual C++ 6.0 IDE 的“新建”对话框

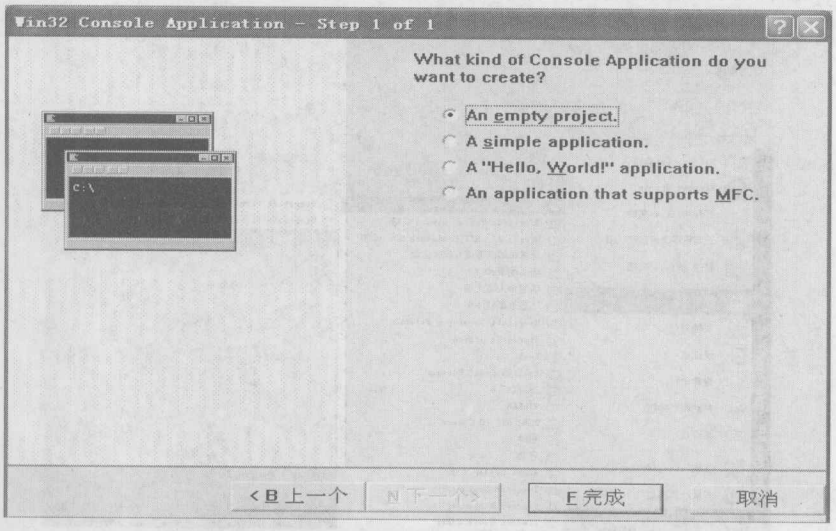


图 1.4　Visual C++ 6.0 IDE 创建控制台项目类型选择

3. 在工程（Project）中添加并编辑源程序

从 Visual C++ 6.0 IDE“工程（Project）”菜单上选择“添加到工程（Add to project）”菜单项，然后单击“新建（new）”下拉菜单项，弹出界面如图 1.5 所示，选择文件类型为 C++ Source File，输入源文件名（如 LT1_1.c，注意加上扩展名.c，若不加则默认扩展名为.cpp），选择保存源文件位置，单击“确定”按钮后将生成一个新的空文件 LT1_1.c，并弹出源文件编辑窗口（如图 1.6 所示），在编辑窗口中输入程序代码并修改，完成后可保存源文件。程序员也可按这种方式向工程中增加其他源文件。

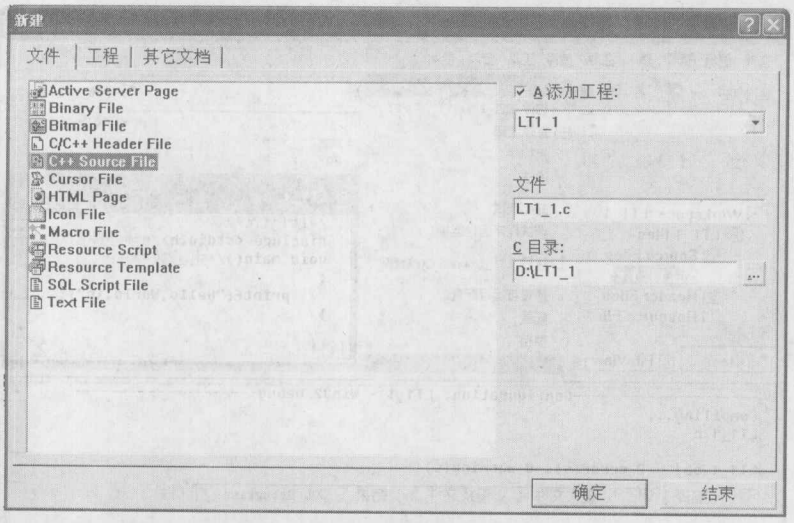


图 1.5 在工程中添加文件的对话框

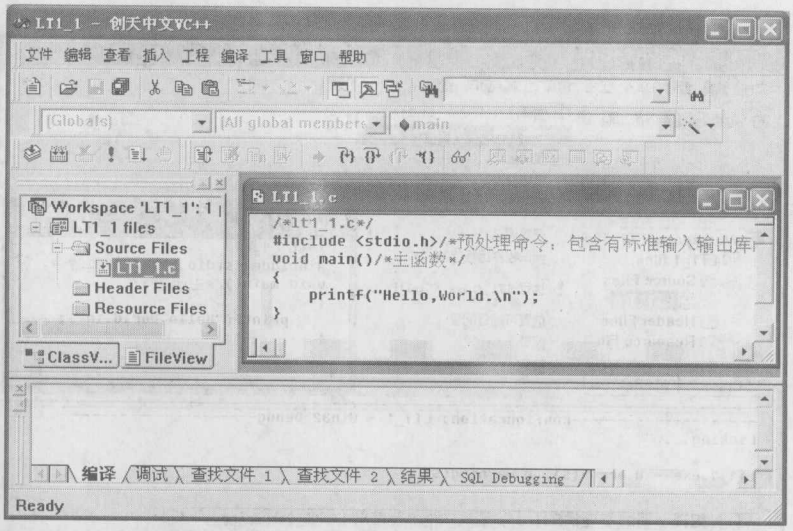


图 1.6 在编辑窗口中输入 C 源程序代码

4. 编译过程

选择下拉菜单——编译（Build）| 编译（Compile），对应的快捷方式为 Ctrl+F7，将生成.obj 目标文件。如图 1.7 所示，单击“编译（Compile）”后，在 IDE 的输出窗口显示“Compiling...LT1_1.c”，表示正在对 LT1 _ 1.c 执行 Compile 操作，执行完后显示“LT1 _ 1.obj – 0 error(s), 0 warning(s)”，表示执行完后生成了目标文件 LT1_1.obj，而且没有 error 错误，也没有 warning 错误。

5. 连接过程

选择下拉菜单——编译（Build）| 构件（Build），对应的快捷方式为 F7，将生成.exe 可执行文件。如图 1.8 所示，单击“构件（Build）”后，在 IDE 的输出窗口显示“Linking...”，表示正在对 LT1 _ 1.obj 执行 Link 操作，执行完后显示“LT1 _ 1.exe – 0 error(s), 0 warning(s)”，表示执行完后生成了可执行文件 LT1_1.exe，而且没有 error 错误，也没有 warning 错误。

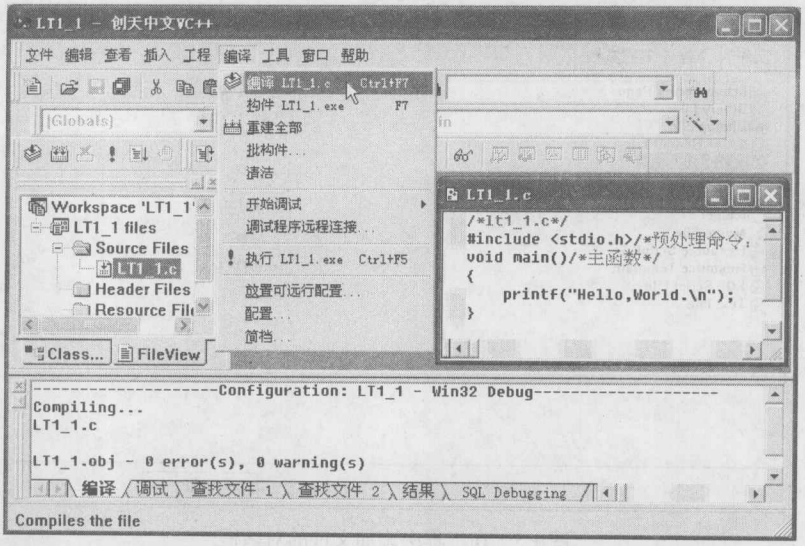


图 1.7　执行“编译（Build）| 编译（Compile）”后输出窗口显示的编译结果

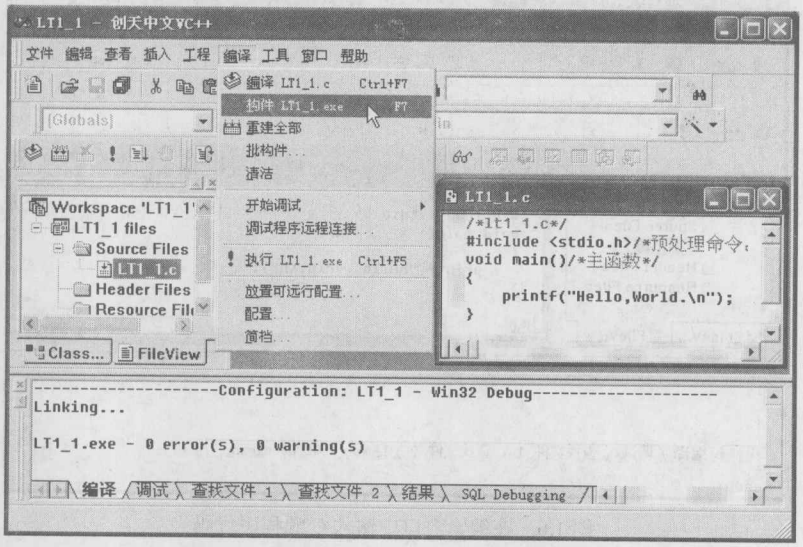


图 1.8　执行“编译（Build）| 生成（Build）”后输出窗口显示的编译结果

6. 执行程序

选择下拉菜单——编译（Build）| 开始调试（Start Debug）| 运行(Go)，对应的快捷方式为 F5，将运行生成的.exe 文件。

也可以选择下拉菜单——编译（Build）| 执行（!Execute），对应的快捷方式为 Ctrl+F5，将运行生成的.exe 文件。

实际上，连接工程的过程还包括对没有编译的源文件进行编译的过程，运行工程的过程还包括对没有编译、连接的源文件进行编译、连接的过程。即如果在输入源程序后，没有对该源程序进行编译就直接连接，或没有进行编译、连接就直接执行，则输出窗口的显示结果均如图 1.9 所示。结果表明，既进行了 Compile，又进行了 Link。

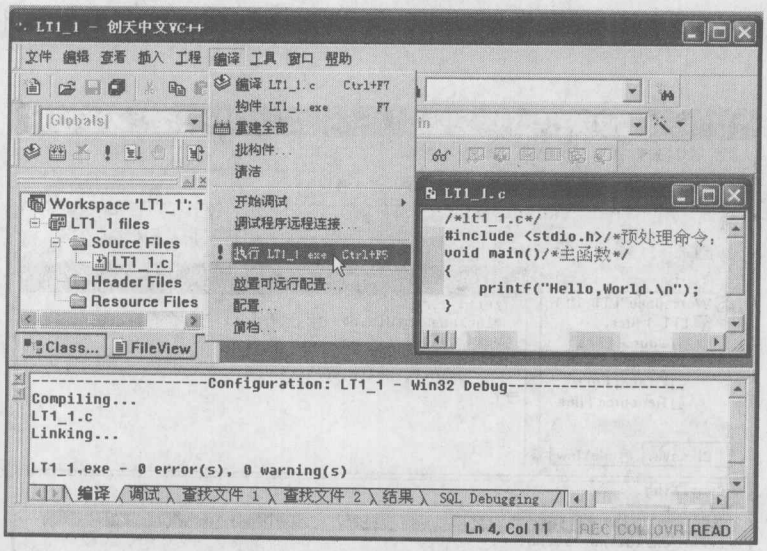


图 1.9 直接执行“编译（Build）| 运行（!Execute）”后输出窗口显示的编译结果

如果用户修改了 C/C++源程序，并重新执行该程序（再一次单击工具栏中的 ! 按钮或按 Ctrl+F5 键），Visual C++ 6.0 IDE 将弹出一个提示信息框，如图 1.10 所示。询问用户是否重新编译并生成相关的目标文件.obj 和可执行文件.exe。单击“是（Y）”按钮，Visual C++ 6.0 IDE 将重新完成编译（Build）的全过程。

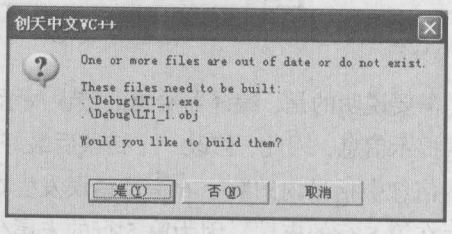


图 1.10 Visual C++ 6.0 IDE 的重建提示信息

1.4.3 程序调试

程序调试是程序设计过程中一个很重要的环节。编译器能找出源程序的语法错误，程序员可以根据错误信息的提示和上下文修改语法错误。但程序逻辑设计错误只能靠程序员利用调试工具对程序进行分析、检查与修改才能完成，这种逻辑错误往往不易发现。下面介绍 Visual C++ 6.0 IDE 查错方法。

1. 查找源程序中的语法错误

C 语言程序的错误主要包括两大类：一类是语法错误；一类是逻辑设计错误。

语法错误是指违背了 C 语言语法规则而导致的错误。语法错误分为一般错误（error）和警告错误（warning）两种。

（1）当用户程序出现 error 错误时，将不会产生可执行程序；

（2）当用户程序中出现 warning 错误时，通常能够生成可执行程序，但程序运行时可能发生错误，严重的 warning 还会引起死机现象。

所以，warning 错误比 error 错误更难于修改，应该尽量消除 warning 错误。

如果程序有语法错误，则在编译时，Visual C++ 6.0 IDE 的编译器将在输出窗口中给出语法错误提示信息，错误提示信息一般还可以指出错误发生所在位置的行号。用户可以在输出窗口中双

击错误提示信息或按 F4 键返回到源程序编辑窗口，并通过一个箭头符号定位到引起错误的语句，如图 1.11 所示。

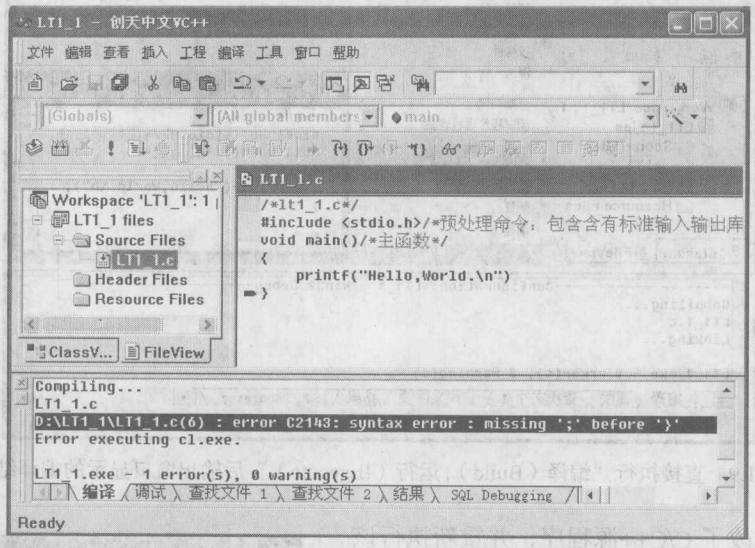


图 1.11　C/C++源程序存在调试时出现的错误信息

需要说明的是，编译器给出的错误提示信息可能不十分准确，并且一处错误往往会引出若干条错误提示信息，因此，修改一个错误后最好马上进行程序的编译或运行。例如，在图 1.11 中，错误提示信息中括号内的数字 6 指示错误发生在第 6 行，指示错误的箭头也指向第 6 行，但实际错误发生在第 5 行的末尾，因为第 5 行的末尾少了一个分号。

如果程序并没有违背 C 语言的语法规则，编译器也没有提示出错，而且程序能够成功运行，但程序执行结果却与原意不符，这类程序设计上的错误被称为逻辑设计错误或缺陷（Bug）。这类错误由于编译器不能给我们出错提示，所以必须利用“调试器（Debug）”对程序进行跟踪调试才能发现错误。

2. 调试器（Debug）

Visual C++ 6.0 IDE 提供了重要的调试工具——Debug，用于查找和修改程序中的逻辑设计错误。“编译”主菜单项下的“开始调试（D）”子菜单中有：Go、Step Into、Run To Cursor 及附加到当前进程（A）...4 个菜单项，它们的功能如表 1.1 所示。单击“编译”主菜单项下的“开始调试（D）”子菜单中的“Step Into”和“Run to Cursor”菜单项，均可以直接启动 Debug，而“Go”菜单项需要预先设置至少一个断点方可启动 Debug。

表 1.1　　开始调试（D）子菜单中的菜单项和功能

菜 单 项	快 捷 键	功　能
Go	F5	程序运行到某个断点、程序的结束或需用户输入的地方
Run to Cursor	Ctrl+F10	程序执行到当前光标处
Step Into	F11	单步执行程序的每一条指令，能进入被调用的函数内部
Step Over	F10	单步执行程序的每一条指令，不进入被调用的函数内部
附加到当前进程(A)...		将调试器与一个正在运行的进程相连接