

C++ 程序设计案例教程

王学颖 李 航 张燕丽 李 晖 编著

- 一个完整应用案例——水果超市管理系统贯穿各章，按照应用系统的开发步骤对该系统进行分模块讲解，充分展示面向对象的编程思想与方法
- 详细介绍C++语言的基本概念，重点叙述面向对象程序设计的特征：封装性（类和对象）、继承性和多态性
- 分析教学中易出现的问题及遇到的困难

21 世纪高等院校计算机应用规划教材

C++程序设计案例教程

王学颖 李 航 张燕丽 李 晖

科 学 出 版 社

内 容 简 介

本书首先简单介绍了面向对象程序设计的基本概念,然后全面系统讲解 C++语言的基本语法和编程方法以及 C++语言面向对象的重要特征(类和对象、继承和派生类、多态性和虚函数)等,最后介绍 ACM-ICPC 实训平台的基本使用方法。本书内含丰富的例题和习题,便于读者学习和巩固知识。

本书可作为高等院校计算机、电子信息及信息管理等相关专业“C++程序设计”课程的教材,也可供初学者自学使用。

图书在版编目(CIP)数据

C++程序设计案例教程/王学颖等编著. —北京:
科学出版社, 2010. 4
21 世纪高等院校计算机应用规划教材
ISBN 978-7-03-027018-4
I. ①C… II. ①王… 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 (2010) 第 044177 号

责任编辑: 张 鑫 / 责任校对: 杨慧芳
责任印刷: 新世纪书局 / 封面设计: 彭琳君

科 学 出 版 社 出 版

北京东黄城根北街 16 号

邮政编码: 100717

<http://www.sciencep.com>

中国科学出版集团新世纪书局策划

北京市艺辉印刷有限公司印刷

中国科学出版集团新世纪书局发行 各地新华书店经销

*

2010 年 4 月 第 一 版 开本: 16 开
2010 年 4 月第一次印刷 印张: 18.25
印数: 1—3 000 字数: 444 000

定价: 29.80 元

(如有印装质量问题, 我社负责调换)

前 言

面向对象程序设计是计算机软件开发人员必须掌握的一门技术，它与传统的结构化程序设计相比，具有很多优点，从根本上改变了人们设计软件的思维方式，使程序设计者不用特别关心具体的数据格式和过程，只需将重点放在要处理对象的设计和 research 上即可。C++语言是一种面向对象的编程语言，同时又兼容 C 语言的特征。用 C++语言编写的程序可读性更好，代码结构更合理，对面向对象技术提供全面支持。C++语言具有简单易学的特点，适合作为面向对象程序设计方法学习的入门语言。

本书是编者多年 C++教学经验的总结，针对编者在教学中出现的问题及遇到的困难进行讲解，全面系统地讲述了 C++语言的基本概念和编程方法，重点叙述面向对象程序设计的特征，即封装性（类和对象）、继承性和多态性。本书以一个完整的应用案例——水果超市管理系统的实现贯穿各章，使读者能够充分认识面向对象的编程方法，并能进行实际操作。在案例中，按照应用系统的开发步骤对系统进行分模块讲解，尤其是对水果超市管理系统中使用的类、对象和继承关系进行了详细的分析和设计，使读者能通过具体的实例充分理解面向对象程序设计方法的实质，同时对应用系统开发有一个初步的认识，并能进行实际操作，对提高读者的编程能力很有帮助。水果超市管理系统的源代码在附录中给出，并已调试通过。

全书共分 10 章，针对各章的知识点提供了相应的例题，并配有大量习题，便于读者对所学知识进行检测和巩固。全书主要内容包括以下 3 部分。

第一部分（第 1~4 章）简单介绍 C++语言的特点和面向对象程序设计的基本概念，详细讲述 C++语言的基本语法规则和功能，包括各种数据类型、控制语句，以及函数、指针和引用等。

第二部分（第 5~7 章）重点叙述 C++语言所支持的面向对象程序设计方法的核心内容：封装性、继承性和多态性。具体内容包括类和对象的定义和使用、对象的赋值和运算、继承性和派生类、多态性和虚函数等。这一部分是学习 C++语言的重点和难点。

第三部分（第 8~10 章）介绍函数模板、类模板和标准模板库（STL）的使用方法，同时对 C++中的异常处理机制和标准库中的异常类进行详细叙述，最后针对 ACM-ICPC 平台及在线实训内容进行简要说明。

书中的所有例题和习题均在 Visual C++ 6.0 环境下运行通过。

本书可作为高等院校计算机、电子信息及信息管理等相关专业“C++程序设计”课程的教材，总学时可安排为 72 学时，其中授课 50 学时，上机实验 22 学时。

本书由王学颖、李航、张燕丽、李晖编著，黄淑伟、穆宝良和周雄伟等参加了本书的编写工作。

由于时间仓促，书中难免存在不足之处，敬请广大读者批评指正。

编者

2010 年 2 月

目

录

第1章 C++程序设计概述..... 1

1.1 C++语言的历史和演进	1
1.1.1 计算机语言的发展	1
1.1.2 C++语言的产生	2
1.1.3 C++语言的主要版本	2
1.2 C++语言的特点	2
1.2.1 面向过程	3
1.2.2 面向对象	3
1.3 面向对象的相关概念	3
1.4 C++程序的基本结构和开发环境	5
1.4.1 C++程序的基本结构	5
1.4.2 C++程序的实现	9
1.4.3 C++程序的开发环境	10
习题	17

第2章 C++语言基础..... 20

2.1 基本数据类型	20
2.2 标识符与关键字	22
2.3 常量与变量	24
2.3.1 常量	24
2.3.2 变量	27
2.4 运算符与表达式	28
2.4.1 运算符	28
2.4.2 表达式	33
2.5 数组	38
2.5.1 数组的定义	38
2.5.2 数组的赋值	39
2.5.3 字符数组	42
2.6 枚举	43
2.6.1 枚举类型定义	43
2.6.2 枚举变量定义	44
2.6.3 枚举变量赋值	44
2.7 结构体	45
2.7.1 结构体类型定义	45

2.7.2 结构体变量的定义和初始化	46
--------------------------	----

2.7.3 结构体变量的引用	47
----------------------	----

2.8 共用体	48
---------------	----

2.9 控制语句	50
----------------	----

2.9.1 选择结构语句	50
--------------------	----

2.9.2 循环结构语句	55
--------------------	----

2.9.3 转向语句	61
------------------	----

2.10 应用实例——水果超市管理系统 菜单设计	63
-----------------------------------	----

2.10.1 系统功能介绍	64
---------------------	----

2.10.2 系统菜单设计	65
---------------------	----

习题	66
----------	----

第3章 指针和引用 71

3.1 指针	71
--------------	----

3.1.1 指针的概念	71
-------------------	----

3.1.2 指针变量的定义	72
---------------------	----

3.1.3 指针变量的运算	72
---------------------	----

3.1.4 指针的指针	75
-------------------	----

3.2 指针与数组	76
-----------------	----

3.2.1 指向数组的指针	76
---------------------	----

3.2.2 指针数组	80
------------------	----

3.3 字符串	82
---------------	----

3.3.1 C风格字符串	82
--------------------	----

3.3.2 string 类型	83
-----------------------	----

3.4 引用	85
--------------	----

3.4.1 引用的概念	85
-------------------	----

3.4.2 指针和引用的区别与联系	86
-------------------------	----

3.5 指针及引用在函数中的作用	88
------------------------	----

3.6 应用实例——水果超市管理系统的 水果基本信息管理	89
---------------------------------------	----

习题	91
----------	----

第4章 函数 96

4.1 函数的定义、声明和类型	96
-----------------------	----

4.1.1 函数的定义	96	5.7.1 对象数组的定义和使用	149
4.1.2 函数的声明	98	5.7.2 对象指针数组	150
4.1.3 函数的类型	99	5.7.3 指向对象数组的指针	151
4.2 函数的参数与调用	100	5.8 子对象和堆对象	152
4.2.1 函数的参数	100	5.8.1 子对象	153
4.2.2 函数的调用	101	5.8.2 堆对象	154
4.3 内联函数	105	5.9 友元	156
4.4 函数的重载	106	5.9.1 友元函数	156
4.5 标识符的作用域	108	5.9.2 友元类	157
4.5.1 作用域的种类	108	5.10 类的作用域和对象的生存期	159
4.5.2 标识符的作用域规则	109	5.10.1 类的作用域	159
4.5.3 全局变量和局部变量	111	5.10.2 对象的生存期	159
4.6 应用实例——水果超市管理系统中 购物车的设计	115	5.11 应用实例——水果超市管理系统 基本类的设计	161
习题	118	5.11.1 “水果类” Fruit 的设计	161
第5章 类和对象	123	5.11.2 “水果总类” FruitKind 的 设计	163
5.1 类的定义	123	5.11.3 “购买水果品种类” CartItem 的 设计	167
5.1.1 类的定义格式	123	5.11.4 “购物车类” Cart 的设计	168
5.1.2 类成员的访问控制	125	习题	169
5.1.3 成员函数的特性	126	第6章 继承	176
5.2 对象的定义和使用	128	6.1 继承的概念	176
5.2.1 对象的定义方法	128	6.1.1 什么是继承	176
5.2.2 对象成员的表示方法	128	6.1.2 派生类的定义格式	177
5.3 构造函数和析构函数	130	6.1.3 派生类对象的结构	177
5.3.1 构造函数	130	6.2 继承方式	178
5.3.2 析构函数	135	6.2.1 公有继承	178
5.4 静态成员	137	6.2.2 私有继承	180
5.4.1 静态数据成员	138	6.2.3 保护继承	181
5.4.2 静态成员函数	139	6.3 派生类的构造函数和析构函数	183
5.5 常对象和常成员	140	6.4 多继承	186
5.5.1 常对象	140	6.4.1 多继承的概念	186
5.5.2 常数据成员	141	6.4.2 多继承下的构造函数	188
5.5.3 常成员函数	141	6.4.3 二义性	191
5.6 对象指针和对象引用	143	6.5 虚基类	195
5.6.1 对象指针	143	6.5.1 虚基类的定义	195
5.6.2 this 指针	147		
5.6.3 对象引用	148		
5.7 对象数组	149		

6.5.2 虚基类的构造函数和析构函数	196
6.6 类型转换	198
6.6.1 派生类到基类的转换	198
6.6.2 基类到派生类的转换	199
6.7 应用实例——水果超市管理系统中类的继承关系	199
6.7.1 “正价水果类” RegularFruit 的设计	199
6.7.2 “特价水果类” DiscountFruit 的设计	201
6.7.3 验证	203
习题	204
第7章 多态	210
7.1 多态的描述	210
7.1.1 什么是多态	210
7.1.2 多态的分类	210
7.1.3 多态的实现方式	211
7.2 运算符重载	212
7.2.1 运算符重载的定义	212
7.2.2 双目运算符重载	213
7.2.3 单目运算符重载	218
7.3 虚函数	223
7.3.1 虚成员函数	226
7.3.2 虚析构函数	227
7.4 抽象类	230
7.4.1 纯虚函数	230
7.4.2 抽象类与具体类	231
7.5 应用实例——水果超市管理系统中虚函数的使用	232
习题	236
第8章 模板和 STL	243
8.1 模板简介	243
8.2 函数模板	243
8.3 类模板	245
8.3.1 类模板的定义	246
8.3.2 类模板用作函数的参数	248
8.3.3 类模板用作基类	248
8.4 STL	249
8.4.1 STL 简介	249
8.4.2 容器	250
8.4.3 迭代子	251
8.4.4 算法	253
习题	254
第9章 异常处理	256
9.1 简介	256
9.2 基本语法	257
9.3 构造函数、析构函数和异常	261
9.4 继承和异常	264
9.5 标准库中的异常类	264
习题	266
第10章 ACM-ICPC 简介及在线实训	267
10.1 ACM-ICPC 简介	267
10.2 在线实训模式	267
10.3 实训练习	269
习题	275
附录 A C++常用运算符的功能、优先级和结合性	276
附录 B 水果超市管理系统源程序清单	277
参考文献	286

第 1 章

C++程序设计概述

CHAPTER 01

C++语言是一种应用较为广泛的面向对象的程序设计语言，用以实现面向对象的程序设计。要进行 C++程序设计，就要了解面向对象程序设计的一些相关概念。本章主要介绍 C++语言的特点和 C++程序的基本构成，并简要介绍面向对象程序设计的相关概念，让大家初步了解面向对象的程序设计方法。

本章要点：

- C++语言的历史和演进
- C++语言的特点
- 面向对象的基本概念
- C++程序的基本结构和开发环境

1.1 C++语言的历史和演进

1.1.1 计算机语言的发展

计算机语言的发展经历了 5 个阶段，即机器语言、汇编语言、高级语言、查询语言和面向对象与自然语言。

机器语言是第一代计算机语言，它直接用二进制代码来编写程序，即用“0”和“1”这样的二进制码组成一串代码。程序不需要经过翻译，计算机就能直接识别，也不需要经过编译，计算机就可以直接运行，因此运行速度最快。

汇编语言也称低级语言，它是第二代计算机语言。汇编语言采用助记符来编写计算机的指令，然后通过汇编程序将助记符代码翻译成机器语言，再由计算机来执行。汇编语言程序的编写与计算机硬件密切相关，对于不同的 CPU，其语法规则也不相同，因此程序的可移植性差。但它克服了直接利用机器语言编程所带来的不方便以及工作量大的弱点，同时程序的运行速度也相当快。

高级语言的语法更接近于人类语言和数学表达式，属于面向过程的语言，是第三代计算机语言。高级语言程序需要经过编译或解释等翻译过程才能由计算机来执行，程序独立于计算机。在众多的高级语言中，C 语言仍是目前学习程序设计的一个主流语言。

查询语言是第四代计算机语言，属于非程序语言，它只描述问题但不叙述解决问题的步骤。查询语言主要包括进行数据库查询的结构化查询语言（Structured Query Language, SQL）或进行工程绘图的 AutoCAD。这种语言必须事先通过预处理器转换成第三代程序设计语言才能编译成可执行码。

面向对象与自然语言属于第五代计算机语言，是一种更高级的语言，这种语言的每个对象拥有自己的属性和方法，对象具有代码重用（Reused）、继承（Inheritance）、封装（Encapsulation）和多态（Polymorphism）等特性。

1.1.2 C++语言的产生

C++语言由 C 语言演变而来，其发展经历了从早期语言到结构化程序设计语言，从面向过程到非过程化程序设计语言的阶段。

- 1967 年, Martin Richards 为编写操作系统软件和编译程序开发了 BCPL 语言 (Basic Combined Programming Language)。
- 1970 年, Ken Thompson 在继承 BCPL 语言许多优点的基础上开发了实用的 B 语言。
- 1972 年, 贝尔实验室的 Dennis Ritchie 在 B 语言的基础上经过充实和完善, 开发出了 C 语言。它是一种面向过程的语言。
- 1980 年, 美国 AT&T 的 Bjarne Stroustrup 开始着手对 C 语言进行扩充和改造, 开发出一种过程性与面向对象相结合的程序设计语言。最初他把这种新的语言叫做“带类的 C”, 经过一系列改进, 1983 年, 被正式命名为 C++ (C Plus Plus)。

1.1.3 C++语言的主要版本

1986 年, 正式发布了第一个商业化的 C++ 版本; C++ 2.0 版本是 1989 年出现的, 它对第一个版本作了重大的改进, 加入了类的多继承性; 1991 年的 C++ 3.0 版本又增加了模板; C++ 4.0 版本则增加了异常处理、命名空间和运行时类型识别 (RTTI) 等功能。1994 年, ANSI 公布了标准 C++ 草案, ANSI C++ 标准草案是以 C++ 4.0 版本为基础制定的, 并于 1997 年正式通过并发布。

值得一提的是, 目前使用的 C++ 编译系统有一些是早期推出的, 并未全部实现 ANSI C++ 标准建议的所有功能。

当前应用较为广泛的 C++ 有 Microsoft 公司的 Visual C++ (简称 VC++) 和 Borland 公司的 Borland C++ (简称 BC++)。本书以 Microsoft Visual C++ 6.0 集成开发环境为例, 介绍 ANSI C++ 标准语言。

1.2 C++语言的特点

C++ 语言从 C 语言进化而来, 是 C 语言的超集。其主要特点表现在两方面: 一是全面兼容 C 语言, 二是支持面向对象的程序设计方法。也就是说, C++ 既可以用于面向过程的结构化程序设计, 也可以用于面向对象的程序设计, 是一种功能强大的混合型程序设计语言。

1.2.1 面向过程

面向过程的程序设计方法是把求解问题中的数据定义为不同的数据结构，以功能为中心进行设计，用一个函数实现一个功能。面向过程的程序设计方法中，所有的数据都是公用的，一个函数可以使用任何一组数据，而一组数据又能被多个函数所使用，函数与其操作的数据是分离的；其控制流程由程序中预定的顺序来决定，通过分析得出解决问题所需要的步骤（功能），然后用函数把这些步骤一一实现，使用的时候再依次对函数进行调用。

面向过程的 C++ 主要体现在 C++ 对 C 语言的全面兼容，C++ 不但保持了 C 语言的全部优点，而且继承了 C 语言的语法规则，多数 C 语言的程序都可以不加修改地移植到 C++ 的环境中使用。

1.2.2 面向对象

面向对象的程序设计方法是把求解问题中所有的独立个体都看成是各自不同的对象，将与对象相关的数据和对数据的操作都封装在一起，数据和操作是一个不可分割的整体。它是以数据为中心来描述系统，其控制流程由运行时各种事件的实际发生来触发，而不再由预定顺序来决定。它建立对象的目的是为了完成一个个步骤，而是为了描述某个对象在整个解决问题的步骤中的行为。

面向对象的 C++ 主要体现在：C++ 语言具有面向对象语言的基本要素，既包括对象和类的概念，又充分支持面向对象方法中的封装性、继承性和多态性 3 个特征。与面向过程的语言相比，类和对象的使用使 C++ 程序中各个模块的独立性、程序的可读性和可移植性更强，程序代码的结构更加合理，更易于程序扩充。这些优点对于设计、编制及调试一些大型的程序和软件尤为重要。

面向对象和面向过程的根本区别在于封装之后，面向对象提供了面向过程不具备的特性，最主要的就是继承和多态。

为进一步加强读者对面向对象程序设计方法的理解，本章先简要介绍几个面向对象的基本概念，更多的概念会在后续章节中详细讲解。

1.3 面向对象的相关概念

面向对象方法是分析问题和解决问题的新方法，其基本出发点就是尽可能按照人类认识世界的方法和思维方式来分析和解决问题。面向对象方法以对象为最基本的元素，是一种由对象、类、封装、继承、消息和多态等概念来构造系统的软件开发方法。

1. 对象（Object）

对象是现实世界中客观存在的某种事物，可以将人们感兴趣或要加以研究的事、物和概念等都称为对象。对象既能表示结构化的数据，也能表示抽象的事件、规则以及复杂的工程实体等。

在面向对象的系统中，对象是一个将数据属性和操作行为封装起来的实体。数据描述了对对象的状态，是对象的静态特性；操作用来描述对象的动态特性，可以操纵数据、改变

对象的状态。

对象之间传递信息是通过消息实现的，当一个对象对另一个对象发出的消息进行响应时，其操作才得以实现。

2. 类 (Class)

类是人们对客观事物的高度抽象。抽象是指抓住事物的本质特性，找出事物之间的共性，并将具有共同特性的事物划分为一类，得到一个抽象的概念。例如，人、汽车、房屋、水果等都是类的例子。

类是一种类型，是具有相同属性和操作的一组对象的集合。类的作用是定义对象，类给出了属于该类的全部对象的抽象定义，而对象则是类的具体化，是符合这种定义的一个类的实例。类还可以有子类和父类，子类通过对父类的继承形成层次结构。

把一组对象的共同特性加以抽象并存储在一个类中的能力是面向对象技术中最重要的一点，是否建立了一个丰富的类库则是衡量一个面向对象程序设计语言成熟与否的重要标志。

3. 封装 (Encapsulation)

封装是面向对象方法的重要特征之一，是指将对象的属性和行为（数据和操作）包裹起来形成一个封装体。该封装体内包含对象的属性和对象的行为，对象的属性由若干个数据组成，而对象的行为则由若干操作组成，这些操作是通过函数来实现的，也称之为方法。

封装体具有独立性和隐藏性。独立性是指封装体内所包含的属性和行为构成了一个不可分割的独立单位。隐藏性是指封装体内的某些成员（数据或者方法）在封装体外是不可见的，既不能被访问，也不能被改变，这部分成员被隐藏了，具有安全性。一般情况下，封装体和外界的联系是通过接口来进行的。

4. 继承 (Inheritance)

继承是面向对象方法的另一个重要特征，是提高重用性的重要措施。继承提供了创建新类的一种方法，表现了特殊类与一般类的关系。特殊类具有一般类的全部属性和行为，并且还具有自己特殊的属性和行为，这就是特殊类对一般类的继承。通常将一般类称为基类（父类），而将特殊类称为派生类（子类）。

使用继承可以使人们对事物的描述变得简单。例如，已经描述了动物这个类的属性和行为，由于哺乳动物是动物的一种，它除了具有动物这个类的所有属性和行为外，还具有自己特殊的属性和行为，这样在描述哺乳动物时就只需要在继承动物类的基础上再加入哺乳动物所特有的属性和行为就可以了。因此，哺乳动物是特殊类，是子类；而动物是一般类，是父类。

继承的本质特征就是行为共享。通过行为共享，可以减少冗余性，很好地解决软件重用性的问题。

5. 消息 (Message)

对象与对象之间是通过消息进行通信的，这种通信机制叫做消息传递。消息传递是通过函数调用来实现的。当一个消息发送给某个对象时，该消息中包含要求接收对象去执行

某些活动的信息，接收到消息的对象经过解释，然后予以响应。发送消息的对象不需要知道接收消息的对象是如何对请求予以响应的。

6. 多态 (Polymorphism)

多态性指的是一种行为对应着多种不同的实现。在同一个类中，同一种行为可以对应不同的实现，如函数重载和运算符重载。同一种行为在一般类和它的各个特殊类中可以具有不同的实现，即不同的对象在收到同一消息时可产生完全不同的结果。多态性的表现就是允许不同类的对象对同一消息做出响应，即同一消息可以调用不同的方法，而实现的细节则由接收对象自行决定。

多态的实现受到继承性的支持，利用类的继承的层次关系，把具有通用功能的消息存放在高层次，而把这一功能行为的不同实现放在较低层次，在这些低层次上生成的对象能够给通用消息以不同的响应。多态性是面向对象方法的又一特征，很好地解决了应用程序中函数同名的问题。

1.4 C++程序的基本结构和开发环境

1.4.1 C++程序的基本结构

为了更好地了解 C++ 程序以及 C++ 程序的基本结构，下面先看几个简单的例子。

例 1.1 输出一行字符：“Welcome to C++ program!”。

```
// my first program in C++
#include <iostream>                //用 cout 输出时需要用此头文件
using namespace std;              //使用命名空间 std
int main()
{
    cout<<"Welcome to C++ program!\n"; //用 C++的方法输出一行
    return 0;
}
```

程序运行后输出结果为：

Welcome to C++ program!

分析：

(1) 第一行“// my first program in C++”是注释 (comments) 行。在 C++ 程序中，所有以“//”开始的程序行都被认为是注释行。这些注释行既可以单独放在一行，也可以出现在一行语句之后。注释写在程序源代码内，用来对程序作简单解释或描述，对程序本身的运行不会产生影响。注意：它是单行注释，不能跨行。

如果要对多行进行注释，可采用“/*.....*/”方式。“/*”符号和“*/”符号之间的所有内容均为注释内容，可以包含多行内容。通常，在编写程序过程中，对于暂时不用的程序段可以用多行注释方式将其屏蔽，需要时再去掉注释，这样可避免误操作而删除有用的代码。

(2) 第二行“`#include <iostream>`”是一个预处理命令。文件 `iostream` 的内容是提供输入或输出时所需要的一些信息。因为这类文件都放在程序开头, 所以称为“头文件”。

注意：在C语言中，所有头文件都带后缀.h（如stdio.h），而按C++标准要求，由系统提供的头文件不带后缀.h（iostream），用户自己编制的头文件可以有后缀.h。在C++程序中也可以使用C语言编译系统提供的带后缀.h的头文件，如“#include <math.h>”。

(3) 第三行“`using namespace std;`”的意思是“使用命名空间 `std`”。C++标准库中类和函数是在命名空间中声明的，因此程序中如果需要使用 C++标准库中的有关内容，就需要用“`using namespace std;`”语句声明，表示要用到命名空间 `std` 中的内容。如果程序中有输入或输出，必须使用“`#include <iostream>`”命令提供信息，同时要用“`using namespace std;`”语句使程序能够使用这些信息，否则程序编译时将出错。本书中所有程序示例的开头都包含这两行语句。

(4) 第四行是主函数 `main()`，它是一个程序执行的入口函数。不管它是在代码的开头、结尾还是中间，此函数中的代码总是在程序开始运行时第一个被执行。主函数的内容紧跟在它的声明之后，由花括号“`{}`”括起来，是函数体，它由若干条 C++ 语句构成，每一条 C++ 语句都必须以“`;`”结尾，表示一条语句结束。

标准 C++ 要求，main 函数必须声明为 int 型，表示 main 函数返回一个整型的函数值，即“int main()”。语句“return 0;”的作用是向操作系统返回 0。如果程序不能正常执行，则会自动向操作系统返回一个非零值，一般为-1。

目前使用的一些 C++ 编译系统并未完全执行标准 C++ 的规定，如果主函数首行写成 “void main()” 也能通过。本书的所有例题都按 C++ 标准规定写成 “int main()”，读者在初学时要养成这个习惯，以免编写的程序在严格遵循 C++ 标准的编译系统中通不过。只要记住：在 main 前面加 int，同时在 main 函数的最后加一条语句 “return 0;” 即可。

(5) 在 C++ 程序中, 一般用 `cout` 进行输出。`cout` 是 C++ 系统定义的对象名, 称为输出流对象, 通常和 “<<” 一起实现输出功能。与 `cout` 对应的还有 `cin` 对象名, 称为输入流对象, 通常和 “>>” 一起实现从键盘输入信息的功能。C++ 也可以使用 C 语言中的 `scanf` 和 `printf` 函数进行输入和输出。

例 1.2 求 x 和 y 两个整数之和。

```
//求两个整数之和
#include <iostream >
using namespace std;
int main()
{
    int x,y,sum;
    cout<<"Input first integer:";
    cin>>x;
    cout<<"Input second integer:";
    cin>>y;
    sum=x+y;
}
```

```
    cout<<"Sum is "<<sum<<endl;    //输出 sum 结果
    return 0;                        //若程序正常结束，向操作系统返回一个 0
}
```

该程序的作用是求两个整数 x , y 之和 sum 。程序中的语句 “ $\text{int } x,y,sum;$ ” 用来声明变量 x , y 和 sum 为 int (整型) 变量。语句 “ $\text{sum}=x+y;$ ” 是一个赋值语句，表示将 x 与 y 的值相加，其结果赋给变量 sum 。在 “//” 后的部分表示注释。“ endl ” 是 C++ 输出时的控制符，表示本行结束需要换行，其作用相当于 “ $\backslash n$ ”。

该程序经过编译、链接和运行后，在屏幕上显示：

```
Input first integer: 7
Input second integer: 8
Sum is 15
```

注意：7和8两个整数是从键盘输入的。

例 1.3 输入两个整数 a 和 b ，输出其中较大的数。

```
#include <iostream>
using namespace std;
int main()
{
    int Max(int x,int y);    //对 Max 函数进行声明
    int a,b,m;
    cout<<"Input a,b:";
    cin>>a>>b;              //从键盘上输入变量 a 和 b 的值
    m=Max(a,b);              //调用 Max 函数，并把结果赋给变量 m
    cout<<"max="<<m<<endl;
    return 0;
}

int Max(int x,int y)        //定义 Max 函数
{
    int z;
    if (x>y) z=x;
    else z=y;
    return(z);
}
```

程序运行后，结果如下：

```
Input a,b: 4 5
max=5
```

该程序由两个函数组成：主函数 $\text{main}()$ 和被调用函数 $\text{Max}()$ 。主函数用来输入两个变量 a 和 b 的值，调用 $\text{Max}()$ 函数时，将两个实际参数 a 和 b 的值分别传给 $\text{Max}()$ 函数的形式参数 x 和 y ，执行 $\text{Max}()$ 函数得到一个最大值，输出其结果。函数 $\text{Max}()$ 的作用是找出 x 和 y 中的较大值赋给变量 z ，通过 return 语句将 z 值返回给主函数的调用处，并赋给变量 m 。

通过以上例题可以看出，C++ 程序由以下几个部分组成。

(1) 预处理命令

C++程序开始经常出现含有以“#”开头的命令，它们是预处理命令。C++提供了3类预处理命令：文件包含命令、宏定义命令和条件编译命令。本书很少涉及后两种预处理命令，因此不做详细介绍，读者需要时可以参考相关书籍。

示例中出现的“`#include <iostream>`”是文件包含命令，是一种常用的预处理命令，其中 `include` 是关键字，“`<>`”中是被包含的文件名。由于 `iostream` 头文件中包含有预定义的提取符“`>>`”和插入符“`<<`”等内容，在编写程序时，若要使用这些运算符就必须包含 `iostream` 头文件。如果在源程序中使用了系统提供的一些函数，如数学函数（`sin()`、`cos()`和指数等），就应该在预处理命令中包含 `math` 头文件，即在源程序开始部分加上“`#include <math>`”。

(2) 输入/输出

C++程序中的输入/输出操作是通过输入/输出流对象 `cin` 和 `cout` 来实现的，在实现的过程中经常会用到运算符和表达式等。

(3) 函数

一个C++程序可以由若干个文件组成，每个文件又可以包含多个函数，函数是构成C++程序的基本单位。C++程序中至少要包含一个主函数 `main()`，并且只能有一个 `main()` 函数。C++程序的执行总是从 `main()` 函数开始，而其余函数的执行则只能通过主函数来调用或被主函数调用的其他函数来调用执行。函数的调用既可以嵌套，又可以递归。被调用的其他函数可以是系统提供的库函数，也可以是用户自定义的函数。例如，例 1.3 的 C++程序就是主函数 `main()` 调用了用户自定义函数 `Max()`。

(4) 语句

语句是构成C++程序的基本单元，而函数就是由若干条语句组成的，但空函数没有语句。语句由单词组成，单词间用空格符分隔，C++中的每条语句必须以分号结束，分号是C++语句的必要组成部分。在使用分号时，初学者一定要注意，不能在编写程序时随意加分号，该有分号的一定要加，不该有分号的一定不能加。

在C++语句中，表达式语句最多，它由一个表达式后加一个分号组成。任何一个表达式加上分号都可以组成一条语句。只有分号而没有表达式的语句为空语句。

除了表达式语句和空语句外，还有复合语句、控制语句和函数调用语句等，将在第2章详细讲解。

(5) 变量

多数程序都需要声明和使用变量。变量的类型有很多种，包括 `int` 型、`float` 型和 `char` 型。`int` 型又分为 `long` 和 `short` 两种，`float` 型中还有 `double` 型和 `long double` 型，`char` 型又分 `signed` 和 `unsigned` 两种。

对象 `cout` 和 `cin` 属于“类”类型。广义地讲，对象包含了变量，即变量也称为一种对象。狭义地讲，将对象看做是类的实例，对象是指某个类的对象。

(6) 其他

一个C++程序除了上述5个部分以外，还有其他组成部分。例如，符号常量和注释信息也是程序的一部分。

C++中尽量把常量定义为符号常量，如用预处理命令中的宏定义命令：


```
#define PI 3.14159
```

其中, **define** 是关键字, 表示定义一个符号常量 **PI**, 它所代表的常量值是 3.14159。

另一种方法是使用常量关键字 **const**, 例如:

```
const double PI=3.14159;
```

这也是将一个符号常量 **PI** 定义为 3.14159。

使用符号常量会带来很多好处: 方便修改、便于移植、增加可读性等。

对于较复杂的或大型的软件, 必须添加注释信息。在 C++ 程序的任何位置都可以插入注释信息, 这些注释信息增加了程序的可读性。

另外, 在编写程序时还需要注意以下两个问题。

(1) 程序的书写必须规范, 同一层语句同列书写。同一层次的花括号必须对应显示在同一列上。内层语句向里缩进, 以便于程序的阅读和修改。

(2) 在 C++ 程序中, 严格区分字母的大小写。

例如:

```
int a,A;
```

表示定义两个不同的变量 **a**、**A**。

1.4.2 C++程序的实现

C++ 源程序的实现与其他高级语言源程序实现的原理是一样的, 一般都要经过编辑、编译、链接和运行 4 个步骤。

1. 编辑

编辑就是将编写好的 C++ 源程序输入到计算机中或者对已经编写好的 C++ 源程序进行修改, 以符合 C++ 语言所规定的语法规则, 并生成磁盘文件的过程。C++ 的源程序与其他语言的源程序一样, 可以用计算机中任何相应的文本编辑软件进行编写, 形成一个程序源文件。在实际应用中, 所选用的 C++ 编译器本身都提供编辑器, C++ 源程序文件可以在其中编写, 十分方便。C++ 的源程序文件名通常以 **.CPP** 为扩展名。

2. 编译

用户编写好的源程序是不能被计算机所识别和运行的, 必须将源程序文件按该语言的语法规则翻译成二进制表示的程序文件, 即目标代码文件, 这一过程称为编译。

C++ 的源程序在使用某种 C++ 语言的编译器进行编译时, 要对源程序的语法和逻辑结构进行分析和检查, 最终生成扩展名为 **.OBJ** 的目标代码文件。

3. 链接

用户编写的程序可能被存放在多个源文件中, 被编译生成的目标代码文件也可能分布在不同的地方, 因此需要把它们链接到一起。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也需要系统提供的库文件中的一些代码, 因此需要把它们链接起来。这种链接工作由编译系统中的连接器来完成, 连接器把目标代码文件和库中的某些文件进行链接, 生成一个可执行文件。可执行文件的扩展名为 **.EXE**, 库文件的扩展名为 **.LIB**。

4. 运行

最后,生成的可执行文件可以在计算机系统中执行。用户可以根据运行结果来判断和检查程序是否正确或进行实际的应用。

一个程序编写好后,在生成可执行文件之前需要改正编译和链接时出现的一切致命错误和警告错误,这样才可能生成无错的可执行文件。当程序中存在警告错误时,也可生成可执行文件,但一般要求改正这些错误后再重新进行编辑、编译、链接和运行,否则有可能造成错误的结果。

C++程序的编辑、编译、链接和运行的实现过程如图 1-1 所示。

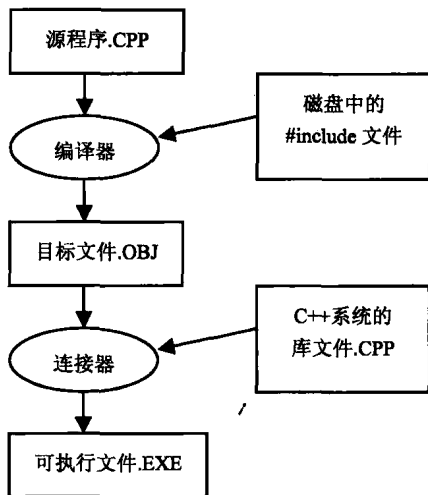


图 1-1 C++程序的实现过程

1.4.3 C++程序的开发环境

就像文字处理要通过 Microsoft Word 之类的软件来完成一样, C++程序的编写也需要一个编辑环境。在众多的 C++开发环境中, Microsoft 公司的 Visual C++ 6.0 是较为流行的可视化编程工具, 它将窗口、对话框、菜单、工具栏和快捷键等集成为一个整体, 用于观察和控制整个开发过程。本书以 Microsoft 公司的 Visual C++ 6.0 为背景, 介绍 C++程序的开发。

1. Visual C++ 6.0 的安装与启动

如果计算机中未安装 Microsoft Visual C++ 6.0, 只需找到 Microsoft Visual Studio 光盘, 执行其中的 setup.exe 文件, 并按屏幕上的提示进行安装即可。

安装结束后, 在 Windows “开始” 菜单的 “程序” 子菜单中就会出现 “Microsoft Visual Studio” 子菜单。在需要使用 Visual C++ 6.0 时, 依次选择 “开始→程序→ Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0” 命令, 屏幕上将显示 Visual C++ 6.0 的版权页, 单击 close 按钮, 弹出 Microsoft Visual C++ 6.0 的主窗口, 如图 1-2 所示。

在 Visual C++ 主窗口的顶部是菜单栏, 左侧为项目工作区窗口, 右侧为程序编辑区窗口。工作区窗口用来显示所设定的工作区的信息, 程序编辑区窗口用来输入和编辑源