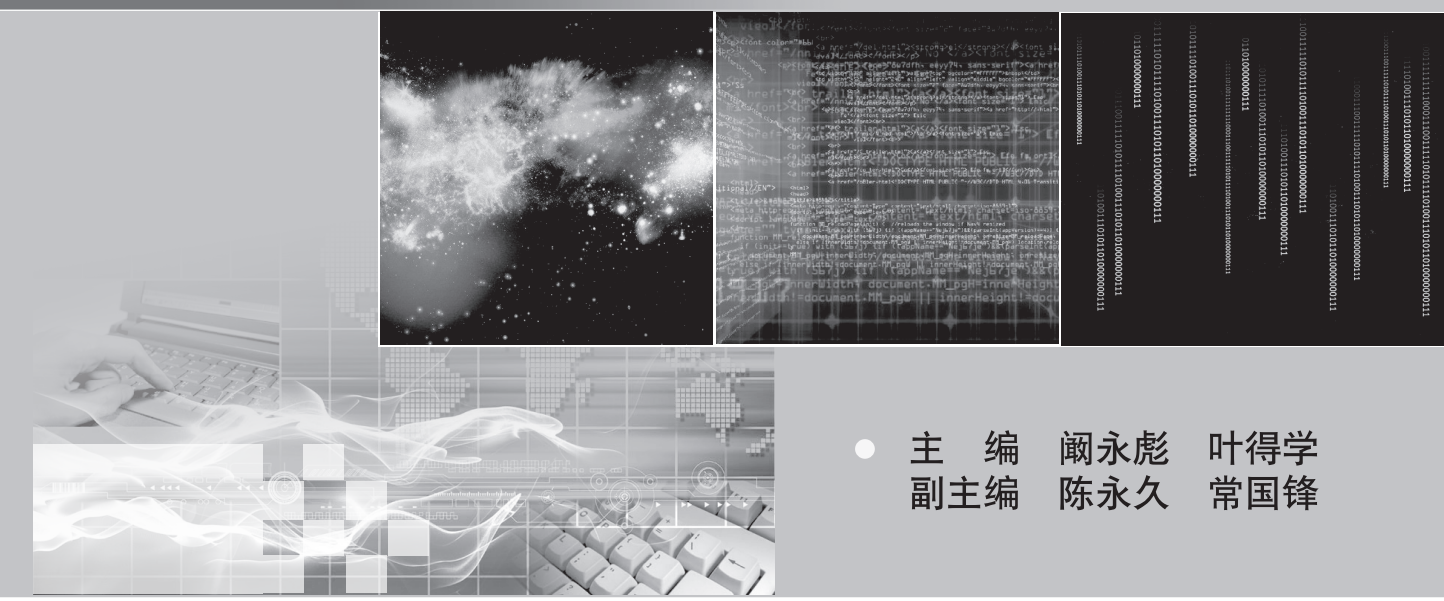
21世纪普通高等教育精品教材

计算机系列

C++程序设计



主　编　阚永彪　叶得学
副主编　陈永久　常国锋

WUHAN UNIVERSITY PRESS
武汉大学出版社

图书在版编目(CIP)数据

C＋＋程序设计/阚永彪,叶得学主编.—武汉:武汉大学出版社,2014.7

21世纪普通高等教育精品教材

ISBN 978-7-307-12740-1

Ⅰ.C… Ⅱ.①阚… ②叶… 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(2014)第004139号

责任编辑:王 师

出版发行:**武汉大学出版社** (430072 武昌 珞珈山)

(电子邮件:cbs22@whu.edu.cn 网址:www.wdp.com.cn)

印刷:三河市延风印装厂

开本:787×1092 1/16 印张:19 字数:395千字

版次:2014年7月第1版 2014年7月第1次印刷

ISBN 978-7-307-12740-1 定价:38.00元

内 容 简 介

本书依据《国家中长期教育改革和发展规划纲要(2010－2020 年)》的指导精神,并结合教育部最新颁布的教学指导要求及普通高等院校教学特点编写而成。

全书共分 11 章,主要包括 C＋＋语言概述,基本数据类型与表达式,C＋＋程序的流程控制,C＋＋函数,数组,指针,自定义数据类型,面向对象程序设计基础,类和对象的创建,类的继承与派生、多态与虚函数,以及流类库和输入输出。

本书可作为本科院校(含独立学院)的计算机专业课教材,也可供成人教育和高职高专院校使用,亦可作为广大青年朋友学习的参考用书。

前　言

本书在贯彻落实《国家中长期教育改革和发展规划纲要(2010—2020年)》的前提下,结合教育部《关于深化教育改革培养适应21世纪需要的高质量人才的意见》,根据普通高等院校教育人才培养目标及要求编写而成。

在以下领域,C++有着根本性的优势:低级系统程序设计、高级系统程序设计、嵌入式程序设计、数值科学计算、通用程序设计以及混合系统设计等。

在所有的编程语言中,C++可以说是最为复杂的。它既是一门传统的编程语言,又是一门新的编程语言。说它是一门传统语言,是因为C++诞生已将近20年了,特别是最近10年来,C++得到了快速的发展。C++是计算机软件领域中覆盖面最为广阔的编程语言,并且,与C++相关的智力投入也是其他任何一门语言所无法比拟的。人们对于C++的研究已经远远超出了对于一门编程语言所应有的关注。所以现在的C++已经非常成熟,有大量的资源、文档书籍、源代码等可供用户使用。说C++是一门新的编程语言,是因为在1998年C++由ISO(International Standards Organization)完成了标准化,从此C++领域有了统一的标准,所有的编译器都将向标准靠拢,或者说,与标准兼容。这有利于程序员写出可移植的C++代码来,同时C++标准也统一了C++标准库。为C++用户提供了最为基本的基础设施。C++经历了多年的发展,终于有了一个相对稳定的版本。所以,用户应该用一种新的眼光来看待C++,而不再简单地把C++认为是C语言的超集。

C++程序设计课程,已经成为许多大学理工科学生的必修课程,本书主要参考全国计算机等级考试二级C++考试大纲的要求进行编写,并适当做了一定的提高和修改。基本要求是:掌握C++语言的基本语法规则;熟练掌握有关类与对象的相关知识;能够阅读和分析C++程序;能够采用面向对象的编程思路和方法编写应用程序;能熟练使用Visual C++6.0集成开发环境编写和调度程序。

教材分11章,第1章主要是对C++语言的概述,让读者对C++有初步的认识;第2章主要介绍了C++的基本数据类型和表达式;第3章主要介绍了C++常用的三种流程控制结构和语句;第4章介绍了C++的函数;第5章介绍了C++的数组;第6章介绍了C++中的指针。这些内容的编排顺序充分考虑到学习者的认知特点,由易到难,由简单到复杂的过程。通过对上述内容的学习,读者会知道,程序是由一系列的动作序列组成的,在C++中,动作被称为表达式,以分号结尾的表达式被称作语句,C++中最小的程序单元是语句,把语句按逻辑语义分组,就形成了一些有名字的单元,这些单元被称为函数。通过对数组和指针的学习,可使读者进一步了解数据在内存中存储方式及管理方式,加深对计算机工作原理的理解。第7章介绍用户自定义数据类型,重点介绍了结构体的用法,这一章是过渡性的知识,为第8章介绍面向对象的程序

设计做个铺垫；第 8 章简要介绍了面向对象的程序设计思想和方法；第 9 章、第 10 章介绍类和对象，以及类的继承和派生的方法；第 11 章介绍了流类库和输入输出。后面 3 章集中讲述基于对象的程序设计——即 C++的类 class 设施的定义以及用法，通常可以用类来定义新的类型，并且操纵这些新的类型可以像内置类型一样容易，通过创建新的类型来描述问题域，C++使程序员能够编写出更易于理解的应用程序。

本书讲解的内容只是 C++入门方面的一些知识，适用于大学低年级的学生作为一门程序设计的入门课程来学习，建议该课程的学习时间至少为 96 个学时，同时要进行大量的上机练习。在学习方法上希望能做到四“多”一“有”。

(1)多看：多看别人写的程序，从简单的程序看起，揣摩别人的思想和意图。

(2)多抄：挑选难度合适的、别人编写好的代码，亲自去尝试一下运行的结果。在不断地借鉴别人的代码过程中，思维会不断升级。

(3)多改：正所谓“青出于蓝胜于蓝”，把个人的思想融入别人的思想中，那么就得到了两种思想。

(4)多实践：不要用纸和笔来写程序。没有人能保证那样写出来的程序一定能执行。一定要勤上机、勤测试，个人的水平才能真正提高。

(5)有风格：一名优秀的程序员都有着自己良好的风格习惯。至于这种良好的习惯如何养成，以后会在各章节陆续介绍。

必要准备——五“要”。

(1)要有一定能学会的信心和坚持到底的决心。

(2)要有足够的时间去经常写程序，经常去实践。长时间不写程序，水平就会退步。

(3)要有良好的身体素质。做程序员很伤身体，废寝忘食更是家常便饭。

(4)要有一定的电脑常识和实践操作基础。

(5)要配置电脑和相关软件。

编　者

目　录

第 1 章　C＋＋语言概述

第 2 章　基本数据类型与表达式

第 3 章　C＋＋程序的流程控制

第4章　C++函数

第5章　数组

第6章　指针

第 7 章　自定义数据类型

第 8 章　面向对象程序设计基础

第 9 章　类和对象的创建

第 10 章　类的继承与派生、多态与虚函数

第 11 章　流类库和输入输出

参考文献

第1章　C++语言概述

学习目标

本章主要介绍C++语言的基本概念，通过本章的学习，要求学生能够：

1. 了解C++语言的基本符号；
2. 了解C++语言的词汇；
3. 掌握C++程序的基本框架；
4. 使用 Visual C++ 6.0 集成开发环境编辑、编译、运行与调度程序。

C++是一门应用非常广泛的计算机程序设计语言。它既支持过程化程序设计，又支持面向对象程序设计。从操作系统、设备控制到数据库、网络、多媒体等众多的应用领域都能看到它的身影。目前，业界应用比较广泛的是它的两个变种：Visual C++和 Borland C++。这两种C++都是在标准C++的基础上做了一些扩展而来的。本书将基于 Visual C++ 6.0版本的开发环境，向读者讲授标准C++的强大功能和编程方法。

1.1　概　　述

C++作为当今业界应用最为广泛的编程语言之一，其涵盖的内容非常广泛多样。本节将首先给读者一个初步的概念。

1.1.1　C++的发展史

C++由美国贝尔实验室的 Bjarne Stroustrup 博士在 20 世纪 80 年代初期发明并实现，最初被称作 C with Classes。1983 年，C with Classes 被更名为C++。1985 年 10 月发布了第一个商业化版本。同年，C++的第一本编程手册《C++程序设计语言》出版。1989 年，发布了第二版。1990 年，出版了 The Annotated C++ Reference Manual。这本书后来成为标准化的基础。稍后又引入了模板、异常处理、命名空间、新的强制转型，以及布尔类型等特性。

从 20 世纪 80 年代到 1995 年。这一阶段C++语言基本上是传统类型上的面向对象语言，并且凭借着接近 C 语言的效率，在工业界使用的开发语言中占据了相当大份额。

从 1995 年到 2000 年，这一阶段由于标准模板库（STL）和后来的 Boost 等程序库的出

现，泛型程序设计在C++中占据了越来越多的比重。当然，由于Java、C#等语言的出现和硬件价格的大规模下降，C++也受到了一定的冲击。

从2000年至今，由于以Loki、MPL等程序库为代表的产生式编程和模板源编程的出现，C++出现了发展历史上又一个新的高峰，这些新技术的出现以及和原有技术的融合，使C++已经成为当今主流程序设计语言中最复杂的一员。

1.1.2 C++的优势

计算机的发明与发展是20世纪最伟大的科学成就之一。作为一种通用的智能工具，计算机具有以下几个特点。

C++语言是由C语言扩充而来。它是一门混合型的语言，既支持传统的结构化程序设计，又支持面向对象的程序设计，这使得C++非常流行。它在编程中支持面向对象。因此，它比C更适合人们的思考方式，并且比C要安全、可靠、有效。但是它并不像Java那样全面实现类的使用，这使得它很灵活，能够适应多种场合的需要。

C++是编译语言，性能良好，效率更高。它不像Java、PHP、ASP等解释型语言需要在运行时转换为机器码，或者读一条语句执行一条。它直接被编译为机器可识别的机器码。

C++编写的程序独立性好。只要C++编译时产生的机器码不依赖外部的动态链接库，C++就可以在不需要安装额外程序的情况下，移动到运行同样操作系统的其他机器和微处理器上。而移动Java程序时需要先安装Java运行期库。这是C++语言成功流行的一个重要原因。

C++语言应用极为广泛。尤其在底层编程和系统级编程上更是C++的传统优势。在数据库和多媒体方面，C++也以其卓越的稳定性而赢得了荣誉。

1.1.3 对面向对象方法的支持

对象是指人们要进行研究的任何事物。面向对象的核心思想是对象和对象间的消息，是建立在“对象”基础上的方法学。它将系统看做对象的组合，每个对象都能接受并处理数据，然后和其他对象通过消息来交换数据。面向对象编程具有以下3个基本特征：封装、继承和多态。

在C++中，用类来封装数据和数据上的操作，外界看不到该类的属性和方法。如果需要的话，系统一般会为每个属性提供set/get函数对。

继承就是从父类继承旧的属性、参数等。C++可以进行公有、私有、保护3种类型的继承。有时为了避免内存的浪费，也可以进行虚继承。在虚继承中，整个继承体系里公有的基类将不会被重复分配空间。

多态指用多种形态来实现。C++中的多态包括函数重载、子类与基类间的覆盖、虚函数、运算符重载以及抽象类等方式。

1.2　开发工具

要想发挥C++的强大功能，就需要一个优秀的集成开发环境，因为一个好的编程环境常常可以起到事半功倍的作用。集成开发环境(IDE)是一个将程序编辑器、编译器、调试工具和其他建立应用程序的工具集成在一起，用于开发应用程序的软件系统。本节将向读者介绍常见的几款C++开发环境，并着重介绍本书所用到的开发环境。

1.2.1　常见的C++开发工具

目前，传统的C++开发工具有 Visual C++、C++ Builder 两种，新近又出现了 Dev-C++、Anjuta、Code::Blocks、Eclipse 等开发工具。它们各有千秋，下面将简单介绍一下它们的特性。

1. Visual C++

Visual Studio 是微软公司推出的开发环境，它是目前最流行的 Windows 平台应用程序开发环境。在 Visual Studio 中可以创建应用程序和网络应用程序，以及智能设备应用程序和 Office 插件等。Visual Studio 对标准C++做了扩展，习惯上将在 Visual Studio 中实现的C++称为 Visual C++，即 VC。VC 是目前使用最广泛的C++开发语言，Visual Studio 也是使用最广泛的开发工具。

2. C++ Builder

C++ Builder 是另一款比较流行的C++开发工具。它由 Borland 公司于 1998 年推出，也是 Windows 下的开发工具。C++ Builder 具有高度安全性、高可靠性、快速性的编译优化方法，所以编译出的软件执行速度很快。所有符合 ANSI/ISO 标准的源代码都可以在 C++ Builder 中编译，而且支持最新的 ANSI C++/C 语言特征。

3. Dev-C++

Dev-C++是 Windows 平台下的开源C++编程环境。它集成了 GCC、MinGW32 等众多自由软件，界面类似 Visual Studio，但体积要小的多。它的缺点是难以胜任规模较大的软件项目，但对于初学者是一个不错的选择。

4. Anjuta

Anjuta 是一款 GNOME 桌面环境下的 C/C++编程的集成环境，也是开源软件。它不仅有项目管理、交互式调试以及强大的代码编辑和语法增色的功能，还可以直接开发 Glade 图形界面的程序。但是，它主要用在 Linux、UNIX 平台下，不支持 Windows 平台。

5. Code::Blocks

Code::Blocks 是一款全功能的、跨平台的 C/C++集成开发环境，属于开源软件。它提

供了众多的工程模板，包括控制台应用、动态链接库、OpenGL 应用、QT 应用、Win32 GUI 应用等。Code::Blocks 的另一过人之处在于它具有丰富的插件，包括代码格式化、类向导、代码补全、代码统计、To-Do 列表，以及 WindowsXP 外观等各种各样的插件。此外，它不仅支持 GNU GCC 编译器，还支持 Visual C++、Borland C++编译器，以及 Digital Mars C 等多种编译器。

6. Eclipse

Eclipse 是目前开源平台中最著名的集成开发环境。最初主要用来支持 Java 语言编程，目前通过插件 CDT 也可以用来开发 C/C++程序。因为它本身只是一个框架，因此插件众多是它的一大特点。这使得 Eclipse 拥有其他支持单一语言的 IDE 环境很难具有的灵活性。

此外，还有很多 IDE 也都支持C++开发，例如 Visual Slick Edit、NetBeans、Understand C 等，读者可以根据自己的需要选择适合自己的开发环境。

1.2.2 Visual C++ 6.0 开发环境简介

Visual C++是微软公司推出的目前使用极为广泛的基于 Windows 平台的可视化集成开发环境。它包含了一个文本编辑器、资源编辑器、工程编译工具、一个增量连接器、源代码浏览器、集成调试工具，以及一套联机文档 MSDN。Visual C++ IDE 一般由 3 部分组成：Developer Studio、MFC 和 Platform SDK。

1. Developer Studio

这是一个集成开发环境，它提供了一个很好的编辑器和很多 Wizard。它不仅可以用来开发 C、C++，还可以用来开发 Visual Basic、VF 等语言。使用 Developer Studio，可以完成创建、调试、修改应用程序等各种操作。

2. MFC

MFC(Microsoft Foundation Classes)表示为微软基础类，它的应用程序的总体结构通常由开发人员从 MFC 类派生的几个类和一个 CWinApp 类对象(应用程序对象)组成。但是用 Visual C++编写代码也并不意味着一定要用 MFC，使用 STL、ATL、编写 SDK 程序一样没有限制。

说明：STL 是 Standard Template Library 的缩写，表示标准模板库。ATL 是 Active Template Library 的缩写，表示活动模板库。

3. Platform SDK

Platform SDK 是以 Microsoft C/C++编译器为核心，配合 MASM，辅以其他一些工具和文档资料。SDK 表示 Software Development Kit，意为软件开发工具包。

Visual C++ 6.0 支持的文件类型众多，详见表 1-1。

表 1-1　文件类型

文件类型	说　明	文件类型	说　明
ActiveServerPage	ASP 文件	BinaryFile	二进制文件
BitmapFile	位图文件	C++SourceFile	C++源程序文件
C/C++HeaderFile	C/C++头文件	CursorFile	光标文件
HTMLPage	HTML 文件	IconFile	图标文件
MacroFile	宏文件	ResourceScript	资源脚本文件
ResourceTemplate	资源模板	SQLScriptFile	SQL 语言脚本文件
TextFile	文本文件		

由于功能强大，Visual C++6.0 预定义的工程类型也非常多，详见表 1-2。

表 1-2　工程类型

工程类型	说　明	工程类型	说　明
ATL COM AppWizard	ATL 程序	Database Project	数据库
Win32 Dynamic-Link Library	Win32 动态链接库	DevStudio Add-inWizard	自动嵌入执行文件宏
Custom AppWizard	自定义程序向导	ISAPI Extension Wizard	Internet 服务器或过滤器
Makefile	Make 文件	MFC ActiveX Control Wizard	ActiveX 控件
MFC AppWizard(dll)	MFC 动态链接库	MFCAppWizard(exe)	MFC 可执行文件
Win32 Application	Win32 程序	Win32 Console Application	Win32 控制台程序
Win32 Static Library	Win32 静态库	Utility Project	该工程作为其他子工程的容器，从而减少子工程的联编时间

1.2.3　Visual C++ 6.0 开发环境的使用

Visual C++提供了一个集源程序编辑、代码编译与调试于一体的开发环境，这个环境称为集成开发环境。通过集成开发环境程序员可以访问C++源代码编辑器、资源编辑器，使用内部调试器，以及创建工程文件。本节将详细讲解 Visual C++ 6.0 集成开发环境各部分的使用方法。

1. 主界面

为了使用 Visual C++ 6.0 开发环境，首先需要单击“开始”|“程序”|Microsoft Visual Studio 6.0|Microsoft Visual C++ 6.0 命令，打开 IDE。主界面如图 1-1 所示。

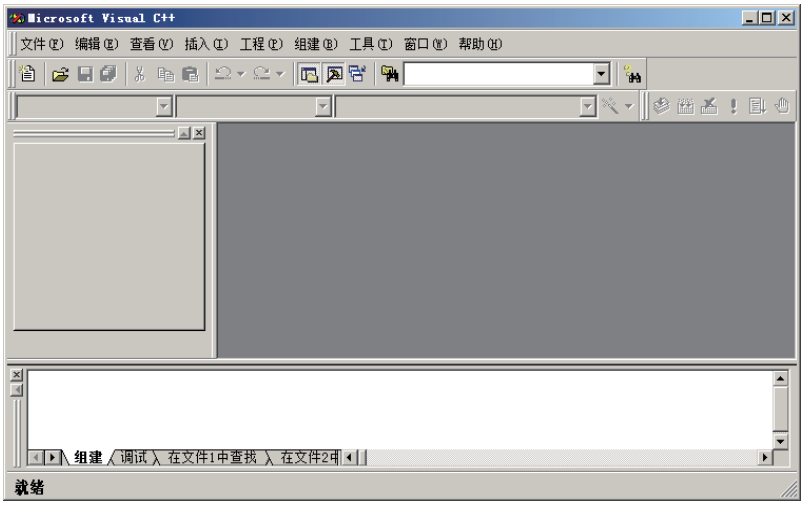


图 1-1　Visual C++ 6.0 IDE

图中上部为菜单条，菜单条下面为工具栏。屏幕左部为工程管理面板，右部为工作区。工作区下部为输出窗口，再往下是状态栏。各菜单的功能如下所示。

- 文件菜单：包括对文件、项目、工作区及文档进行文件操作的相关命令或子菜单。
- 编辑菜单：除了常用的剪切、复制、粘贴命令外，还有为调试程序设置的 Breakpoints 命令，完成设置、删除、查看断点。此外还有为方便程序员输入源代码的 List Members、Type Info 等命令。
- 查看菜单：该菜单中的命令主要用来改变窗口和工具栏的显示方式、检查源代码、激活调试时所用的各个窗口等。
- 插入菜单：该菜单包括创建新类、新表单、新资源及新的 ATL 对象等命令。
- 工程菜单：该菜单可以创建、修改和存储正在编辑的工程文件。
- 组建菜单：该菜单用于编译、创建和执行应用程序。
- 工具菜单：该菜单允许用户简单快速地访问多个不同的开发工具，如定制工具栏与菜单、激活常用的工具(Spy++等)或者更改选项等。

2. 创建工程

工程是一个文件。该文件记录了程序中包含的文件和集成环境的配置参数。因此为了编写一个程序，首先就要创建一个工程。选择 File|New 命令，打开新建工程对话框，如图 1-2所示。

图 1-2　创建工程

该界面左部由 4 个选项卡组成。

Files 选项卡用来创建单个的文件，如头文件、程序文件、资源文件等。

Projects 选项卡用来创建工程，图中列出的是 Visual C++ 6.0 所支持的各种类型的工程。

Workspaces 选项卡可以创建一个工作空间，一个工作空间可以包含多个工程。如果编写的软件系统比较庞大，模块众多，就可以利用工作空间来统一管理。

Other Documents 选项卡可以用来创建一些其他类型的文档，这要取决于操作系统都安装了什么样的软件。例如，如果安装了 Office，这里就会出现 Word、Excel 等文档。在右部可以输入要创建的工程名称和存放位置。一般默认是 create new workspace 选项，如果有旧的工作空间，还可以选择加入到该空间内。

技巧：每个工程类型都是一种代码的模板。如果预定义的模板不能满足程序员的要求，可以选择 Custom AppWizard 定义特定的模板。

3. 工具栏

工具栏是一组直观、快捷的图形化按钮和编辑框，熟练使用工具栏可以大大提高工作效率。Visual C++中包含很多种工具栏，默认为图 1-3 所示的工具栏。

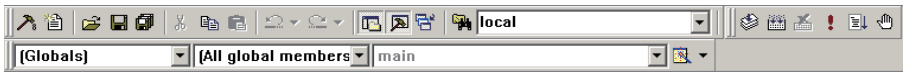


图 1-3　工具栏

一般来讲，工具栏会根据当前工作的不同而不同。例如，调试程序时，会出现调试工具栏；编写数据库程序时，会出现数据库工具栏。如果要添加新的工具栏，只需右击工具栏，然后在弹出的快捷菜单中选中需要的功能，它就会出现在工具栏上。

4. 工程管理面板

工程管理面板包括 3 个选项卡，每个选项卡的功能如下所示。

ClassView 选项卡显示工程中使用的类、函数、全局变量等，双击可以跳转到对应的代码处，如图 1-4 所示。

ResourceView 选项卡显示工程中使用的资源，双击可以编辑该资源，如图 1-5 所示。

FileView 选项卡显示工程中使用的文件。文件按类型管理，双击可以进行编辑，如图 1-6所示。

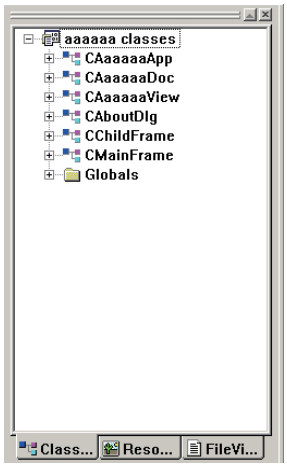


图 1-4　ClassView 选项卡

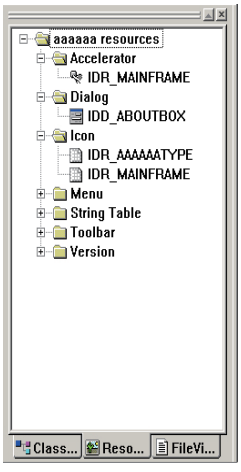


图 1-5　ResourceView 选项卡

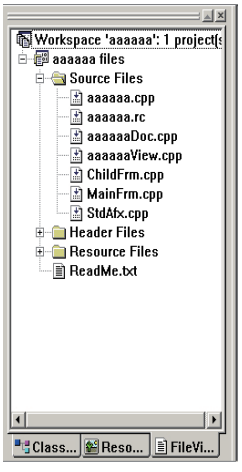


图 1-6　FileView 选项卡

5. 编译运行

编译是对写好的程序进行检查，如果检查无误就会生成目标代码。生成目标代码后，还需要进行链接才能生成最终的可执行程序，然后才能执行程序。这些操作都要通过 Builder 菜单来完成。该菜单主要用于编译、创建和执行应用程序，如图 1-7 所示。

菜单中 Compile 命令是编译当前文件，对当前文件进行语法检查。Build 命令是链接当前工程，生成可执行文件。如果要编译全部文件，就要选择菜单中的 Build All 命令。当编译好文件后，可以单击 Execute 命令来执行程序。

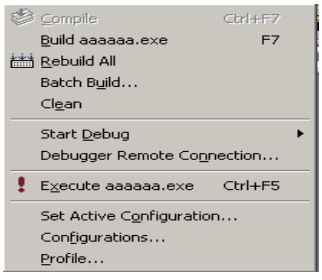


图 1-7　编译运行

1.2.4　Visual C＋＋ 6.0 使用示例

在这一小节将给读者演示一个典型的C＋＋程序的开发过程，该程序用C＋＋语法输出字符串 Hello world!。

(1) 选择“开始”|“程序”|Microsoft Visual Studio 6.0|Microsoft Visual C＋＋ 6.0 命令，启动 Visual Studio 6.0 集成开发环境，如图 1-8 所示。

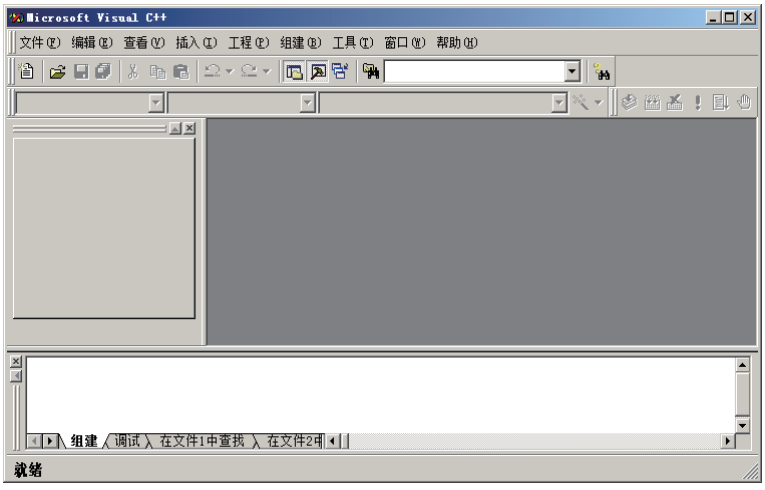


图 1-8　Visual Studio 6.0 集成开发环境

（2）选择 File|New 命令，打开 New 对话框，如图 1-9 所示。该对话框主要用来选择要创建的工程类型，并选择工程文件的保存位置和工程名称。

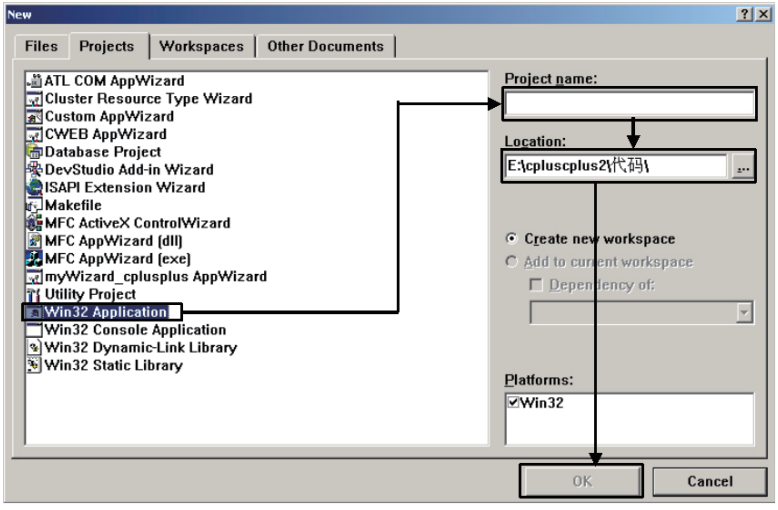


图 1-9　工程类型选择

（3）选择 Win32 Console Application 选项，该选项表示要创建 Windows 下的 32 位控制台应用程序。然后在 Project name 文本框中输入工程名称 test，在 Location 文本框中输入工程保存的位置。最后单击 OK 按钮，将打开如图 1-10 所示的对话框。

图 1-10　选择控制台程序的类型

(4) 在这个窗口可以选择要创建的控制台程序的类型。如果上个窗口选择的工程类型不同,这里将出现不同的选项。该窗口实际上是选择要使用的模板。

An empty project:表示创建一个空工程,该工程不会自动生成程序文件,仅包含环境配置文件;

A simple application:表示创建一个简单程序,仅是一个简单的程序框架,不包含任何有用的代码;

A"Hello,World!"application:表示创建一个有一条输出语句的简单程序;

An application that supports MFC:表示是带有 MFC 支持的程序框架。

注　意

An application that supports MFC 选项只是在程序中加入了 MFC 的头文件支持,具体的使用还需要程序手工增加。如果在创建新工程时,选择创建 MFC 型工程,则系统会自动创建一个 MFC 的使用框架。

出于演示的需要,这里选择第 3 种。单击 Finish 按钮,出现图 1-11 的对话框。

图 1-11　工程摘要

(5) 该窗口显示了工程的一个摘要,说明该自动生成的工程都包含哪些文件。单击 OK 按钮就创建了一个简单的控制台应用程序,如图 1-12 所示。

该窗口显示了刚才创建的控制台应用程序。左边的工程管理栏内,ClassView 选项卡显示了程序的类结构图,FileView 选项卡显示了文件结构图。右边就是要编写程序的地方。

(6) 选择左边工程管理栏中的 FileView 页,双击 Stdafx.h,打开该文件,如图 1-12 所示。在右边的代码区内找到 stdio.h,将它修改为 iostream。

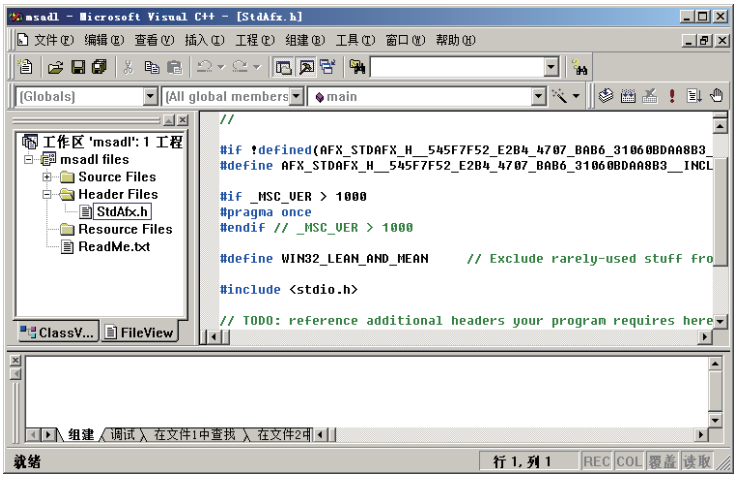


图 1-12　主窗口

说明:stdio.h 是 C 语言的输入输出支持头文件,iostream.h 是C++的输入输出头文件。创建 32 位控制台应用程序时,Visual Studio 6.0 默认使用的 I/O 头文件是 stdio.h,因此需要改成 iostream.h。这是将负责输入输出的头文件引入,为了在后面使用 cin 和 cout 输入输出流。修改后的 stdafx.h 文件如下所示。

```
// stdafx.h : include file for standard system include files,
// or project specific include files that are used frequently, but
// are changed infrequently
#if ! defined(AFX_STDAFX_H__AE3AD910_2C76_44B4_A864_1212C9114CAD__
includeD_)
#define AFX_STDAFX_H__AE3AD910_2C76_44B4_A864_1212C9114CAD__includ-
eD_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
# define WIN32 _ LEAN _ AND _ MEAN // Exclude rarely - used stuff from
Windows headers
#include <iostream.h>
```

```
// TODO: reference additional headers your program requires here
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif
// ! defined(AFX_STDAFX_H__AE3AD910_2C76_44B4_A864_1212C9114 CAD__includeD_)
```

(7) 经过上述修改后,还需要修改 test.cpp 文件,该文件是程序的主要实现文件。双击 test.cpp 文件打开它,如下所示。

```
// test1.cpp : Defines the entry point for the console application.
#include "stdafx.h"
int main(int argc, char * argv[])
{
    printf("Hello World! \n");
    return 0;
}
```

上述代码是创建新工程时自动生成的,该代码使用一条 printf 语句输出了字符串 Hello world!。printf 语句是 C 语言中的输出语句,C++中的输出语句则使用 cout 流。因此,需要将代码区中的 printf 语句修改为如下代码:

```
cout<<"Hello world! "<<endl;
```

(8) 修改后的 test.cpp 文件如下所示。

```
// test1.cpp : Defines the entry point for the console application.
#include "stdafx.h"
int main(int argc, char * argv[])
{
    cout<<"Hello world! "<<endl;
    return 0;
}
```

经过上述步骤就建立了一个输出字符串 Hello world! 的简单控制台应用程序。要想运行它还需要进行编译和链接。编译操作负责对源程序进行语法检查,然后生成目标代码。链接操作负责将目标代码和程序用到的库程序链接成可执行文件。在 Visual Studio 6.0 中,Builder 菜单提供了编译、链接和执行程序的操作。编译和链接的步骤如下所示。

①选择 Builder|Compile 命令,编译源程序。如果输出窗口内的输出内容如下所示,则

表示没有语法错误，且正确生成了目标代码。

```
----------Configuration: test - Win32 Debug----------
Compiling...
Skipping...(no relevant changes detected)
test.cpp
test.obj - 0 error(s), 0 warning(s)
```

该输出内容表示检测到 0 个错误，0 个警告，生成目标代码 test.obj。

②对该工程进行链接，以生成可执行文件。选择 Builder|Build 命令，执行链接操作。如果输出如下所示，则表示链接正常，生成可执行文件正常。

```
----------Configuration: test - Win32 Debug----------

test.exe - 0 error(s), 0 warning(s)
```

该输出说明链接时检测到 0 个错误，0 个警告，生成可执行文件 test.exe。

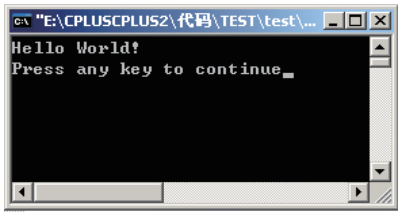


图 1-13　test.exe 的执行结果

生成可执行文件后就可以执行该程序。执行一个 32 位控制台程序即可以从 Windows 的“资源管理器”中双击打开，也可以从命令行手工执行打开。在 Visual Studio 6.0 中，还可以选择 Builder|Execute 命令，执行该工程，如图 1-13 所示。

图中第一行是可执行程序 test.exe 的输出，第二行则不是。第二行是由 Visual Studio 6.0 开发环境执行完 test.exe 程序后输出的，表示“按任意键继续”，即按任何一个键就可返回到开发环境中去。到此为止，本节就完整演示了在 Visual Studio 6.0 中开发一个典型的C++程序的过程。

1.3　程序结构

程序结构是程序的组织结构，指该程序语言特定的语句结构、语法规则和表达方式，其内容包括代码的组织结构和文件的组织结构两部分。只有严格遵守这种规则，才能编写出高效、易读的程序。否则写出的代码将晦涩难懂，甚至不能被正确编译运行。

1.3.1　第一个程序

这一节将通过一个简单的程序讲解C++程序的基本结构，同时说明C++程序中输入输出操作的方法，以方便对后续章节的学习。

例 1.1 实现从命令行读入一个整数，然后加 1 再输出结果。

```
/*第一部分*/
//这是一个演示程序，它从命令行读入一个整数，然后加 1 再输出
/*第二部分*/
#include <iostream.h>
/*第三部分*/
int main()
{
    int x;
    cout<<"输入整数:";
    cin>>x;
    x=x+1;
    cout<<"x=x+1="<<x<<endl;
    return 0;
}
```

C++程序通常会包括示例中所示的三个部分。

1. 第一个部分

第一部分是整个文件的注释，指出该文件的作用和版权等信息。

2. 第二部分

第二部分是预处理部分，既在编译前要处理的工作。这里是以 #include 说明的头文件包含代码 #include <iostream.h>，它指示编译器在预处理时，将文件 iostream.h 中的代码嵌入到该代码指示的地方。其中 #include 是编译指令。头文件 iostream.h 中声明了程序需要的输入输出操作的信息。

注 意

在 Visual Studio 中，有时还会看到 #include <iostream> 的引入方式，这也是 Visual Studio 中鼓励使用的方式。但是采用这种方式时，还需要用“using namespace std;”引入 std 命名空间；否则系统会提示 cin 和 cout 找不到。

3. 第三部分

第三部分是代码的主要部分，它实现了一个函数，结构如下：

```
int main()
{
    …
    return 0;
```

```
}
```

其中，花括号包含了实现该函数所需的代码。C++规定每个可执行程序都有且只能有一个名为 mian 的函数，它表示了程序的入口点。当C++程序执行时，首先执行该函数，然后从该函数内调用其他需要的操作。下面依次分析每行代码的功能。

第 1 条代码“int x;”表示定义一个对象，并命名为 x。末尾的分号表示这条代码到此结束。

第 2 条语句使用 cout 流输出一行文字。其中，cout 是一个代表标准输出设备（在这里指显示器）的对象，它是C++中预定义的系统对象。当程序要向输出设备输出内容时，就需要在程序中使用该对象。输出的运算符用“<<”表示，它表示将运算符右边的内容输出到运算符左边的对象上。代码中运算符左边的内容用双引号包含起来，这表示它是一个字符串。因此，代码 cout<<"输入整数:";将在标准输出设备上输出字符串文字“输入整数:”。

第 3 条语句“cin>>x;”中，cin 是一个代表标准输入设备（一般指键盘）的对象，也是C++中预定义的对象。当程序需要从输入设备接收输入时，就需要在程序中使用该对象。输入的运算符是“>>”，它表示将从运算符左边接收的输入放到右边的对象中。当程序执行到该代码处时，将停止并等待来自标准输入设备的输入。输入完毕后，按下 Enter 键，cin 接收输入并将输入放到相应的对象中，然后跳到下一条代码开始执行。

第 4 条语句“x=x+1;”中，“+”号表示加法运算，即将加号两边的对象相加。“=”号表示赋值，它将符号右边的运算结果放到符号左边的对象中。因此，该代码表示将对象 x 保存的值加 1 再放回 x 中。

第 5 条语句仍然是一条在标准输出设备上输出文字的代码。它包含 3 个输出运算符，第 1 个操作输入了文字“x=x+1=”。第 2 个运算符输出对象 x 保存的值。第 3 个运算符的右边是 endl，它表示一个“回车换行”操作。因此，该代码在输出前两个文字后就将光标跳到新的一行上去。

第 6 条语句“return 0;”是一条跳出程序的代码。它表示从程序跳出并返回到操作系统，同时带有一个数字 0 作为返回值。

注　意

有些编译器并不要求 main 函数一定返回值，如 Visual C++。但是，有些编译器却要求必须有返回值，如开源软件 MinGW。

1.3.2　文件的组织

当程序较小时，用一个文件就可以保存所有代码。但是有实际用途的程序一般都不会太小。所以，通常会将程序分成几个文件分别保存，再通过包含语句放到一起。这种做法既有利于模块化开发，又有利于代码的重用。

C++的程序的文件类型有.h 和.cpp 两种，前者是头文件，后者是代码的实现文件。头

文件中包含了类、函数、常量、全局变量等的声明，使用时用#include语句在程序的预处理部分包含进来即可。代码的实现文件是对头文件中声明的类、函数等的具体实现。不需要显示包含.cpp文件，当包含了与它对应的.h文件后，编译器会自动去找同名的.cpp文件。

1.4 编码规范

编码规范是指编写代码时应该遵守的文字约束。一个好的编码规范不仅能够明显改善代码的可读性、可管理性，还可以提高程序的可靠性、可修改性、可维护性，以及一致性，从而保证代码的质量。当开发工作需要团队合作时，优秀、统一的编码规范更是合作能否成功的关键。

1.4.1 风格指南

(1)程序块采用缩进风格。每个缩进为4个空格位，要用Tab键缩进，而不要混合使用空格和Tab键。

(2)每个函数不要太长。如果真的需要写很长，就将它拆成多个简单的函数。

(3)避免使用过深的if语句嵌套。这样会增加代码的复杂度，降低可理解性。可以考虑改成函数来调用。

(4)双目运算符的前后加空格。

(5)单目运算符前后不加空格。

(6)尽量不要写太长的语句。如果需要很长，可以考虑分行，更好的办法是拆成多条语句。

(7)使用类时，每个模块中只放一个类。

(8)功能相对独立的程序块之间或for、if、do、while、switch等语句前后应加一空行。

(9)If、while、for、case、default、do等语句各自占一行。

(10)一行不要写多条语句。

(11)当表达式中有多个运算符时，尽量用括号来明确标出优先级。

以上总结了11项指南，实际编程时还有更多的规定。程序员可以根据需要酌情调整。

注意：程序的风格对程序的功能没有影响，编译器会忽略掉程序中的注释。

1.4.2 注释的使用

注释是为了增加程序的可读性和美观，给程序提供注解和备忘录。注解不能太多，也不能太少。太多有碍阅读，太少不利于注解代码。C++中的注释有行注释和块注释两种。行注释用“//”，一次注释一行。块注释用“/* */”，星号中间是被注释的块，可以是多行，也可以是单行。

例 1.2　演示单行注释的使用方法。

```
//计算 x 和 y 的和
z=add(x,y);
```

如果用块注释可以这么写：

```
/*计算 x 和 y 的和*/
```

当需要注释多条语句时，虽然也可以用多条行注释，但是建议使用块注释。

例 1.3　块注释的使用举例。

```
/*
计算 x 和 y 的和
函数 add 接收两个整型参数，返回它们的和
*/
int add(int x, int y)
{
    return (x+y);
}
```

为了写出规整的注释，需要遵循下述几条规则。

不要滥用注释，只在必要的地方写注释，注释要准确、易懂、简洁。

注释要与被注释的内容一致，只能描述被注释的内容，而不能描述其他。

注释要放在被注释内容的上方，或者被注释语句的后面，以符合阅读习惯。

函数头部应该进行注释，列出函数的功能、可接收的参数、返回值等。

要对.h 和.cpp 文件进行注释，注释应该列出文件名、作者、功能、版本等。

对重要的语句、代码块、变量和操作等要给予充分明确的注释。

下面再给出对函数 add 的完整注释。

```
/*
* 函数名称:add
* 参　　数:接收两个整型参数
* 返 回 值:返回整型值
* 函数功能:计算两个整型参数的和并返回其值
* 作　　者:XXX
* 版 本 号:0.0.1
* 修改日期:XXXX.XX.XX
*/
```

注释的内容可以根据需要酌情增减，但是一定要清晰明确，遵循上述几条原则。

1.4.3 美化代码

代码不仅是写给编译器看的，也是写给其他同事或同行业的人看的。所以在保证程序正确性的前提下，一定要注意代码的优雅和整洁。这既显示了程序员良好的素养，又显示了其优秀的编程风格。对代码的美化主要通过规范严谨的缩进和必要的注释。

例 1.4 寻找 3 个数中最大数的代码。

```
/*
 * 函数名称:MaxIn3
 * 参 数:接收 3 个整型参数
 * 返回值 :无
 * 函数功能:找出 3 个整型数中较大的数
 * 作 者:XXX
 * 版本号 :0.0.1
 * 修改日期:XXXX.XX.XX
 */
void MaxIn3(int x,int y,int z)
{
    int num=0; //存放最大数

    //选择最大数
    if(x>y)
    {
        //x 较大
        if(x>z)
        num=x;    //x 是最大数
        else
        num=z;    //z 是最大数
    }
    else
    {
        //y 较大
        if(y>z)
        num=y; //y 是最大数
        else
        num=z; //z 是最大数
    }
```

```
    //返回最大数
    return num;
}
```

这里给出了一个例程作为参考，读者可以根据自己的需要删减。通常，IDE 都会带有专门的格式化代码的工具，也有专门的第三方工具可供使用。但是这些都只是对代码的缩进进行格式化，至于注释还需要自己合理安排和增加。

技　巧

如果希望写出更规范和优美的注释，可以参考开源软件 doxygen 的格式。该软件提供了一个内容丰富的风格规范，其更大的好处是可以用该软件将源代码中符合要求的注释导出成文当。

1.5　标识符

标识符就是为变量、函数、类以及其他用户对象所起的名称。本节主要讲解C++中保留的标识符以及如何定义用户自己的标识符。

1.5.1　保留字

保留字就是系统已经预定义的标识符，不能再用来定义为其他意义，也被称作关键字。C++中常见的保留字见表 1-3。

表 1-3　C++中的常见保留字

_asm	abstract	bool	break	case
catch	while	char	class	const
continue	default	delete	do	double
else	enum	explicit	extern	false
float	for	friend	goto	if
inline	int	long	namespace	new
operator	private	protected	public	struct
class	register	return	short	signed
sizeof	static	switch	template	this
throw	true	try	typedef	union
unsigned	using	virtual	void	volatile

保留字是系统预留的符号，这些符号已经被赋予特定的意义，所以程序员只能直接使用

它们而不能修改其定义。例如，保留字 int 和 float 分别用来表示整型和浮点型数据类型，for 和 while 用来表示循环语句。

1.5.2 标识符命名

标识符的名称可以是一个字符，也可以是多个字符。第一个字符必须是字母或下画线，后跟字母、数字、下画线的组合。标识符区分大小写。长度不得大于 32 个字符，而通常是前 8 个字符有效。例如，a、_a、a12 等都是标识符。但是 1、1a、。a、&a1 等都不是合法的标识符。标识符的命名除了满足字符组合方面的原则外，还要遵循下述原则。

1. 一致性

同一个模块内部的标识符命名要一致。例如，如果规定变量的首字母大写，用全部大写表示常量，那么整个模块内都应该这么写。

2. 准确性

用词要准确，可以望文生义，避免概念模糊或形式相近的标识符。例如，定义 Total 表示合计比随意用一个变量来表示要明确得多。MyFun、temp 等模糊概念的变量也要避免。

3. 长度短，信息多

在保持准确性的前提下，要力争长度短、信息多，即用最短数目的字符数表示尽可能多的信息。例如，用 Total 表示合计，而不用 TotalOfNumbers。

上机实训

练习使用 Visual C++ 6.0 开发环境，仿照 1.2.4 节的步骤，参照 1.3.1 节上机实现下述程序，并观察其运行情况。

要求：提示使用者从命令行输入 3 个数，然后按与输入相反的顺序输出。

代码如下所示：

```
#include <iostream.h>
int main()
{
    int x;
    int y;
    int z;
    cout<<"输入第 1 个数:";
    cin>>x;
    cout<<"输入第 2 个数:";
    cin>>y;
```

```
    cout<<"输入第 3 个数:";
    cin>>z;
    cout<<z<<end;
    cout<<y<<endl;
    cout<<x<<endl;
    return 0;
}
```

本章小结

本章主要讲述了C++的一些基本知识,以及编程中需要注意的事项。本书的代码都是在 Visual C++ 6.0 开发环境下完成的,因此着重讲解了 Visual C++ 6.0 开发环境及其使用,并给出了一个简单的例子。另外,本章对于C++程序的基本结构和编码中应该注意的规范也给出了说明。标识符是编程中必须要用到的,本章也作了一定的说明。

本章习题

1. C++中.h 文件和.cpp 文件有什么用处?
2. 程序中必须有的函数是什么?
3. C++有几种注释方式,分别表示什么意思?
4. 写出 5 个合法的变量名称。
5. 下面的变量名称是否合法? 为什么?

X,x,_,123,_123,a*,char,123a,a b,1aaa,const,int

6. 什么是保留字? 请举例说明。