21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

C++程序设计大学教程

C++ Programming

王春玲 编著

- 强调面向对象的概念
- 注重程序分析与设计
- 重视理论，突出实践

高校系列

人民邮电出版社
POSTS & TELECOM PRESS

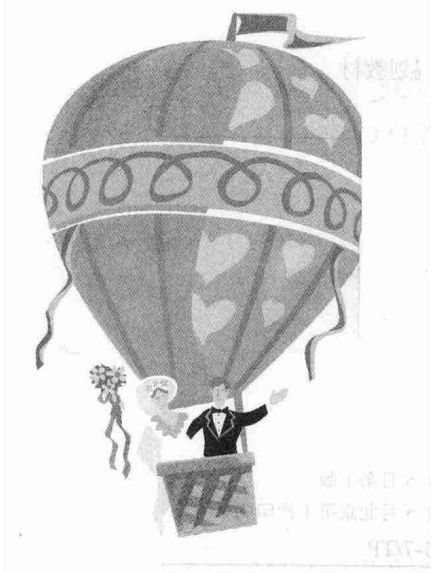
21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

C++程序设计大学教程

C++ Programming

王春玲 编著



高校系列

人民邮电出版社

北京

图书在版编目（CIP）数据

C++程序设计大学教程 / 王春玲编著. —北京：人民邮电出版社，2009.5
21世纪高等学校计算机规划教材
ISBN 978-7-115-20553-7

Ⅰ. C… Ⅱ. 王… 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（2009）第038583号

内 容 提 要

本书是一本易学易用的C++程序设计大学教程。全书覆盖了C++的基础知识，透彻讲解了C++的核心技术，并附以典型实例，另配有实验指导。书中主要内容包括面向对象程序设计的基本思想，数据类型、运算符和表达式，基本控制结构，函数，数组、指针与引用，类和对象，继承和派生，虚函数与多态性，运算符重载，模板，C++流和异常处理等。

本书既可作为高等学校C++语言程序设计的教材，也可作为程序设计爱好者的参考用书。

21世纪高等学校计算机规划教材

C++程序设计大学教程

◆ 编　　著　王春玲
责任编辑　滑　玉
执行编辑　武恩玉

◆ 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
中国铁道出版社印刷厂印刷

◆ 开本：787×1 092　1/16
印张：18
字数：474千字　　2009年5月第1版
印数：1－3 000册　　2009年5月北京第1次印刷

ISBN 978-7-115-20553-7/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

C++语言是目前应用较广的一种优秀的高级程序设计语言，它既保留了对传统的结构化程序设计方法的支持，同时又增加了对面向对象程序设计方法的完全支持，是一种具有代表性的面向对象的程序设计语言。

目前，关于 C++语言的教材大多是围绕其理论知识的讲解展开的，然而，要想学好 C++语言，除了掌握其基本理论知识外，还必须加强实践环节。也就是说，对于 C++语言和程序设计来讲，实践和理论是同等重要的。因此，本书的特色不仅仅是理论知识内容全面，而且突出实践，具体体现为：

内容全面　本书全面、系统地讲述了 C++语言的基本概念、语法知识和面向对象程序设计的方法，并结合作者多年的教学经验对 C++语言面向对象的有关概念：类和对象、数据封装、继承和派生、虚函数和多态性等进行了详尽的介绍。

实例丰富　本书每一章节针对重点语法都给出了实例，并详细地加以说明，对基本概念的讲解简明、易懂；每章末都附有相应的习题，以进一步强化读者所学的理论知识。书中所有的例题和习题都在 Visual C++ 6.0 开发环境中调试完成，保证代码的正确性。

突出实践　本书为每一章都设计了与理论知识配套的详细的实验指导，使读者能够真正提高学习效率，做到学以致用。

全书共分为 12 章，主要内容如下。

第 1 章主要介绍与面向对象程序设计有关的基本概念、基本知识。

第 2 章主要介绍 C++语言的数据类型、常量和变量、运算符和表达式以及数据类型转换。本章是 C++语言学习的语法基础。

第 3 章主要介绍 C++语言程序的基本控制结构，重点介绍了 C++语言各种语句的使用方法。

第 4 章主要介绍函数，不仅包括函数的基本概念、定义、说明、调用等的实现方法，还包括函数的递归调用、函数重载、内联函数等知识。

第 5 章主要介绍构造数据类型，其中包括数组、指针、引用、结构体和共用体。

第 6 章主要介绍面向对象程序设计中的两个最基本的概念：类和对象。重点是如何建立一个类以及类的使用方法。

第 7 章主要介绍继承和派生，通过继承可以在已有类的基础上派生新的类，从而实现了软件重用。

第 8 章主要介绍虚函数和多态性。数据封装、继承性和多态性共同组成了面向对象程序设计的三大机制。

第 9 章主要介绍运算符重载的意义，以及实现运算符重载的方法。

第 10 章主要介绍函数模板和类模板的概念、定义以及使用。

第 11 章主要介绍 C++流的概念、数据输出的格式控制和文件流。

第12章主要介绍C++的异常处理机制。

本书在编写过程中得到了陈志泊教授及王旻龙先生的大力帮助和支持。另外，王娜、陈航、刘寅等同学也参加了部分程序的调试工作，在此表示衷心的感谢！

由于编者水平有限，书中不妥之处，恳请广大读者提出宝贵意见。作者电子邮箱：wang_chunling@126.com。

编　者

2008年11月

目 录

第 1 章 绪论……1

1.1 程序设计初步……1
1.1.1 程序设计语言……1
1.1.2 面向对象程序设计……1
1.2 C++的产生与发展……3
1.3 C++的特点……4
1.4 C++程序的基本组成……4
1.5 C++程序的开发过程……5
1.5.1 Visual C++ 6.0 集成开发环境简介……5
1.5.2 C++程序的开发过程……6
1.5.3 控制台应用程序实例……7
1.6 实验指导……9
习题……10

第 2 章 基本数据类型、运算符与表达式……12

2.1 数据类型……12
2.1.1 基本数据类型……12
2.1.2 类型修饰符……13
2.2 常量……13
2.2.1 整型常量……14
2.2.2 实型常量……14
2.2.3 字符常量……14
2.2.4 字符串常量……15
2.2.5 符号常量……15
2.2.6 逻辑常量……16
2.2.7 枚举常量……16
2.3 变量……16
2.3.1 标识符的命名规则……17
2.3.2 变量的定义……17
2.3.3 变量的使用……18
2.3.4 常变量……18
2.4 运算符和表达式……19
2.4.1 算术运算符和算术表达式……19
2.4.2 自增、自减运算符……20
2.4.3 赋值运算符和赋值表达式……21
2.4.4 sizeof 运算符……22
2.4.5 关系运算符和关系表达式……23
2.4.6 逻辑运算符和逻辑表达式……23
2.4.7 条件运算符……24
2.4.8 位运算符……25
2.4.9 逗号运算符……25
2.4.10 运算符的优先级与结合性……26
2.5 数据类型转换……26
2.5.1 隐式类型转换……27
2.5.2 显式类型转换……27
2.6 数据的输入与输出……27
2.6.1 数据的输入 cin……28
2.6.2 数据的输出 cout……28
2.7 实验指导……30
习题……31

第 3 章 C++的控制语句……33

3.1 C++语句概述……33
3.2 C++程序的 3 种基本结构……34
3.3 if 语句……36
3.3.1 单分支 if 语句……36
3.3.2 双分支 if 语句……37
3.3.3 多分支 if 语句……38
3.3.4 if 语句的嵌套……39
3.4 switch 语句……40

3.5 循环语句……42
3.5.1 while 循环语句……42
3.5.2 do-while 循环语句……43
3.5.3 for 循环语句……44
3.5.4 循环语句的嵌套……46
3.5.5 3 种循环语句的比较……47
3.6 限定转向语句……48
3.6.1 goto 语句……48
3.6.2 break 语句……48
3.6.3 continue 语句……49
3.6.4 return 语句……50
3.7 实验指导……51
习题……54

第 4 章 函数……57

4.1 函数的定义……57
4.2 函数的返回值……58
4.3 函数的调用……58
4.3.1 函数的调用形式……58
4.3.2 调用函数的前提条件……59
4.3.3 函数定义和函数说明的区别……60
4.3.4 函数的嵌套调用……60
4.4 函数的参数……61
4.4.1 形式参数与实际参数……61
4.4.2 参数的传递方式……61
4.4.3 默认参数……61
4.5 函数的递归调用……62
4.5.1 函数递归调用的概念……62
4.5.2 函数递归调用的条件……63
4.6 内联函数……64
4.6.1 内联函数的定义方法……64
4.6.2 内联函数与普通函数的区别和联系……65
4.6.3 对内联函数的限制……65
4.7 函数重载……65
4.7.1 函数重载的定义……65
4.7.2 匹配重载函数的顺序……66
4.7.3 定义重载函数时的注意事项……67
4.8 变量的作用域与生存期……67
4.8.1 局部变量……68
4.8.2 静态局部变量……68
4.8.3 全局变量……69
4.8.4 静态全局变量……71
4.9 编译预处理……72
4.9.1 宏定义命令……72
4.9.2 文件包含命令……72
4.9.3 条件编译命令……73
4.10 实验指导……74
习题……75

第 5 章 构造数据类型……78

5.1 数组……78
5.1.1 一维数组……78
5.1.2 二维数组……81
5.1.3 字符数组与字符串……84
5.1.4 字符串函数……87
5.2 指针……89
5.2.1 指针变量……89
5.2.2 const 指针……91
5.2.3 用指针作为函数的参数……92
5.2.4 指针与数组……93
5.2.5 指针与字符串……96
5.2.6 指针数组与多重指针……97
5.3 引用……98
5.3.1 引用的定义……98
5.3.2 引用的使用……98
5.3.3 用引用作为函数的参数……99
5.3.4 如何使一个被调函数同时返回多个值……100
5.4 结构体和共用体……102

5.4.1 结构体 …… 102
5.4.2 共用体 …… 104
5.5 动态内存分配与释放 …… 106
5.5.1 申请分配内存 …… 106
5.5.2 释放内存 …… 107
5.6 类型定义 typedef …… 107
5.7 实验指导 …… 108
习题 …… 117

第 6 章 类和对象 …… 120

6.1 类 …… 120
6.1.1 类的定义 …… 120
6.1.2 类成员的访问权限 …… 121
6.1.3 类的数据成员 …… 122
6.1.4 类的成员函数 …… 122
6.2 对象 …… 124
6.2.1 对象的定义 …… 124
6.2.2 对象的指针 …… 125
6.2.3 访问对象的成员 …… 125
6.2.4 this 指针 …… 127
6.2.5 对象的作用域与生存期 …… 128
6.3 构造函数与析构函数 …… 129
6.3.1 构造函数 …… 129
6.3.2 构造函数的重载 …… 130
6.3.3 默认构造函数 …… 131
6.3.4 有缺省参数的构造函数 …… 132
6.3.5 析构函数 …… 134
6.3.6 拷贝构造函数 …… 135
6.3.7 浅拷贝与深拷贝 …… 136
6.4 对象成员与对象数组 …… 139
6.4.1 对象成员 …… 139
6.4.2 对象数组 …… 141
6.5 静态成员 …… 143
6.5.1 静态数据成员 …… 143
6.5.2 静态成员函数 …… 145
6.6 友元函数和友元类 …… 147
6.6.1 友元函数 …… 147
6.6.2 友元类 …… 149
6.7 常对象与常成员 …… 150
6.7.1 常对象 …… 150
6.7.2 常数据成员 …… 151
6.7.3 常成员函数 …… 152
6.8 实验指导 …… 154
习题 …… 160

第 7 章 继承和派生 …… 170

7.1 继承的概念 …… 170
7.2 单继承 …… 171
7.2.1 单继承的定义方式 …… 171
7.2.2 派生类的成员构成 …… 172
7.2.3 继承方式对基类成员的访问属性控制 …… 173
7.2.4 派生类的构造函数 …… 177
7.2.5 有子对象的派生类的构造函数 …… 182
7.2.6 派生类的析构函数 …… 183
7.3 多重继承 …… 184
7.3.1 多重继承的定义方式 …… 184
7.3.2 多重继承的二义性 …… 186
7.3.3 虚基类及其派生类的构造函数 …… 189
7.4 实验指导 …… 192
习题 …… 197

第 8 章 虚函数与多态性 …… 203

8.1 多态性 …… 203
8.1.1 多态的类型 …… 203
8.1.2 多态的实现 …… 203
8.2 赋值兼容规则 …… 204
8.3 用基类指针指向公有派生类对象 …… 205
8.4 虚函数 …… 207
8.5 纯虚函数与抽象类 …… 209

8.5.1 纯虚函数……209
8.5.2 抽象类……210
8.5.3 使用纯虚函数与抽象类的注意事项……211
8.6 实验指导……212
习题……216

第 9 章 运算符重载……221

9.1 运算符重载的概念……221
9.2 重载为类的成员函数……221
9.3 重载为类的友元函数……223
9.4 运算符重载的限制……224
9.5 典型运算符重载……225
9.5.1 赋值运算符的重载……225
9.5.2 单目运算符的重载……226
9.5.3 I/O 运算符的重载……228
9.6 实验指导……229
习题……232

第 10 章 模板……235

10.1 函数模板……235
10.1.1 函数模板的定义……236
10.1.2 函数模板的使用……236
10.2 类模板……238
10.2.1 类模板的定义……239
10.2.2 类模板的使用……239
10.3 标准模板库……242
10.3.1 范型化程序设计……242
10.3.2 标准模板库……243
10.3.3 容器……243
10.3.4 迭代器……245
10.3.5 算法……248
10.4 实验指导……249
习题……250

第 11 章 流……253

11.1 流概述……253
11.2 数据输出的格式控制……254
11.2.1 域宽控制……254
11.2.2 填充字符控制……255
11.2.3 数制控制……256
11.2.4 浮点数控制……257
11.2.5 对齐方式控制……257
11.3 文件流……258
11.3.1 文件的打开和关闭……259
11.3.2 文件的操作……259
11.4 实验指导……264
习题……265

第 12 章 异常处理……268

12.1 异常的概念……268
12.2 C++异常处理机制……268
12.2.1 异常处理的语法……268
12.2.2 使用多条 catch 语句……270
12.2.3 捕获所有类型的异常……271
12.3 限制异常与重抛异常……272
12.3.1 限制异常……272
12.3.2 重抛异常……273
12.4 异常处理中对象的构造和析构……273
12.5 实验指导……275
习题……276

附录……278

附录 A C++关键字……278
附录 B ASCII 码字符表……279

参考文献……280

第 1 章 绪论

1.1 程序设计初步

计算机本身并不知道如何解决一个问题，必须由人事先把解决问题的步骤设计好，编写成程序输入到计算机中，计算机才能在程序的控制下按照人的意图解决相应的问题。因此，人们必须用计算机能够接受的语言，告诉计算机对什么样的数据进行什么样的运算。程序设计语言就是人们为了表达程序而设计出来的计算机能够接受的人工语言。

1.1.1 程序设计语言

计算机诞生初期，人们必须使用机器语言或汇编语言编写程序。机器语言只能存储和识别二进制的数据和指令，因此也称为二进制语言。汇编语言相对于机器语言主要的进步是用含义较鲜明的符号代替机器语言中的二进制编码，但汇编语言指令与机器语言指令是一一对应的，编写起来仍然很烦琐。

世界上第一种计算机高级语言诞生于 1954 年，它是用于科学计算的 Fortran 语言。随后出现了多种计算机高级语言，常见的有 Basic、Algol、Cobol、Ada、C、Pascal、C++和 Java 等。高级语言的表示方法要比低级语言更接近于待解问题的表示方法，其特点是在一定程度上与具体机器无关，易学、易用、易维护。

本书主要介绍 C++语言。C++语言是在 C 语言的基础上逐渐发展起来的，是目前应用较广的一种优秀的高级程序设计语言。

1.1.2 面向对象程序设计

面向对象程序设计（Object-Oriented Programming，OOP）方法强调直接以问题域（现实世界）中的事物为中心来思考和认识问题，并按照这些事物的本质特征将其抽象为对象，以作为构成软件系统的基础。这样，在现实世界中有哪些值得注意的事物，在程序中就有哪些对象与之对应。由于程序与现实世界间具有极强的对应关系，因此，程序可以用对象的概念很自然地进行思考，从而大大减小了软件开发的难度。

1. 面向对象程序设计的有关概念

在面向对象程序设计中，包括了对象、类、封装、继承、消息传递和多态性等基本概念。

（1）**对象（Object）** 对客观事物的数值化描述，就称为对象。每个对象都具有属性（Attribute）

和方法（Method）这两方面的特征。对象的属性描述了对象的状态和特征，对象的方法说明了对象的行为和功能，并且对象的属性值只应由这个对象的方法来读取和修改，两者结合在一起就构成了对象的完整描述。

例如，在一个学生学籍管理系统中，可以把每一位同学看成是一个对象，其学号、姓名、班级和已修学分等都是对象的属性，同时每一位同学都有修改学分的功能（即方法），如某位同学新学期选修了一门课程，则可以通过修改学分的方法使学分增加。

（2）**类（Class）** 具有相似属性和行为的一组对象，就称为类。可见，有了类的概念以后，就可以对具有共同特征的事物进行统一描述。

例如，在学生学籍管理系统中，张三、李四等同学都是“学生”，都具有学号、姓名、班级和已修学分等属性，所以，可将所有同学抽象成一个类，即“学生”类。

（3）**封装（Encapsulation）** 在面向对象程序设计方法中，封装具有两方面的含义：一方面是指将对象的属性和方法形成一个不可分割的整体，另一方面是指“数据隐藏”，即对象只应保留有限的对外接口（即和外界联系的方法），并尽可能隐藏对象内部的具体细节。也就是说，通过封装，在对象和外界之间建立了一道屏障，使得外界只能通过对象所提供的接口（即对象的方法）与之发生联系，而外界不能以其他方式直接修改对象的属性值。

例如，某台电视机就是“电视机”这个类中的一个对象，它有尺寸、颜色和亮度等属性，同时还有一些能够由用户（外界）进行选台、调节亮度等按钮，即与外界的接口。当需要调节这台电视机的亮度时，用户可以通过调节亮度按钮进行调节，对用户（外界）来说，不需要知道调节电视机亮度的具体细节。

（4）**继承（Inheritance）** 在面向对象程序设计中，允许在已有类的基础上通过增加新特征而派生出新的类，这称为继承。其原有的类称为基类（Base class），而新建立的类称为派生类（Derived class）。

例如，车是一个类，而小轿车就是车的一个派生类，奥迪小轿车又是小轿车的一个派生类。由此可以看出，派生类自动继承了基类的属性和方法，在整个类的继承关系中，越是上层的类越简单、越一般，而越是下层的类越具体、越详细。因此，可以在基类的基础上增加一些属性和方法来构造出新的类。

在面向对象程序设计中定义新的类时，只要将新类说明为某个类的派生类，则该派生类会自动地继承了这个基类的属性和方法，这些内容在新类中可以直接使用而不必重新定义。这显然减少了软件开发的工作量，也实现了代码的重用。

（5）**消息（Message）** 在面向对象程序设计中，对象描述了客观实体，因此，对象之间也是相互联系的。当一个对象需要另外一个对象提供服务时，它向对方发出一个服务请求，而收到请求的对象会响应这个请求并完成指定的服务。这种向对象发出的服务请求就称为消息。

消息传递与对象封装间有着密切的联系。封装使对象成为一些各负其责、互不干扰的单位，而消息传递则为对象提供了唯一合法的动态联系的途径，使它们成为一个相互配合的有机整体。

（6）**多态性（Polymorphism）** 在通过继承而派生出的一系列类中，可能存在一些名称相同，但实现过程和功能不同的方法（Method）。

多态性是指当程序中的其他部分发出同样的消息时，按照接收消息对象的不同能够自动执行类中相应的方法。其优点是用户不必知道某个对象所属的类就可以执行多态行为，从而为程序设计带来更大方便。

2. 面向对象程序设计方法

面向对象程序设计（OOP）方法将设计目标从模拟现实世界的行为转向了模拟现实世界中存在的对象及各自的行为。OOP 方法将“对象”作为系统中最基本的运行实体，“对象”中封装了描述该对象的特殊属性（数据）和行为方式（方法）。整个程序即由各种不同类型的对象组成，各对象既是一个独立的实体，又可通过消息相互作用，对象中的方法决定要向哪个对象发消息、发什么消息以及收到消息时如何进行处理等。

一个对象的内部结构如图 1-1 所示。

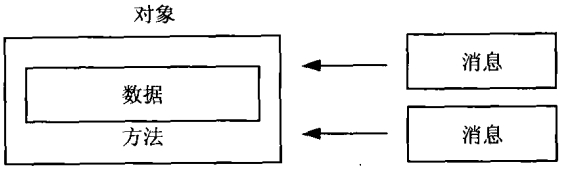


图 1-1　对象的内部结构

3. 面向对象程序设计方法的特点

（1）OOP 方法以“对象”或“数据”为中心。但这时的数据与传统的被动数据不同，它具有“行动”的功能，而这种动作是在对象接收了消息时发生的。由于对象自然地反映了应用领域的模块性，因此具有相对稳定性，可以被用做一个组件去构成更复杂的应用，又由于对象一般封装的是某一实际需求的各种成分，因此，某一对象的改变对整个系统几乎没有影响。

（2）引入了“类”（Class）的概念。类与类以层次结构组织，属于某个类的对象除具有该类所描述的特性外，还具有层次结构中该类上层所有类描述的全部性质。在 OOP 方法中称这种机制为继承。

（3）OOP 方法的模块性与继承性保证了新的应用程序设计可在原有对象的数据类型和功能的基础上通过重用、扩展和细化来进行，而不必从头做起或复制原有代码。这种特性大大减少了重新编写新代码的工作量，同时降低了程序设计过程中出错的可能性，达到事半功倍的效果。

1.2　C++的产生与发展

C++语言是在 C 语言的基础上为支持面向对象程序设计方法而开发的一种程序设计语言。C++语言包括 C 语言的全部特征和优点，既保留了对传统的结构化程序设计方法的支持，同时又增加了对面向对象程序设计方法的完全支持。

C 语言是由贝尔实验室的 Dennis Ritchie 在 Basic 语言的基础上开发出来的。设计 C 语言的最初目的是将其用做 UNIX 操作系统的描述语言。因此，C 语言的产生和发展与 UNIX 操作系统有着十分密切的联系。1972 年在一台 DEC PDP-11 计算机上实现了最初的 C 语言。C 语言的严谨设计，使得把用 C 语言编写的程序移植到大多数计算机上成为可能。到 20 世纪 70 年代末，C 语言已经演化为现在所说的“传统 C 语言”。C 语言在各种计算机上的快速推广导致产生了许多 C 语言版本。这些版本虽然是类似的，但通常是不兼容的。为了明确地定义与机器无关的 C 语言，1989 年美国国家标准协会制定了 C 语言的标准（ANSI C）。至此，C 语言以其独有的特点风靡了全世界。

（1）语言简洁、紧凑，使用方便、灵活。C 语言只有 32 个关键字，程序书写形式自由。

（2）具有丰富的运算符和数据类型。

（3）可以直接访问内存地址，能进行位操作，因而能够胜任开发操作系统的工作。

（4）生成的目标代码质量高，程序运行效率高。

（5）代码的可移植性好。

然而，C 语言在盛行的同时，也暴露出了它的局限性。

（1）数据类型检查机制相对较弱，这使得程序中的一些错误不能在编译阶段被及时发现。

（2）C 语言本身几乎没有支持代码重用的语言结构，因此，一个程序员精心设计的程序很难为其他程序所用。

（3）当程序的规模达到一定程度时，程序员很难控制程序的复杂性，因此，不适合开发大型的软件。

为了满足管理程序的复杂性的需要，1980 年，贝尔实验室的 Bjarne Stroustrup 开始对 C 语言进行改进和扩充，最初的成果称为“带类的 C”，1983 年正式命名为 C++，在经历了 3 次 C++修订后，于 1994 年制定了 ANSI C++标准的草案。这个草案于 1998 年被国际标准化组织（ISO）批准为国际标准（ISO/IEC 14882）。C++在 20 多年的发展过程中不断完善，至今仍是一门充满活力的程序设计语言。

1.3 C++的特点

C++语言的特点主要表现在两个方面：一方面是对 C 语言的全面兼容，另一方面是对面向对象程序设计方法的完全支持。

C++语言包括了 C 语言的全部特征和优点，是一个更好的 C 语言。它不仅保持了 C 语言的简洁、高效、接近汇编语言等特点，而且对 C 语言的数据类型系统进行了改进和扩充，增强了其数据类型检查机制，使编译系统能够检查出更多的数据类型错误。

C++语言最突出的特点是对面向对象程序设计方法的完全支持。虽然与 C 语言的兼容使得 C++语言具有结构化程序设计和面向对象程序设计的双重特点，但是它在概念上是和 C 语言完全不同的语言，应该注意按照面向对象的思想去编写程序。

1.4 C++程序的基本组成

下面来看一个最简单的 C++程序。

【例 1-1】 一个最简单的 C++程序。

```
//Li1_1.cpp
#include <iostream>                    //包含头文件 iostream
using namespace std;                   //使用名字空间 std
int main()
{
    cout<<"Hello C++!"<<endl;          //输出字符串
    return 0;
}
```

程序的运行结果为

```
Hello C++!
```

这个程序虽然很短，但是却包含了 C++程序的几个基本组成部分。

1. 注释

注释是程序员对程序进行的注解和说明，目的是提高程序的可读性。编译系统在对程序进行编译的时候忽略注释。在 C++中有如下两种注释方法。

（1）行注释“//”：注释的内容从“//”开始，到本行末尾结束。

（2）块注释“/*”和“*/”：注释的内容从“/*”开始，到“*/”结束。

2. 编译预处理

C++中所有以“#”开头的代码，都是编译预处理。如程序中第 1 行代码

```
#include <iostream>
```

的作用是将头文件 iostream 中的内容加到程序中。iostream 文件中设置了 C++的 I/O 相关环境，并定义了输入输出流对象 cin 和 cout 等。

3. 标准名字空间

程序中的第 2 行代码

```
using namespace std;
```

的意思是使用标准名字空间 std。“using namespace”是针对命名空间的指令。iostream 是一个标准函数库，cout 是标准函数库提供的一个对象，标准函数库在 namespace 说明书中被指定为“std”。因此如果想要在代码中使用标准函数库，就必须加入此行代码。

4. 程序主体

每个 C++程序有且仅有一个名称为 main 的函数。main 函数（也称主函数）是整个程序运行时的入口。无论主函数处在程序的什么位置，程序都是从它开始执行。main 函数前面的 int 表示它的返回值类型为整型，意思是程序执行结束后要向操作系统返回一个整型值。

所有的函数体都由一对大括号“{}”括起来，其内容由若干条语句构成，函数体的内容决定了该函数的功能。如程序中的第 5 行代码

```
cout<<"Hello C++!"<<endl;
```

的作用是将字符串“Hello C++!”输出到屏幕上。其中，cout 是标准输出流对象；“<<”是插入运算符，可以连续使用；endl 的作用是按 Enter 键换行。

另外，函数体中的每一条语句后面都要有“;”，表示一条 C++语句的结束。

1.5　C++程序的开发过程

C++程序的开发过程通常包括编辑、编译、连接、运行和调试等步骤。目前能完成 C++程序开发的集成开发环境有很多，如 Microsoft 公司的 Visual C++、Borland 公司的 C++ Builder 和 IBM 公司的 Visual Age C++等。迄今为止，Visual C++是功能最强、使用范围最广的软件开发工具。本书将以 Microsoft 公司的 Visual C++ 6.0 为工具来讲解 C++程序的开发。

1.5.1　Visual C++ 6.0 集成开发环境简介

Visual C++ 6.0 是 Microsoft 公司在 1998 年推出的基于 Windows 9x 和 Windows NT 的优秀集成开发环境（IDE，Integrated Development Environment）。Visual C++ 6.0 由许多组件组成，包括编辑器、编译器、调试器以及应用程序向导（AppWizard）、类向导（ClassWizard）等开发工具，这些开发工具通过一个名为 Developer Studio 的组件集成为一个和谐的开发环境，可以在此环境中轻松地完成创建源文件、编辑资源以及对程序的编辑、编译、连接、运行和调试等各项工作。

单击任务栏中的【开始】按钮，在弹出的菜单中，选择【程序】→【Microsoft Visual Studio 6.0】→【Microsoft Visual C++ 6.0】菜单命令，就可以进入 Visual C++ 6.0 的集成开发环境的主窗口，如图 1-2 所示。

图 1-2　Visual C++ 6.0 集成开发环境的主窗口界面

由图 1-2 中可以看出，该集成开发环境由标题栏、菜单栏、工具栏、源代码编辑窗口、工作区窗口、输出窗口和状态栏等部分组成。

屏幕的最上端是标题栏，它用于显示应用程序的名称和打开的文件名称。标题栏的下面是菜单栏和工具栏。工具栏下方有两个窗口，左边是工作区窗口或工作空间（Workspace），右边是源代码编辑窗口。在工作区窗口和源代码编辑窗口下方是输出窗口（Output），它主要用于显示项目建立过程中生成的各种信息。屏幕的最下面是状态栏，它给出了当前操作或所选择命令的提示信息。

另外，在程序调试过程中，Visual C++ 6.0 可以为不同的调试信息创建不同的窗口，比如：观察窗口（Watch Window）、变量窗口（Variable Window）、寄存器窗口（Register Window）、存储器窗口（Memory Window）和调用堆栈窗口（Call Stack Window）等。

1.5.2　C++程序的开发过程

一般地，C++程序的开发流程可用图 1-3 来表示。

1. 程序的编辑

程序的编辑是程序开发过程中的第一步，它主要是将源程序输入到计算机中，并做必要的修改。任何一种文本编辑器都可以完成这项工作。在 Visual C++ 6.0 中，可以使用源代码编辑窗口进行 C++程序的编辑工作，在源代码编辑窗口中提供了自动缩进、关键词亮色、调用提示、查找和替换等一系列功能，用户使用起来非常方便。

C++源程序的扩展名为.cpp，即当用户完成源程序的编辑工作后，需将源程序以.cpp 为扩展名进行保存。

2. 程序的编译

程序的编译是将程序的源代码转换为机器语言代码。C++是一种高级程序设计语言，它的语法规则与汇编语言和机器语言相比更接近人类自然语言的习惯，而机器语言是计算机能够“看”懂的唯一语言。因此，如果要让计算机“看”懂一个 C++程序，就必须使用一种叫做“编译器”的工具，将一个 C++程序“翻译”成机器指令。

C++源程序经过编译以后生成以.obj 为扩展名的目标文件。如果一个 C++程序由多个源程序文件组成，应将它们分别进行编译，从而形成多个目标文件。

3. 程序的连接

编译后的程序是不能由计算机执行的，还需要进行连接。程序的连接是将多个目标文件以及库中的某些文件连在一起，生成扩展名为.exe 的可执行文件。

在编译和连接程序时，都会对程序中存在的语法错误和连接错误进行检查，并将检查出的信息显示在屏幕上。

4. 程序的运行

在程序的编译和连接工作成功地完成之后，就可以运行程序，来观察程序是否符合所期望的运行结果。

5. 程序的调试

如果程序的运行结果与期望不符，说明源程序中存在语义错误。这时，需要使用调试器对可执行程序进行跟踪调试来查找错误发生的原因。

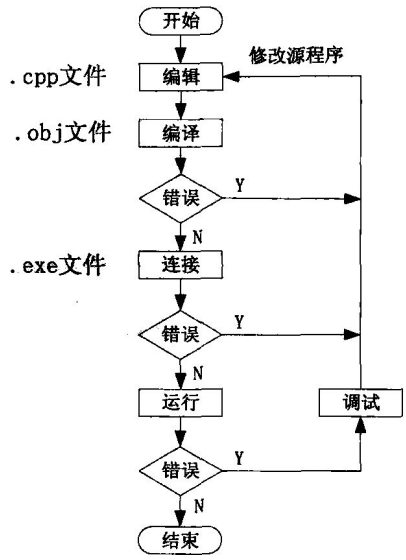


图 1-3　C++程序的开发流程

1.5.3　控制台应用程序实例

下面以【例 1-1】为例来介绍控制台应用程序的开发过程。

（1）启动 Visual C++ 6.0 集成开发环境。单击任务栏中的【开始】按钮，在弹出菜单中选择菜单【程序】→【Microsoft Visual Studio 6.0】→【Microsoft Visual C++ 6.0】，就可以进入 Visual C++ 6.0 的集成开发环境的主窗口，如图 1-2 所示。

（2）建立新工程。在 Visual C++6.0 环境下，所有应用程序都包含在一个工程中，所以建立任何应用程序的第一步就是建立一个工程。选择菜单【File】→【New】(Ctrl+N 组合键)，弹出“New”对话框，如图 1-4 所示。

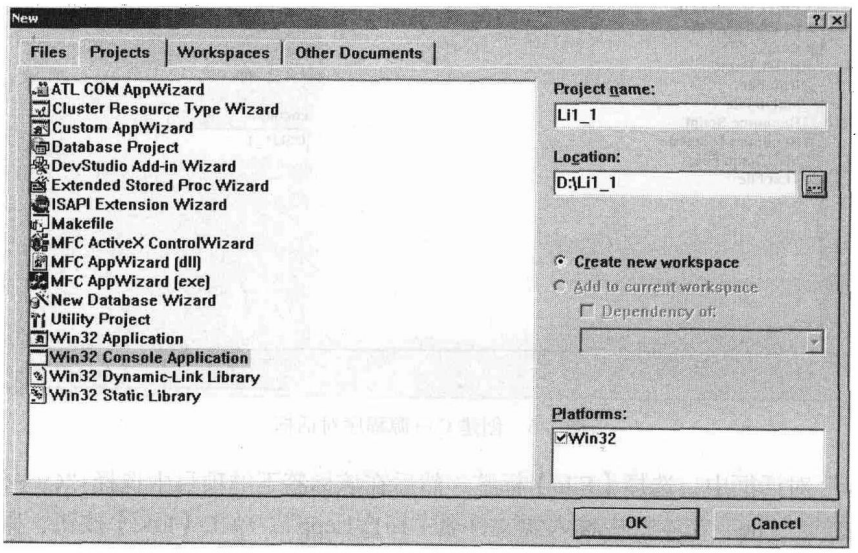


图 1-4　“New”对话框

在“New”对话框中，选择【Projects】标签。选择其中的“Win32 Console Application”工程，然后在右边“Project name”文本框内输入工程名“Li1_1”，在“Location”文本框中输入工程存放位置“D:\Li1_1”，以上步骤完成后，单击【OK】按钮，弹出图 1-5 所示的对话框。

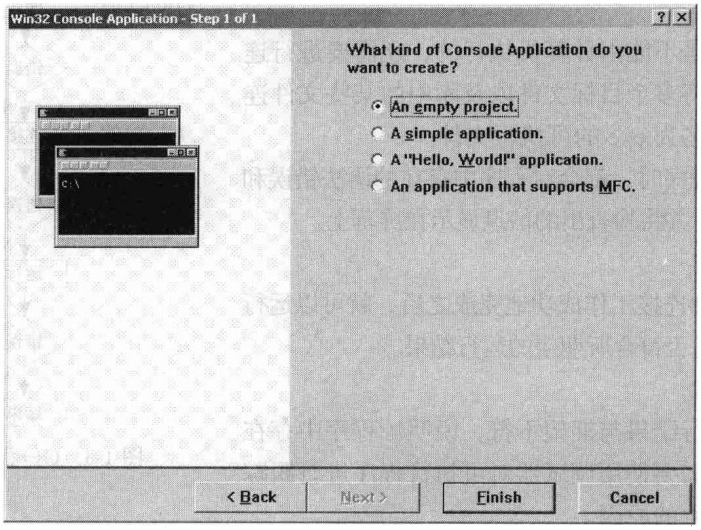


图 1-5　控制台应用程序对话框

在图 1-5 所示的对话框中选择“An empty project”单选钮，表示选择空工程，单击【Finish】按钮，弹出“New Project Information”对话框，在确认工程建立信息后，单击【OK】按钮，至此，新工程建立完成。

（3）添加源文件。新建的空工程中没有任何内容，要在该工程中创建一个 C++源程序文件。选择菜单【Project】→【Add to Project】→【New】，弹出图 1-6 所示的“New”对话框。

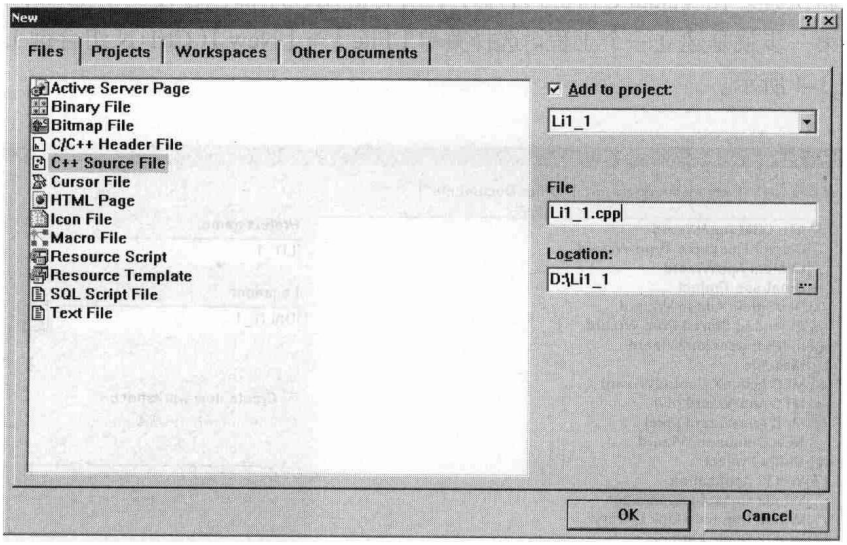


图 1-6　创建 C++源程序对话框

在“New”对话框中，选择【File】标签，然后在该标签下的项目中选择“C++ Source File”项，并在右边的“File”文本框中输入源文件名“Li1_1.cpp”，单击【OK】按钮，则建立了一个文件名为“Li1_1.cpp”的 C++源程序文件。