



工业和信息化普通高等教育“十二五”规划教材立项项目

21世纪高等教育计算机规划教材



C/C++ 程序设计

C/C++ Programming

■ 梁海英 主编

■ 董延华 孙静 于萍 刘哲 副主编

— 典型实例精挑细选

— 实用角度出发、多种方式讲解

— 各章习题针对性强、题型丰富



人民邮电出版社
POSTS & TELECOM PRESS



工业和信息化普通高等教育“十二五”规划教材立项项目

21世纪高等教育计算机规划教材

COMPUTER

C/C++ 程序设计

C/C++ Programming

■ 梁海英 主编

■ 董延华 孙静 于萍 刘哲 副主编



人民邮电出版社

北 京

图书在版编目 (CIP) 数据

C/C++程序设计 / 梁海英主编. — 北京 : 人民邮电出版社, 2013.9
21世纪高等教育计算机规划教材
ISBN 978-7-115-32494-8

I. ①C… II. ①梁… III. ①C语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆CIP数据核字(2013)第182288号

内 容 提 要

本书以面向应用型人才培养为目标,以非传统的组织结构为创新点,以全程伴随上机实践为特色,简洁、通俗、直观、易懂地讲述 C/C++程序设计。第1章~第3章讲述了 C 语言的背景知识、上机环境以及基础知识,包括数据类型、常量、变量和表达式,以及顺序、选择、循环三大结构及其编程。第4章~第6章介绍 C 语言的重点部分,包括数组、函数和指针。第7章~第10章介绍 C++的提高部分,包括类与对象、类的继承与多态性、对话框和常用控件等知识。

全书直接采用 C++的 cin 和 cout 进行输入/输出,摒弃了 C 语言中的 printf 函数和 scanf 函数调用。从实用的角度出发,内容选取先进精准、组织循序渐进、讲解文字精练;各部分辅助图表、结合实例、深入浅出、结构清晰;典型实例精挑细选、算法分析流程图化、程序结构错落有致、程序结果真实有效;各章习题针对性强、题型丰富;详细介绍了开发环境 Visual C++ 6.0 的使用方法,全部例题均在此环境中成功运行。

本书可作为高等学校非计算机专业本科生的计算机通识教材,也可作为计算机相关专业的程序设计入门教材、计算机技术的培训教材,或者作为全国计算机等级考试的参考用书和编程爱好者自学 C++的教材。

-
- ◆ 主 编 梁海英
 - 副 主 编 董延华 孙 静 于 萍 刘 哲
 - 责任编辑 许金霞
 - 责任印制 彭志环 杨林杰
 - ◆ 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 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京天宇星印刷厂印刷
 - ◆ 开本: 787×1092 1/16
印张: 14.5 2013年9月第1版
字数: 378千字 2013年9月北京第1次印刷
-

定价: 34.00 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

Visual C++是优秀的计算机程序设计语言。它的功能相当强大,既适合结构化程序设计,又适合面向对象程序设计。本教材以使学生理解和掌握 C++程序设计的基本概念和方法为主导思想,力求通过简单算法的讲解学习 C++编程。希望通过对本书的学习,读者能够正确理解 C++语言中的面向对象编程方法,基本掌握 Visual C++语言中的语法、词法、程序基本结构,并能够利用 Visual C++编写简单程序。

本书结构共分三大部分:第一部分主要讲解 C/C++语言的基本知识,包括 C 语言的基本词法和语法规则、基本数据类型和表达式、程序控制结构;第二部分主要讲解 C 语言的重点部分,包括数组、函数和指针;第三部分主要讲解面向对象程序设计思想和 Windows 可视化编程,主要内容包括类与对象的概念和定义格式、对象的赋值和引用、友元、类的继承和派生、多态和虚函数、对话框和常用控件等。书中的所有例题都在 Visual C++ 6.0 版本的编译系统下运行通过,每章后都附有习题以便读者及时对所学知识进行巩固。

本书面向应用型人才培养,内容安排由简到难,逐步深入,免得学习者失去学习信心。另外,一开始就使学习者可以上机实践,之后全程理论和实践互补学习,有利于掌握程序设计的技巧,提高编程能力。扎实地掌握好 C 语言的编程后,可自然过渡到 C++编程。C++全国高校计算机水平考试,迫切需要一本适合的教材,于是有了这本 C/C++结合的书。本教材所有例题的输入/输出直接采用 C++的 cin 和 cout,摒弃了 C 语言中的 printf 和 scanf 函数调用。仍愿意使用 printf 和 scanf 函数的学习者,也可以选用本书,将例题中的输入/输出换回 C 语言的函数,也是一个很好的训练。

在本书的编写过程中,得到了同事们热心帮助和支持。参加本书内容编写、程序调试、课件制作、习题收集、答案制作、内容审校等工作的有梁海英、董延华、孙静、于萍、刘哲、朱宏、张伟、王发斌、陈晓明等,在此向他们表示衷心的感谢!

本书在编写过程中参考了有关的 Visual C++教材、文献和网站内容,并引用了一些材料,在此对这些作者表示衷心的感谢。

由于书稿涉及的内容多、范围广,尽管作者已经尽了最大的努力,但由于时间仓促,书中难免存在不妥之处,请读者原谅,并提出宝贵意见。

编 者
2013 年 5 月

目 录

第 1 章 C++概论	1	2.6 构造数据类型	30
1.1 C++语言特点	1	2.6.1 结构体	30
1.1.1 程序和程序设计语言	1	2.6.2 共用体	35
1.1.2 C++语言的特点	2	2.6.3 枚举	37
1.2 C++程序的实现	3	习题二	39
1.3 C++程序结构的特点	5	第 3 章 控制结构	41
1.3.1 一个简单的 C++语言程序	5	3.1 顺序结构	41
1.3.2 C++程序结构及书写格式	6	3.2 选择结构	42
1.4 Visual C++ 6.0 主窗口	7	3.2.1 if 语句	42
1.5 C++上机过程	9	3.2.2 switch 语句	47
习题一	11	3.3 循环结构	48
第 2 章 数据类型、运算符和 表达式	13	3.3.1 while 语句	49
2.1 基本数据类型	13	3.3.2 do...while 语句	50
2.1.1 整型 (int)	13	3.3.3 for 语句	52
2.1.2 字符型 (char)	14	3.3.4 break 和 continue 语句	54
2.1.3 浮点型 (float)	14	3.4 程序设计举例	56
2.1.4 布尔型 (bool)	14	习题三	57
2.1.5 空型 (void)	14	第 4 章 数组	60
2.2 常量和变量	14	4.1 一维数组	60
2.2.1 常量	14	4.1.1 一维数组的定义	60
2.2.2 变量	16	4.1.2 一维数组元素的引用	61
2.3 输入输出	21	4.1.3 一维数组的初始化	61
2.4 运算符和表达式	24	4.1.4 一维数组的输入输出	61
2.4.1 算术运算符和算术表达式	24	4.2 二维数组	62
2.4.2 关系运算符和关系表达式	26	4.2.1 二维数组的定义	62
2.4.3 逻辑运算符和逻辑表达式	27	4.2.2 二维数组元素的引用	62
2.4.4 赋值运算符和赋值表达式	27	4.2.3 二维数组的初始化	63
2.4.5 逗号运算符和逗号表达式	28	4.2.4 二维数组的输入输出	63
2.4.6 sizeof 运算符	28	4.3 字符数组和字符串	64
2.5 数据类型的转换	29	4.3.1 字符数组的定义	64
2.5.1 自动数据类型转换	29	4.3.2 字符数组的初始化	64
2.5.2 强制数据类型转换	29	4.3.3 字符数组的输入输出	66
		4.3.4 常用的字符串处理函数	67

4.4 应用举例.....	68	6.5 引用.....	112
习题四.....	73	6.6 应用举例.....	114
第5章 函数	76	习题六.....	116
5.1 标准函数.....	76	第7章 类与对象	118
5.2 函数的定义.....	76	7.1 面向对象程序设计的概念.....	118
5.3 函数的调用.....	77	7.2 类.....	119
5.4 函数的原型.....	78	7.2.1 类的声明.....	120
5.5 函数参数.....	80	7.2.2 类成员的定义.....	121
5.5.1 参数的传递方式.....	80	7.3 对象.....	122
5.5.2 默认参数.....	81	7.3.1 对象的定义.....	122
5.6 递归函数.....	81	7.3.2 对象成员的引用.....	123
5.6.1 递归函数.....	81	7.4 构造函数和析构函数.....	124
5.6.2 递归调用的执行过程.....	82	7.4.1 构造函数.....	125
5.7 变量的作用域和存储类.....	83	7.4.2 析构函数.....	126
5.7.1 变量的作用域.....	83	7.5 内联函数.....	128
5.7.2 变量的存储类.....	86	7.6 静态成员.....	128
5.8 编译预处理.....	90	7.6.1 静态成员数据.....	128
5.8.1 宏定义.....	90	7.6.2 静态成员函数.....	130
5.8.2 文件包含.....	90	7.7 对象数组和对象指针.....	131
5.8.3 条件编译.....	91	7.7.1 对象数组.....	131
5.9 应用举例.....	91	7.7.2 对象指针.....	133
习题五.....	93	7.7.3 this 指针.....	133
第6章 指针	97	7.8 友元.....	134
6.1 指针的概念.....	97	7.8.1 友元函数.....	135
6.1.1 地址与指针.....	97	7.8.2 友元成员函数.....	136
6.1.2 指针定义.....	98	7.8.3 友元类.....	137
6.2 对指针变量的操作.....	98	习题七.....	138
6.2.1 指针的运算.....	98	第8章 继承与多态性	143
6.2.2 new 和 delete.....	101	8.1 继承.....	143
6.3 指针与数组.....	102	8.1.1 单继承.....	143
6.3.1 用指针访问一维数组.....	102	8.1.2 多继承.....	145
6.3.2 用指针访问二维数组.....	104	8.2 派生类的构造函数和析构函数.....	149
6.3.3 用指针访问字符串.....	105	8.3 重载.....	152
6.3.4 指针数组.....	106	8.3.1 函数重载.....	152
6.4 指针与函数.....	108	8.3.2 运算符重载.....	154
6.4.1 指针作为函数的参数.....	108	8.4 多态性.....	158
6.4.2 数组名作为参数.....	111	8.4.1 虚函数.....	159
6.4.3 指针函数.....	111	8.4.2 纯虚函数和抽象类.....	162

习题八	165	10.3.3 应用举例	199
第 9 章 对话框	173	10.4 列表框	200
9.1 MFC 应用程序	173	10.4.1 列表框的创建	200
9.1.1 MFC 编程	173	10.4.2 列表框的通知消息	201
9.1.2 MFC 应用程序框架类型	175	10.4.3 列表框的操作	201
9.2 创建和使用对话框	178	10.4.4 应用举例	202
9.2.1 创建对话框	178	10.5 组合框	204
9.2.2 控件的添加和布局	180	10.5.1 组合框的类型	204
9.2.3 创建对话框类	182	10.5.2 组合框的数据输入	205
9.2.4 调用对话框	183	10.5.3 组合框的操作	205
9.3 通用对话框和消息对话框	185	10.5.4 组合框的消息	206
9.3.1 通用对话框	185	10.5.5 应用举例	206
9.3.2 消息对话框	188	10.6 滚动条	208
习题九	190	10.6.1 滚动条的结构	209
第 10 章 常用控件	191	10.6.2 滚动条的消息和基本操作	209
10.1 控件的使用	191	10.6.3 应用举例	210
10.1.1 控件的创建	191	10.7 旋转按钮	211
10.1.2 控件的消息和消息映射	192	10.7.1 旋转按钮的创建	212
10.1.3 控件的数据交换 (DDX) 和数据 校验 (DDV)	193	10.7.2 旋转按钮的操作	212
10.2 静态控件和编辑框	194	10.7.3 应用举例	213
10.2.1 静态控件	194	10.8 进展条	215
10.2.2 编辑框	195	10.8.1 进展条的操作	215
10.2.3 应用举例	196	10.8.2 应用举例	215
10.3 按钮控件	198	10.9 列表控制	217
10.3.1 按钮的创建和消息	198	10.9.1 列表控制的建立	218
10.3.2 按钮的操作	198	10.9.2 列表控制的操作	218
		10.9.3 列表控制的数据结构	219
		10.9.4 应用举例	220
		习题十	224

第 1 章

C++ 概论

C++ 是一种应用广泛的面向对象的程序设计语言。C++ 是由 C 发展而来的，它保留了 C 语言的所有优点，既可以用于面向过程的结构化程序设计，又可以实现面向对象的程序设计。本章主要讲述 C++ 的特点和语法、C++ 程序的实现过程和 Visual C++ 6.0 集成开发环境。

1.1 C++ 语言特点

在具体讲述 C++ 语言的特点之前，先来了解一下什么是程序和程序设计语言。

1.1.1 程序和程序设计语言

自 1946 年世界上第一台电子计算机问世以来，计算机科学及其应用得到了迅猛的发展，计算机已被广泛地应用于人类生产、生活的各个领域，成为人们日常工作、生活、娱乐的一种现代化工具，计算机已将人类带入了一个崭新的信息技术时代。对于当代大学生而言，掌握一门计算机程序设计语言是十分必要的。那么，什么是计算机呢？大多数人对计算机可能并不陌生，但是不熟悉计算机的人可能会感到它很神秘。其实，计算机只不过是一种具有内部存储能力、在程序的控制下自动工作的电子设备。一台计算机由硬件系统和软件系统两大部分组成，硬件是物质基础，而软件可以说是计算机的灵魂。没有安装软件的计算机只能是一台“裸机”，什么也干不了。有了软件才能成为一台真正的“电脑”。而所有软件都是用计算机语言编写的。人们将需要计算机做的工作写成一定形式的计算机能够识别的指令，并把它存储在计算机内部的存储器中，当人们给出指令后，它就会按照指令操作顺序自动工作。人们把这种可以连续执行的一条条指令的集合称为程序，而程序是用程序设计语言来编写的。程序设计语言按照语言级别可以分为低级语言和高级语言。低级语言有机器语言和汇编语言。高级语言则主要有过程式语言（如 C、Basic 以及 Pascal 等）、面向对象语言（如 C++、Java 等）、应用式语言（如 Lisp）以及基于规则的语言（如 Prolog）等。

1. 机器语言

机器语言是计算机硬件可以直接识别的语言，它完全用 0 和 1 组成的代码表示，也是最低层的程序设计语言。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。机器语言是面向机器的。不同的计算机硬件（主要是 CPU），其机器语言是不同的，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。有了机器语言，人们就可以用机器语言编写程序，然后输入计算机，计算机就可以通过运行程序来体现人们的意图，计算或处理相应的问题。

2. 汇编语言

由于机器语言是面向具体机器的，所以其程序缺乏通用性，编写程序的过程繁琐复杂，易出错，错了又不易查找和修改，编出的程序可读性差。繁杂的机器代码程序很难记忆，未受过专门训练的人又不易掌握，这严重阻碍了计算机的应用和发展。于是，人们又在机器语言的基础上研制了汇编语言。汇编语言采用符号（称为指令助记符）表示指令，比机器语言的指令代码易于记忆。

用汇编语言编写的程序（又称源程序）经汇编器加工处理后，就可转换成在计算机上可以直接执行的机器语言程序。汇编语言实质上是机器语言的符号化形式，仍属面向机器的一种低级语言。

3. 高级语言

由于汇编语言也依赖于计算机的硬件体系，且助记符量大，难以记忆，于是人们又发明了更加易用的所谓高级语言。这种语言的语法和结构更类似普通英文，且通用性好，不必对计算机的指令系统有深入的了解就可以编写程序。

采用高级语言编写的程序在不同型号的计算机上只需做某些微小的改动便可运行，对这些高级语言程序只要采用对应计算机上的编译程序重新编译，即把由高级语言编写的源程序转换成机器能够接受的机器语言程序，这些程序就可以在不同的机器上运行。这种具有翻译功能的程序称为编译程序。每一种高级语言都有与之对应的编译程序。

1.1.2 C++语言的特点

C++语言于 20 世纪 80 年代由贝尔实验室设计并实现。它是在 C 语言的基础上发展起来的，既支持传统的面向过程的程序设计，又支持面向对象的程序设计。

C 语言是在 B 语言的基础上由美国 Bell 试验室的 Dennis.M.Ritchie 于 1972 年设计实现的，改进了 B 语言的缺陷，其设计目标是保持 BCPL 和 B 语言的精练性及接近硬件的特点，并且增加这些语言的通用性。C 语言由于简洁，功能强大，运行速度快，一直是非常重要的程序设计语言之一。相较于其他的高级语言，C 语言的特点主要表现在：①功能强，应用广泛；②语句简洁，表达能力强；③运算符丰富；④数据结构丰富，具有现代化语言的各种数据结构；⑤具有结构化的控制语句；⑥程序设计自由度高；⑦C 语言允许直接访问物理地址，能够进行位操作，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，既有高级语言的功能，又有低级语言的功能；⑧生成的目标代码质量高，程序执行效率高；⑨可移植性好。

除了上述优点之外，C 语言也有它的局限性：①C 语言的类型检查机制相对较弱，有些错误不能在编译阶段检查出来；②C 语言本身几乎没有支持代码重用的语言结构；③当程序的规模达到一定的程度时，程序员就很难控制程序的复杂性。

与 C 语言不同，C++是一种广泛使用的面向对象的程序设计语言。C++语言包括了 C 语言的所有特征、属性和优点（如高效、灵活性），同时改进了 C 语言的一些不足，并且支持面向对象的程序设计。C++语言的特点：

①保持与 C 语言兼容；②可读性更好，代码结构更合理；③生成代码的质量高；④可重用性、可扩充性、可维护性和可靠性有所提高；⑤支持面向对象的机制。

C++语言中与面向对象有关的特征如下。

（1）类和数据封装

C++支持数据封装，将数据和对该数据的操作函数封装在一起作为一种数据类型，称为类。同时提供一种对数据访问严格控制的机制，封装体通过操作接口与外界交换信息。

(2) 结构作为一种特殊的类

在C语言中可以定义结构体,但是这种结构只包含数据,而不包含函数。C++中的类是数据和函数的封装体。在C++中,结构可以作为一种特殊的类。

(3) 构造函数和析构函数

构造函数是类内和类同名的成员函数,创建对象时对类的数据成员进行初始化。析构函数的功能是释放对象。

(4) 私有、保护和公有成员

C++类中可以定义三种不同访问控制权限的数据成员。它们分别是私有(`private`)、保护(`protected`)和公有(`public`)成员。私有成员只有类本身定义的函数才能访问,而类外的其他函数不可以访问。保护成员是只有派生类可以访问,而在类外不可以访问的成员。公有成员是在类外也可以访问的成员,是该类与外界的接口。

(5) 对象和消息

对象是类的实例。对象之间通过消息来实现合作,共同完成某一任务。每个对象根据收到消息的性质来决定需要采取的行动,以响应这个消息。

(6) 友元类和友元函数

类中的私有成员是不允许类外的任何函数访问的。但是友元打破了类的这一限制,破坏了类的封装性,它可以访问类的私有成员。友元可以是类外定义的整体类,称为友元类;也可以是类外的函数,称为友元函数。

(7) 运算符和函数名重载

函数名重载和运算符重载都属于多态。多态是指相同的语言结构可以代表不同类型的实体,或者对不同类型实体进行操作。C++允许相同的运算符或标识符代表多个不同实现的函数,这就称为标识符或运算符重载。用户可以根据需要定义标识符重载或运算符重载。

(8) 派生类,继承性

一个类可以根据需要生成派生类。派生类继承了基类的所有方法,同时还可以定义新的不包含在父类中的方法。派生类包含从父类继承过来的数据成员和自己特有的数据成员。

(9) 虚拟函数,多态性,动态联编

C++可以定义虚函数,通过虚函数实现动态联编。动态联编是多态的一个重要特征。多态性形成由父类和它们的子类组成的一个树型结构。在这个树中的每一个子类可接收一个或多个具有相同名字的消息。当一个消息被这个树中的一个类的一个对象接收时,这个对象动态地决定给予子类对象的消息的某种用法。多态中的这一特性允许使用高级抽象。

1.2 C++程序的实现

一般把由高级语言编写的程序称为源程序。这种程序不能在机器上直接运行,只有把它转换为由二进制代码表示的目标程序后,才能在机器上运行。与其他高级语言一样,C++程序的实现要经过编辑、编译、组建和运行几个步骤,其过程如图1.1所示。

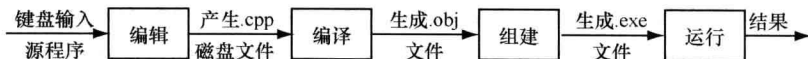


图 1.1 C++程序的实现过程

1. 编辑

编辑是将编写好的 C++源程序输入到计算机中,并生成磁盘文件的过程。C++程序的编辑可以利用计算机软件所提供的某种编辑器进行编辑,然后将 C++程序的源代码存放到磁盘文件中,磁盘文件的扩展名是.cpp。

在实际应用中,所选用的 C++编译器本身都提供编辑器的功能,使用所选用的 C++编译器中的编辑器来编写 C++源程序是十分方便的。例如,Microsoft Visual C++ 6.0 版本就提供编辑功能,将 C++程序输入后,指定文件名便可存入磁盘文件。然后,选用编译菜单项,便可编译执行。其他的 C++编译器也都有编辑功能,可用它来进行源程序编辑,不必再去选用其他编辑软件。编辑器所采用的编辑方法都大致相同,采用全屏幕编辑方法。插入、覆盖、删除等简单操作都与 Word 相同或相近,也有块操作功能。例如,删除块、复制块、移动块都可以通过编辑菜单中的菜单项进行。

2. 编译

C++是以编译方式实现的高级语言。C++程序的实现,必须使用某种 C++语言的编译器对程序进行编译。

编译器的功能是将程序的源代码转换成为机器代码的形式,称为目标代码。源程序进行编译时,首先要经过预处理过程。如果源程序中有预处理命令,则先执行这些预处理命令,执行后再进行后面的编译过程。如果程序中没有预处理命令,就直接进行后面的编译过程。

C++编译过程主要是进行词法分析和语法分析的过程,又称源程序分析。这个阶段基本与机器硬件无关,主要进行的是对程序的语法结构进行分析,发现不符合要求的语法错误,并及时报告给用户,显示在屏幕上。在这个过程中还要生成一个符号表。最终生成目标代码程序,完成编译阶段的任务。整个编译过程主要完成以下工作。

① 词法分析。主要是对由字符组成的单词进行词法分析,检查这些单词使用的是否正确,删除程序中的冗余成分。单词是程序使用的基本符号,是最小的程序单元。按照 C++所使用的词法规则逐一检查,并登记造册。发现错误,及时显示错误信息。

② 语法分析。语法又称文法,主要是指构造程序的格式。分析时按该语言中使用的文法规则分析检查每条语句是否有错误的逻辑结构,如发现有错误,便及时通报用户。

③ 符号表。符号表又称字典。它用来映射程序中的各种符号及它们的属性,例如,某个变量的类型、所占内存的大小和所分配的内存的相对位置等。该表是在进行词法分析和语法分析时生成的,它在生成中间代码和可执行的机器代码时使用。

④ 错误处理程序。在进行词法分析和语法分析过程中将所遇到的语法错误交给该程序处理,该程序根据所出现的错误的性质分为警告错和致命错显示给用户,并且尽可能指出出错的原因,供用户修改程序时参考。

⑤ 生成目标代码。将词法分析和语法分析的结果以及使用符号表中的信息,由中间代码进而生成机器可以执行的指令代码,又称为目标代码。将这些代码以.obj 为扩展名存在磁盘文件中,称为目标代码文件。这种文件中的代码机器可以识别,但是计算机并不能直接执行,还需要对它进行组建,才能生成可执行文件。

3. 组建

编译后的目标代码文件还不能由计算机直接执行。因为编译器对每个源文件分别进行编译,如果一个程序有多个源文件,编译后这些源文件的目标代码文件还分布在不同的地方,因此需要把它们组建到一起。即使该程序只有一个源文件,这个源文件生成的目标代码文件还需要系统提供的库文件中的一些代码,因此,也需要把它们组建起来。总之,基于上述原因,将用户程序生

成的多个目标代码文件和系统提供库文件中某些代码组建在一起，还是十分必要的。这种组建工作由编译系统中的组建程序（又称组建器）来完成。组建器将由编译器生成的目标代码文件和库中的某些文件组建处理，生成一个可执行文件，可执行文件的扩展名为.exe，因此，有人又称它为 EXE 文件。库文件的扩展名为.lib。

4. 运行

一个 C++ 的源程序经过编译和组建后生成了可执行文件。运行可执行文件的方法很多，一般在编译系统下有运行功能，通过选择菜单项便可实现。也可以在 MS-DOS 系统下，在 DOS 提示符后，直接输入可执行文件名，按回车键即可执行。有时需要给出可执行文件的全名，包含路径名，扩展名一般省略。如果需要参数，还应在命令行中命令字的后面输入所需要的参数。

程序被运行后，一般在屏幕上显示出运行结果。用户可以根据运行结果来判断程序是否还有算法错误。编好一个程序后，在生成可执行文件之前需要改正编译和组建时出现的一切致命错和警告错，这样才可能生成无错的可执行文件。在程序中存在警告错时，也会生成可执行文件，但是一般要求改正警告错后再去运行可执行文件。有的警告错会造成结果的错误。

1.3 C++程序结构的特点

本节通过几个例子让大家对 C++ 程序的结构有一个大体上的认识，以对 C++ 语言程序的构成有一个初步的了解。

1.3.1 一个简单的 C++ 语言程序

首先介绍一个简单的 C++ 程序，使读者对 C++ 程序有一个大概的了解。下面的例子虽然简单，但反映了一般 C++ 程序的特点以及基本的组成。

例 1.1 编写一个 C++ 程序，其功能是显示字符串 “This is my first C++ program.”。其 C++ 程序如下：

```
#include <iostream>    //包含头文件 iostream.h
using namespace std;   //使用命名空间 std
int main()             //主函数
{
    cout << "This is my first C++ program.\n"; //输出一行字符
    return 0;
}
```

这个程序的运行结果是在显示器屏幕的当前光标位置处显示句子：

This is my first C++ program.

程序的第 1 行：`#include <iostream>` 通常称为命令行。命令行必须用符号 “#” 开头。一对尖括号中的 `iostream` 是系统提供的文件名，全称是 `iostream.h`，文件中包含着有关输入输出函数的信息。调用不同的标准库函数，应包含不同的头文件。随着课程的深入，将在以后的章节中陆续介绍相关的头文件。

第 2 行语句 `using namespace std;` 的功能是在使用系统库时使用命名空间 `std`。

第 3 行中 `main` 是主函数名，其后的一对圆括号中间可以是空的，但是这一对圆括号不能省略。`main()` 是主函数的起始行。一个 C++ 程序可以包含任意多个不同名的函数，但是必须有且只有一个主函数，一个 C++ 程序总是从主函数开始执行的。

主函数后面由一对花括号{}括起来的部分是主函数体，其中的语句是实现程序的具体功能。函数体用左花括号“{”开始，右花括号“}”结束。其中可以有定义部分和执行部分，定义部分主要是对要用到的变量进行说明，执行部分主要是实现程序的具体功能。执行部分的语句称为可执行语句，必须放在说明部分之后，语句的数量不限，程序中的这些语句向计算机系统发出操作命令。C++的每一条定义语句和执行语句都要以分号“;”作为结束，分号是C++语句的一部分。

在程序中可以对程序进行注释，编译器在对程序进行编译时忽略注释的内容。注释有两种方法。第1种方法为行注释，以“//”开始到本行结束的任何内容均为注释。第2种为块注释或段注释，用符号“/*”和“*/”括起来的内容为注释，“/*”和“*/”必须成对出现，“/”和“*”之间不能有空格。注释可以用中文，也可以用西文。注释可以出现在程序中任何需要的地方，注释部分对程序的运行不起作用，使用注释的目的只是使程序员可以在源程序中插入一些说明解释性的内容。在注释中可以说明变量的含义、程序段的功能，主要是帮助人们阅读程序。

1.3.2 C++程序结构及书写格式

下面再举几个例子让大家熟悉C++程序的结构与书写格式。

例 1.2 已知矩形的两条边，求矩形的面积。

程序如下：

```
#include <iostream.h>
void main()
{
    double a,b,area;
    a=1.2;
    b=1.5;
    area=a*b;
    cout<<"a="<<a<<"\tb="<<b<<"\tarea="<<area<<endl;
}
```

程序的运行结果为：

```
a=1.2    b=1.5    area=1.8
```

以上程序中main()后一对花括号括起来的部分称为函数体，其中，程序的第4行为函数的说明部分，第5行到第8行是函数的执行部分，第8行为输出语句，功能是输出a、b和area的值。

例 1.3 从键盘输入一个直角三角形的两直角边a和b的长度，求其斜边长度。

程序如下：

```
# include <iostream>
# include <cmath>
using namespace std;
void triangle(double x, double y)
{
    double z;
    z=sqrt(x*x+y*y);
    cout<<"hypotenuse="<<z<<endl;
}
void main()
{
    double a,b;
    cout<<"input a and b:";
    cin>>a>>b;
    triangle(a,b);
}
```

程序运行结果为：

```
input a and b: 3 4
```

```
hypotenuse=5
```

这个程序包含了两个函数：一个是用于计算直角三角形斜边长度的函数 `triangle()`，一个是主函数 `main()`。主函数中首先要求从键盘输入两直角边 `a` 和 `b` 的值，然后调用计算斜边长度的函数 `triangle()`。

从上面几个例子可以看出每个程序都包含命令行，它们都以“#”开头，其作用是提供标准的库函数、用户自定义类库和函数。每个程序都有一个 `main()` 函数，它是程序的入口，每个程序都从这里开始执行。除了 `main()` 函数外，还可以定义其他的函数，整个 C++ 程序可以说是由若干函数组成。每个程序都包含对变量和函数的说明，同时还有输入和输出功能。

一个 C++ 程序的一般格式如下：

```
#include <..... >           //命令行
函数原型                     //程序中用到的函数的说明
全局数据定义                 //程序中用到的全局数据的定义
void main()
{
    ...                       //main() 函数体，由多条语句组成
}
函数定义                     //程序中用到的函数的具体实现
```

其中函数原型给出了函数的说明，包括函数返回值类型、函数名和参数；函数的定义部分给出了函数的具体实现。一个 C++ 程序并不一定严格按照上述要求来写，有些部分可以没有。

1.4 Visual C++ 6.0 主窗口

Visual C++ 6.0 集成开发环境是一个功能强大的程序开发环境，开发环境主窗口由标题栏、菜单栏、工具栏、项目工作区、主工作区、输出窗口和状态栏组成，如图 1.2 所示。

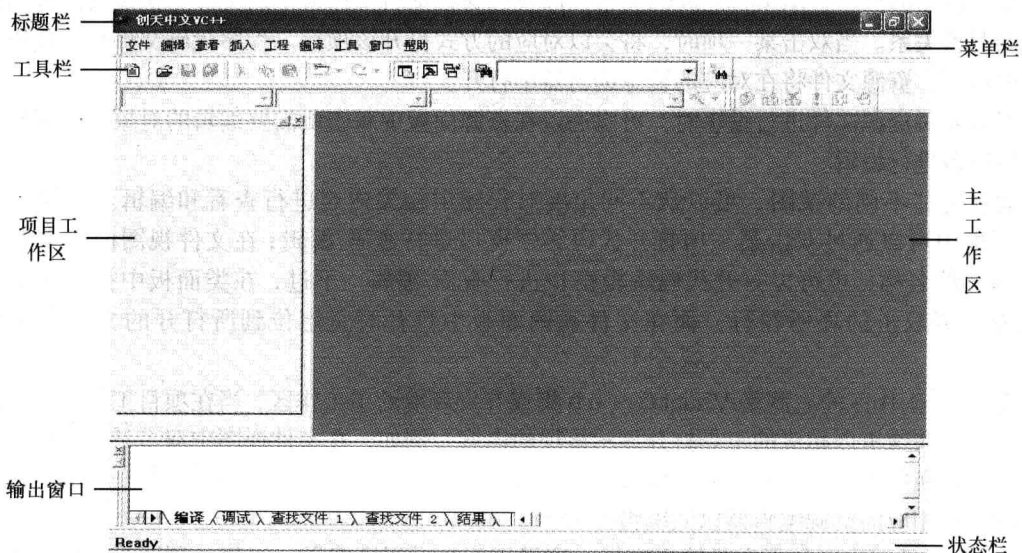


图 1.2 Visual C++ 6.0 集成开发环境

- ① 标题栏位于主窗口顶端。标题栏显示应用程序项目名称和当前打开的文件名称。

- ② 菜单栏位于标题栏下方，包含集成开发环境中所有功能命令。
- ③ 工具栏位于菜单栏的下方，提供了实现集成开发环境某些命令的快捷方式。命令以图标形式出现在工具栏中，要执行某个命令只需按下相应的图标按钮即可。
- ④ 工具栏下方左侧是当前工程的项目工作区，包含三个标题面板，如图 1.3 所示。

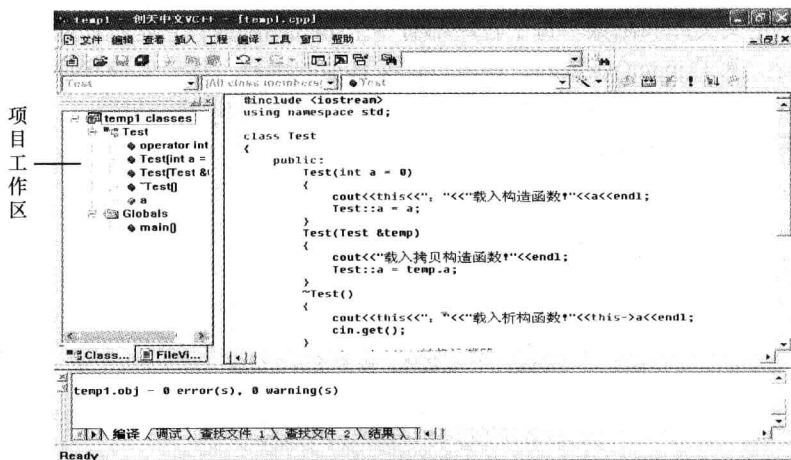


图 1.3 项目工作区

从左到右依次为类视图 (ClassView)、资源视图 (ResourceView) 和文件视图 (FileView)，可以通过单击下方的标签进行面板的切换。

类视图面板中显示程序中所定义的 C++ 类。通过此面板可以针对类、类成员进行快速定位与编辑；还可以添加新类，创建函数或声明方法等。

资源视图面板显示程序中所用到的资源文件，包括菜单、工具栏、图标、位图、对话框等。通过该面板还可以添加或删除资源文件。

文件视图面板中显示项目之间的关系，以及包含在项目工作区中的文件。这里给出各项目文件间的逻辑关系。当双击某一项时，将会以对应的方式打开该项目。程序代码文件将在代码编辑窗口中打开，资源文件将在对应的资源编辑器中打开。

当需要对资源文件进行编辑时，可以通过在资源面板中双击所需要编辑的对象，打开相应的资源编辑器进行编辑。

通过这些不同的视图，可以从不同角度对程序中相关内容进行查看和编辑。例如，在类视图面板中可以通过双击某一项打开代码编辑窗口进行查看/编辑；在文件视图面板中双击某一代码文件名称，也可以打开代码编辑窗口进行查看/编辑。不过，在类面板中打开时是直接定位光标到双击的项所在行，而在文件视图面板中打开时是定位到所打开的文件当前编辑位置。

⑤ 项目工作区的右侧是 Visual C++ 6.0 集成开发环境的主工作区，当在项目工作区中双击某一项时，将在这里以相应的工具打开显示该项的内容。例如，在文件视图中双击某一代码文件名称项时，将在主工作区打开代码编辑窗口进行编辑；而在资源视图面板中双击某一项时，则将在该区域打开相应的资源编辑器以供编辑。

⑥ 在项目工作区的下方是输出窗口，这里用于显示程序编译、组建、调试等过程中的输出内容。

⑦ Visual C++ 6.0 集成开发环境的最下方是状态栏。根据当前工作的不同，状态栏中将显示出相应的信息，如光标位置、插入/改写方式等。

1.5 C++ 上机过程

前面已经说过一个 C++ 程序的实现需要经过编辑、编译、组建和运行几个步骤,下面以 Visual C++ 6.0 为环境介绍 C++ 的上机过程。

1. 源程序的输入

用户在纸上写好的 C++ 程序,只有输入到计算机内经过处理后才能运行,这样首先就要输入源程序到计算机,建立源程序文件。

(1) 启动 Visual C++ 6.0 开发环境

在“开始”菜单中选择“程序”,然后选择“Microsoft Visual Studio 6.0”,再选择“Microsoft Visual C++ 6.0”,显示 Visual C++ 6.0 开发环境主窗口,如图 1.2 所示。Visual C++ 6.0 开发环境主窗口也可以通过直接双击 Windows 桌面上的“Microsoft Visual C++ 6.0”图标来启动。

(2) 创建 C++ 文件

在主窗口的菜单栏,单击“文件”菜单项,然后选择“新建”命令,显示“新建”对话框,如图 1.4 所示。

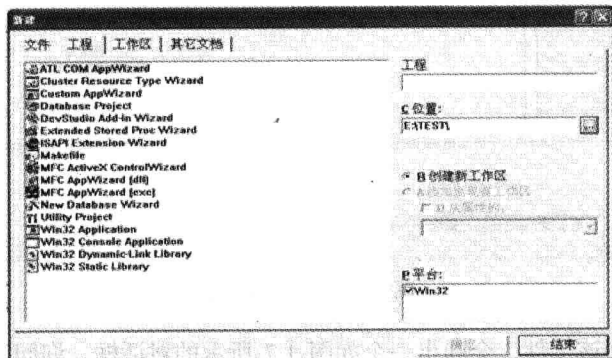


图 1.4 “新建”对话框

单击“新建”对话框的“文件”标签,在“文件”选项卡中选择“C++ Source File”一项,并在右边的“文件”文本框中填入文件名称,如 test1,可以省略文件的扩展名.cpp;同时在“目录:”文本框中输入希望保存文件的目录名,如图 1.5 所示。

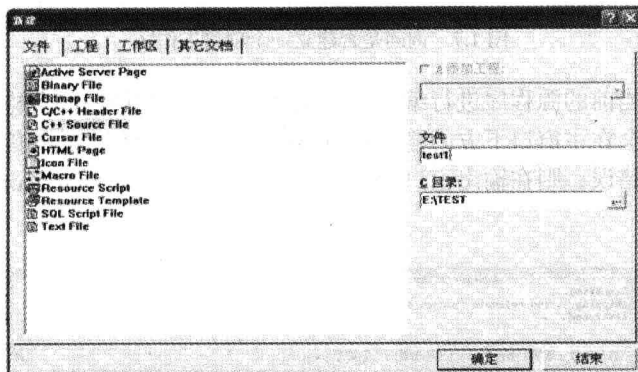


图 1.5 “新建”对话框(输入文件名和目录名)

单击“确定”按钮，完成新建 C++源程序文件，返回 Visual C++ 6.0 主窗口（这时可以在编辑窗口输入源程序）。

接下来在编辑窗口中输入源程序，源程序输入完成并检查无误后，应将它保持在磁盘文件中，其过程是：在菜单栏中选择“文件”，再在下拉菜单中选择“保存”命令。

2. 编译

用户输入源程序后，计算机还不能立即执行，还必须对源文件进行编译。对 C++源程序进行编译的操作过程是：单击主菜单中的“编译”，在下拉菜单中选择“Compile<文件名>”命令，如图 1.6 所示。

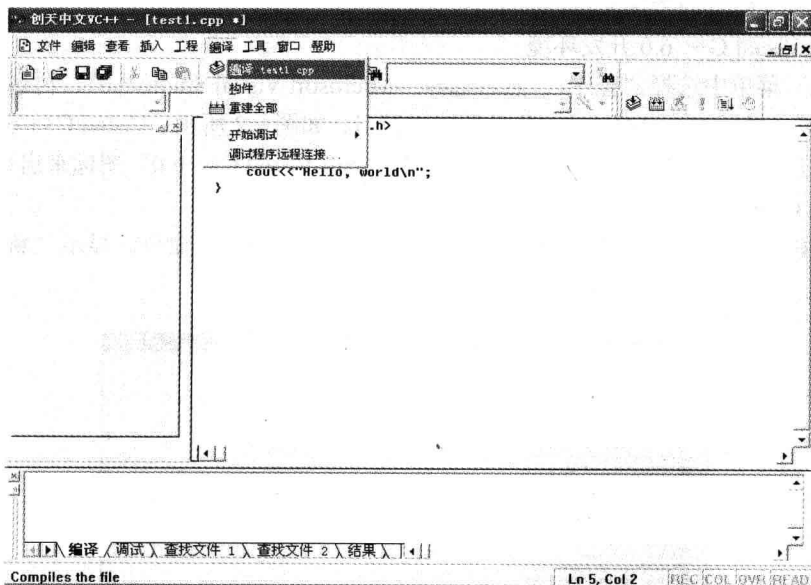


图 1.6 选择编译菜单

如果是第一次编译该文件，会弹出一个如图 1.7 所示的对话框，询问是否建立一个项目工作区，单击“是”按钮即可。

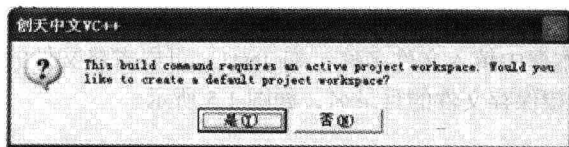


图 1.7 询问是否建立一个项目工作区

这时系统开始对当前的源程序进行编译。在编译过程中，编译系统会检查源程序中有没有错误。如果有错误，就会在主窗口下方的输出窗口给出错误的位置和性质等信息，以使用户对程序进行修改；如果没有错误，则在输出窗口给出“文件名.obj- 0 error(s), 0 warning(s)”这样的无错误信息，如图 1.8 所示。

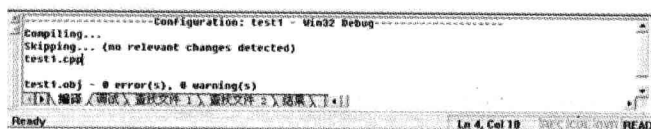


图 1.8 编译信息