21世纪高等教育计算机规划教材

COMPUTER

C++ 程序设计

C++ Programming

刘艳菊 主编

迟立颖 张凌宇 陈淑鑫 副主编

堵秀凤 主审

- 总结多年教学经验及常见问题
- 结构清晰，内容由浅入深，重点突出
- 配有大量习题和答案，锻炼学生实际应用能力

人民邮电出版社

POSTS & TELECOM PRESS

21世纪高等教育计算机规划教材

COMPUTER

C++ 程序设计

C++ Programming

刘艳菊 主编
迟立颖 张凌宇 陈淑鑫 副主编
堵秀凤 主审

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

C++程序设计 / 刘艳菊主编. -- 北京 : 人民邮电出版社, 2013.2
21世纪高等教育计算机规划教材
ISBN 978-7-115-30779-8

Ⅰ. ①C… Ⅱ. ①刘… 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013458号

内 容 提 要

本书全面、系统地介绍了 C++程序设计的基本概念、语法和程序设计方法，详细地讲解了 C++程序设计中的数据类型、基本控制语句、数组、函数、指针、类和对象的定义与应用、继承和派生、多态性、虚函数、输入/输出流、模板等内容。本书重点突出、内容精练；便于自学，读者可以边学边练，快速掌握知识点。

本书不仅可以作为高等学校“C++程序设计”课程的教材，也可供自学 C++程序设计的人士参考。

21 世纪高等教育计算机规划教材

C++程序设计

◆ 主　　编　刘艳菊
副 主 编　迟立颖　张凌宇　陈淑鑫
主　　审　堵秀凤
责任编辑　许金霞

◆ 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
北京艺辉印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：19.75　　2013 年 2 月第 1 版
字数：517 千字　　2013 年 2 月北京第 1 次印刷

ISBN 978-7-115-30779-8

定价：38.00 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

C++广泛用于开发应用程序和解决实际问题，支持面向对象程序设计，同时也支持面向过程程序设计，因此功能非常强大，对于没有程序设计语言基础的学生来说有一定的学习困难。本书是作者结合多年的教学经验，在总结教学过程中学生遇到的各种问题的基础上编写而成。全书各章节结构清晰，内容由浅入深，重点突出，适合初学者使用，同时每章配有大量习题和相应的习题答案，使学生能在实际应用中掌握C++程序设计方法。

本书系统地讲述了C++语言的基本概念和编程方法，全书共有12章。第1章主要讲述面向对象程序设计的基本思想、面向对象的概念、C++程序的基本结构以及Visual C++集成环境的使用。第2章主要讲述数据类型、运算符和表达式，是C++程序设计的基础。第3章主要讲述程序设计的基本框架，包括顺序结构、选择结构和循环结构，经过本章的学习可以编程解决一些简单的问题。第4章为数组，是C++中的一个简单的构造结构的数据类型，包括一维数组、多维数组、字符数组和字符串的定义及使用方法。第5章为函数，C++程序就是由若干个函数组成的，本章主要讲述了预处理、函数的定义、函数间参数的传递以及几种特殊的函数（内联函数、函数重载等）。第6章为指针，主要讲述利用指针变量访问各种数据类型的方法以及指针在函数中的参数传递。前面六章主要讲述的是面向过程的知识。第7章讲述面向对象的知识，介绍类和对象的定义和使用方法。第8章为类和对象的更高层次的应用，主要介绍了类和指针、类和数组的关系及使用方法，通过例题的应用对类和对象深入学习。第9章为类的继承和派生，讲述了类的继承和派生的概念和应用。第10章为多态和虚函数，由例题引入多态的概念和虚函数的应用。第11章为C++的I/O流类库，介绍了C++中数据流的概念，与外部设备的输入输出操作。第12章为模板，介绍了函数模板和类模板的概念及简单应用。

本书中的所有例题和习题都在Visual C++ 6.0环境下调试运行通过，对掌握C++程序设计的基本方法和编程应用有很大帮助。

全书由刘艳菊、迟立颖、张凌宇和陈淑鑫编写。其中，刘艳菊编写了第4章、第5章，第7章～第9章以及附录Ⅰ和附录Ⅱ，迟立颖编写了第3章、第10章、第11章，张凌宇编写了第1章、第2章，陈淑鑫编写了第6章和第12章。全书由刘艳菊统稿，堵秀凤教授审阅。

由于编者水平有限，书中难免有疏漏之处，敬请广大读者批评指正。

编　者

2013年1月

目　录

第 1 章　程序设计语言 Visual C++概述 …… 1

1.1　程序设计的思想 …… 1

1.1.1　结构化程序设计 …… 1

1.1.2　面向对象程序设计的基本思想 …… 2

1.1.3　面向对象程序设计方法的应用 …… 3

1.2　C++的词汇 …… 4

1.2.1　字符集 …… 4

1.2.2　标识符 …… 4

1.2.3　关键字 …… 4

1.3　C++程序的构成 …… 5

1.3.1　C++程序的基本框架 …… 5

1.3.2　C++程序设计风格 …… 7

1.4　C++程序实现与编程 …… 7

1.4.1　输入并编译一个新的 C++程序 …… 8

1.4.2　C++程序的开发过程 …… 10

习题一 …… 12

第 2 章　数据类型及表达式 …… 14

2.1　基本数据类型 …… 14

2.1.1　整型（int） …… 15

2.1.2　浮点型数据(float) …… 16

2.1.3　字符型数据(char) …… 17

2.1.4　布尔型(bool) …… 17

2.1.5　空值型(void) …… 17

2.2　常量 …… 18

2.2.1　数值常量 …… 18

2.2.2　字符常量 …… 20

2.2.3　符号常量 …… 23

2.2.4　逻辑常量 …… 24

2.3　变量 …… 24

2.3.1　变量定义 …… 24

2.3.2　变量的存储属性 …… 25

2.3.3　常变量 …… 25

2.4　运算符 …… 26

2.4.1　算术运算符 …… 26

2.4.2　关系运算符 …… 29

2.4.3　逻辑运算符 …… 29

2.4.4　位运算符 …… 30

2.4.5　赋值运算符 …… 30

2.4.6　其他运算符 …… 31

2.5　表达式 …… 34

2.5.1　表达式中的优先级和结合性 …… 35

2.5.2　表达式的种类 …… 35

2.5.3　表达式中的类型转换 …… 37

2.6　复合数据类型 …… 38

2.6.1　枚举类型 …… 38

2.6.2　结构类型 …… 39

2.6.3　联合类型 …… 39

2.6.4　用户自定义类型 …… 39

习题二 …… 40

第 3 章　程序结构 …… 42

3.1　顺序结构 …… 42

3.1.1　声明语句 …… 42

3.1.2　表达式语句 …… 42

3.1.3　空语句和复合语句 …… 43

3.1.4　基本输入/输出语句 …… 43

3.2　选择结构 …… 44

3.2.1　if 语句 …… 44

3.2.2 switch 语句 …… 49
3.3 循环结构 …… 51
3.3.1 while 循环语句 …… 51
3.3.2 do…while 循环语句 …… 52
3.3.3 for 循环语句 …… 53
3.3.4 循环嵌套 …… 55
3.3.5 转向语句 …… 58
习题三 …… 61

第 4 章 数组 …… 72
4.1 数组概述 …… 72
4.2 一维数组 …… 72
4.2.1 一维数组的定义 …… 73
4.2.2 一维数组的应用 …… 74
4.3 二维数组 …… 76
4.3.1 二维数组的定义 …… 76
4.3.2 二维数组的应用 …… 77
4.4 字符数组 …… 79
4.4.1 字符数组的定义 …… 79
4.4.2 字符数组的初始化 …… 80
4.4.3 字符串 …… 80
4.4.4 字符串处理函数 …… 81
4.4.5 字符数组的应用 …… 83
习题四 …… 84

第 5 章 预处理、函数和作用域 …… 90
5.1 编译预处理 …… 90
5.1.1 宏定义 …… 90
5.1.2 文件包含 …… 93
5.1.3 条件编译 …… 94
5.2 函数的定义与声明 …… 95
5.2.1 函数的定义 …… 96
5.2.2 函数的调用 …… 96
5.2.3 函数的声明 …… 98
5.3 函数参数的传递 …… 99
5.3.1 值传递 …… 99
5.3.2 地址传递 …… 101
5.3.3 引用传递 …… 103
5.4 数组作为函数参数 …… 104
5.4.1 形参和实参均为数组名 …… 104
5.4.2 形参和实参均为对应数组指针 …… 104
5.4.3 实参用数组名形参用引用 …… 105
5.5 函数的嵌套调用和递归调用 …… 106
5.5.1 函数的嵌套调用 …… 107
5.5.2 函数的递归调用 …… 108
5.6 内联函数 …… 111
5.7 函数重载 …… 112
5.8 作用域 …… 113
5.8.1 作用域的类别 …… 114
5.8.2 全局变量与局部变量 …… 115
5.8.3 变量的存储类别 …… 115
5.8.4 内部函数与外部函数 …… 117
5.9 C++的系统函数 …… 118
习题五 …… 119

第 6 章 指针 …… 124
6.1 内存空间的访问方式 …… 124
6.2 指针类型 …… 125
6.3 引用类型 …… 126
6.4 指针运算 …… 129
6.4.1 指针赋值运算 …… 129
6.4.2 指针传值传址交换运算 …… 130
6.4.3 指针加减运算 …… 132
6.4.4 指针比较 …… 133
6.5 指针与数组 …… 134
6.5.1 用指针访问数组 …… 134
6.5.2 指针与字符串 …… 137
6.6 指针与函数 …… 138
6.6.1 函数指针 …… 138
6.6.2 指针用做函数参数 …… 139

6.6.3　指针函数……140
6.7　C++语言的动态存储分配……141
习题六……142

第 7 章　类和对象的基础……146
7.1　类与对象的概念……146
7.1.1　对象的概念……146
7.1.2　类的概念……146
7.2　类与对象的定义……147
7.2.1　类的定义……147
7.2.2　成员函数的定义……148
7.2.3　对象的定义……149
7.2.4　访问类成员……149
7.3　成员函数的特性……151
7.3.1　内联函数和外联函数……151
7.3.2　成员函数重载……152
7.3.3　设置参数的缺省值……153
7.4　构造函数和析构函数……154
7.4.1　构造函数……154
7.4.2　析构函数……155
7.4.3　复制构造函数……156
7.5　静态成员……158
7.5.1　静态数据成员……158
7.5.2　静态成员函数……159
7.6　友元……160
7.6.1　友元函数……160
7.6.2　友元类……161
7.7　类的作用域与对象的生存期……163
7.7.1　类的作用域……163
7.7.2　对象的生存期……164
习题七……166

第 8 章　类和对象的应用……173
8.1　类和指针……173
8.1.1　指向类对象的指针……173
8.1.2　指向类的成员的指针……174
8.1.3　this 指针……175
8.2　类和数组……176
8.2.1　对象数组……176
8.2.2　对象指针数组……178
8.2.3　指向对象数组的指针……179
8.3　常类型……180
8.3.1　常对象……180
8.3.2　常成员函数……180
8.3.3　常数据成员……181
8.4　子对象与堆对象……182
8.4.1　子对象……183
8.4.2　堆对象……184
习题八……186

第 9 章　继承性和派生类……190
9.1　基类和派生类……190
9.1.1　继承的概念……190
9.1.2　派生类的定义格式……191
9.1.3　派生类的继承特性……192
9.2　派生的构造函数和析构函数……195
9.2.1　单继承派生类的构造函数和析构函数……195
9.2.2　多继承派生类的构造函数和析构函数……197
9.3　虚基类……199
9.3.1　多重继承中可能存在的二义性问题……199
9.3.2　虚基类的定义……200
9.3.3　虚基类的构造函数……201
习题九……203

第 10 章　多态性和虚函数……207
10.1　函数重载……207
10.2　运算符重载……208

10.2.1 运算符重载的必要性……208
10.2.2 运算符重载的规则……209
10.2.3 运算符重载的两种形式……210
10.2.4 运算符重载的实例……214
10.3 静态联编和动态联编……219
10.3.1 静态联编……219
10.3.2 动态联编……220
10.4 虚函数……220
10.5 纯虚函数和抽象类……223
10.5.1 纯虚函数……223
10.5.2 抽象类……225
10.6 虚析构函数……230
习题十……232

第 11 章 C++的 I/O 流类库……238
11.1 流类库……238
11.2 标准输出……240
11.2.1 预定义的插入符……241
11.2.2 put()成员函数……242
11.2.3 write()成员函数……242
11.3 标准输入……243
11.3.1 预定义提取符……243
11.3.2 get()成员函数……244
11.3.3 read()成员函数……246
11.4 格式控制 I/O 操作……247
11.4.1 流的格式化标志……248
11.4.2 格式化输出函数……250
11.4.3 操作子……250
11.5 插入符和提取符的重载……252
11.6 字符串流……255
11.6.1 ostrstream 类的构造函数……255
11.6.2 istrstream 类的构造函数……256
11.7 磁盘文件的 I/O 操作……256
11.7.1 磁盘文件的打开和关闭……257
11.7.2 文件流状态的判别……259
11.7.3 文本文件的读和写……261
11.7.4 二进制文件的读和写……263
11.7.5 随机文件的读和写……265
习题十一……269

第 12 章 模板……274
12.1 函数模板……274
12.1.1 函数模板……274
12.1.2 模板函数……275
12.1.3 模板实参的省略……276
12.2 类模板……277
12.2.1 类模板……277
12.2.2 模板类……279
12.2.3 类模板的继承与派生……280
习题十二……281

习题答案……284
习题一……284
习题二……284
习题三……285
习题四……288
习题五……289
习题六……290
习题七……290
习题八……292
习题九……292
习题十……293
习题十一……297
习题十二……301

附录 I 常用系统函数……302
附录 II ASCII 表……305
参考文献……307

第 1 章 程序设计语言 Visual C++概述

计算机能完成预定的任务是硬件和软件协同工作的结果，而计算机的这种根据不同要求完成不同功能的“智能”，主要是由于计算机软件的“可编程”。也就是说，同样的硬件配置，安装不同的软件就能完成不同的工作。程序设计是一门技术，需要相应的理论、方法和工具来支持。程序设计就是用计算机语言编写一些代码来驱动计算机完成特定的功能，也就是用计算机能理解的语言来告诉计算机做什么。一般情况下，这个过程包括：问题描述、算法设计、代码编制和调试运行。整个开发过程都需要编制文档，以便管理。

C++语言是在 C 语言的基础上建立起来的，它包括了 C 语言的全部语言成分，同时又添加了尤为重要的面向对象编程的完全支持。它既可进行过程化程序设计，也可进行面向对象程序设计。C++语言具备了许多 C 语言不支持的新功能和新特性，是目前编程人员使用最广泛的语言工具。本章主要讲解面向对象程序设计的特点，C++程序的结构、语法，C++编程的基本方法和上机实践操作步骤。

1.1 程序设计的思想

早期的程序设计语言是面向数值计算的，程序规模通常较小。随着计算机硬件技术的发展，其速度和存储容量不断提高，成本急剧下降，但要解决的问题却越来越复杂，程序规模也越来越大。这样的程序必须由多个程序员密切合作才能完成。由于旧的程序设计方法很少考虑程序员间相互合作的需要，因此编写程序的错误随着软件规模的增加而迅速增加，造成调试时间和成本迅速上升，甚至使得某些软件产品因调试成本过高而报废，产生了通常所说的“软件危机”。结构化程序设计的方法就是在这种背景下产生的。

1.1.1 结构化程序设计

随着计算机硬件性能的提高，程序设计的目标不应再集中于如何充分发挥硬件的效率。新的程序设计方法应以能设计出结构清晰、可读性强、易于分工合作编制和调试的程序为基本目标。结构化程序设计思想认为，好的程序应具有层次化的结构，应该采用“逐步求精”的方法，只使用顺序、分支、循环等基本程序结构，通过组合、嵌套来编写。

程序一般由若干子程序构成，而子程序又是由语句构成的。对于程序员来说，程序设计工作的一个主要内容，就是如何将解决问题的算法，用某种语言，按照一定的结构编写成语句和子程序来完成。结构化设计方法是以模块化为中心，将待开发的软件系统划分为若干个相互独立的模

块，这样就使完成每一个模块的工作变得单纯而明确，为设计一些较大的软件打下了良好的基础。由于模块间相互独立，所以在设计一个模块时，不会受到其他模块的干扰，因而可将一个复杂的大问题分解为若干个简单的小问题来处理，即编写一系列简单的小模块。模块的独立性还为扩充已有的系统、建立新系统带来了不少方便。按照结构化程序设计方法设计出的程序具有结构清晰、可读性好、易于修改、易于扩充、容易调试的优点。结构化程序设计包括 3 种结构：顺序结构、选择结构和循环结构。详细内容将在本书第 3 章学习。

结构化程序设计的基本思想是采用“自顶向下、逐步求精”的程序设计方法和“单入单出”的控制结构。具有结构化特点的程序，实际上是由一些具有相对独立功能、结构清晰、容易理解的小程序模块串联起来的顺序结构。在进行具体的程序设计时，可以将这些相对独立的小程序用函数编程手段定义成“模块”，即将程序模块化。程序模块化的优点如下。

（1）便于将复杂的问题转化为个别的小问题，从而容易实现“各个击破”。

（2）便于从抽象到具体地进行程序设计。对问题采用模块化解法时，可以提出许多抽象的层次。在抽象的最高层，使用自然语言来描述；在抽象的较低层，则采用比较具体的方法来描述；最后，在抽象的最低层可以用直接编程的方式实现。

（3）便于测试和维护。采用模块化原则设计程序时，为了得到一组最佳模块，应当遵循信息隐蔽的原则分解软件。即某个模块所包含的信息（函数和数据）其他模块不需要知道，即不能访问，以体现模块的独立性。“隐蔽”意味着模块化可以通过定义一组独立的模块来实现，这些独立的模块彼此之间仅仅交换那些为了完成系统功能所必需的信息。在测试和以后的维护期间，当需要软件进行修改时，如果某一模块之间的接口不变，每个模块内部的具体细节可以任意修改。由于疏忽而引起的错误传播到其他部分的可能性很小。

（4）便于理解分析程序。在对模块化程序进行分析时，由于每个模块功能明确，彼此独立，所以可以采用自底向上的分析方法，首先确定每个模块的功能，进而完成整个程序。

1.1.2 面向对象程序设计的基本思想

在面向对象的程序设计技术（Object-Oriented Programming，OOP）出现前，程序员们一般采用面向过程的程序设计方法（Process-Oriented Programming，POP）。面向过程的程序设计方法采用函数来完成对数据结构的操作，但又将函数和所操作的数据结构分离开来。但函数和它所操作的数据是密切相关的，特定的函数往往对特定的数据结构进行操作；如果数据结构发生改变，则相应的函数也要发生变化。这样就使得面向过程的程序设计方法编写出来的大程序不但难于编写，而且也难于调试、修改和维护。

面向对象的程序设计是一种重要的程序设计方法，它能够有效地改进结构化程序设计中存在的问题。在结构化的程序设计中，要解决某个问题，就要确定这个问题能够分解为哪些函数，数据能够分解为哪些基本的类型，即思考方式是面向机器实现的，不是面向问题的结构，需要在问题结构和机器实现之间建立联系。面向对象的程序设计方法的思考方式是面向问题的结构，它认为现实世界是由对象组成的，而问题求解的方法与现实世界是对应的。因此，采用面向对象的程序设计方法来解决某个问题，则需先确定此问题由哪些对象组成，这些对象之间如何相互作用。

面向对象程序设计是通过对数据和代码建立分块的内存区域，对程序进行模块化的一种程序设计方法。这些模块被用作样板——类，将其实例化成对象。面向对象的程序设计有 3 个特征，即封装、继承和多态。因此，面向对象程序设计方法要求语言必须具备抽象、封装、继承、多态性等关键要素。

封装性是指将数据和算法捆绑成一个整体，这个整体就是对象，描述对象的数据被封装在其内部，若需要存取数据可通过对象提供的算法来进行操作，而无需知道对象内部的数据是如何表示和存储的。这种思想被称为信息隐藏。例如，手机的使用者无需知道手机内部电路的具体工作原理及结构，就可以用它来接听电话，封装性和数据隐藏从根本上解决了结构化程序设计中数据和算法一致性差的问题。

继承性是指一种事物保留了另一种事物的全部特征，并且具有自身的独有特征，继承改变了过去传统的非面向对象程序设计中对不再适合要求的用户自定义数据类型进行改写甚至重写的方法，克服了传统程序设计方法对编写出来的程序无法重复使用而造成资源浪费的缺点。例如，赛车继承了汽车所具有的属性，赛车又具备其独有的特征。

多态性是指当多种事物继承来自一种事物时，同一种操作在它们之间表现出不同的行为，若在一个使用面向对象思想编写的绘图程序中可能含有四种类型的对象，分别表示抽象概念——形状、具体概念——三角形、矩形、圆形对象都继承了形状对象的全部特征。

本书各章将详细介绍面向对象的程序设计方法。

简单地说，面向对象的程序设计方法可以分成以下 4 个步骤。

（1）找出问题中的对象和类。

（2）确定每个对象和类的功能，具有哪些属性，提供哪些方法等。

（3）找出这些对象和类之间的关系，确定对象之间的消息通信方式、类之间的继承和复合等关系。

（4）用程序代码实现这些对象和类。

可见，面向对象程序设计是将问题抽象成若干类，将数据与对数据的操作封装在一起，各个类之间可能存在着继承关系，对象是类的实例，程序是由对象组成的，通过对象之间相互传递消息、进行消息响应和处理来完成功能。面向对象的程序设计可以较好地克服结构化程序设计存在的问题，可以开发出健壮、易于扩展和维护的应用程序。

1.1.3　面向对象程序设计方法的应用

C++和 C 语言均诞生于贝尔实验室，Bjarne Stroustrup 博士在 C 语言中引入了面向对象的思想，并将这种语言命名为 C++。C++是 C 语言的扩展，是 C 语言的一个超集，根据 Stroustrup 博士自己的说法，C++是一个“更好的 C 语言”。它是一种混合型的语言，既支持传统的结构化程序设计，又支持面向对象程序设计。

C++设计的初衷是为了扩充 C 语言并引入面向对象程序设计思想。C 语言虽然有其强大的功能，但作为一种结构化编程语言，当程序量相当大时，其局限性不可避免地暴露出来。而 C++以其对面向对象程序设计方法的支持，成为设计和开发大规模软件的强有力工具。同时，C++在设计时充分考虑了与 C 语言的兼容性，使用大量基于 C 语言的开发工作得以继承和发展，许多 C 语言的编写代码不需修改就可为 C++语言所用，而且原来用 C 语言编写的众多库函数和实用程序也可以用于 C++语言中。C++和 C 的主要区别是 C++对数据抽象和面向对象程序设计方法的支持。C++允许数据抽象，支持封装、继承、多态等特征。C 语言程序的设计一般采用自上而下、逐步求精的方式进行软件开发，而 C++则是兼有自下而上和自上而下两种方式。C++语言支持几乎所有的面向对象程序设计特征，具备开发领域的新思想和新技术主要包括：

（1）抽象数据类型；

（2）封装和信息隐藏；

（3）以继承和派生方式实现程序的重用；

（4）以运算符重载和虚函数来实现多态性；

（5）以模板来实现类型的参数化。

目前，C++语言已被广泛用于程序设计的众多领域。实践证明，C++尤其适用于大、中型软件的开发。

1.2 C++的词汇

基本符号本身一般没有什么含义，而由它们按照一定规则组合成的单词却能表达出某种语义。C++程序是一个字符序列，能够出现在程序中的字符是有限的，即 C++程序是由一定字符集的字符构成的。

1.2.1 字符集

字符是一些可区别的符号，C++语言规定任何 C++程序只能由下列字符构成。

（1）大小写的英文字母：A～Z、a～z 共 52 个符号。

（2）数字字符：0~9 共 10 个符号。

（3）特殊字符：! ? # %^& *_ + = - ～ < > / \ " : ; . , ( ) [ ] { }及空格（space）共 28 个符号。

以上三类符号共计 92 个，它们组成了 C++语言的基本符号集合。

1.2.2 标识符

在高级程序设计语言中，用来标识变量、符号常量、函数、数组、类型等实体名字的有效字符序列称为标识符（identifier）。标识符就是一个名字，是用户为程序中各种需要命名的“元素”所起的名字。不同的语言，对于标识符的组成形式的规定也各不相同。C++标识符的定义很灵活，但作为标识符必须满足以下规则。

（1）所有标识符必须由一个字母 a～z、A～Z 或下划线_开头。

（2）标识符的其他部分可以用字母、下划线或数字 0~9 组成。

（3）标识符中大小写字母是区分的，表示不同的符号。

（4）标识符不宜过长，C++语言不限制标识符的长度，多数编译器只有前 32 个字符有效，若程序中的标识符超过了这个长度，超出的部分被忽略不计。

（5）标识符不能与任意一个关键字同名。尽量使用有含义的单词作为标识符，用户定义的标识符不能采用系统的保留字，保留字是指系统已预定义的标识符，包含关键字和设备字等。

下面列出的是合法的标识符：

Hello _key_1 M_max BASIC _TEST_S

下面是不合法的标识符：

0123 5small key? m.max C++ U.S.A #sys

1.2.3 关键字

关键字是C++语言预定义的具有特殊意义的标识符，是由C++语言本身预先定义好的一类单词。

程序员不能随意使用关键字，只能按照 C++语言为它们定义的意义来使用，所以关键字也称为保留字。有些关键字代表计算机的动作，有的表示语言预定义的某种数据类型，有的用于标识某个程序段。表 1-1 所示为 ANSI C 标准规定的 32 个关键字，表 1.2 所示为 ANSI C++标准补充的 29 个关键字。

表 1.1　　ANSI C 标准规定的关键字

auto	break	case	char	const	continue	default	do
double	else	enum	extern	float	for	goto	if
int	long	register	return	short	signed	sizeof	static
struct	switch	typedef	union	unsigned	void	volatile	while

表 1.2　　ANSI C++标准补充的关键字

bool	catch	class	const_cast	delete	dynamic_cast	explicit
false	friend	inline	mutable	namespace	new	operator
private	protected	public	reinterpret_case	static_cast	template	this
throw	true	try	typeid	typename	using	virtual
wchar_t						

每个关键字在 C++语言中都具有特殊的含义，并实现一定的功能。所以不能将上述关键字再当作其他类型的单词使用。

1.3　C++程序的构成

Visual C++是 Microsoft 公司推出的目前使用极为广泛的基于 Windows 平台的可视化编程软件。Visual C++ 6.0 是在以往版本不断更新的基础上形成的，由于其功能强大、灵活性好、完全可扩展以及具有强有力的 Internet 支持，在各种 C++语言开发工具中脱颖而出，成为目前最为流行的 C++语言集成开发环境。

C++是一个复杂的程序设计语言。为帮助读者养成良好习惯，编写更清晰、更易读易懂、更易维护、更易测试和调试的代码，本书从编写第一个 C++程序设计开始，就应该遵循一定的程序设计风格，通过长期的训练和积累，使得良好的程序设计风格成为习惯。

1.3.1　C++程序的基本框架

一个 C++程序可以由一个程序单位或多个程序单位构成。每个程序单位作为一个文件。在程序编译时，编译系统分别对各个文件进行编译，因此，一个文件是一个编译单元。在一个程序单元中，可以包括以下几个部分。

（1）预处理命令。

（2）全局声明部分（在函数外的声明部分）。在这部分中包括对用户自己定义的数据类型的声明和程序中所用到的变量的定义。

（3）函数。函数是实现操作的部分，因此函数是程序中必须有的和最基本的组成部分。每一个程序必须包括一个或多个函数，其中必须有一个（而且只能有一个）主函数（main 函数）。

（4）类（class）是 C++新增加的重要的数据类型，是 C++对 C 的最重要的发展。有了类，就可以实现面向对象程序设计方法中的封装、继承、派生、多态等功能。在一个类中可以包括数据

成员和成员函数，它们可以被指定为私有的（private）、公有的（public）属性和保护的（protected）。私有的数据成员和成员函数只能被本类的成员函数所调用。

一个函数由两部分组成。

（1）函数首部，即函数的第一行。包括函数名、函数类型、函数属性、函数参数（形参）名、参数类型。一个函数名后面必须跟一对圆括号，括号中的函数参数可以缺省。

如：int max(int x,int y)

（2）函数体，即函数首部下面的大括号内的部分。如果在一个函数中有多个大括号，则最外层的一对{ }为函数体的范围。

函数体一般包括如下几个部分。

（1）局部声明部分（在函数内的声明部分），包括对本函数中所用到的类型、函数的声明和变量的定义。对数据的声明既可以放在函数之外（其作用范围是全局的），也可以放在函数内（其作用范围是局部的，只在本函数内有效）。

（2）执行部分，由若干个执行语句组成，用来进行有关的操作，以实现函数的功能。

当然，在某些情况下也可以没有声明部分。甚至可以既无声明部分，也无执行部分。如 void demp( ){ }它是一个空函数，什么也不干，但这是合法的。

语句包括两类：一类是声明语句，如“int a,b;”，用来向编译系统通知某些信息（如类型、函数和变量的声明或定义），但它并不引起实际的操作，是非执行语句；另一类是执行语句，用来实现用户指定的操作。C++对每一种语句赋予一种特定的功能。语句是实现操作的基本成分，显然，没有语句的函数是没有意义的。C++语句必须以分号结束，如“c=a+b;”，分号是语句的一个组成部分，注意没有分号的就不是语句。

一个 C++程序总是从 main 函数开始执行的，而不论 main 函数在整个程序中的位置如何，main 函数可以放在程序文件的最前头，也可以放在程序文件的最后，或在一些函数之前，在另一些函数之后。

【例 1.1】 C++一个最简单的输出语句程序。

```
//一个最简单的C++程序
#include <iostream.h>
void main( )
{
    cout<<"Welcom to Visual C++.\n";
}
```

此程序的运行结果是在屏幕上显示：

```
Welcom to Visual C++.
```

上述程序虽然只有 6 行，但却包含了每一个 C++程序都具备的几个基本组成部分。下面详细介绍该程序：

第 1 行是注释语句。该注释语句用来说明整个程序的功能。注释用来对程序进行注解和说明。注释不是程序的有效部分，对程序不起任何作用，但给程序添加合适的注释是一种良好的程序设计风格，注释被视为程序的一个重要组成部分。在 C++程序中，注释有两种方式：一是以“//”开头，表示该行的后续部分都是注释；另一种即 C 语言原有的注释方式，以“/*”开头，以*/结尾，二者之间的所有字符都是注释。

第 2 行语句是一条预处理指令，这不是 C++的语句，它以“#”开头与 C++语句相区别，行的末尾没有分号。#include<iostream.h>是一个“包含命令”，它的作用是将文件 iostream 的内容包

含到该命令所在的程序文件中，代替该命令行。文件 iostream 的作用是向程序提供输入或输出时所需要的一些信息。iostream 是 i-o-stream 3 个词的组合，它告诉预处理器要在程序中包括“输入输出流”的头文件 iostream.h 内容，“.h”称为“头文件”（head file）。该文件包含了与输入输出流相关的类，类型和函数的说明。如果程序中要使用这些输入输出流功能，必须包括这个文件。在程序进行编译时，先对所有的预处理命令进行处理，将头文件的具体内容代替#include 命令行，然后再对该程序单元进行整体编译。

第 3 行语句是每个 C++程序都包含的主函数语句。main 后面的括号表示它是一个函数。C++程序由一个或多个函数组成，其中有且只有一个 main 函数。即使 main 不是程序中的第一个函数，C++程序都是从函数 main 开始执行的。main 左边的关键字 void 表示函数无返回值。

第 4 行的左花括号{与第 6 行的右花括号}表示函数的开头和结尾。注意 C++所有语句最后都应当有一个分号。

第 5 行语句是一个输出语句。cout 是由 c 和 out 两个单词组成，是 C++用于输出的语句。cout 实际上是 C++系统定义的对象名，称为输出流对象。用 cout 和“<<”实现输出的语句简称为 cout 语句，“<<”是“插入运算符”，与 cout 配合使用，在本例中它的作用是将运算符“<<”右侧双撇号内的字符串" Welcom to Visual C++"插入输出的队列 cout（输出的队列也称作“输出流”），C++系统将输出流 cout 的内容输出到系统指定的设备（通常为显示器）中。

C++程序书写格式自由，一行内可以写几个语句，一个语句可以分写在多行上。C++程序没有行号，也不像 FORTRAN 或 COBOL 那样严格规定书写格式（语句必须从某一列开始书写）。一个好的、有使用价值、可读性强的源程序都应当加上必要的注释。

1.3.2　C++程序设计风格

所谓程序设计风格，是指借助于好的设计方法编写结构好的程序。在编写源程序时，往往需要采用各种措施来提高程序的可读性、可理解性和可修改性，以利于程序的查错、测试、维护修改及交流。

（1）以简洁明了的方式编写 C++程序是一种良好的程序设计习惯。通常称这种编写程序的方法为 KIS（keep it simple，保持简洁）。

（2）缩排规则。所谓缩排规则就是使程序的书写格式应能较好地反映出该程序的层次结构。例如，处于同一层的语句都从同一个字符位置开始书写；将每个函数的整个函数体在定义函数体的花括号中缩排一级，可使程序中的函数结构更明显，使程序更易读。

（3）标识符选择能够反映相关功能和特征的单词来命名，以便于见名知意。

（4）注释是增加可读性、可理解性的常用措施。注释虽然不是程序的有效部分，但是，给程序添加合适的注释是一种良好的程序设计风格，注释应该视为程序的一个重要组成部分。

（5）注意大小写英文字母，C++语言中是区分英文字母的大小写的。

（6）输出信息。利用 C++的输出语句尽量将要输出的信息组织得直观清晰、布局合理，使之形象化、表格化、页面化、自动成文，便于使用者看懂和存档。

1.4　C++程序实现与编程

C++的流行使得许多软件厂商都提供了自己的 C++集成开发环境，称为 C++ IDE，著名的有

Borland 公司的 C++ Builder，IBM 公司的 Visual Age For C++，Microsoft 公司的 Visual C++等。其中，Visual C++ 6.0 是当今 Windows 操作系统下最流行的 C++集成开发环境之一。Visual C++ 6.0 分为标准版、专业版和企业版 3 种，但其基本功能是相同的。为统一起见，本书称为 Visual C++ 6.0，如图 1.1 所示，并将其置于 Windows XP 操作系统中。本书的程序实例均用 Visual C++ 6.0 调试通过。

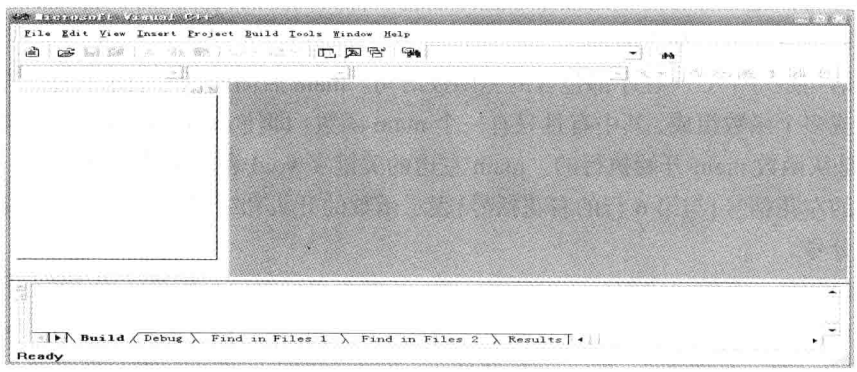

图 1.1　Visual C++ 6.0 编译环境

1.4.1　输入并编译一个新的 C++程序

在 Visual C++ 6.0 中，用应用程序向导创建和编连一个控制台应用程序，可按下列步骤操作。

1. 启动 Windows XP 操作系统

打开计算机，启动 Windows XP 操作系统。

2. 启动 Visual C++ 6.0

选择“开始”→“程序”→Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0，运行 Visual C++ 6.0。第一次运行时，将显示“当时的提示”对话框。单击“下一个提示”按钮，可看到有关各种操作的提示。如果取消选中“再启动时显示提示”复选框，那么下一次运行 Visual C++ 6.0，将不再出现此对话框，如图 1.2 所示。

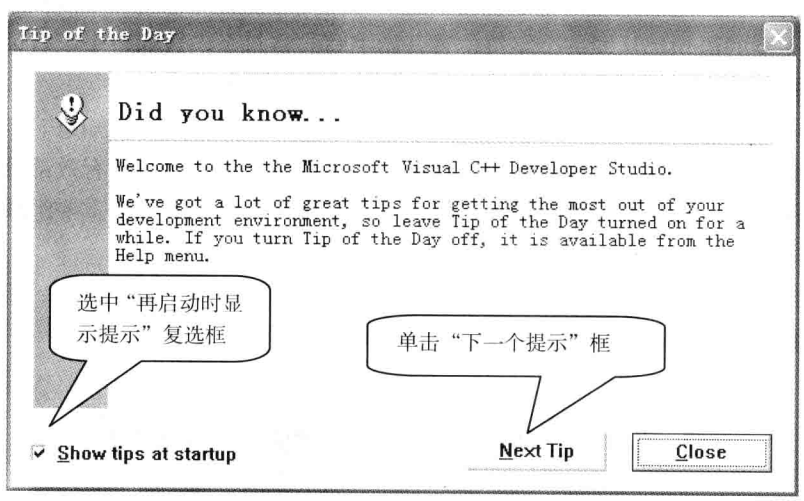

图 1.2　“当时的提示”对话框

3. 开发环境界面

开发环境界面由标题栏、菜单栏、工具栏、项目工作区窗口、文档窗口、输出窗口、状态栏等组成，如图 1.3 所示。

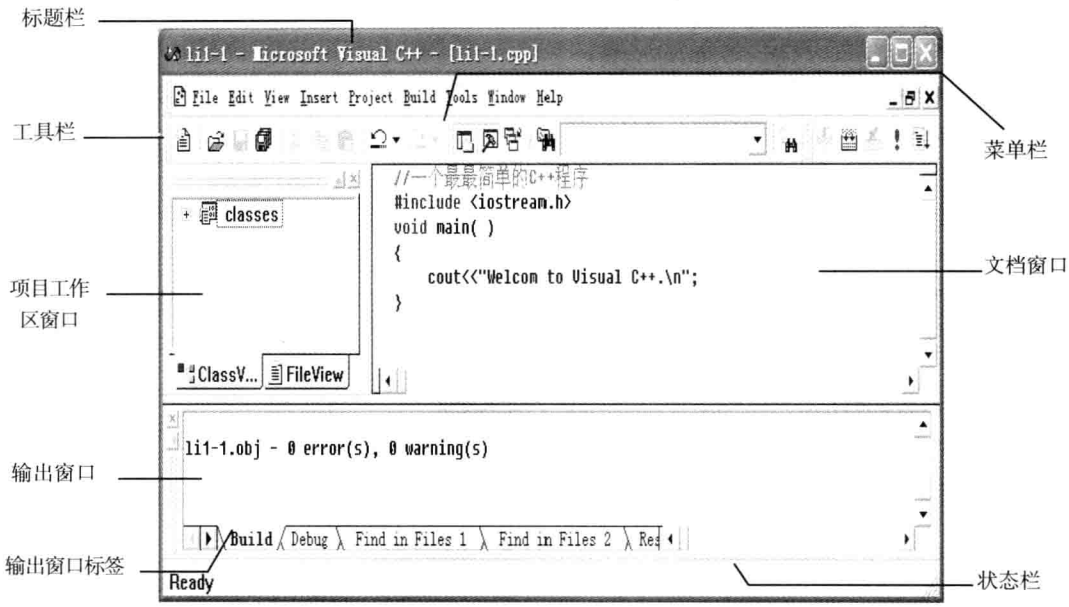

图 1.3　Visual C++ 6.0 开发环境

标题栏一般有“最小化”、“最大化”或“还原”、“关闭”按钮，单击“关闭”按钮将退出开发环境。标题栏上还显示出当前被操作的文档的文件名。

菜单栏包含了开发环境中几乎所有的命令，它为用户提供了文档操作、程序的编译、调试、窗口操作等一系列功能。菜单中的一些常用命令还被排列在相应的工具栏上，以便用户更好地操作。

项目工作区窗口包含用户项目的一些信息，包括类（ClassView 页面）、项目文件（FileView 页面）、资源（ResourceView 页面）等。在项目工作区窗口中的任何标题或图标处单击鼠标右键，都会弹出相应的快捷菜单，包含当前状态下的一些常用操作。

文档窗口一般位于开发环境窗口的右边，各种程序代码的源文件、资源文件、文档文件等都可以通过文档窗口显示出来。

输出窗口一般出现在开发环境窗口的底部，包括编译（Build）、调试（Debug）、查找文件（Find in Files）等相关信息的输出。这些输出信息以多页面标签的形式出现在输出窗口中，例如“编译”页面标签显示的是程序在编译和连接时的进度及错误信息。

状态栏一般位于开发环境的最底部，它用来显示当前操作状态、注释、文本光标所在的行列号等信息。

4. 编辑 C++源程序

单击 File 菜单→new→File→双击 C++ Source File，打开一个新的编辑对话框窗口，即可输入 C++源程序。

5. 保存 C++源程序

单击 File 菜单→Save 或 Save As，弹出“保存为”对话框。选择存放文件的路径后，在“文