

25

TP312.43  
X47

计算机综合培训教程丛书

# 从 C 到 C++ 进阶教程

肖任重 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

## 内 容 简 介

本书不仅详细讲解 C 语言的基本概念、语法规则和基本的 C 语言程序设计方法,还重点解析了 C++ 语言中关于面向对象程序设计思想的基本概念和方法,包括类与对象、虚函数、多态性和运算符重载等内容。作者拥有多年使用 C/C++ 进行开发和培训的经验,站在初学者的立场上,将全书内容分为 12 讲,按照快速培训的目标做了精心安排,力求通过丰富的例题来介绍枯燥的语法知识。同时附有大量的习题,以供读者课后练习。

本书适合于 C 和 C++ 的快速培训,读者对象为 C 和 C++ 的初学者,也适合于对 C 语言已有一定了解,希望进一步学习 C++ 语言的读者使用。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。  
版权所有,侵权必究。

### 图书在版编目(CIP)数据

从C到C++进阶教程/肖任重编著. —北京:电子工业出版社, 2002.2

(计算机综合培训教程丛书)

ISBN 7-5053-7457-5

I.从… II.肖… III.C语言—程序设计—教材 IV.TP312

中国版本图书馆CIP数据核字(2002)第004520号

责任编辑:曹建 特约编辑:逢积仁

印刷者:北京大中印刷厂

出版发行:电子工业出版社 <http://www.phei.com.cn>

北京市海淀区万寿路173信箱 邮编 100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:18.75 字数:459千字

版 次:2002年2月第1版 2002年2月第1次印刷

印 数:5000册 定价:24.00元

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系。联系电话:(010)68279077

# 前 言

本书面向的是那些渴望了解当前最为流行的编程语言——C/C++的读者。在计算机编程语言迅猛发展的今天，如何用最短的时间去掌握一门编程语言，是作者长期以来一直在思考的问题，这也是快速培训的目标之一。本书试图将作者的思考所得表现出来。全书尽可能使用大量的例程去介绍枯燥的语法，因为作者相信阅读源代码是研习编程语言最快和最好的办法之一。同时将大量篇幅放在最能体现该编程语言特点的地方和最重要的基本概念上。对于一些在实际中几乎不会用到的概念，只用很少的笔墨轻轻带过或者干脆略去不谈。语言方面尽可能简洁，浅显易懂，避免使用晦涩的专业术语，充分考虑到初学者的实际特点。书中所有例子程序均在 Visual C++ 6.0 环境下调试通过，并给出运行结果，使得读者在没有条件上机的情况下，也不影响对全书的阅读。

全书由 12 讲组成，每讲分成若干课。每讲开头都有“本讲概要”提示本讲的主要内容，后面均附有练习题，以巩固学习者的学习效果；每讲最后还给出“本讲难点”，以突出培训重点和学习难点。

讲授或自学完全书内容约需 30 个学时。各讲学时安排建议如下表：

	课时安排	每讲所需学时
第 1 讲	共 2 课内容，需 1 学时	1 学时
第 2 讲	第 1, 2 课内容需 1 学时，第 3 课内容需 1 学时	2 学时
第 3 讲	第 1, 2 课内容需 1.5 学时，第 3, 4 课内容各需 1 学时	3.5 学时
第 4 讲	共 4 课内容，需 2 学时	2 学时
第 5 讲	第 1 课内容需 1 学时，第 2, 3 课内容和第 4, 5 课内容各需 1 学时	3 学时
第 6 讲	共 3 课内容，需 1 学时	1 学时
第 7 讲	共 4 课内容，各需 1 学时	4 学时
第 8 讲	第 1 课内容需 2 学时，第 2, 3 课内容共需 2 学时	4 学时
第 9 讲	共 2 课内容，需 2 学时	2 学时
第 10 讲	共 2 课内容，需 2 学时	2 学时
第 11 讲	第 1 课内容需 1 学时，第 2 课内容需 1.5 学时	2.5 学时
第 12 讲	共 3 课内容，各需 1 学时	3 学时
	总学时	30 学时

由于作者水平有限，加之付梓仓促，书中难免存在一些疏漏和不足之处，希望广大读者不吝批评指正！

肖任重  
2002 年 1 月

# 目 录

<b>第 1 讲 C 语言概论</b> .....	1
<b>第 1 课 C 语言概论</b> .....	1
一、C 语言的由来 .....	1
二、C 语言的特点 .....	1
<b>第 2 课 简单的 C 语言程序</b> .....	2
一、简单例子 .....	3
二、C 语言程序的格式 .....	4
三、建立与运行 C 程序的过程 .....	5
四、用 MS Visual C++ 6.0 运行 C/C++ 程序的上机步骤 .....	6
<b>习题一</b> .....	10
<b>第 2 讲 数据</b> .....	11
<b>第 1 课 标识符、常量及变量</b> .....	11
一、标识符 .....	11
二、常量及变量 .....	12
<b>第 2 课 简单数据类型</b> .....	14
一、整型数据 .....	14
二、实型数据 .....	15
三、字符型数据 .....	16
四、不同数据类型间的转换 .....	17
<b>第 3 课 运算符和表达式</b> .....	18
一、运算符的优先级别和结合规则 .....	19
二、运算符和表达式 .....	19
<b>习题二</b> .....	26
<b>第 3 讲 语句与程序的流程结构</b> .....	28
<b>第 1 课 C 语言程序的语句</b> .....	28
一、表达式语句 .....	28
二、函数调用语句 .....	28
三、控制语句 .....	29
四、复合语句 .....	29
五、空语句 .....	29
<b>第 2 课 顺序结构</b> .....	30
一、赋值语句 .....	30
二、数据输出语句 .....	31

三、数据输入语句 .....	36
第3课 判断选取结构 .....	40
一、if 语句 .....	40
二、switch 语句 .....	45
第4课 循环控制结构 .....	48
一、while 语句 .....	48
二、do-while 语句 .....	50
三、for 语句 .....	50
四、转移语句 .....	54
习题三 .....	56
第4讲 数组 .....	60
第1课 一维数组的定义及引用 .....	60
一、一维数组的定义 .....	60
二、一维数组的初始化 .....	61
三、一维数组的引用 .....	61
第2课 二维数组的定义及引用 .....	63
一、二维数组的定义 .....	63
二、二维数组的初始化 .....	63
三、二维数组的引用 .....	64
第3课 字符数组与字符串 .....	65
一、字符数组的定义 .....	65
二、字符数组的初始化 .....	66
三、字符数组的引用 .....	66
四、字符串和字符串结束标志 .....	66
五、字符数组的输入输出 .....	67
六、字符串处理函数 .....	68
习题四 .....	71
第5讲 函数 .....	73
第1课 函数的定义和声明 .....	73
一、函数概述 .....	73
二、函数的定义 .....	75
三、函数参数和函数的值 .....	76
四、函数的声明 .....	80
第2课 函数的调用 .....	81
一、函数的调用 .....	81
二、函数的嵌套调用 .....	82
三、函数的递归调用 .....	82
四、数组作为函数参数 .....	83
第3课 传值方式 .....	87

第4课 变量的作用域和存储类别 .....	90
一、变量的作用域 .....	90
二、存储类别 .....	92
第5课 外部函数和内部函数 .....	97
一、内部函数 .....	97
二、外部函数 .....	97
习题五 .....	99
<b>第6讲 编译预处理</b> .....	100
<b>第1课 宏定义</b> .....	100
一、不带参数的宏定义 .....	100
二、带参数的宏定义 .....	102
三、宏定义和函数 .....	104
<b>第2课 文件包含</b> .....	104
<b>第3课 条件编译</b> .....	106
习题六 .....	108
<b>第7讲 指针</b> .....	110
<b>第1课 指针概论</b> .....	110
一、指针的概念 .....	110
二、变量的指针和指针变量 .....	111
<b>第2课 指针与数组</b> .....	115
一、指向数组的指针 .....	115
二、指针与字符数组 .....	119
三、指针与多维数组 .....	122
<b>第3课 指针与函数</b> .....	126
一、指针作为函数的参数 .....	126
二、函数的返回值为指针 .....	131
三、指向函数的指针 .....	132
<b>第4课 指针数组和指针的指针</b> .....	134
一、指针数组 .....	134
二、指针的指针 .....	136
三、命令行参数 .....	137
习题七 .....	139
<b>第8讲 结构体、共用体及位操作</b> .....	143
<b>第1课 结构体</b> .....	143
一、结构体的定义 .....	143
二、结构体类型变量的说明 .....	145
三、结构体变量成员的表示方法 .....	147
四、结构体变量的赋值 .....	147
五、结构体变量的初始化 .....	148

六、结构体数组 .....	149
七、结构体指针变量 .....	154
第2课 共用体 .....	172
一、共用体的定义 .....	172
二、共用体变量的说明 .....	173
三、共用体变量的赋值和使用 .....	174
第3课 枚举类型与位运算 .....	176
一、枚举类型 .....	176
二、类型定义符 typedef .....	178
三、位运算 .....	180
习题八 .....	186
第9讲 文件 .....	189
第1课 文件概论 .....	189
一、C语言文件的概念 .....	189
二、文件的指针 .....	190
三、文件的打开与关闭 .....	190
第2课 文件的读写 .....	192
一、读字符函数 fgetc .....	192
二、写字符函数 fputc .....	194
三、读字符串函数 fgets .....	195
四、写字符串函数 fputs .....	196
五、数据块读写函数 fread 和 fwrite .....	197
六、格式化读写函数 fscanf 和 fprintf .....	198
七、rewind 函数和 fseek 函数 .....	200
八、告诉位置指针当前位置的函数 ftell .....	202
九、文件检测函数 .....	202
习题九 .....	205
第10讲 由C迈向C++ .....	206
第1课 C与C++ .....	206
一、C++概述 .....	206
二、由C到C++的转变 .....	208
三、C++的新特性 .....	210
第2课 从面向过程到面向对象 .....	223
习题十 .....	226
第11讲 类与对象 .....	228
第1课 类的定义 .....	228
一、从结构体到类 .....	228
二、类的成员 .....	236
第2课 对象 .....	238

一、对象的声明 .....	238
二、构造函数和析构函数 .....	238
三、静态类成员 .....	245
四、对象行为和性能控制 .....	249
五、对象组织 .....	253
习题十一 .....	255
<b>第 12 讲 继承与多态性 .....</b>	<b>258</b>
<b>第 1 课 派生类 .....</b>	<b>258</b>
一、派生类的声明 .....	258
二、多基派生 .....	262
三、派生类的构造和析构函数 .....	262
四、虚基类 .....	265
<b>第 2 课 虚函数 .....</b>	<b>266</b>
一、多态性与虚函数 .....	266
二、纯虚函数与抽象类 .....	272
三、多态性和虚函数的归纳 .....	275
<b>第 3 课 运算符重载与转换 .....</b>	<b>275</b>
一、运算符重载 .....	275
二、转换 .....	280
习题十二 .....	283
<b>读者意见反馈表 .....</b>	<b>287</b>



# 第 1 讲 C 语言概论

## 本讲概要:

- 📖 C 语言的由来
- 📖 C 语言的特点
- 📖 C 语言的基本结构
- 📖 C 语言编译的过程
- 📖 C 语言上机步骤

## 第 1 课 C 语言概论

### 一、C 语言的由来

C 语言（通常简称 C）是美国贝尔实验室的 Dennis.M.Ritchie 于 1972 年设计实现的。C 语言直接来源于 B 语言，但它的根源可以追溯到 ALGOL60。ALGOL60 结构严谨，其设计者非常注重语法和分程序结构。因此对于后来许多重要的程序设计语言，如 PASCAL,PL/1,SIMULA67 产生过重要的影响。但由于它是面向过程的语言，与计算机硬件相距甚远，不适合编写系统软件。1963 年英国剑桥大学在 ALGOL60 的基础上推出更接近硬件的 CPL 语言，但 CPL 太复杂，难于实现。1967 年，剑桥大学的 Martin Rinchards 以 BCPL 为基础，设计了更简单也更接近硬件的 B 语言（取 BCPL 第一个字母）。B 语言是一种解释性语言，功能上也不够强大，为了很好地适应系统程序设计的要求，Ritchie 把 B 语言发展成称之为 C 的语言（取 BCPL 的第二个字母）。C 语言不仅保持了 BCPL 和 B 的特点（如精练，接近硬件等），又克服了它们的缺点（如过于简单，数据无类型等）。1973 年 K.Thompson 和 Dennis.M.Ritchie 使用 C 改写了 UNIX 代码，并在 PDP-11 计算机上加以实现，给出了 UNIX 版本 5，这一版本奠定了 UNIX 系统的基础，使 UNIX 逐渐成为最重要的操作系统之一。

由于 C 语言简洁，功能强大，运行速度快，一直是计算机领域非常重要的语言之一。

### 二、C 语言的特点

较之其他高级语言，C 语言有明显的特点。这些特点主要表现在：



(1) 功能强, 应用面广。C 语言绘图能力强, 能处理汉字, 与其他高级语言链接, 与汇编语言混合编程。既可以用来编制大型的系统软件, 如 UNIX 等, 也可以广泛应用于编制各种应用软件。

(2) 语句简洁, 表达能力强。C 语言的语句种类很少, 只有 10 种语句, 而表达式形式多样, 功能强, 是学习 C 语言的重点和难点之一。

(3) 运算符丰富。C 语言的运算符包含的范围非常广泛, 共有 34 种运算符。C 语言把括号、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。

(4) 数据结构丰富, 具有现代化语言的各种数据结构。C 语言的数据类型有: 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结构(如链表、队列、栈、树等)的运算。尤其是指针类型数据, 使用起来非常地灵活、多样。

(5) 具有结构化的控制语言(如 if...else 语句、while 语句、do...while 语句、switch 语句、for 语句)。用函数作为程序模块来实现程序的模块化, 是实现结构化的理想语言, 符合结构化编程风格要求。

(6) 程序设计自由度大, 对变量范围和存储空间的存取制约小。比如: 对数组下标越界不做检查, 由程序员自己保证程序的正确。对变量类型的使用非常地灵活, 例如整数类型和字符类型以及逻辑类型的数据可以通用。一般的高级语言语法检查比较严, 能检查出几乎所有的语法错误。而 C 语言允许程序编写者有比较大的自由空间, 因此放宽了语法的检查, 程序设计能很自由地进行, 这给程序设计带来很大灵活性, 但这既是优点, 又是缺点。因为太灵活, 初学者往往比较难以掌握, 不易编出正确的程序。这对于已经学过一门高级语言的人来说, 问题不大, 对于没有这个基础的初学者而言, 相对困难一些。可以说, 一个不熟练的人员, 编一个正确的 C 语言程序会比编一个其他高级语言的程序要难一些。

(7) C 语言允许直接访问物理地址, 能够进行位(bit)操作, 能够实现汇编语言的大部分功能, 可以直接对硬件进行操作。因此 C 语言既有高级语言的功能, 又具有低级语言的许多特点。

(8) 生成目标代码质量高, 程序执行效率高。一般只比汇编程序生成的目标代码效率低 10%~20%。

(9) 移植性好。由于 C 语言在语法上仅仅定义了最基本的控制语句, 而输入、输出和字符处理作为外部函数, 在编译或链接时才结合到所编写的程序上使用, 所以移植性好。

## 第 2 课 简单的 C 语言程序

本课通过介绍几个简单的 C 语言程序来了解 C 语言的特性。这几个程序由简到难, 表现了 C 语言源程序在组成结构上的特点。虽然有关内容还未介绍, 但可从这些例子中了解到组成一个 C 语言源程序的基本部分和书写格式。



## 一、简单例子

### [例 1.1]

```
01: main()  
02: {  
03:     printf("Hello,World\n");  
04: }
```

该程序的运行将在屏幕上输出下面的文字:

Hello,World

说明: `main` 是主函数的函数名;表示这是一个主函数。每一个 C 语言源程序都必须有,且只能有一个主函数(`main` 函数)。第 3 行调用函数语句, `printf` 函数的功能是把要输出的内容送到显示器去显示。`printf` 函数是一个由系统定义的标准函数,可在程序中直接调用。

### [例 1.2]

```
01: #include<math.h>  
02: #include<stdio.h>  
03: main()  
04: {  
05:     double x,s;  
06:     printf("input number:\n");  
07:     scanf("%lf",&x);  
08:     s=sin(x);  
09:     printf("sine of %lf is %lf\n",x,s);  
10: }
```

程序运行结果如下。

input number:

1

sine of 1.000000 is 0.841471

程序说明如下。

第 01 行: `include` 称为文件包含命令。我们将在第 6 讲预编译中介绍;

第 02 行: 扩展名为 `.h` 的文件也称为头文件或首部文件;

第 03 行: `main` 函数的定义。函数的定义格式将在第 5 讲中介绍;

第 05 行: 定义两个实数变量,以便被后面的程序使用;

第 06 行: 显示提示信息;

第 07 行: 从键盘获得一个实数 `x`;

第 08 行: 求 `x` 的正弦,并把它赋给变量 `s`;

第 09 行: 显示程序运算结果;



第 04, 10 行: 使用大括号将 main 主函数的函数体括起来。

程序的功能是从键盘输入一个数  $x$ , 求  $x$  的正弦值, 然后输出结果。在 main( ) 之前的两行称为预处理命令。预处理命令还有其他几种, 这里的 include 称为文件包含命令, 其意义是把双引号 " " 或尖括号 < > 内指定的文件包含到本程序中来, 成为本程序的一部分。被包含的文件通常是由系统提供的, 其扩展名为 .h。因此也称为头文件或首部文件。C 语言的头文件中包括了各个标准库函数的函数原型。因此, 凡是在程序中调用一个库函数时, 都必须包含该函数原型所在的头文件。在本例中, 使用了三个库函数: 输入函数 scanf, 正弦函数 sin, 输出函数 printf。sin 函数是数学函数, 其头文件为 math.h 文件, 因此在程序的主函数前用 include 命令包含了 math.h。scanf 和 printf 是标准输入输出函数, 其头文件为 stdio.h, 在主函数前也用 include 命令包含了 stdio.h 文件。

需要说明的是, C 语言规定对 scanf 和 printf 这两个函数可以省去对其头文件的包含命令。所以在本例中也可以删去第二行的包含命令 #include <stdio.h>。

同样, 在例 1.1 中使用了 printf 函数, 也省略了包含命令。在例题中的主函数体中又分为两部分, 一部分为说明部分, 另一部分为执行部分。说明是指变量的类型说明。例题中未使用任何变量, 因此无说明部分。C 语言规定, 源程序中所有用到的变量都必须先说明, 后使用, 否则将会出错。这一点是编译型高级程序设计语言的一个特点, 与解释型的 BASIC 语言是不同的。说明部分是 C 语言源程序结构中很重要的组成部分。本例中使用了两个变量  $x$ ,  $s$ , 用来表示输入的自变量和 sin 函数值。由于 sin 函数要求这两个变量必须是双精度浮点型, 故用类型说明符 double 来说明这两个变量。说明部分后的四行为执行部分或称为执行语句部分, 用以完成程序的功能。

从上面我们可以总结出 C 语言源程序的结构特点:

- (1) 一个 C 语言源程序可以由一个或多个源文件组成。
- (2) 每个源文件可由一个或多个函数组成。
- (3) 一个源程序不论由多少个文件组成, 都有且只能有一个 main 函数, 即主函数。
- (4) 源程序中可以有预处理命令(include 命令仅为其中的一种), 预处理命令通常应放在源文件或源程序的最前面。
- (5) 每一个说明, 每一个语句都必须以分号结尾。但预处理命令, 函数头和花括号 ({} ) 之后不能加分号。
- (6) 标识符、关键字之间必须至少加一个空格以示间隔。若已有明显的间隔符, 也可不再加空格来间隔。

## 二、C 语言程序的格式

C 语言程序的一般格式如下:

```
#include <头文件名> /*包含程序中引用的库函数*/
.....
#define 宏名 宏体 /*宏定义部分*/
.....
#include "外部文件名" /*外部函数所在文件*/
```



```

.....
全局变量说明
.....
main() /*主函数部分*/
{
    局部变量定义部分
    执行语句部分
}
存储类型 数据类型 函数名 1(参数表)
{
    局部变量定义部分
    执行语句部分
}
.....

```

在上面的格式中，除了 `main` 函数是必不可少的，其他都是可选的。下面就是一个最简单的 C 程序：

```

main ()
{
}

```

这个程序的作用是什么都不做，但它的的确确是一个合法的 C 程序，可以编译，可以运行（就是没有任何运行结果）。

上述的函数定义可以有很多个，它们可以放在一个文件中，也可以放在不同的文件里面。与主函数放在一个文件中的函数称为内部函数，不与主函数同在一个文件中的函数称为外部函数。在上面的一般格式中涉及到了我们尚未提到的许多概念，大家可以不用担心，在以后的讲解中，我们会逐一介绍。现在只需了解主函数部分就行了。

### 三、建立与运行 C 程序的过程

编写好一个 C 程序完成运行得到正确的结果，一般要经过以下几个步骤，编辑源程序、编译、链接、执行，如图 1.1 所示。

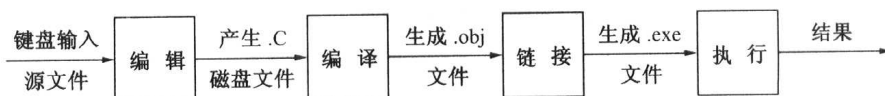


图 1.1 C 语言可执行文件的创建过程

(1) 编辑源程序。编辑源程序即是指使用文本编辑软件将源程序输入计算机，并保存在磁盘文件中。一般习惯使用 Microsoft Visual C++ 6.0 的集成开发环境 (IDE) 来完成编辑的功能。当然也可以是其他的集成开发环境，如 Borland C++ Builder 5.0 等，当然 Windows 自带的记事本也是可以的，只不过要注意文件的扩展名一定要使用 `.c` (如果是 C++



源代码的话，一定要使用.cpp）。

(2) 编译。编译的过程就是将源程序翻译成二进制的目标代码，生成扩展名为.obj 的目标文件。在这个过程中，编译系统会自动地对源程序进行语法检查，如果发现行错，会在屏幕上显示出错信息，此时需要对源程序进行相应的修改，纠正错误后再重新进行编译，直到顺利通过编译为止。

(3) 链接。链接的过程是将二进制的目标代码和系统的标准模块（库函数）等进行链接处理，然后生成可执行文件。这种可执行文件的后缀名为.exe。

(4) 执行。执行的过程就是通过运行 exe 文件以得到期望输出结果的过程。

## 四、用 MS Visual C++ 6.0 运行 C/C++ 程序的上机步骤

Visual C++ 6.0（通常简称 VC 6.0）是由 Microsoft 公司开发的一套具有强大功能的编程工具，具有很高的市场占有率。在本套教程中将使用这套工具来完成对 C/C++ 程序的编辑、编译、链接和执行的全过程。因为本书面向的读者是那些渴望了解 C/C++ 语言（C++ 语言通常简称 C++）的初学者，因此对于这套工具只需了解一些简单的操作即可。如果你的手头上使用的编程工具不是 Visual C++ 6.0，请参考相应工具的说明文档。

### （一）Visual C++ 6.0 的启动和关闭

VC 6.0 在安装完成以后。通过单击 Windows 上“开始”→“程序”→“Microsoft Visual Studio 6.0”→“Microsoft Visual C++ 6.0”即可启动 VC 6.0。启动后的界面如图 1.2 所示。如果要关闭 VC 6.0，直接单击窗口最右上角“关闭”按钮即可。

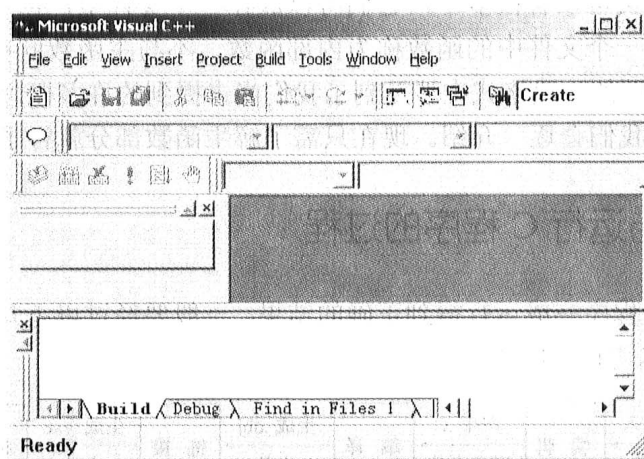


图 1.2 Visual C++ 6.0 的集成开发环境

### （二）编辑源程序

分两种情况。

情况一，建立编辑新的源程序。按如下方法操作：

(1) 选择菜单 File→New 命令，打开新建对话框，如图 1.3 所示。

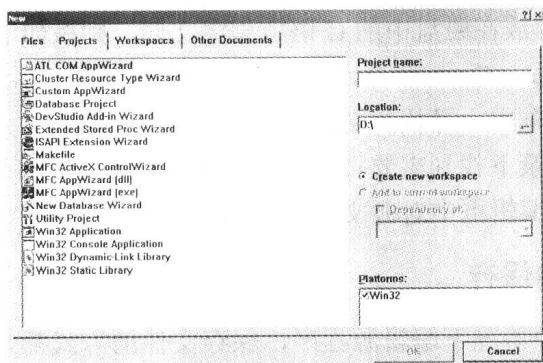


图 1.3 新建对话框

(2) 在对话框上边的选项卡中选择“Files”。

(3) 选中“C++ Source File”一项。

(4) 在右边的“File”文本框中输入该源程序的文件名，注意如果是 C 语言文件的话，需要手工输入.c 的后缀名，否则默认的后缀名为.cpp。同时在“locate”文本框中输入你希望保存文件的目录名，如图 1.4 所示。

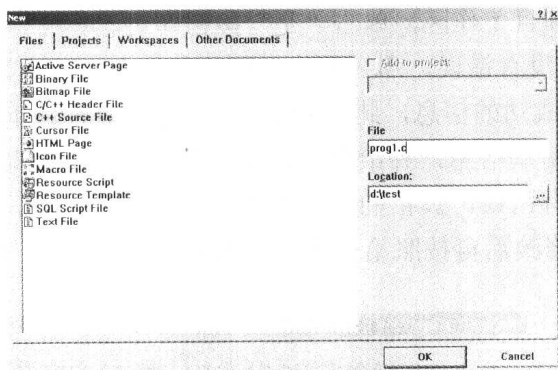


图 1.4 输入文件名

(5) 单击“OK”按钮，此时在编辑框（图 1.5）中逐行输入源程序即可，每输入完一个程序行，要按一下回车(Enter)键。在输入过程中，发现有错误，随时可进行修改。

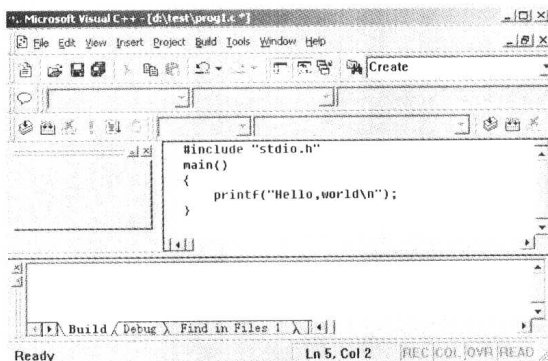


图 1.5 输入源程序



情况二，编辑已存放在磁盘中的 C 语言程序文件，则按如下方法操作。

- (1) 选择菜单 File→Open 命令，激活文件对话框。
- (2) 在文件对话框中，寻找源文件所在的目录。
- (3) 在相应的目录下找到需要打开的文件，然后单击“打开”按钮。
- (4) 此时可以在编辑框中对源文件进行修改。

### (三) 源文件的保存

不管是在上述哪种情况下编辑源程序，保存文件的方式都是一样的：单击工具栏上一个磁盘图样的图标。当鼠标在该图标上做短暂停留时，会有一个“Save(Ctrl+S)”的简短说明。或者也可以选择菜单 File→Save 命令，来达到同样的目的。

### (四) 源文件的编译和链接

源文件的编译也分为两种情况：单文件程序和多文件程序。

情况一，如果 C/C++ 程序只有一个源文件构成，则可以采用以下的方式进行编译。

(1) 当源文件编辑完成以后（我们不妨假设该文件名为 prog1.c），选择菜单 Build→Compile prog1.c 文件，如果是第一次编译该文件，VC 会弹出一个如图 1.6 所示的对话框，问是否需要建立一个项目工作区，单击“是”按钮即可。

(2) 如果源程序没有错误，那么工作界面的下方会出现诸如“prog1.obj - 0 error(s), 0 warning(s)”表示编译成功的信息，此时 prog1.c 文件成功的编译成为 prog1.obj 文件了。否则就会显示出相应的错误信息，并给出错误的可能位置（之所以说是可能位置，是指 VC 编译器告诉你哪一行有错误，真正的错误并不一定就是那一行，这一点需要大家在实际调试中去慢慢体会），修改后再按照第一步的步骤继续进行。

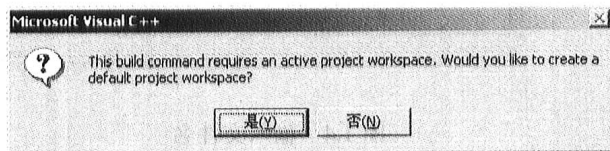


图 1.6 是否建立一个项目工作区

情况二，如果编写的程序由多个文件构成，比如两个文件，如例 1.3 所示。

[例 1.3]

```
01: /*This is prog1.c*/
02: int max(int a,int b);
03: main()
04: {
05:     int x,y,z;
06:     x=3;y=4;
07:     z=max(x,y);
08:     printf("maxmum=%d",z);
```





```
09: }
01: /*This is prog2.c*/
02: int max(int a,int b)
03: {
04:     if(a>b)
05:         return a;
06:     else
07:         return b;
08: }
```

那么就需要按照下面的步骤进行。

- (1) 在编辑文件前首先创建项目文件，选择菜单 **File**→**New** 命令，弹出新建对话框。
- (2) 在对话框的选项卡上选择“**Projects**”。
- (3) 然后选中“**Win32 Console Application**”一项。
- (4) 在“**locate**”文本框中输入或选择希望保存项目文件的目录名，同时在右边的“**Projectname**”文本框中输入该项目名（假设为 **test**），单击“**OK**”按钮。
- (5) 在这一步里选择“**An empty project**”，单击“**Finish**”按钮。
- (6) 最后给出即将创建的项目文件的信息，单击“**OK**”按钮完成创建该项目文件，后面需要将两个文件加入进去。
- (7) 选择菜单 **Project**→**Add to project**→**New** 命令，打开新建对话框。
- (8) 根据编辑源文件情况一的方式，加入并编辑源文件 **prog1.c**。
- (9) 重复（7）、（8）两步加入并编辑源文件 **prog2.c**，如果还有其他文件的话，继续重复（7）、（8）两步。
- (10) 加入所有的文件以后，编译的情况和情况一就完全相同了。具体可以参考情况一的步骤进行编译。

链接的过程非常简单，如果是情况一，在编译完成以后，可以单击菜单选择 **Build**→**Build prog1.exe** 命令来完成链接过程，生成 **exe** 可执行文件。如果是情况二，则可以选择直接选取菜单 **Build**→**Build test.exe** (**test** 为项目名) 命令一次性完成编译和链接的过程（其实，单文件程序也可以一次性完成编译和链接的过程，但是要在已经建立起工作区的前提下）。

### （五）程序的运行

经过编译成功后的程序，便可运行了。操作步骤是：选择菜单 **Build**→**Execute 文件名.exe** 即可。

需要特别指出的是，虽然系统允许执行未经编译的程序，即可以直接在修改文件后，并经过编译和链接，就直接选择菜单 **Build**→**Execute 文件名.exe**，来运行程序。但实际上并不是可以不要编译便可直接运行源程序，而是系统首先检查有没有最新的可运行的 **exe** 文件，若有则直接运行；否则就检查是否存在最新的 **obj** 文件，若有，则链接成 **exe** 后运行；若没有，则自动先进行编译链接后再运行。

建议读者按前述步骤循序渐进地进行，特别对于初学者，尤其应该这样。在编程过程中，错误是难免的，如果输入完源程序，来不及存盘就急急忙忙运行程序，往往会由于某