应用型特色

高等学校应用型特色规划教材

GAODENGXUEXIAOYINGYONGXINGTESEGUIHUAJIAOCAI

C++程序设计与应用开发

王继民 柴春来 编著
余宇峰 俞 佳
严云洋 主审

- 以基础理论—实用技术—实训为主线
- 用任务来驱动，按教与学的实际需要取材谋篇
- 每一章都精心设置“案例实训”
- 配备丰富的免费教学资源——电子课件与案例实训资源包

清华大学出版社

TP312/2985
2008

高等学校应用型特色规划教材

C++程序设计与应用开发

王继民　柴春来　编著
余宇峰　俞　佳
严云洋　主审

清华大学出版社
北　京

内 容 简 介

本书由浅入深、系统全面地介绍了利用 C++程序设计语言进行结构化/面向对象/泛型程序设计的方法与开发技巧。全书共分 12 章，内容包括 C++语言概述、基本数据类型和表达式、语句和流程控制、函数和预处理、自定义数据类型、类和对象、运算符重载、继承/多态性与虚函数、输入/输出、异常处理、泛型机制—模板、项目实践等。

本书以“基础理论—实用技术—实训”为主线组织编写，每一章都设置了“案例实训”，以便于读者能够掌握该章的重点并提高实际操作能力。本书实例丰富、结构清晰、易教易学，对易混淆和实用性强的内容进行了重点的提示和讲解。

本书配有立体化教学资源包下载资源。提供电子教案，便于老师教学使用；提供所有习题答案，方便读者自学自测；并提供源代码及素材(包括案例实训内容)，便于学生上机调试；此外，特别编写了多种版本的综合项目实训(约 80 页篇幅)，以提高读者的应用开发能力。下载地址为 http://www.wenyuan.com.cn。

本书既可作为大中专院校的教材，也可作为各类培训班的培训教程，还可供使用 C++进行程序设计的软件开发人员阅读和参考。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。
版权所有，侵权必究。侵权举报电话：010-62782989　13701121933

图书在版编目(CIP)数据

C++程序设计与应用开发/王继民，柴春来，余宇峰，俞佳编著；严云洋主审. —北京：清华大学出版社，2008.9

ISBN 978-7-302-18209-2

Ⅰ.C…　Ⅱ.①王…②柴…③余…④俞…⑤严…　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(2008)第 108252 号

责任编辑：章忆文　宋延清
封面设计：杨玉兰
版式设计：北京东方人华科技有限公司
责任印制：何　芊
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京密云胶印厂
装 订 者：三河市新茂装订有限公司
经　　销：全国新华书店
开　　本：185×260　**印　张**：29.75　**字　数**：719 千字
版　　次：2008 年 9 月第 1 版　　**印　　次**：2008 年 9 月第1 次印刷
印　　数：1～4000
定　　价：39.80 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：(010)62770177 转 3103　　产品编号：029965－01

丛 书 序

二十一世纪人类已迈入“知识经济”时代，科学技术正发生着深刻的变革，社会对德才兼备的高素质应用型人才的需求更加迫切。如何培养出符合时代要求的优秀人才，是全社会尤其是高等院校面临的一项急迫而现实的任务。

为了培养高素质应用型人才，必须建立高水平的教学计划和课程体系。在教育部有关精神的指导下，我们组织全国高校计算机专业的专家教授组成《高等学校应用型特色规划教材》系列学术编审委员会，全面研讨计算机和信息技术专业的应用型人才培养方案，并结合我国当前的实际情况，编审了这套《高等学校应用型特色规划教材》丛书。

编写目的

配合教育部提出要有相当一部分高校致力于培养应用型人才的要求，以及市场对应用型人才需求量的不断增加，本套丛书以“理论与能力并重，应用与应试兼顾”为原则，注重理论的严谨性、完整性，案例丰富、实用性强。我们努力建设一套全新的、有实用价值的应用型人才培养系列教材，并希望能够通过这套教材的出版和使用，促进应用型人才培养的发展，为我国建立新的人才培养模式做出贡献。

已出书目

本丛书陆续推出，滚动更新。现已出版如下书目：

- Visual Basic 程序设计与应用开发
- Visual FoxPro 程序设计与应用开发
- Java 程序设计与应用开发
- Visual C++程序设计与应用开发
- Visual C# .NET 程序设计与应用开发
- C 语言程序设计与应用开发
- 计算机应用基础(等级考试版)
- 计算机网络技术与应用
- 微机原理与接口技术
- Windows XP+Office 2003 实用教程
- C++程序设计与应用开发

丛书特色

- 理论严谨，知识完整。本丛书内容翔实、系统性强，对基本理论进行了全面、准确的剖析，便于读者形成完备的知识体系。
- 入门快速，易教易学。突出“上手快、易教学”之特点，用任务来驱动，以教与学的实际需要取材谋篇。
- 学以致用，注重能力。将实际开发经验融入基本理论之中，力求使读者在掌握基

本理论的同时，获得实际开发的基本思想方法，并得到一定程度的项目开发实训，以培养学生独立开发较为复杂的系统的能力。

- 示例丰富，实用性强。以实际案例和部分考试真题为示例，兼顾应用与应试。
- 深入浅出，螺旋上升。内容和示例的安排难点分散、前后连贯，并采用循序渐进的编写风格，层次清晰、步骤详细，便于学生理解和实现。
- 提供教案，保障教学。本丛书绝大部分教材提供电子教案，便于老师教学使用，并提供源代码下载，便于学生上机调试。

读者定位

本系列教材主要面向普通高等院校和高等职业技术院校，适合本科和高职高专教学需要；同时也非常适合编程开发人员培训、自学使用。

关于作者

丛书编委特聘请执教多年、且有较高学术造诣和实践经验的名师参与各册之编写。他们长期从事有关的教学和开发研究工作，积累了丰富的经验，对相应课程有较深的体会与独到的见解，本丛书凝聚了他们多年的教学经验和心血。

互动交流

本丛书贯穿了清华大学出版社一贯严谨、科学的图书风格，但由于我国计算机应用技术教育正在蓬勃发展，要编写出满足新形势下教学需求的教材，还需要我们不断地努力实践。因此，我们非常欢迎全国更多的高校老师积极加入到《高等学校应用型特色规划教材》学术编审委员会中来，推荐并参与编写有特色、有创新的应用型教材。同时，我们真诚希望使用本丛书的教师、学生和读者朋友提出宝贵意见或建议，使之更臻成熟。联系信箱：Book21Press@126.com。

《高等学校应用型特色规划教材》编审委员会

E-mail: Book21Press@126.com

此为试读，需要完整PDF请访问：www.ertongbook.com

《高等学校应用型特色规划教材》系列
学 术 编 审 委 员 会

主　　编　吴文虎(清华大学)
许卓群(北京大学)
王　珊(中国人民大学)
杨静宇(南京理工大学)
曹进德(东南大学)

副 主 编　李勇智　许　勇　王士同

总 策 划　清华大学出版社第三事业部

执行策划　何光明

编　　委　(按姓氏笔画排序)

马世伟　方厚加　毛红梅　王　健　王士同
王国全　王建国　王继民　王维民　王景玉
史国川　史春联　刘廷章　刘志高　刘家琪
华继钊　汤学华　许　勇　严云洋　何光明
吴　婷　吴小俊　宋正虹　张　宏　李　胜
李　海　李千目　李亚非　李勇智　杨　明
杨帮华　陈亦望　周　松　於东军　俞　飞
姚昌顺　姜萍萍　赵　明　赵明生　温阳东
童爱红　戴仕明

前　言

C++是当今非常流行的一种支持结构化程序设计、面向对象程序设计以及泛型程序设计的高级程序设计语言，它适合于作为系统描述语言使用，即可用来编写系统软件，也可用来编写应用软件。C++是20世纪80年代初由贝尔实验室在C语言的基础上借鉴其他面向对象程序设计语言的特性而开发的。

本书由浅入深、系统全面地介绍了C++程序设计语言的基础理论以及利用C++进行应用程序开发的各种知识。

(1) 与同类图书相比，本书具有下列特色和优点：

- 结构清晰，知识完整。依据高校教学大纲来组织内容，同时覆盖最新版本的所有知识点，并将实际经验融入基本理论之中。
- 入门快速，易教易学。突出“上手快、易教学”的特点，以任务来驱动，根据教与学的实际需要取材谋篇。
- 学以致用，注重能力。以基础理论—实用技术—实训为主线编写，每一章都设置了案例实训，以便于读者掌握该章的重点及提高实际操作能力。
- 示例丰富，实用性强。通过大量的示例，分步骤细致地进行讲解，突出可操作性和实用性。
- 配有立体化教学资源包下载资源。提供电子教案，便于老师教学使用；提供所有习题答案，方便读者自学自测；并提供源代码及素材(包括案例实训内容)，便于学生上机调试；此外，特别编写了多种版本的综合项目实训(约 80 页篇幅)，以提高读者的应用开发能力。下载地址为 http://www.wenyuan.com.cn。

(2) 全书共分12章，各章的主要内容如下：

- **第1章** 介绍程序设计语言的发展、C++程序设计语言的历史，以及使用 Visual C++ 6.0 进行简单C++程序开发的方法。
- **第2章** 介绍C++的基本数据类型、变量的定义和使用、运算符优先级和结合性以及类型转换。
- **第3章** 介绍C++的各种控制语句、如何设计选择结构/循环结构的C++程序。
- **第4章** 介绍C++函数的概念、函数的定义/调用/参数传递、变量的作用域和生命期，讲解如何利用命名空间减少程序实体之间的命名冲突、如何利用递归函数简化问题的解决、如何利用重载函数来提高程序的易读性/利用内联函数提高程序的效率，以及编译预处理的过程。
- **第5章** 介绍如何在C++程序中通过自定义数据类型来描述复杂的事物，包括枚举、数组、指针、联合、结构以及引用等。
- **第6章** 介绍类和对象、面向对象编程的基本思想，讨论如何通过定义类、创建对象来描述问题域中的事物、对象的创建/初始化/撤消、对象成员的访问。
- **第7章** 介绍运算符重载的基本思想、通过成员函数和全局函数来实现运算符重

载的方法、常见运算符的重载实现。

- **第 8 章** 介绍继承的基本原理，以及如何通过继承来创建新类，继承与组合的区别，多重继承中要注意的问题等。
- **第 9 章** 介绍多态性的基本原理、C++中继承和虚函数对多态的支持，并介绍一种实现多态性的原理。
- **第 10 章** 介绍 C++中输入/输出的知识、输入/输出流的体系结构、C++标准库对输入/输出的支持、基于控制台和文件以及字符串的输入/输出的过程。
- **第 11 章** 介绍 C++的异常处理，包括异常的抛出、捕获和处理。
- **第 12 章** 介绍 C++模板机制对泛型编程的支持，以及 C++标准库的结构和简单应用。

本书可作为大中专院校的教材，也可作为各类培训班的培训教程，还可供使用 C++进行程序设计的软件开发人员阅读和参考。

本书由王继民(河海大学)、柴春来(浙江工商大学)、余宇峰(河海大学)、俞佳(江苏省行政学院)编著，由严云洋主审，全书框架结构由何光明拟定。

本书第 1、2、3、5 章由俞佳编写，第 4 章由柴春来编写，第 6、7、8、9、10、11 章由王继民编写，第 12 章及综合项目案例由余宇峰编写。另外，感谢蒋延辉、徐亮、陈坚、杨明、许勇、赵传申、许娟、李海、史春联、吴婷、王珊珊、陈玉旺、陈智等同志的关心和帮助。

限于作者水平，书中难免存在不当之处，恳请广大读者批评指正。任何批评和建议请发至：Book21Press@126.com。

编 者

2008 年 7 月

目　录

此为试读,需要完整PDF请访问: www.ertongbook.com

第 1 章　C++语言概述

本章要点

- 程序设计方法和程序设计语言的发展
- C++语言的历史
- C++语言的字符集、标识符和关键字
- 简单 C++程序的构成及开发

1.1　程序设计概述

程序设计是为计算机编制程序的过程，是将人类的自然语言(如汉语、英语等)所描述的问题及解决问题的方案转化为用计算机语言来描述的过程。从现代软件工程的角度来看，程序设计是指软件生命周期(软件产品在开发过程中所经历的一系列阶段，包括可行性分析、需求分析、软件设计、程序实现、测试、维护等)中程序实现阶段的工作，涉及到程序设计方法和程序设计语言等内容。

1.1.1　程序设计方法

我们编写的程序由两个主要方面构成。

- 算法的集合：就是将指令组织成程序来解决某个特定的问题。
- 数据的集合：算法在这些数据上操作，以提供问题的解决方案。

综观计算机发展的历史，这两个方面(算法和数据)一直保持不变，发展演化的是它们之间的关系，就是所谓的程序设计方法。目前常用的程序设计方法主要包括结构化程序设计、面向对象程序设计以及泛型程序设计。

1. 结构化程序设计(Structural Programming)

结构化程序设计是以功能为中心，基于功能分解的程序设计方法。一般采用自顶向下，逐步求精的方法，将一个复杂的系统功能逐步分解成由许多简单的子功能构成，然后分别对子功能进行编程实现。一个程序由一些子程序构成，每个子程序对应一个子功能，实现了功能抽象。子程序描述一系列的操作，是操作的封装体。结构化程序的执行过程体现为一系列子程序的调用。在程序中，数据处于附属地位，它独立于子程序，在调用子程序时，数据作为参数传递给子程序使用。

可以用一个式子来描述结构化程序的本质特征：程序=算法+数据结构。

式中的“算法”是对数据加工步骤的描述，而“数据结构”是对算法所加工的数据的描述。早期的程序大都采用了结构化程序设计方法，其不足之处在于：数据与操作分离，缺乏对数据的保护；功能会随着需求的改变而变化，而功能子程序的重新设计往往会导致整个程序结构的变动，使得程序难以维护；子程序的设计往往是针对某个应用而设计的，

它们很难用于其他的应用程序，导致难以重复使用。复用往往以一个一个的子程序为单位。本书将以第 2~5 章来描述 C++语言对结构化程序设计方法的支持。

2. 面向对象程序设计(Object-Oriented Programming)

20 世纪 70 年代，由于软件危机的出现，结构化程序设计越来越不能满足大型程序设计的要求，程序设计的焦点从结构化程序设计方法转移到抽象数据类型的程序设计上，现在通常称为面向对象的程序设计。一个面向对象的程序由一些对象构成，对象是由一些数据及可施于这些数据的操作所构成的封装体，对象的特征由相应的类来描述，一个类可以从其他的类继承。面向对象程序的执行过程体现为各个对象之间互相发送和处理消息。面向对象的程序可以描述为：程序=对象/类+对象/类+……。其中，对象/类=数据+操作。

在面向对象程序设计中，把数据和对数据的操作封装在一起，对数据的操作必须通过相应的对象来进行，从而加强了数据的保护。对象在问题的求解领域是相对稳定的实体，由对象构成的程序能够适应软件需求的变化。在面向对象程序设计中软件的复用以类为单位。本书将以第 6~11 章来讲述 C++语言对面向对象程序设计的支持。

3. 泛型程序设计(Generic Programming)

简单地说，泛型程序设计是一种将类型参数化的思维模式。面向对象程序设计关注的是程序的数据方面，而泛型程序设计关注的是算法方面，两者的侧重点不同。面向对象程序设计用来管理和实现大型项目，而泛型程序设计一般用来实现通用的任务，如数据排序等。其中的“泛”字表示根据类型来产生代码。

C++中所描述的数据有很多类型，如整型、浮点型、字符型及自定义类型等，如果希望在系统中为不同类型的数据实现排序功能，通常的做法是为每种类型的数据编写单独的排序代码。而泛型程序设计则可以编写统一的泛型排序代码，然后在对每种实际类型的数据排序时，调用该泛型排序代码。本书第 12 章将介绍 C++语言对泛型程序设计的支持。

1.1.2 程序设计语言

程序设计语言从总体上可划分为低级语言和高级语言两种，其中低级语言包括机器语言、汇编语言等，高级语言包括 Fortran、Basic、Pascal、Java、C 和 C++等。

1. 机器语言

计算机是无知觉、无生命的机器，它的中央处理单元(CPU)具有高速运算的能力，能够不知疲倦地帮我们进行计算和判断，最后得出结果。人类在交流和理解问题时通常使用自然语言，如汉语、英语等。而计算机是不懂自然语言的，要让计算机帮助我们解决问题，必须将问题以及解决问题的步骤利用计算机能够理解的机器语言来描述，这个过程就是编程，如图 1-1 所示。

图 1-1 编程模型

机器语言是机器指令的集合。不同型号 CPU 的计算机都有自己的指令集合，如 Intel CPU 和 AMD CPU 能够识别的机器语言是不同的。即使采用的同样是 Intel CPU，如 286、386、486、586 或是 Core2，它们的机器语言也有些区别。因此以某种 CPU 的机器语言编写的程序难以向另一种型号的 CPU 移植。

另外，由于机器码指令是用许多二进制数来表示的。例如，字母'A'表示为 1010，数字 9 表示为 1001 等，难记忆，易弄错，并且难以检查程序和调试程序。因此，用机器语言编程必然很繁琐，非常消耗精力和时间，工作效率很低。

2. 汇编语言

为了减轻使用机器语言编程的困扰，人们进行了一种有益的改进：用一些简洁的英文字母、符号串来替代一个特定指令的二进制串，比如，用 ADD 代表加法，用 MOV 代表数据传递等。这样一来，人们能够很容易读懂并理解程序在干什么，纠错和维护也都变得方便了，这种程序设计语言就称为汇编语言。

如有一个简单的问题：计算表达式 a×b+c−d 的值。用自然语言描述的解决步骤如下。

(1) 计算 a 与 b 的乘积赋给 result。

(2) 计算 result 与 c 的和赋给 result。

(3) 计算 result 与 d 的差赋给 result。

(4) 最后获得的 result 值为表达式的值。

使用汇编语言编写程序来解决该问题的代码为：

```
mov eax, a
mul eax, b
add eax, c
sub eax, d
mov result, eax
```

其中：

mov eax, a：表示将 a 的值送到 CPU 中一个叫 eax 的存储空间中。

mul eax, b：表示将 eax 中的值和 b 相乘后再送到 eax 中。

add eax, c：表示将 eax 和 c 中的值相加，结果送到 eax 中。

sub eax, d：表示将 eax 存储空间中的值减去 d，结果送到 eax 中。

mov result, eax：表示将 eax 的值送到一个叫 result 的内存空间中。

result 中存放的即为表达式的计算结果。

可以看出，问题的汇编语言描述(汇编源程序)比机器语言描述容易理解。但是 CPU 是不懂汇编语言的，我们必须使用某个翻译程序(通常称为汇编程序)，将汇编源程序转换成机器能够理解的机器语言(这个过程一般称为汇编)。这样的转换程序通常由专门的软件公司来提供，全世界的程序员都可以使用。利用汇编语言进行编程的过程如图 1-2 所示。

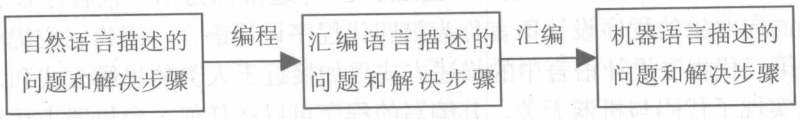

图 1-2　汇编语言编程模型

虽然汇编语言较机器语言已有很大的改进，但仍是低级语言，它有两个主要缺点：

- 涉及太多的机器硬件细节。
- 与具体的计算机相关，因为汇编语言中的指令基本上与机器语言的指令一一对应，只是采用符号简化了程序员的记忆。

因此，汇编语言也被称为面向机器的语言。早期的计算机由于速度慢、内存小，衡量程序质量高低最重要的指标是机器执行的效率，从而使汇编语言得到广泛的应用。目前在底层硬件编程方面，程序执行效率仍十分重要，针对计算机特定硬件而编制的汇编语言程序精炼而质量高，能准确发挥计算机硬件的功能和特长，所以在一些系统软件(如操作系统、硬件控制以及三维图形引擎等)的开发中，仍是一种常用而强有力的工具。

3. 高级语言

为了进一步提高编程效率，改进程序的可读性、可维护性，又出现了许多高级语言，这些编程语言接近于数学语言或人类的自然语言，同时又不依赖于具体的计算机硬件，编出的程序能在所有机器上通用，如 Fortran、Basic、Pascal、Java、C 和 C++等。

利用高级语言来描述问题相对于汇编语言要简单许多，如采用高级语言(C++语言)所编写的计算表达式 a×b+c−d 结果的程序为：

```
result = a * b + c  - d;
```

与汇编语言类似，也需要使用专门的翻译程序——即 Compiler(编译器)或 Interpreter(解释器)将高级语言的源程序翻译成机器语言后才能运行。这样的翻译程序也可以由专门的软件公司(如微软、Borland 等)来编写。但是将自然语言描述的问题转换成使用高级语言来描述要相对容易得多，利用高级语言编程的过程如图 1-3 所示。

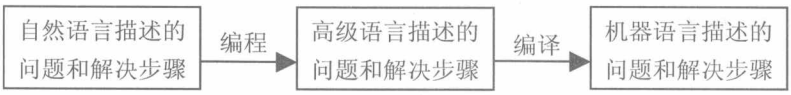

图 1-3 高级语言编程模型

利用高级语言编写的程序在执行效率上可能会比用汇编语言编写的程序稍微逊色一些，但是面对如今软件越来越复杂、功能越来越庞大以及多人合作开发等状况，程序的易维护性、可读性等对软件开发的成败非常重要，代码选择上降低的些许效率，可以通过提高硬件的性能来弥补。

高级语言比低级语言更加抽象、简洁，其优点如下：

- 一条高级语言的指令相当于多条机器语言的指令。
- 用高级语言编写的程序与英语非常接近，易于学习。
- 用高级语言编写程序并不需要某种计算机硬件的专门知识。

在高级语言中，像 Fortran、Basic、Pascal、C 等通常称为第三代程序设计语言，而 C++、Java 等面向对象的程序设计语言称为第四代程序设计语言，因为它们提供了类、对象等概念，比第三代程序设计语言中的描述方式更加接近于人类的思维方式和习惯。其中 Java 语言真正实现了代码与机器无关，其编写的程序可以在任何一台机器上运行，常用于网络编程。因此，也称 Java 是一种面向 Internet 的程序设计语言。

从以上介绍的程序设计语言的发展过程可以看出：程序设计语言发展的总体趋势是描述问题的方式越来越接近于人的自然思维方式，这样可以降低程序员的编程工作强度。而源程序代码到机器语言指令的转换则留给某些大公司开发的编译软件去完成。高级语言虽然比低级语言更容易描述问题，但采用这些语言还必须按照计算机解决问题的方式来描述问题，对于大型的软件系统开发，程序设计仍然非常困难。因此，人们还在努力设计更高级、更加接近自然语言的编程语言，以便能够用更自然的方式来进行程序设计。

在程序设计语言的发展过程中，语言提供的描述问题的方式的改变，也带来了软件复用粒度的增加，如在 Pascal、C 等第三代程序设计语言中，复用以函数为单位，而到了第四代程序设计语言(如 C++、Java)中，软件复用可以以类为单位。

1.2　C++语言的发展历史

C++是当今非常流行的一种支持结构化程序设计、面向对象程序设计以及泛型程序设计的高级程序设计语言。它适合于作为系统描述语言，既可用来编写系统软件，也可用来编写应用软件。它是 20 世纪 80 年代初由贝尔实验室在 C 语言的基础上，借鉴其他面向对象程序设计语言的特性而开发的。

在 C 语言推出之前，操作系统等系统软件主要是用汇编语言编写的(如著名的 Unix 操作系统)，由于汇编语言依赖于计算机硬件，因此程序的可移植性和可读性就比较差。为了提高程序的可读性和可移植性，最好能采用高级语言来编写这些系统软件。然而，一般的高级语言难以实现汇编语言的某些功能(如直接对硬件进行操作、对内存地址进行操作和位操作等)。人们设想能否有一种能集一般高级语言和低级语言特性于一身的语言呢？于是，C 语言便应运而生了。

C 语言最早的原型是 Algol 60，1963 年，剑桥大学将 Algol 60 发展成为 CPL (Combined Programing Language)；1967 年，剑桥大学的 Matin Richards 对 CPL 语言进行了简化，产生了 BCPL 语言；1970 年，美国贝尔实验室的 Ken Thompson 将 BCPL 进行了修改，并取名为 B 语言，意思是提取 CPL 的精华(Boiling CPL down to its basic good features)，并用 B 语言写了第一个 Unix 系统；1973 年，AT&T 贝尔实验室的 Dennis Ritchie 在 BCPL 和 B 语言的基础上设计出了一种新的语言，取 BCPL 中的第二个字母为名称，这就是大名鼎鼎的 C 语言。随后不久，Unix 的内核和其上的应用程序全部用 C 语言改写，从此，C 语言成为 Unix 环境下使用最广泛的主流编程语言。

1979 年，C++之父 Bjarne Stroustrup 借鉴面向对象程序设计语言 Simula 中“类”的概念，开始研究增强的 C 语言，使其支持面向对象的特性，由此产生了带类的 C。后来 C 标准委员会决定为这个版本的 C 起个新的名字，那时征集了很多种名字，最后采纳以 C 语言中的++运算符来体现它是 C 语言的进步，所以就叫 C++。C++语言在使用过程中，由 ANSI(美国国家标准化委员会)、ISO(国际标准化组织)等标准化组织进行了不断的标准化。在标准化过程中，添加了很多新的语言特性，以使这种语言更加容易使用，同时也建议去掉了一些过时的语言特性。1998 年，ANSI 和 ISO 先后批准 C++语言成为美国国家标准和国际标准，同年正式发布了 C++语言的国际标准 ISO/IEC：98-14882，各软件商推出的 C++编译器都支持该标准，并有不同程度的拓展。C++发展历史如图 1-4 所示。

此为试读，需要完整PDF请访问：www.ertongbook.com