普通高等教育“十二五”规划教材

C++面向对象程序设计基础教程

付永华　主　编
黄　玲　副主编

中国电力出版社
CHINA ELECTRIC POWER PRESS

普通高等教育“十二五”规划教

C++面向对象程序设计基础教程

主　编　付永华
副主编　黄　玲
编　写　傅尔胜　李　向　宋　涛
　　王亚楠　肖　连　杨金霞
主　审　李建文

中国电力出版社
CHINA ELECTRIC POWER PRESS

内 容 提 要

本书为普通高等教育“十二五”规划教材。本书具有以下特点：一，源于教学经验，符合教学规律——结合多年教学经验，融合各学校该课程的教学大纲，采用很多便于读者巩固所学知识的教学特征，设计了教学体系结构和教学内容，符合教学规律；二，实例易于理解——在保证程序结构严谨、清晰的基础上，尽量用生活实例来解释理论，而且大胆采用了趣味横生的一些例子，将知识点囊括其中，提高学生的学习兴趣；三，注重实践和提高——增加了设计部分，并提供了所有的源程序和设计思路，供学有余力的读者学习。

本书可作为普通高等院校程序设计基础课程的本、专科教材（可以根据本科、专科教学要求的不同进行适当取舍），对致力于数据库系统、交互式界面、应用平台、分布式系统、网络管理、CAD 技术、人工智能等领域的开发人员亦有参考价值。

图书在版编目（CIP）数据

C++面向对象程序设计基础教程 / 付永华主编. —北京：中国电力出版社，2012.12

普通高等教育“十二五”规划教材

ISBN 978-7-5123-3790-9

Ⅰ. ①C… Ⅱ. ①付… 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（2012）第 286166 号

中国电力出版社出版、发行

（北京市东城区北京站西街 19 号 100005 http://www.cepp.sgcc.com.cn）

航远印刷有限公司印刷

各地新华书店经售

*

2012 年 12 月第一版 2012 年 12 月北京第一次印刷

787 毫米×1092 毫米 16 开本 19.5 印张 474 千字

定价 **36.00** 元

前　言

自从20世纪60年代，面向对象这一概念被提出到现在，不但在编程方面，已经形成成熟的思想体系，成为软件开发的主流技术，而且，在信息科学、软件工程、人工智能和认知科学等领域，也产生了重要的影响。这种技术将客观世界直接映射到面向对象的世界，是一种全新的认知世界的方式。这种认知，可以通过编程语言来描述或者解释。

支持面向对象的语言很多，从最古老的Simula67语言和Smalltalk语言，到前几年比较流行的C#，每一种新的支持面向对象的语言的出现，都会掀起编程领域的一阵浪花。在这些璀璨的浪花中，一直持久澎湃的，应该属于C++。

“真正的程序员用C++，聪明的程序员用Delphi”。在编者上大学的时候，这句话就流行于编程领域。现在，因为各种因素，在网络时代，Delphi王朝已日渐没落，但C++风采依旧。

作为一门成熟的重量级语言，C++以强大灵活的语言机制、深邃的内涵、广博丰富的外延吸引了许多程序员。伴随着C++成为ANSI标准，C++和C++支持的面向对象的概念和应用已超越了程序设计和软件开发，扩展到很宽的范围。在数据库系统、交互式界面、应用平台、分布式系统、网络管理、CAD技术、人工智能等领域都可以看到C++的身影。

C++是一种高效应用的程序设计语言，它既可进行过程化程序设计，也可进行面向对象程序设计。学好C++，很容易触类旁通其他语言，C++架起了通向强大、易用、真正的软件开发应用的桥梁。因此，在深受程序员广泛喜爱的同时，基于技术的发展和学生的需求，目前，很多大学把支持面向对象编程的C++作为大学生的第一门语言，而支持结构化的语言，据了解，在国内的大学里，基本已经难觅踪迹。

但是，在进行“面向对象程序设计”、“C++高级语言程序设计”等课程的教学、课后辅导时，编者发现学生并不能够很好地学习和掌握C++，一开始劲头很足，打算学好C++，后来渐渐觉得课本晦涩，例题无趣，便失去了兴趣。很多初涉C++的程序员亦称接触到的教材或者书籍难觅佳品。

而且，有的人认为，编程无需教材，只要在网上找一些例子，多练习、多调试就够了，这样做是不错的，而且也应该如此。但这种情况，往往会造成不严谨，导致程序可阅读性不强。甚至，因为没有扎实的理论基础，对某些细节的东西，把握不到位，可能导致某些隐患。基于此，编者一直希望有一本严谨、务实，而又适用于各方面人员的教材。

该书的几位编者一直主讲“面向对象程序设计”、“C++高级语言程序设计”、“C 语言程序设计”课程，有着多年的、丰富的一线教学经验，主持、参与了大量的横向课题、系统开发，有丰富的实践经验。知道学生最需要的是什么，深知如何去讲授才能让学生更好地理解知识。

从讲义到现在的书稿，已有三届学生、四个专业和近千名学生用过，可以说历经学生试用、反馈、修改、锤炼。经过对比分析，该书具有以下特点。

（1）例题经过精心设计，不是简单的C语言或者其他语言的重复。

（2）例题实用性较强，尤其是很多例题选用了游戏小程序，趣味性很强，避免了学生因为面向对象的抽象的理论知识从而产生的厌学心理。

（3）所有例题都经过运行调试，确保学生正确输入即可使用，代码注释丰富，利于学习。

（4）书中的“注意”、“规则”部分是几位编者通过多年教学、上机指导积累下来的宝贵经验，是重点、难点的总结。

（5）尝试用生活中的现象解释纯粹理论知识，增加可读性。

（6）该书的讲义有试用期，历经实践教学锤炼。

（7）力图跟踪国内外最新技术，增加课堂授课和实际应用的联系。

（8）提供在线的学习帮助系统。

（9）采用对错例题对比法，提前将学生容易犯的错误着重指出，提高了上机的质量和学生上手速度。

（10）第11章为学生提供实用、完整的程序代码，供能力突出的学生进一步研究提高。

（11）对部分不重要且其他教材中讲述比较详细的内容一笔带过，或者采用图的形式描述，侧重突出。

郑州航空工业管理学院王亚楠编写了第1章和第2章，郑州航空工业管理学院付永华编写了第3章和第11章，哈尔滨理工大学的黄玲编写了第4章、第5章和附录，郑州航空工业管理学院的傅尔胜编写了第6章，郑州航空工业管理学院李向编写了第7章，郑州航空工业管理学院宋涛编写了第8章，郑州航空工业管理学院肖连编写了第9章，河南省图书馆的杨金霞编写了第10章。郑州航空工业管理学院的王亚楠和付永华负责全书的程序调试。全书由郑州航空工业管理学院付永华统稿，由陕西科技大学李建文主审。

本书成书也几经波折，从该书初成讲义到出版，将近三年。

本书从构思体系到丰富内涵，一点一滴地积累，居然也有了四十余万文字；将打印的修改稿收集到一块儿，居然也充满了装显示器的纸箱；屏幕上，鼠标要拖动一会儿才能看到最后。看文字如同流水般轻轻划过屏幕，非常高兴。如果它能遂编者的心愿——为读者带来帮助，有些许提高，这是编者最大的欣慰。

真的要诚挚地表达编者的谢意：

感谢郑州航院信息科学学院刘永院长提供工作的支持、写作的灵感、多年编程的经验和部分源码的支持。

感谢李建文教授于百忙之中，审阅全书，并提出中肯的意见。

感谢刘冬青、谢晶和马晓林等同学，感谢她们将同学的意见汇总，及时和编委进行沟通，从学生的角度，提出自己的感受。

感谢所有正在帮助和曾经帮助编者的所有人。

限于水平本书存有许多错误和不足，敬请读者斧正，编者不胜感激。

编者信箱：fuyonghua_12@zzia.edu.cn。

编　者

2012年11月

目　录

第1章　面向对象技术暨C++概述

1.1　面向对象技术概述

面向对象技术是一种全新设计和构造软件的技术，它使计算机解决问题的方式更符合人类的思维方式，更能直接地描述客观世界。通过增加代码的可重用性、可扩充性和程序自动生成功能来提高编程效率，并且大大减少软件维护的开销，已经被越来越多的软件设计人员所接受。希望读者通过本章的介绍，读者能从宏观上了解面向对象技术。本章首先介绍面向对象技术的基本概念、基本特征，面向对象与面向过程程序设计的区别，然后介绍目前流行的几种面向对象程序设计语言，特别强调C++对面向对象技术的支持及其发展现状，其中还涉及.NET技术。

1.1.1　面向对象技术的基本概念

1. 面向对象技术概述

面向对象技术是一种新的软件技术，其概念来源于程序设计。从20世纪60年代提出的面向对象的概念，到现在已发展成为一种比较成熟的编程思想，并且逐步成为目前软件开发领域的主流技术。同时，它不仅局限于程序设计方面，而且已经成为软件开发领域的一种方法论。它对信息科学、软件工程、人工智能和认知科学等都产生了重大影响，尤其在计算机科学与技术的各个方面影响深远。面向对象技术，可以将客观世界直接映射到面向对象解空间，从而为软件设计和系统开发带来革命性影响。

2. 面向对象与面向过程的区别

在面向对象程序设计（Object Oriented Programming，OOP）方法出现之前，程序员用面向过程的方法开发程序。面向过程的方法把密切相关、相互依赖的数据和对数据的操作相互分离，这种实质上的依赖与形式上的分离使得大型程序不但难于编写，而且难于调试和修改。在多人合作中，程序员之间很难读懂对方的代码，更谈不上代码的重用。由于现代应用程序规模越来越大，对代码的可重用性与易维护性的要求也相应提高，面向对象技术便应运而生了。

面向对象技术是一种以对象为基础，以事件或消息来驱动对象执行处理的程序设计技术。它以数据为中心而不是以功能为中心来描述系统，数据相对于功能而言具有更强的稳定性。它将数据和对数据的操作封装在一起，作为一个整体来处理，采用数据抽象和信息隐蔽技术，将这个整体抽象成一种新的数据类型——类，并且考虑不同类之间的联系和类的重用性。类的集成度越高，就越适合大型应用程序的开发。另一方面，面向对象程序的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，更符合实际。事件驱动程序执行围绕消息的产生与处理，靠消息循环机制来实现。更重要的是，可以利用不断扩充的框架产品MFC（Microsoft Foundation Classes），在实际编程时采用搭积木的方式来组织程序，站在“巨人”肩上实现自己的愿望。面向对象的程序设计方法使得程序结构清晰、简单，提高了代码的重用性，有效地减少了程序的维护量，提高了软件的开发效率。

例如，用面向对象技术来解决学生管理方面的问题，重点应该放在学生上，要了解在管理工作中，学生的主要属性，要对学生做些什么操作等，并且把它们作为一个整体来对待，形成一个类，称为学生类。作为其实例，可以建立许多具体的学生，而每一个具体的学生就是学生类的一个对象。学生类中的数据和操作可以提供给相应的应用程序共享，还可以在学生类的基础上派生出大学生类、中学生类或小学生类等，实现代码的高度重用。

在结构上，面向对象程序与面向过程程序有很大不同，面向对象程序由类的定义和类的使用两部分组成，在主程序中定义各对象并规定它们之间传递消息的规律，程序中的一切操作都是通过向对象发送消息来实现的，对象接到消息后，启动消息处理函数完成相应的操作。

类与对象是面向对象程序设计中最基本且最重要的两个概念，有必要仔细理解和彻底掌握。它们将贯穿全书并且逐步深化。

3. 面向对象技术的基本特征

面向对象技术强调在软件开发过程中面向客观世界或问题域中的事物，采用人类在认识客观世界的过程中普遍运用的思维方法，直观、自然地描述客观世界中的有关事物。面向对象技术的基本特征主要有抽象性、封装性、继承性和多态性。

这些特征将放第 7 章去详细讲解。

1.1.2 面向对象程序设计语言和工具简介

20 世纪 60 年代，出现了最早的面向对象程序设计语言 Simula67 语言，具有了类和对象的概念，被公认为是面向对象语言的鼻祖。随后又出现了纯面向对象程序设计语言，如美国 Xerox Palo Alto 研究中心推出的 Smalltalk，它完整地体现并进一步丰富了面向对象的概念。进而出现了混合型面向对象程序设计语言，如 C++，这类语言一般是在其他语言的基础上开发出来的；还有与人工智能语言结合形成的面向对象程序设计语言，如 LOOPS、Flavors 和 CLOS；以及适合网络应用的面向对象程序设计语言，如 Java 语言等。下面简要介绍几种目前常用的面向对象程序设计语言。

1. 混合型面向对象程序设计语言 C++

C++是 AT&T Bell 实验室的 Bjarne Stroustrup 博士于 20 世纪 80 年代早期提出的，是迄今为止商业上最受欢迎的混合型面向对象程序设计语言。C++兼容了 C 语言并弥补了其缺陷，支持面向过程程序设计方法；增加了面向对象的能力，支持面向对象程序设计方法。许多软件公司都为 C++设计编译系统。如 AT&T、Apple、Sun、Borland 和 Microsoft 等，国内最为流行的是 Visual C++。同时，许多大学和公司也在为 C++编写各种不同的类库，其中 Borland 公司的 OWL（Object Windows Libray）和 Microsoft 公司的 MFC（Microsoft Foundation Class）是优秀的代表作，尤其是 MFC 在国内外都得到广泛应用。

C++被数以十万计的程序员应用到几乎每个领域中。早期的应用趋向于系统程序设计，有几个主要操作系统都是用 C++写出的：Compbell、Rozier、Hamilton、Berg、Parrington，更多系统用 C++做了其中的关键部分。C++还用于写设备驱动程序，或者其他需要在实时约束下直接操作硬件的软件。许多年来，美国的长途电话系统的核心控制依赖于 C++。图形学和用户界面是使用 C++最深入的领域，如 Apple Macintosh 或 Windows 的基本用户界面都是 C++程序。此外，一些最流行的支持 UNIX 中 X 的库也是用 C++写的。

本书就是以 C++为平台，描述面向对象技术。

2. 纯面向对象程序设计语言 Java

Java 是由 SUN 公司的 J.Gosling、B.Joe 等人在 20 世纪 90 年代初开发出的一种纯面向对象程序设计语言。Java 是标准的又是大众化的面向对象程序设计语言。首先，Java 作为一种解释型程序设计语言，具有简单性、面向对象性、平台无关性、可移植性、安全性、动态性和健壮性，不依赖于机器结构，并且提供了开发的机制，具有很高的性能；其次，它最大限度地利用了网络，Java 的应用程序（Applet）可在网络上传输，可以说是网络世界的通用语言；最后，Java 还提供了丰富类库，使程序设计者可以方便地建立自己的系统。因此，Java 具有强大的图形、图像、动画、音频、视频、多线程及网络交互能力，使其在设计交互式、多媒体网页和网络应用程序方面大显身手。

Java 程序有两种类型。一种是可在 Web 网页上运行的 Applet，称为小应用程序。考虑到网络环境、连接速度等原因，Applet 一般都比较小，适合客户端下载，很多网站利用 Java 开发出了商业网络平台，实现交互运行，还有大量的 Applet 嵌入到网页，使页面变得更加活泼生动。但 Applet 不能单独运行，必须嵌入在 HTML 文件中，由 Web 浏览器执行。另一种是 Application，即应用程序，可完成任何计算任务，运行时不必借助于 Web 浏览器，可单独执行。

Java 从 C++发展而来。Java 摒弃了 C++中许多不合理的内容，真正做到了面向对象。在 Java 中，一切都是对象。Java 通过 new 运算符创建对象，通过 new 运算符返回的对象引用来操纵对象，而不是直接操作指针，这样可以防止程序员的误操作而导致的错误。Java 通过内存垃圾收集机制，自动管理内存，不需要程序员显式地释放所分配的内存，从而大大减轻了程序员的负担。Java 与 C++都有类的概念，其最大的差异是 C++支持多重继承，而 Java 只支持单重继承。Java 抛弃多重继承是为了使类之间的继承关系更加清晰，不会造成任何混乱。

3. 可视化程序设计语言 Visual Basic

1991 年 Microsoft 公司推出了基于 BASIC 语言的可视化面向对象开发工具 Visual Basic，标志着软件设计和开发技术一个新时代的开始。在其带动下，相继产生了 Visual C++、Visual J++、Visual FoxPro 及 Borland Delphi、Power Builder 等众多可视化开发工具。这些工具的共同特点是，提供了 Windows 界面下一些常用界面元素样本。

所谓可视化技术一般是指软件开发阶段的可视化和对计算机图形技术和方法的应用。这里是指前者，即可视化程序设计，是应用可视化开发工具开发图形用户界面（GUI）应用程序的方法。软件开发人员不需编写大量代码去描述界面元素的外观和位置，而只需选定特定界面元素的样本，并用鼠标拖放到屏幕的窗体上，然后再通过不同的方法，编写一些容易理解的事件处理程序，就可完成应用软件的设计。

在 Visual Basic 中，既继承了 BASIC 语言所具有的语法简单、容易学习、容易使用、数据处理能力强的特点，又引入了面向对象、事件驱动的编程机制和可视化程序设计方法，大大降低了开发 Windows 应用程序的难度，有效地提高了应用程序开发的效率。同时，Visual Basic 还兼顾了高级编程技术，不仅可以编写功能强大的数据库应用程序、多媒体处理程序，还可以用来建立客户与服务器应用程序、通过 Internet 访问遍及全球的分布式应用程序、创建 ActiveX 控件及与其他应用程序紧密集成。它可以实现 Windows 的绝大部分高级功能，如多任务、多文档界面（MDI）、对象的链接与嵌入（OLE）、动态数据交换、动态链接库（DLL）子程序的调用等，尤其是动态链接技术，使得 Visual Basic 可以调用 Windows 系统

的各种资源。

但是，Visual Basic 存在语法不严格、开发出的系统稳定性较低的缺点。相对 Visual C++语言来说，Visual Basic 面向系统底层的编程能力有限，不适合开发系统监控程序，不适合设备驱动程序的开发。比较而言，Visual C++虽然学习起来难度较大，但开发出的系统稳定性高，同时还能使用 Visual C++做一些 Windows 系统下特殊应用的开发，如设备驱动程序等。

4. Visual C++

随着 C++逐渐成为 ANSI 标准，它迅速成为程序员最广泛使用的工具，其开发环境也随之不断地推出。Visual C++从 1.0 发展到 6.0 等版本，软件系统逐渐庞大，功能日益完善。

（1）Visual C++ 6.0。1986 年 Borland 公司开发了 Turbo C++，而后又推出了 Borland C++。Microsoft 公司于 20 世纪 80 年代中期在 Microsoft C 6.0 的基础上开发了 Microsoft C/C++ 7.0，同时引进了类库 MFC 1.0 版本，完善了源代码。这些版本都是依赖于 DOS 环境，或在 Windows 下的 DOS 模式下运行。随后 Microsoft 公司推出了 Microsoft C/C++ 8.0，即 Visual C++1.0 版本，它是 Microsoft 公司推出的第一个真正基于 Windows 环境下的可视化集成开发环境，将编辑、编译、连接和运行集成为一体。由于 Internet 的流行，在 4.0 版本中，Visual C++引进了为 Internet 编程而设计的新类库。5.0 版本也增加了一些新类，但注意力更多地集中在改善产品的界面上，以提供一个更好的在线帮助系统、更高级的宏能力和对在开发者组内进行类和其他代码共享的支持。6.0 版本在功能上做了进一步的改进。

Visual C++ 一直是用于创建高性能的 Windows 和 Web 应用程序与 Web 服务的最佳语言。Microsoft 公司自喻 Visual C++是所有开发语言及工具中的“旗舰”。Visual C++不仅是 C++语言的集成开发环境，而且与 Win32 紧密相连，利用 Visual C++可以完成各种各样的应用程序的开发，从底层软件直到上层直接面向用户的软件，而且 Visual C++强大的调试功能也为大型应用程序的开发提供了有效的排错手段。

（2）Visual C++.NET。Visual C++.NET 是 Microsoft 的新一代 Visual C++语言。2000 年 6 月 22 日，Microsoft 公司正式推出了 Microsoft.NET（以下简称.NET），使 Microsoft 公司现有的软件在 Web 时代不仅适用于传统的 PC，而且也能够满足新设备的需要，如蜂窝电话及个人数字助理（Personal Digital Assistant，PDA）等。

（3）2.C#与.NET。当 Microsoft 公司推出组件对象模型（Component Object Model，COM），通过将组件改变为通用、集成型的构件，开发人员逐渐地从过去的繁杂编程事务中解脱出来，可以选择自己最得心应手的编程语言进行编程。然而，软件组件与应用程序之间的结合仍然是松散的，不同的编程语言与开发平台限制了模块间的互用性，其结果是产生了日益庞大的应用程序与不断升级的软硬件系统。

为了将快速的应用程序开发与对底层平台所有功能的访问紧密结合在一起，Microsoft 公司推出了 C#。C#是专门为.NET 应用而开发的面向对象程序设计语言，C#为 C++程序员提供了快捷的开发方式，使得程序员能够在.NET 平台上快速开发各种应用程序。同时，C#忠实地继承了 C 和 C++的基本特征——强大的控制能力及其他优点。

使用 C#语言设计的组件能够用于 Web 服务，通过 Internet，可以被运行于任何操作系统上的任何编程语言所调用。C#运行于.NET 平台之上，其各种特性与.NET 有着密切联系。它没有自己的运行库，许多强大的功能均来自.NET 平台的支持。

据了解，C++、Java 和 C#是应用最广泛的三种支持面向对象技术的编程语言，其中 C++

的历史最为悠久，在面向对象技术方面支持更多的特性，如运算符重载、多重继承等。因此，本书将采用 C++语言来介绍面向对象的编程技术。

1.2　C++ 初　　窥

C++包含三部分相对独立的编程技术：以 C 语言为代表的面向过程的编程技术；C++向 C 语言中增加了类之后的面向对象编程技术；通过模板来实现的泛型编程。本章将带读者了解这些知识。但首先，请想一想为什么 C++从 C 中继承了这么多内容。我们使用 C++编程的一个主要原因就是要使用它的面向对象技术，但是你必须要首先了解一些 C 语言的基本编程技术：基本数据类型、运算符、控制结构和语法规则等。因此，如果你已经学会了 C 语言，那么再学习 C++将会比较从容，但也绝不是你想象的那样简单，多学几个关键字和结构就能精通 C++了。本书将指引着读者一步一步地学习 C++知识，每一阶段都经过精心的安排，最大化地降低学习 C++的难度。

本书假设读者从未学过 C 语言，因此在内容上既包含 C 语言的基础知识，也包含 C++的高级知识。使用本书，可以同时学到 C 和 C++的知识。如果你已经会 C 语言了，那么你将发现，本书的前半部分将是很好的复习 C 语言的资料。在这些章节中，既包含了后面将用到的重要概念，也讲明了 C 和 C++的区别。当你 C 语言基础扎实之后，你将开始学习 C++的高级知识。那时，你将了解类和对象的概念，以及 C++中是如何实现的。你还将学习模板，这也是非常重要的技术，能大大增加代码的重用性。

本书并没有打算写成一本无所不包的 C++手册，它并没有去探索语言中的所有细节。但是你将学到 C++中绝大部分重要的特性，包括模板，异常和命名空间。

下面让我们先简单地了解一下 C++的背景知识。

1.3　C++ 简　　史

在过去的几十年里，计算机技术以不可思议的速度在进化发展。今天的笔记本电脑，无论是运算速度还是存储器的容量，都要强于 20 世纪 60 年代的大型机。编程语言也是一样，虽然变化不是特别显著，但这些变化都十分重要。更大、更强的计算机孕育了更大、更复杂的程序。随着程序规模的不断增大，程序设计过程的管理变得日益困难，程序部署后的维护工作也同样越来越烦琐。

20 世纪 70 年代，C 语言和 Pascal 语言开创了结构化编程的时代。C 语言同时还具有代码结构紧凑、运行效率高的特点，并且能够直接操纵硬件。这些特点使得 C 语言在 20 世纪 80 年代成为具有统治地位的编程语言。同时，20 世纪 80 年代还诞生了其他的一些编程模式，也就是 C++和 SmallTalk 语言提供的面向对象编程技术（OOP）。下面让我们进一步了解一下 C 语言和 OOP。

1.3.1　C 语言

20 世纪 70 年代，贝尔实验室的 Dennis Ritchie 正在参与一个重要的项目，那就是 UNIX 操作系统的开发。为了顺利完成这个项目，他需要一种简洁高效并且能直接操纵硬件的编程语言。按惯例，程序员一般都会选择汇编语言来完成这类任务。但是汇编语言太低级了，与

计算机的 CPU 密切相关。如果你想把一个汇编语言编写的程序移植到另一个型号的计算机上，那么你几乎需要重新编写整个程序。而 UNIX 操作系统的设计目标是能够运行在各种类型的计算机上面，因此，人们需要一种高级别的语言，这种语言主要是用来解决各种问题，并且与硬件细节无关。因此，用这种语言编写出来的程序，拿到其他类型的计算机上运行时，只需要用该计算机的专用编译器重新编译一下即可，而无需重新编写整个程序。Ritchie 设想了一种语言，既有低级语言的高效性和硬件访问特性，又有高级语言的通用性和可移植性，于是他创造了 C 语言。

1.3.2 C 语言的设计哲学

由于 C++是在 C 语言的基础之上增加了新的编程哲学，因此，我们有必要先了解一下 C 语言的编程哲学。一般来讲，编程语言主要处理两个方面的东西——算法和数据。数据涵盖了一个程序所使用和处理的信息，而算法是程序在解决问题时所用到的方法。和大多数主流编程语言一样，C 语言在创建之初是一个过程式编程语言，这意味着它更重视算法的编写。

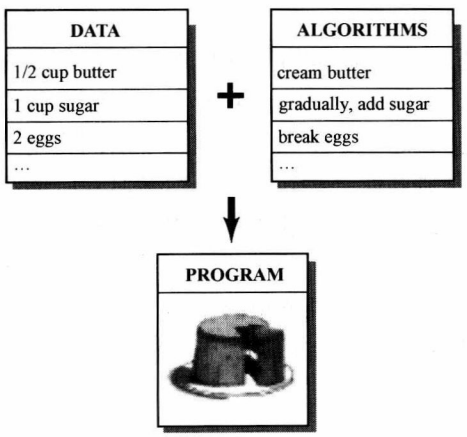

图 1.1 算法 + 数据 = 程序

从概念上讲，过程式语言首先设计出解决一个问题都需要哪些操作，然后用语言实现这些操作。一个程序事先制订好一系列的过程让计算机执行，从而得出一个特定的结果。这就好比一个食谱制订了一系列的操作，最终烘烤出一个蛋糕一样，如图 1.1 所示。

像 Fortran 和 BASIC 这种早期的过程式编程语言，它们在程序的组织结构方面有很大的问题。例如，程序中经常用到分支语句，根据判断结果的不同，来选择不同的代码块执行。许多早期的编程语言在处理这种结构时，如果问题比较复杂，则代码逻辑就会非常混乱。这种程序阅读起来非常困难，对它进行维护和修改简直就是一个灾难。因此，计算机科学家们制定了一种编程规范，称之为结构化编程。C 语言充分吸收了这种编程规范的特点。在处理分支结构时，C 语言仅仅提供了少量几个含义明确的关键字，如 for、while、if else 等。

自顶向下的设计方式是另一个新的概念。在 C 语言中，处理复杂问题的时候，都是先将它分解为若干个含义明确、易于处理的子问题。如果某个子问题仍然规模太大，则可以继续将它分解。在 C 语言中，提供了“函数”这一功能来帮助你实现问题的分解。你可能已经注意到了，结构化程序设计反映了这样一种设计过程：将问题分解为若干个操作，通过这些操作来最终解决问题。

1.3.3 C++的变化：面向对象的编程

尽管结构化编程对于程序的清晰性、可靠性及可维护性都有了大幅的改进，但是面对大型程序，它仍然有些力不从心。OOP 带来了一种新的方法来应对这种挑战。与强调算法的过程式编程不同，OOP 强调的是数据。传统的编程方式是让问题去适应编程语言，而 OOP 则尽量让语言去适应问题，其思想就是设计各种数据形式来与问题进行匹配，充分地表达出问题的特点。

在 C++中使用“类”这一概念来描述一种新的数据形式，而“对象”是针对类的定义，

赋予具体的数据之后形成的一种数据结构。例如，类可以用来定义一个公司的高管的公共属性（姓名、头衔、年薪等），而对象用于描述一个具体人（马化腾，首席执行官，300 万）。一般来讲，类用来定义描述一个对象都需要哪些数据，并且定义了都可以对这些数据进行哪些操作。例如，你正在编写一个绘画程序，有一个功能是可以绘制一个矩形，那么你应该定义一个矩形类，该类可能包含如下属性，分别是左上角坐标，长、宽、线条颜色、线条风格、填充颜色等。在矩形类中，还应该定义相应的操作，如移动矩形、旋转矩形、改变填充颜色、调整大小等。当上述一切都实现之后，你在程序中想绘制一个矩形时，只需要定义一个矩形对象就可以了，然后通过各种方法来对其进行改变。如果你想绘制两个矩形，那么你需要定义两个对象，每个对象分别代表一个矩形。

在面向对象的编程模式中，首先需要定义一些类，来充分描述待解决问题中的方方面面。例如，在编写一个绘图程序的时候，你可能会定义直线类、矩形类、椭圆类、毛笔、铅笔、钢笔、画刷等这些类。接下来在编写程序的时候，就可以充分利用这些类提供的功能，方便地完成相应操作。由此可见，在 OOP 编程中，我们首先在较低的层次上进行编码，如定义一些类，然后进入到较高的层次上，如考虑一下程序的逻辑模式等。因此，这种编程模式被称为自底向上的方式。

面向对象的编程不仅仅是定义和使用类这么简单，它还包含更多的内容。例如，OOP 编程可以帮你创建可重用的代码，大大降低你的工作量。信息隐藏可以帮你避免一些不恰当的数据访问。多态允许你对运算符和函数进行重载，从而在运行时确定具体的工作代码。继承允许你根据已有的类来生成新的类。由此可见，OOP 编程引入了大量新的概念，与之前的过程式编程有很大的不同。之前在编程时，我们总把精力集中在具体的细节上，而现在，我们更多的是关注高层次的概念。之前是自顶向下的编程，而现在是自底向上的编程。本书将通过大量的、浅显易懂的例子来引导读者学习相关知识。

当然，设计一个优秀的类是十分困难的。幸运的是，C++中已经提供了大量的设计优良、功能实用的类来供你使用。同时，一些公司和组织也提供有类库供你使用，比如 Microsoft 公司就提供了 MFC 来帮助你方便地创建 Windows 下的图形应用程序。C++的一大特色就是它允许你重用那些设计精良、久经考验的成熟的类库。

1.3.4　C++与泛型编程

泛型编程是 C++支持的另一个编程模式，它和 OOP 技术相互配合，大大地增加了代码的重用性。OOP 主要是将各种编程元素抽象成类，而泛型是对数据类型进行抽象，从而使得同一段代码可以处理不同的数据类型。OOP 的优势是使大型项目的管理变得更容易，而泛型是处理常用任务的有利工具，比如排序、合并链表等。“泛型”的含义就是类型的独立性。C++中有很多数据类型，有整型、实型、字符型、布尔型，还有用户自定义的各种类型。如果你要对各种数据类型进行排序，则按照以前的编程方式，你必须针对每一种数据类型编写一个排序函数。而泛型技术允许你只编写一个排序函数，排序的数据类型指定为“泛型”，在调用这个函数时，将各种具体的数据类型代入，都可以使得函数完美地工作。由此可见，代码的重用性大大增加，编程工作量显著减少。在 C++中使用模板技术来实现泛型。

1.3.5　C++的起源

和 C 语言一样，C++也是源自于贝尔实验室，于 20 世纪 80 年代初由 Bjarne Stroustrup 发明。在谈到自己发明 C++的原因时，Bjarne Stroustrup 这样说：“我设计 C++的目的是让我

和我的朋友无需再使用汇编、C 或其他高级语言来编写程序。C++的设计目标就是让每一个程序员都感到写程序是一件简单有趣的事情。”

在设计 C++的时候，Bjarne Stroustrup 最关心的问题是一定要让它易于使用。Bjarne Stroustrup 更愿意在 C++中加入一些编程中能实际用到的特性，而不是那些理论上很美好的东西。之所以选择在 C 语言的基础上进行改进，是因为 C 语言很简洁，很适合于系统编程，应用广泛并且与 Unix 操作系统关系十分密切。C++的 OOP 特性是受到了 Simula67 语言的启发。Bjarne Stroustrup 在向 C 语言中添加面向对象特性和泛型特性的时候，几乎没有改变 C 语言原有的内容。因此 C++是 C 语言的超集，也就是说，在 C 语言环境下运行良好的程序，也同样能在 C++环境下运行。C 和 C++之间或多或少还是有一些差别的，但不是十分重要。C++程序同样可以使用专门为 C 语言设计的函数库。这些特性使得 C++的推广变得非常容易。

C++的命名源自于 C 语言中的“++”运算符，该运算符的功能是在变量原有值的基础上加 1。因此，C++从名字上可以理解为是 C 语言的一个增强版本。

一个计算机程序就是把一个真实的问题转化为若干个计算机能够处理的模块。C++的面向对象特性使得它很容易对问题进行描述，而它包含的 C 语言特性又使得它很容易地操纵硬件。这种能力上的组合使得 C++迅速被大家所接受。

C++的泛型特性并不是一开始就有的，而是在 C++取得了阶段性成功之后，Bjarne Stroustrup 才添加进去的。只有当模板技术被广泛应用之后，人们才发现模板技术是 OOP 技术的一个重要的增强，有些人甚至认为模板比 OOP 更重要。OOP、泛型加上 C 语言的特性，这一切都表明 C++是一门非常重视实用性的语言，这也是它能取得成功的因素之一。

1.4 可移植性和标准

如果一个程序在更换平台之后，则能够顺利编译并正确执行，我们就说这个程序具有“可移植性”。

要想实现程序的可移植性会遇到很多困难，首先就是硬件的不兼容。如果一个程序用到了硬件的某些细节，那么它很难实现可移植性。为了减小移植程序的难度，对于直接访问硬件的代码，我们应该将它单独作为一个模块来编写。这样，在移植程序的时候，只需要把这个模块重新编写，而其他代码可以不做修改。在本书中我们将不会涉及这类代码。

另一个影响程序可移植性的是方言问题。先不说编程语言，就是我们日常说话中，同一个意思都会有不同的表达方式。普通话说“好”，郑州话说“中”，商丘话说“管”，其实都是一个意思。同样的，在编程语言中也存在方言问题。不同的编译器提供商，如 Intel 和 Microsoft，在实现 C++的时候都尽可能相互保持统一，但是，若没有一个公共的标准，不同公司之间想保证 100%统一是不可能的，每个公司的 C++都会有自己的方言。因此，美国国家标准化委员会（ANSI）在 1990 年成立了一个小组来开发 C++语言标准（C 语言标准也是这个委员会制定的）。国际标准化组织（ISO）不久之后也加入到这个项目中，共同开发 C++标准。

经过几年的努力，该标准于 1998 年制定完成，俗称 C++98。这个标准不仅规范地描述了 C++已经存在的特性，还对语言进行了扩充，加入了异常、运行时类型检查（RTTI）、模板和标准模板类库（STL）。在 2003 年，C++标准的第二个版本发布了。新版本标准是对上一个版本的修订——修正了一些印刷错误，明确了一些有歧义的地方。新标准并没有增加新

特性。这个版本的 C++标准俗称 C++ 03。

C++还在继续演变进化。ISO 委员会于 2011 年 8 月发布了 C++ 11 标准。和 C++ 98 一样，C++ 11 标准为语言增加了许多新特性。它的目标是消除所有的不一致性，并且让语言变得更容易学习和使用。这个标准先前被称做 C++0x，当时人们认为 x 的取值应该是 7 或 8，也就是人们期待着该标准在 2007 年或 2008 年完成。但实际上标准的制定工作进展非常缓慢，直到 2011 年才完成。为了掩盖这种进度缓慢的尴尬，人们又将 0x 解读为十六进制数字，按照这种解释，这个标准在 2015 年之前出来都是正常的。按照这种解释，ISO 组织出色地完成了任务，在计划时间内发布了标准。

ISO C++标准又将 ANSI C 标准吸收进来，因为 C++被认为是 C 语言的一个超集。这意味着凡是合法的 C 语言程序，都应该是合法的 C++程序。虽然在 ANSI C 标准和当前的 C++标准之间存在着一些的不同之处，但数量非常少。实际上，在 ANSI C 中还引入了一些首先出现在 C++中的内容，如 const 修饰符。

在 ANSI C 标准出台之前，人们将 Kernighan 和 Dennis Ritchie 写的一本书《C 语言程序设计》当做事实标准，这个标准通常被称为 K&R C。ANSI C 标准出台之后，K&R C 标准现在一般被称为经典 C。

ANSI C 标准不仅定义了 C 语言的语法规范，同时还定义了一个 C 语言的标准库，所有的 C 语言提供商都必须支持该标准库，C++也同样支持该标准库。另外，ISO C++还为 C++定义了一个标准的类库。

综上所述，C++语言的发展经历了如下阶段。

（1）一开始，并没有书面标准，人们将 Bjarne Stroustrup 写的《C++程序设计》作为事实标准来遵照执行。

（2）C++ 98 标准出台，向语言中添加了许多新特性。该标准将近有 800 多页。

（3）C++ 11 标准出台，该标准长达 1300 多页。

1.5　创建一个程序的步骤

假如你想写一个 C++程序，你该如何编写并使得它能够运行呢？根据你的计算机所使用的操作系统的不同，还有你使用的编译器的不同，其中的步骤可能会不一样。但是，无论任何操作系统还是编译器，都少不了如下几个步骤，只是操作方式上有所不同而已。

（1）使用一个文本编辑器编写 C++代码，并将结果保存为文件。这个文件被称之为 C++源代码。

（2）用编译器对源代码进行编译，也就是使用一个称之为“编译器”的软件，对源代码进行处理，将其转化为与特定机器相关的机器指令。转变后的程序被称为目标代码。

（3）将目标代码与其他的一些代码进行连接。例如，C++程序一般都会使用标准库。标准库中包含了常用功能的目标代码，当你要实现诸如“打印信息到屏幕上”、“求一个数字的平方根”等功能时，可以直接调用相关代码。之后通过连接的方式，将你写的代码、库代码，还有平台相关的一些启动代码组合成一个可执行程序，如图 1.2 所示。

在本书中你将会经常遇到“源代码”这一概念，因此，请牢记这一概念。

本书中的大部分代码都是通用的，平台无关的，按照 C++98 编写，能够在所有的系统中