国外计算机科学教材系列

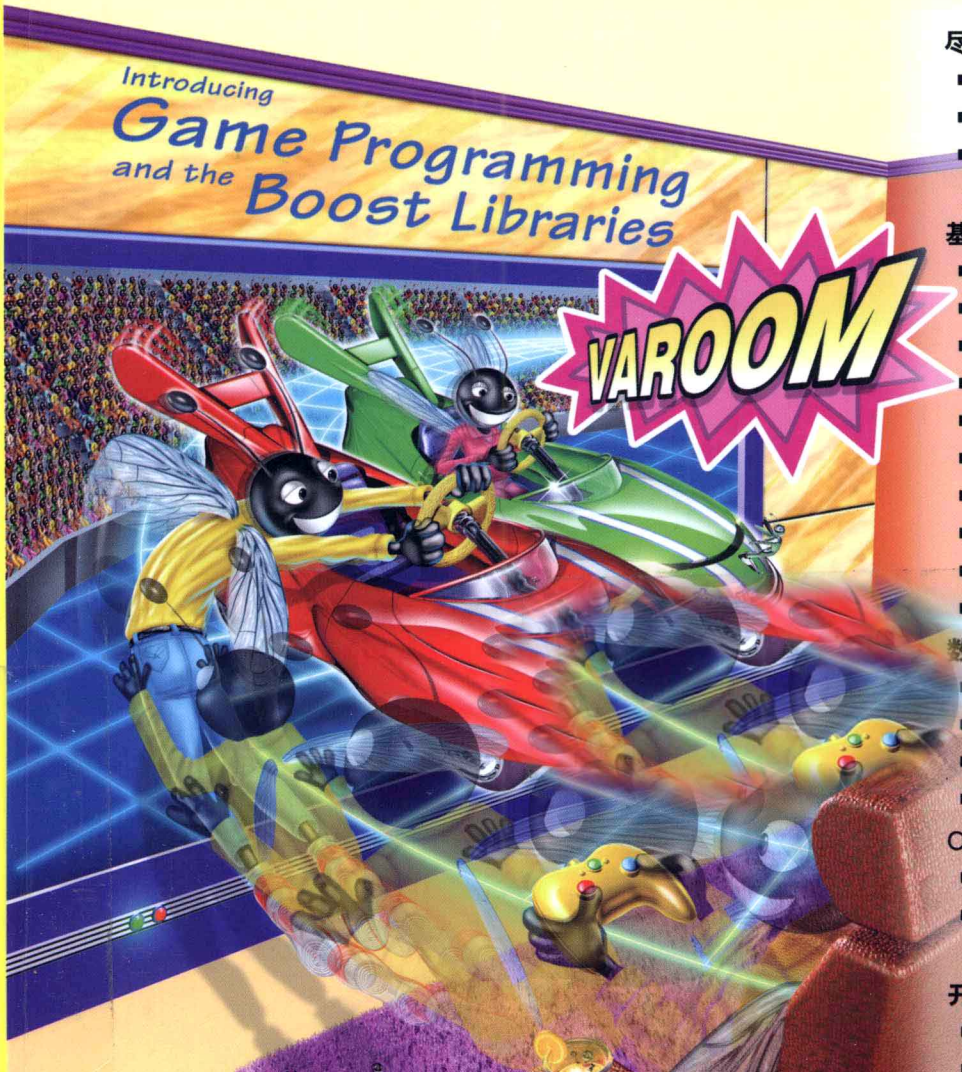
光盘内容包括Microsoft Visual C++ 2008 Express Edition安装程序

PEARSON

C++大学教程

——（第六版）（英文版）——

C++ How to Program, Sixth Edition



尽早接触类/对象/面向对象编程
- 类，对象，封装
- 继承，多态性
- 完整的面向对象编程实例研究：Time类，GradeBook类，Employee类

基础知识
- 历史，硬件，软件
- 输入/输出流，类型，运算
- 控制语句，函数
- 数组，vector类模板
- 指针，引用
- 字符串类，C风格的字符串
- 运算符重载
- 异常处理，文件
- 位运算及字符操作
- GNU C++调试器与Visual Studio调试器

数据结构
- 递归，查找，排序
- 链表，队列，堆栈，树
- 模板
- 标准模板库：容器，迭代器和算法

OOD/UML2 ATM实例研究（选学）
- 确定类，属性，状态，活动，操作，协作
- 示图：用例图，类图，状态图，活动图，通信图，顺序图

开源的C++库
- 使用Orge进行游戏编程
- Boost C++库与C++的将来

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

DEITEL®

[美] P. J. Deitel
H. M. Deitel 著

国外计算机科学教材系列

C++ 大学教程

（第六版）（英文版）

C++ How to Program

Sixth edition

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内容简介

本书是一本讲解C++编程的优秀教材，全面介绍了面向对象编程的原理和方法，详细分析了与C++编程有关的技术，具体包括类与对象、控制语句、函数与递归、数组、指针、运算符重载、继承、多态、模板、流输入/输出、异常处理、文件处理、搜索与排序、数据结构、标准模板库等内容，本书的同步学习网站上还包含了更多的扩展内容。全书以“活代码”方式详细分析每个知识要点，是初学者和中高级程序员学习C++编程的理想用书。

本书可作为高等院校相关专业的编程语言教材和C++编程教材，也是软件设计人员学习C++编程的理想读物。

版权贸易合同登记号　图字：01-2010-5782

图书在版编目（CIP）数据

C++大学教程 = C++ How to Program：第6版：英文/（美）戴特尔（Deitel, P. J.）著.
北京：电子工业出版社, 2011.1
国外计算机科学教材系列
ISBN 978-7-121-12197-5

Ⅰ.①C…　Ⅱ.①戴…　Ⅲ.①C语言－程序设计－高等学校－教材－英文　Ⅳ.①TP312

中国版本图书馆CIP数据核字（2011）第216396号

策划编辑：冯小贝
责任编辑：许菊芳
印　　刷：涿州市京南印刷厂
装　　订：涿州市桃园装订有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编：100036
开　　本：787 × 1092　1/16　　印张：67　　字数：1716千字
印　　次：2011年1月第1次印刷
定　　价：128.00元（含CD光盘1张）

凡所购买电子工业出版社的图书有缺损问题，请向购买书店调换；若书店售缺，请与本社发行部联系。联系及邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

“诚实地对待每个细节，这正是你能力之所在。”

—修女 Teresa

欢迎大家进入C++编程世界，开始学习《C++大学教程》（第六版）。在Deitel & Associates公司，我们为Pearson Education（培生教育出版公司）编写程序设计语言类的教材和专业书籍，提供全球范围的企业培训，并进行Web 2.0互联网业务的开发。这本书正是我们的成果之一，它不仅反映了C++语言的发展变化，而且体现了更好的关于编程的教学和学习方法。和前版相比，所有章节都有了显著的调整。此前言的“本书导读”部分将帮助教师、学生和专业人士了解本书所覆盖的C++和面向对象编程知识。

新增特性与更新特性

下面是我们对《C++大学教程》第五版和第六版本所做的更新。

- **游戏编程。**新增了一章专门介绍游戏编程。现在计算机游戏产业的收入已经超过了首映电影业务的收入，该产业为有志于从事游戏编程职业的学生创造了许许多多的就业机会。第22章“使用Orge进行游戏编程”，介绍了如何使用开源的Ogre 3D图形引擎软件进行游戏编程和图形绘制。首先讨论游戏编程所涉及的一些基本问题，然后展示如何使用Ogre开发一个简单的游戏，它的玩法与一款最初由Atari在1972年开发的经典视频游戏Pong相似。我们将演示如何创建一个三维彩色图形场景、平滑绘制移动的物体、使用计时器来控制动画的速度、检测物体之间的碰撞、添加声效、接受键盘输入和显示文本输出等。
- **C++的将来。**增加了第23章来考虑未来的C++——介绍了Boost C++库，技术报告1（TR1）和C++0x。这个免费的Boost开源库是由C++社区（C++ community）的成员开发的。TR1描述了对C++标准库的修改建议，其中很多是基于当前的Boost库的。C++标准委员会正在修订C++标准。新标准的主要目标是令C++更容易学习，提高库的构建能力，并增强与C程序设计语言的兼容性。最近的标准是在1998年公布的。目前称为C++0x的新标准化工作始于2003年。除了对核心语言的变更外，新标准极可能包括TR1中的许多库。我们对Boost库进行概述，并提供有关“正则表达式”和“智能指针”库的代码示例。正则表达式用于在文本中匹配特定的字符模式，它们可以用于验证数据以保证它具有特定的格式，可以用于用另外的字符串替换一个字符串中的一部分，可以用于拆分字符串等。C和C++代码中的许多常见错误都是与指针有关的。指针是第8章中学习的一种强大的编程能力，而智能指针将通过对标准指针提供附加的功能来帮助避免一些错误。
- **大幅度的内容修订。**所有章节都经过了有意义的校正和提升。我们不断改进，使语言表达更为清晰与准确。同时，也调整了C++术语的表达方式，使其与定义该语言的ANSI/ISO C++标准文档相适应。
- **尽早接触类和对象。**本书在第1章介绍对象技术的基本概念和术语，在第3章学生即开始开发自定义的、可重用的类及对象。无论从书的开篇，还是贯穿整本书，只要适合，这本书都在体现面向对象编程。尽早地讨论类和对象可以使学生直接“考虑对象”和更彻底地掌握这些概念。面向对象编程不是一件简单的事情，但是用面向对象的思想编写程序是一种乐趣，相信学生很快就能体会到。
- **综合的实例研究。**书中提供了若干横跨多个章节的实例研究，它们往往都基于本书前面引入的类，然后在后面的章节中继续阐述新的编程概念。这些实例研究包括第3～7章中GradeBook（成绩簿）类的开发，第9～10章中Time（时间）类的开发，第12～13章中Employee（雇员）类的开发，以及既出现在第1～7章、第9章和第13章，又出现在附录G的可选学的OOD/UML ATM（自动取款机）实例研究。
- **完整的GradeBook实例研究。**GradeBook实例研究不断增强早期提出的类，它用第3～7章中的类和对象一步一步地创建GradeBook类。GradeBook类表示了教师的成绩簿，并且可以根据一群学生的成绩执行各种计算，例如计算平均成绩，找出最高和最低的成绩，以及打印条形图等。

- **统一建模语言2（UML 2）。**统一建模语言已经成为面向对象系统设计者首选的图形建模语言。本书中所有的UML图示都符合UML 2规范。我们使用UML类图可视化地表示出类以及它们之间的继承关系，使用UML活动图展示各个C++控制语句中的控制流。尤其是在可选学的OOD/UML ATM实例研究中，大量使用了UML。
- **可选学的OOD/UML ATM实例研究。**这个可选学的OOD/UMLATM实例研究出现在第1～7章、第9章和第13章的软件工程实例研究部分，非常适合于第一门和第二门程序设计课程的教学。组成该实例研究的9个章节循序渐进地介绍如何采用UML语言进行面向对象的设计。首先介绍UML 2的一个精简子集，然后指导大家体验第一项设计，这项设计是专为面向对象设计或编程的新手而准备的。这一实例研究的目标是通过逐步展开面向对象的设计，帮助学生增强在第1章开始学习并在第3章实现的面向对象编程的概念。这个实例研究已通过一个精通OOD / UML的杰出团队的审查，团队的专家来自学术界和工业界。而且，该实例研究并不是一个简单意义上的练习，更确切地说，应该是一个精心设计的、一环扣一环的学习过程。此过程通过对一个完整的877行C++代码实现的详尽诠释而结束。后面将对这个实例研究的9个部分进行详细的介绍。
- **多个源文件程序的编译和连接过程。**第3章包含了编译和连接过程的详细图示和讨论，该过程生成一个可执行的应用程序。
- **函数调用栈的解释。**第6章通过提供函数调用栈和活动记录的详细讨论（含有图示），来解释C++是如何记录正在运行的函数，内存是如何保存函数的自动变量，以及函数在完成它执行的任务后是如何知道返回何处的。
- **C++标准库中的string和vector对象。**通过使用string类和vector类，可以在书中较前面的示例程序中更多地体现面向对象的性质。
- **string类。**在整本书中，对于大多数字符串的操作都使用string类，而非C风格的基于指针的char *字符串。但是，在第8章、第10章和第11章中我们仍讨论char *字符串的有关内容。这样，给学生练习指针操作的机会，举例说明如何用new和delete进行动态内存分配，构建自己的String类，并使学生今后有能力应对C和C++遗留代码中的char *字符串问题。
- **类模板vector。**整本书都使用类模板vector，而不是C风格的基于指针的数组操作。尽管如此，仍会在第7章中继续讨论C风格的基于指针的数组，为学生以后能够处理遗留的C和C++代码做准备，同时也为第11章中建立自定义的Array类奠定基础。
- **继承和多态性的协调处理。**第12章和第13章经过仔细调整，使用Employee类层次结构来介绍继承和多态性，使得关于继承和多态性的处理更清楚，更容易被刚刚接触OOP的学生所接受。
- **对多态性工作“知其然也知其所以然”的讨论和图示。**第13章通过详细的图示和说明，解释了C++内部是如何实现多态性、虚函数和动态绑定的，使学生对这些机制的实际工作原理有非常深刻的理解。更重要的是，它有助于学生认识到实现多态性是要付出代价的，即需要额外占用内存和处理器时间，从而使他们能够决定什么时候应该使用多态性，什么时候应避免使用。
- **标准模板库（STL）。**如果基于软件重用性的观点，这一部分应该是本书中最重要的章节了。STL定义了功能强大的、基于模板的和可重用的组件，这些组件实现了许多常见的数据结构及用于处理这些数据结构的算法。第21章介绍了STL，并讨论它的三个关键组件：容器、迭代器和算法。我们展示了使用STL组件可以提供极强大的表达能力，并可以将多行的程序代码简化为单条语句。
- **遵循ANSI/ISO C++标准。**我们根据最新的ANSI/ISO C++标准文档审核了相关的文字表述，以实现完整性和精确性。[注意：如果读者需要更进一步地了解C++技术的细节内容，请阅读C++标准文档。C++标准文档有一份PDF格式的电子版本（文档序列号为INCITS/ISO/IEC 14882-2003），可以从webstore.ansi.org/ansidocstore/default.asp购买。]
- **调试器附录。**本书中放入了两个“使用调试器”的附录——附录I和附录J。
- **多平台的代码测试。**我们在各种流行的C++平台上测试了本书的代码实例。对大多数情况而言，书中所有例子在主流的符合标准的编译器上都能成功运行。
- **不同平台上的错误和警告。**对于一些故意设置错误来阐述关键概念的程序，我们列出了不同操作平台上的错误信息。

上述的一切内容都已经过学术界和工业界著名专家的细致评审，他们与我们就《C++大学教程》第五版和第六版的工作进行了意义深远的合作。

我们相信本书及其辅助材料将为学生和专业人员提供一次集知识性、趣味性、挑战性和娱乐性于一体的C++教育体验。书中包括了一整套教辅材料，帮助教师最大限度地发挥学生的学习能力。

在阅读本书的过程中，如果有任何疑问，请发送电子邮件到deitel@deitel.com，我们将会尽快回复。为了获取本书的更新和所有C++支持软件的具体情况，以及关于Deitel出版物和服务的最新资讯，请访问www.deitel.com。请赶快在www.deitel.com/newsletter/subscribe.html上注册免费的DEITEL Buzz Online电子邮件时事通信，并经常在www.deitel.com/ResourceCenters.html查看所列的越来越多的C++及相关资源中心。每周我们都会在时事通信中公布最新的资源中心。同时，也请告诉我们您所感兴趣的其他资源中心。

教学方法

《C++大学教程》(第六版)含有大量的实例。这本书着重考虑良好的软件工程原则，并强调程序的清晰性。我们通过实例来教学。作为教育工作者，我们在全球范围内讲授工业界最前沿的程序设计语言和软件相关知识。Harvey M. Deitel博士有着20年的高校教学经验和19年的工业界教学经历。另一作者Paul J. Deitel也具有16年的工业界教学资历。他们针对来自政府机构、工业界、军事部门和学术界的Deitel & Associates公司的不同客户，从不同的层次讲授C++课程。

采用活代码方式。《C++大学教程》(第六版)拥有大量的“活代码”实例。通过这种方式，对于每个新概念，都用完整的、能实际运行的C++应用程序进行介绍，程序代码之后直接附有一个或者多个运行示例，用于展示程序的输入/输出。这种风格体现了我们讲授和编写程序的方法，称为“活代码”(Live-Code)方式。

语法的颜色标记。与大多数C++集成开发环境或代码编辑器类似，我们对所有的C++代码都通过不同颜色来标记其中的语法。这样，大大提高了代码的可读性——本书提供的完整的、可执行的C++程序拥有20 000行代码，可读性的确是一个特别重要的目标。进行语法颜色标记的约定如下[①]：

```
注释采用绿色显示
关键字采用深蓝色显示
错误采用红色显示
常量和字符串采用浅蓝色显示
其他所有的代码采用黑色显示
```

代码高亮显示。每个程序中的关键代码段用灰色的矩形块标记。

使用不同的字体和颜色突出重点。我们对关键的术语以及索引页中术语定义出现的地方都用加粗的蓝色字体来表示，从而方便查找；用加粗的Helvetica字体来强调是屏幕上出现的元素(例如File菜单)，而用Lucida字体来显示C++程序的内容(例如，`int x = 5;`)。

Web访问。《C++大学教程》(第六版)的所有源代码实例都可以从下面的地址下载：

```
http://int.prenhall.com/deitel
```

下载所有的实例后，请在阅读相应的教材内容时运行一下每个程序。然后可以对这些例子做一些改动，通过程序的运行马上看到改动的效果，这是增强学习C++效果的一种立竿见影的好方法。

学习目标。在每章的开始部分列出了一系列学习目标，目的是使大家知道每章的学习主题，并且在学完每章后，有机会确定自己是否到达预期的目标。

名人名言。在每章的学习目标之前，我们引用了一些名人名言。有些是幽默的，有些富有哲理，有些则提供了有趣的见解。希望读者能够细细品味它们所蕴含的与章节内容相关的深层意义。

提纲。每章的提纲有助于大家以自上而下的方式学习这些内容。这样可以预见到即将学习什么，进而轻松、有效地调整学习节奏。

插图/图表。本书包含了大量的图表、表格、线条图、程序及程序输出。我们采用UML活动图表示控制语句中的控制流，用UML类图表示类的数据成员、构造函数和成员函数。在可选学的OOD/UML ATM实例研究中，更是广泛使用了6种主要的UML图。

① 由于印刷原因，英文影印版中取消了颜色区分——编者著。

编程提示。本书还包含了很多编程提示，目的在于帮助学生将学习重点放在程序开发的关键部分。这些提示和实践经验是两位作者累积60年的编程和教学经验总结。我的一位数学专业的学生说，这种方法就像在数学中着重标明公理、定理、引理和推论一样，同样为编写一个好的软件提供了基础。

良好的编程习惯

良好的编程习惯关注的技术将有助于编写出更加清晰、更易理解和更易维护的程序。

常见的编程错误

学生总是犯一些常见的错误。指出这些常见的编程错误可以减少犯类似错误的可能性。

错误预防技巧

这些提示包含发现和消除程序错误的各种建议。事实上，许多这样的提示描写的是如何在开始编写C++程序的第一时间就防止错误。

性能提示

学生喜欢使他们的程序“超值”（turbo charge），快速而高效。这些提示提供了一些强调提高程序性能的方法，使程序运行得更快，或者它们所占用的内存最少。

可移植性提示

书中所包含的可移植性提示将有助于编写出可以运行在不同平台上的代码，同时也使大家明白C++是如何实现其高度可移植性的。

软件工程知识

软件工程知识这一部分的提示突出了关于软件体系结构和设计的问题，这些问题往往影响到整个软件系统的构建，特别是对一些大型的系统而言。

摘要列表。每章都是以附加的教学内容作为结尾的。本书的一个教学特色，是在每章结束部分都以清单的形式逐节汇总了本章要点。

术语。在这一部分提供了一个按字母顺序排列的术语清单，列出了每章中所定义的重要术语。

自测题及答案。每章都包含了为自学而精心设计的大量自测题及其答案。

练习题。每章最后都提供了大量的练习题，包括：

- 对一些重要术语和概念的简单回顾
- 在代码实例中找出错误
- 编写单条的C++语句
- 编写小块的C++函数和类
- 编写完整的C++的函数、类和程序
- 开发大型的课程项目

如此大量的练习题有助于教师根据学生的特殊需要来调整课程，以及在不同的学期布置不同的作业等。教师可以利用这些题目来安排家庭作业、小测验、大型考试和课程项目。如果需要更多的其他可能的练习和项目资源，可以访问我们的编程项目资源中心（www.deitel.com/ProgrammingProjects/）。

注意：请不要给我们写信，要求访问Pearson Education教师资源中心。只有采用这本教材进行授课的高校教师才可以访问该中心。教师只能通过Pearson Education的代理商获得访问资格[①]。

成千条的索引项。本书包含了规模庞大的索引。索引是非常有用的，尤其当你以此书作为参考书的时候。

本书导读

这一部分将带领大家了解一下在《C++大学教程》（第六版）中可以学到的许多C++功能。图1给出了各个章节之间的依赖关系。我们建议沿着箭头所指的顺序来学习各个主题，当然，也可以按照其他的路线。这本书可广泛适用于各个层次的C++课程授课中。在网络上搜索“syllabus”、“C++”和“Deitel”等关键字，会找到不少关于这本书的最新版本的教学大纲。

① 国内教师申请教辅资源的方法，请参见本书末尾的“教学支持说明”页。

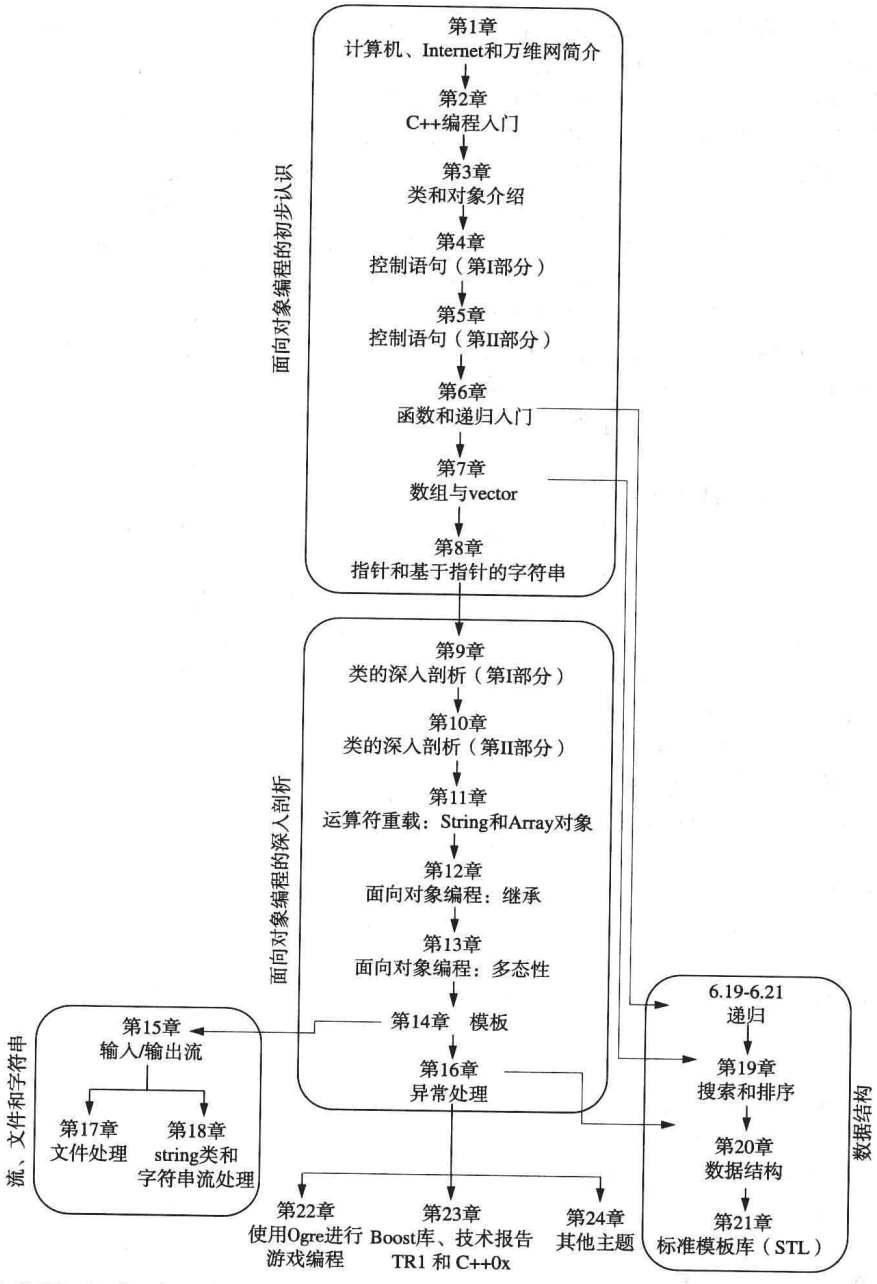


图 1　本书各章之间依赖关系的示意图

第 1 章：计算机、Internet 和万维网简介，讨论什么是计算机，它们是如何工作的，以及在其上是如何编程的。这一章简单介绍了程序设计语言发展的历史：从最早的机器语言，发展到汇编语言，再发展到高级语言；讨论了 C++ 程序设计语言的起源；包含了一个典型的 C++ 编程环境的介绍，并且带领大家在 Windows 和 Linux 操作系统上“试运行”了一个典型的 C++ 应用程序。本章还介绍了基本的面向对象技术的概念和术语，以及统一建模语言。

第 2 章：C++ 编程入门，简单介绍了如何运用 C++ 程序设计语言编写应用程序。这一章引导从未接触过编程的人了解基本的编程概念和程序结构。此章中的程序演示了如何在屏幕上显示数据及如何从键盘读入数据。第 2 章以详细介绍判断的处理和数学运算作为结束。

第3章：类和对象介绍，很自然地将类和对象较早地引见给大家。这一章有助于学生从一开始就顺利接受面向对象的思想。这部分工作是在一支优秀的评审队伍的指导下完成的，这支队伍由工业界和学术界著名的专家组成。我们通过一系列简单的实例来介绍类、对象、成员函数、构造函数和数据成员。在本章中，我们探索出了一个很好的工程化的框架，用来组织用C++编写的面向对象程序。首先，用一个简单实例来引出类的概念。然后，通过精心策划的7个循序渐进可执行的完整程序来展示如何创建和使用自己定义的类。这些例子从关于成绩簿类的开发这一完整实例研究作为出发点，教师可以使用成绩簿类来维护学生的考试成绩。在后续的几章中不断增强这个实例研究，最终版本在第7章中得以实现。GradeBook类实例研究描述了如何定义一个类，以及如何使用它去创建对象；讨论了如何声明和定义成员函数以实现类的一些行为，如何声明数据成员来实现类的属性，以及如何调用对象的成员函数使它们执行任务。在这一章，还介绍了C++标准库中的类string，并创建string对象来保存GradeBook对象所表示的课程的名字；解释了类的数据成员和函数的局部变量之间的差别，以及如何使用构造函数来保证对象的数据在对象创建时就已被初始化；说明了通过将类定义和使用类的客户代码（如函数main）相分离，可以提高软件的可重用性。第3章也介绍了另一条基本的软件工程原则——将接口从实现中分离出来，并包含了一张详细的示意图和相应的讨论，解释了产生可执行程序的编译和连接过程。

第4章：控制语句（第I部分），重点在于创建有用的类的程序开发过程。这一章讨论了如何面对问题陈述，继而开发出一个可解决问题的C++程序，这当然包括用伪代码实现的那些中间步骤。本章介绍了一些简单的控制语句，例如用于判断的if和if...else语句和用于循环的while语句。我们利用第3章的GradeBook类来考究计数器控制的循环和标记控制的循环，并介绍了C++的自增、自减和赋值运算符。这一章包含GradeBook类的两个增强版，每个都基于第3章的最后版本。这些版本各自都包含了一个用控制语句计算一组学生成绩平均分的成员函数。第一个版本中的成员函数采用计数器控制的循环，由用户输入10名学生的成绩，然后计算出平均分。第二个版本的成员函数则采用标记控制的循环，由用户输入任意多名学生的成绩，然后计算平均分。该章还使用简单的UML活动图来呈现通过每个控制语句的控制流。

第5章：控制语句（第II部分），继续以实例的方式讨论C++中的控制语句，包括for循环语句、do...while循环语句、switch选择语句、break语句和continue语句。我们创建了GradeBook类的一个增强版本，它使用switch语句对用户输入的A、B、C、D和F五个等级的成绩个数分别进行统计。这个版本采用标记控制循环来输入成绩。在从用户那里读入成绩的同时，一个成员函数就在修改记录相应等级成绩个数的特定数据成员。之后，GradeBook类的另一个成员函数使用这些数据成员，显示一份基于这些输入成绩的汇总报告。这一章还进行了逻辑运算符的讨论。

第6章：函数和递归入门，深入了解对象及其成员函数。我们讨论了C++标准库函数，并进一步探讨了学生如何构建自己的函数的问题。在第6章中阐述的技术对于编写正确组织的程序是至关重要的，尤其是那些在现实世界应用中很可能由系统程序员和应用程序员开发的大型程序或软件。“分而治之”策略是解决复杂问题的有效手段，其做法是将一个复杂的问题分解成较简单的、相互作用的部分。本章的第一个例子继续进行GradeBook类实例研究，其中列举的函数具有多个参数。学生将会喜欢这一章中关于随机数和模拟方面的知识，以及关于掷骰子游戏的讨论。在解决这些问题的时候，恰到好处地应用了控制语句。本章讨论了所谓的“C++是更好的C”的话题，其中包括内联函数、引用参数、默认实参、一元作用域分辨运算符、函数重载和函数模板等。还介绍了C++的按值调用和按引用调用的功能。头文件表介绍了很多头文件，这些文件是大家在整本书中经常用到的。在这个新的版本中，通过对函数调用堆栈和活动记录的详细讨论（含有图示），解释了C++如何能够跟踪正在运行的当前函数，函数的自动变量如何在内存中进行维护，以及函数在完成它执行的任务后如何知道返回何处。这一章为递归的学习奠定了扎实的基础，并提供了一个表格，总结了会在本书其余部分介绍的大量的递归例子和练习题。有的教材喜欢将递归安排在最后一章，但我们认为这个主题最好均匀地分布于全书各处进行讲解。在本章末尾的大量练习题中，包括了几个经典的递归问题，如Hanoi塔问题等。

第7章：数组与vector，解释如何处理值的列表和表格。我们讨论了将数据组织为具有相同类型的数据元素的数组问题，并表明数组使对象执行其任务更方便。本章前面的部分仍然使用C风格的基于指针的数组。这样的数组，正如大家将在第8章中所看到的，可以视为是指向内存中数组内容的指针。在这一章数组还会以成熟的对象形式出现，我们将介绍C++标准库中的vector类模板，它是一个具有很强健壮性的数组数据结构。本章列举了许多一维数组和二维数组的例子。这章中的例程涉及各种常见的数组操作、直方图打印、数据排序和将数组传给函数等。在本章中有两节含有最后一些关于GradeBook类实例研究的内容。在这两节中，我们在程

序执行期间使用数组来保存学生的成绩。该类的以前版本只是对用户输入的一系列成绩进行处理，并没有在类的数据成员里保存这些成绩。这次则利用数组使GradeBook类的对象在内存中维护这些成绩，从而省去了重复输入这些成绩的过程。本章中GradeBook类的第一个版本，利用一个一维数组来存储这些成绩，并且可以产生一份报告，其中包含了这些成绩的平均分、最高分、最低分以及一张表示成绩分布的直方图。本章中GradeBook类的第二个版本(也是关于这个实例研究的最后一个版本)，则用一个二维数组来存储很多学生一学期里多门考试的成绩。这个版本既可以计算单个学生的学期平均分，又可以得到整个学期所有成绩的最高分和最低分。当然该版本的类也可以产生一张关于这个学期整体成绩分布的直方图。这一章的另一个重要特色就是对基本的排序和搜索技术的讨论。该章后面的练习题包括了大量有趣且富有挑战性的问题，例如改进的排序技术、一个简单的航班订票系统的设计、海龟图概念入门（因LOGO语言而著名）、骑士周游和八皇后问题等，这些都涉及人工智能领域广泛使用的启发性编程的概念。这些练习题总结了许多递归问题，包括选择排序、回文、线性搜索、八皇后问题、数组打印、逆序打印字符串及在数组中查找最小值等。

第8章：指针和基于指针的字符串，介绍C++语言中功能最强大的特性之一——指针。这一章详细解释了指针运算符、按引用调用、指针表达式、指针算术运算、指针和数组之间的关系、指针数组和函数指针等。我们通过讲授指针和const一起使用的方法，实践以最少的特权建立更健壮的软件这一原则。讨论了使用sizeof运算符来确定在程序编译期间数据类型或者数据项的字节大小。在C++中，指针、数组和C风格的字符串之间有着密切的联系。因此，我们介绍了基本的C风格的字符串操作的概念，并讨论了一些最常用的C风格字符串处理函数，例如getline（输入一行文本）、strcpy和strncpy（复制一个字符串）、strcat和strncat（连接两个字符串）、strcmp和strncmp（比较两个字符串）、strtok（将一个字符串“标记”为不同的部分）和strlen（返回一个字符串的长度）等。在这个新版教材中，我们尽量用string对象（在第3章中已介绍）来代替C风格的char *基于指针的字符串。但是，第8章中仍然使用char *字符串，目的在于帮助读者掌握指针，并为将来的职业生涯做好准备。因为在以后的工作中，可能会遇到大量三十多年来实现的C遗留代码。所以，大家非常有必要熟悉C++中这两种最盛行的创建和操作字符串的方法。许多人发现，迄今为止在基础的程序设计课程中，指针这部分内容无疑是最难学的。在C和“原始C++”中，数组和字符串其实就是指针，它们指向内存中数组和字符串的内容（甚至函数的名字也是指针）。通过这一章的认真学习，大家能深入地理解指针这一主题。此章配有富有挑战性的练习题。练习题包括对经典龟兔赛跑问题的模拟、洗牌和发牌问题的算法、递归快速排序和递归走迷宫等。另外，还包括一个特殊的部分，命名为“建立自己的计算机”。这个部分解释了机器语言的编程，并实际开发一个项目，目的是设计和实现一个计算机模拟器，让学生编写和运行机器语言程序。本教材的这个独一无二的特色，对于那些想要知道计算机的真正工作原理的读者是特别有用的。事实上，我们的学生很喜欢这个项目，并经常做出实质性的改进，其中许多的改进已体现在练习题中。另一个很特别的部分包含了颇具挑战性的字符串操作练习题。这些练习题涉及文本分析、字处理、以各种格式打印日期、支票保护、写入支票金额的大写形式、摩尔斯编码和公制-英制单位转换等。

第9章：类的深入剖析（第I部分），继续讨论面向对象的编程。本章通过内容充实的Time类的实例研究来阐述类成员的访问、接口与实现的分离、访问函数和工具函数的使用、用构造函数初始化对象、用析构函数撤销对象、通过默认的“逐个成员复制”进行赋值以及软件的重用性等内容。学生将明白在对象的生命周期中构造函数和析构函数的调用顺序。Time类实例研究的另一个版本向大家演示了在成员函数返回对private(私有)数据成员的引用时所产生的问题，它破坏了类的封装性。这一章的练习题向学生提出了挑战，要求开发关于Time（时间）、Date（日期）、Rectangle（矩形）和游戏TicTacToe（三连棋）的类。通常，学生都很喜欢玩游戏的程序。那些爱好数学的读者会更喜欢创建诸如Complex类（针对复数）、Rational类（针对有理数）和HugeInteger类（针对任意大的整数）等的练习题。

第10章：类的深入剖析（第II部分），继续学习类并讲解其他的面向对象编程概念。这一章讨论了常量对象的声明和使用、常量成员函数、组成（composition）(指构建类的过程，它用其他类的对象作为自己的成员)、friend（友元）函数和friend类（它们具有访问类的private和protected成员的特权）、this指针（它允许对象知道自己的地址是什么）、动态内存分配、static（静态）类成员（用于保存和操作类范围内的数据）、常用的抽象数据类型的例子(数组、字符串和队列)、容器类和迭代器等。在const对象的讨论中，我们提到了关键字mutable。它的用法非常微妙，允许对const对象中的“不可见”的实现进行修改。讨论了使用new和delete进行动态内存分配。当执行new失败时，程序会按照默认的方式中止，因为在标准的C++中new“抛出一个异常”。为了展

开对static类成员的讨论，本章给大家展示了一个基于视频游戏的实例。我们强调了在类的客户面前隐藏类的实现细节的重要性。为此，接下来讨论代理类。代理类是在类的客户面前隐藏实现（包括类头文件中的private数据）的有效手段。这一章的练习题包括了开发一个储蓄存款账户的类和一个拥有整数集合的类。

第11章：运算符重载：String和Array对象，介绍C++课程中最受欢迎的主题之一。学生的确对这一章的内容抱有浓厚的兴趣。他们发现，本章是对“精雕细琢”深入讨论类的第9章和第10章内容的完美补充。通过运算符重载（operator overloading），程序员可以告诉编译器如何将现有的运算符应用于新类型的对象。C++已经知道如何将这些运算符用于内置类型的对象，例如整数、浮点型数据和字符型数据。但是，假如我们新创建了一个类String，那么在String对象之间使用加号会是什么意思呢？很多程序员对字符串使用加号（+），意思是连接相邻的两个字符串。在第11章中，大家将学习如何重载加号。这样，当一个表达式中的两个String对象之间书写了加号时，编译器将产生对一个“运算符函数”的函数调用，该函数将连接这两个String对象。这一章讨论了运算符重载的基本知识、运算符重载时的限制、类成员函数重载和非成员函数重载的比较、一元和二元运算符重载及不同类型间的转换等。第11章的一个特点是收集了许多有意义的实例研究，包括数组类、String类、日期类、大整数类和复数类（最后这两个类及其完整源代码出现在练习题中）。喜欢数学的学生应该会喜欢创建polynomial（多项式）类的练习题。这里材料与大多数的程序设计语言和课程不同。运算符是一个复杂的主题，但也是一个丰富多彩的主题。合理地使用运算符重载会使您的类“锦上添花”。对于已在使用C++标准库中的string类和vector类模板的学生来说，对具有类似功能的Array类和String类的讨论显得特别有价值。这一章的练习题鼓励学生为Complex类、Rational类和HugeInteger类添加运算符重载功能，这样可以方便地使用运算符符号操作这些类的对象，就如同在数学中一样，而不是像第10章中的练习题那样要求采用函数调用。

第12章：面向对象编程：继承，介绍面向对象程序设计语言最基本的性能之一——继承（inheritance）。继承是一种软件重用形式。通过继承，程序员可以吸纳现有类的性能并适当增加一些新的性能，从而快速、较容易地开发新的类。在Employee类层次结构的实例研究中，经过充分修订过的这一章通过连续的5个例子，示范了private数据和protected数据的继承，以及有关继承的良好软件工程等内容。首先，以一个具有private数据成员和public（公有）成员函数的类作为示范的开始，这些成员函数操纵类的这些数据。接下来，实现了具有附加性能的第二个类，故意且乏味地复制第一个例子的大多数代码。第三个例子开始涉及继承和软件重用的知识，采用第一个例子里的类作为基类，快速且简单地在新派生的类中继承了基类的数据和函数。这个例子介绍了继承的机制，并说明了派生类不能直接访问其基类的private成员。这样就促使我们给出第四个例子，在其基类中引入了protected数据，并示范了派生类确实可以访问从基类继承的protected数据。最后一个例子详细说明了正确的软件工程，它将基类的数据定义为private，使用基类的public成员函数（被派生类所继承）在派生类中操纵基类的private数据。本章讨论了基类和派生类、protected成员、public继承、protected继承、private继承、直接基类、间接基类、基类与派生类中的构造函数和析构函数，以及与继承有关的软件工程等。在这一章中，还比较了继承（“is-a”关系）与组成（“has-a”关系），并介绍了“uses-a”和“knows-a”关系。

第13章：面向对象编程：多态性，介绍面向对象编程的另一种基本性能，即多态性行为。全面修订的第13章基于第12章讲授的继承概念，关注的是类层次结构中各个类之间的关系及这些关系所形成的强大处理能力。当许多类通过继承而与一个公共的类相关时，每一个派生类对象都可视为一个基类对象。这样一来，程序可以按照一种简单而一般的方式，即独立于派生类对象的具体类型来编写。新的类型的对象可以被同一个程序处理，从而使系统更具扩展性。多态性使程序能够消除复杂的switch逻辑，取而代之的是更简单的“直线”逻辑。例如，一个视频游戏的屏幕管理器可以发送一条draw（绘图）消息给绘制对象链表中的每个对象。每个对象都知道如何去绘制自己。只要一个新类的对象知道如何绘制自己，那么它可在不改变程序的情况下直接加入程序。这一章讨论了如何利用virtual（虚）函数实现多态性行为，区分了抽象类（由它不能实例化对象）和具体类（由它可以实例化对象）。抽象类适合为遍布层次结构中的各个类提供一个可继承的接口。通过回顾第12章的Employee类层次结构，举例说明了抽象类和多态性行为。引入了一个抽象基类Employee，而CommissionEmployee类、HourlyEmployee类和SalariedEmployee类直接继承Employee类，BasePlusCommissionEmployee类间接继承Employee类。在过去，我们的专业客户都坚持要我们提供更深层次的解释，能够更清楚地说明在C++中多态性是如何实现的，以及当用这个功能强大的特性编程时执行时间和内存“开销”的情况。为此，我们给出了一张示意图，明明白白地解释了C++编译器为支持多态性而自动构建的vtable（virtual函数表）。最后，介绍了运行时类型信息（run-time type information，RTTI）和动态强制类型转换，它们使程序能够在运行时确定

一个对象的类型，然后对该对象做相应的操作。利用RTTI和动态强制类型转换，我们给某个特定类型的雇员增加了10%的工资，然后计算这些雇员的所得。而对于其他的雇员类型，则可以通过多态性来计算他们的所得。

第14章：模板，讨论C++在软件重用方面比较强大的特色之一，也就是模板（template）。利用函数模板和类模板，编程者可以用一段统一的代码，指定一整套相关的重载函数（这些函数称为函数模板特化）或一整套相关的类（这些类称为类模板特化）。这种技术称为泛型编程（generic programming）。函数模板在第6章中做了介绍，在这一章对此继续进行讨论，并给出相应的例子。我们可以为堆栈类编写单一的类模板，然后让C++生成各个类模板特化，例如“int类型的堆栈”类、“float类型的堆栈”类、“string类型的堆栈”类，等等。本章讨论如何对类模板使用类型参数、非类型参数和默认类型，还探讨了模板与其他C++的特性（如重载、继承、友元、static成员等）之间的关系。这一章的练习题很具挑战性，要求学生编写各种各样的函数模板和类模板，并在完整的程序中应用它们。此外，在第21章对标准模板库（STL）的容器、迭代器和算法进行讨论时，将再次强化对模板的理解和应用。

第15章：输入/输出流，全面讨论标准C++的输入/输出性能。这一章既讨论了足够保证大部分常见的输入/输出操作执行的一系列功能，又概括了其余的功能。许多输入/输出的特性都是面向对象的。输入/输出的这种风格利用了其他C++特性，例如引用、函数重载和运算符重载等。C++的各种输入/输出能力包括：用流插入运算符输出、用流提取运算符输入、类型安全的输入/输出、有格式的输入/输出和无格式的输入/输出（出于对性能的考虑）等。用户可以通过重载流插入运算符（<<）和流提取运算符（>>），对自定义类型的对象明确规定如何进行输入/输出。这种可扩展性是C++最有价值的特色之一。C++提供了各种流操纵符，可以执行格式化的任务。本章讨论的流操纵符，能够提供诸如以各种进制显示整数、控制浮点数的精度、设置域宽、显示小数点和尾随零、对齐输出、设置和取消格式状态和设置填充字符等功能。此外，这一章还给出了一个创建用户自定义的输出流操纵符的例子。

第16章：异常处理，讨论程序员如何通过异常处理，能够编写出鲁棒性强、容错性好并且适用于关键业务和关键任务环境的程序。这一章讨论异常处理适用于何时，介绍具有try语句块、throw语句和catch处理器的异常处理的基本性能，指出怎样及何时重新抛出异常，解释如何编写异常说明和处理未预料到的异常，以及讨论异常与构造函数、析构函数和继承之间的重要关系。本章中的练习题向学生展示C++异常处理能力的多样性和强大功能。我们讨论重新抛出异常，并阐明当内存用尽时new如何会失败。当new失败的时候，许多早期的C++编译器返回默认的0值。我们通过抛出一个bad_alloc（坏分配）异常，展示new操作失败的另一种新形式。这一章还演示了如何使用set_new_handler函数，指定一个要被调用的定制函数，该定制函数将处理有关内存耗尽的情况；讨论了如何使用auto_ptr类模板，隐式地删除动态分配的内存，从而避免内存泄漏。本章最后给出标准库的异常层次结构。

第17章：文件处理，讨论创建和处理顺序文件和随机存取文件的技术。这一章首先介绍数据的层次结构，从位到字节、到字段、再到记录，最后至文件。接着，说明从C++的角度如何看待文件和流。我们讨论顺序文件，并构建程序来展示如何打开和关闭文件，如何在文件中顺序地存储数据，以及如何从文件中顺序地读取数据。然后，讨论随机存取文件，并构建程序来说明如何创建随机存取的文件，如何随机地对文件进行数据的读和写，以及如何从随机存取的文件中顺序地读取数据。本章的实例研究将顺序和随机地存取文件的技术结合到一个完整的事务处理程序中。参加我们举办的企业研讨会的学生提到，在学习了关于文件处理的这部分内容后，他们具备了编写实际有价值的文件处理程序的能力，这些程序在他们的企业能够立即得到应用。这一章的练习题要求学生实现各种各样的程序，它们可以构建与处理随机文件和顺序文件。

第18章：string类和字符串流处理，讨论C++由内存的字符串输入数据和向内存的字符串输出数据的能力。这些能力通常称为核内格式化（in-core formatting）或者字符串流处理。string类是标准库中必不可少的组成成分。出于以下几点原因，在第8章及后续章节中继续进行关于C风格的基于指针的字符串讨论。第一，这部分内容可以增强读者对指针的理解；第二，在未来十年左右，C++编程者必须有能力对近三十年积累起来的大量C遗留代码进行阅读和修改。这些遗留代码和工业界多年来已编写的大部分代码一样，都将字符串作为指针来处理。第18章讨论string对象的赋值、连接和比较，说明如何确定各种各样string对象的特性，例如string对象的大小、容量、是否为空等，讨论如何重新设置string对象的大小。我们考虑各种不同的“查找”函数，它们能够在string对象中查找某个子字符串（向前或者向后搜索string对象）展示了对于一个选自于字符组成的string对象中的字符，如何查找它在string对象中的第一次出现或最后一次出现，以及对非出自于string对象中的一个

字符，如何查找它的第一次或者最后一次的出现。这一章还介绍了如何对 string 对象进行替换、删除和插入操作，如何将 string 对象转化成 C 风格的 char * 字符串。

第 19 章：搜索和排序，讨论计算机科学中最重要的两类算法。对于每一类，我们考虑多种特定的算法，并就它们的内存和处理器的开销（采用大 O 标记法，表明一个算法解决问题的难易程度）进行比较。搜索数据涉及确定一个值（称为搜索值）是否在已有的数据中。如果在其中，就找到那个值的位置。在这一章的例子和练习题中，我们讨论各种不同的搜索算法，包括二分查找法，递归版本的线性查找法和二分查找法。通过这些例子和练习题，本章讨论了归并排序法、冒泡排序法、桶排序法、递归的快速排序法等。

第20章：数据结构，讨论用来创建和操纵动态数据结构的技术。这一章首先讨论自引用类和动态内存分配，然后讨论如何创建和维护各种动态数据结构，包括链表、队列、堆栈和树等。对于每一种数据结构，都提供了完整的可运行的程序，并给出输出示例。本章非常有助于学生掌握指针，其中列举了大量的间接引用和双重间接引用（一个非常难的概念）使用的例子。学生在使用指针时比较苦恼的问题是指针概念比较抽象，难以形象化地理解相关的数据结构，以及它们的节点是如何链接在一起的。我们通过一些图解来说明这些链接及其创建的顺序。二叉树堪称是学习指针和动态数据结构最经典有效的示例。这个例子创建一个二叉树，强制进行重复值的删除，并进行了递归的前序、中序和后序的遍历。当学生学习和实现了这个例子后，就会有很强的成就感，特别是当他们看到中序遍历以排序后的顺序打印出节点值时。本章包含了大量内容充实的练习题，其中最精彩的部分就是“特殊小节：构建自己的编译器”。这部分使学生一步步进行中缀到后缀转化程序和后缀表达式求值程序的开发。接着，修改后缀求值算法来生成机器语言代码。这个编译器把这些代码放在一个文件中（运用第 17 章中的技术）。然后，学生在第 8 章构建的软件模拟器上运行由这里的编译器生成的机器语言。这一部分的众多练习题包括了递归地搜索链表、递归地逆向打印链表、二叉树节点的删除、二叉树的层序遍历、打印树、编写部分优化编译器、编写解释器、插入和删除链表的任意一个节点、实现没有尾指针的链表和队列、分析二叉树的搜索和排序的性能、实现一个索引链表的类及一个使用队列的超市模拟系统。在学习完第 20 章之后，读者应该为第 21 章中 STL 容器、迭代器和算法的学习做好了准备。STL 容器是预先打包好的、模板化的数据结构。几乎所有的编程者将发现在绝大多数应用中都会用到它们。STL 对实现重用性而言是一个巨大的飞跃。

第21章：标准模板库（STL）。整本书我们都在讨论软件重用的重要性。认识到许多数据结构和算法被C++编程者普遍使用，C++标准委员会在C++标准库中增加了标准模板库（STL）。STL定义了功能强大的、基于模板的、可重用的组件，它们实现了许多常见的数据结构及用于处理这些数据结构的算法。STL为采用模板的泛型编程概念提供了有力的证明（模板已在第14章做了介绍，在第21章进行了更深入细致的说明）。本章介绍STL并讨论它的三个关键组成部分——容器（流行的模板化的数据结构）、迭代器和算法。STL容器是能够存储任何数据类型对象的数据结构。我们将看到有三种容器类型——首类容器、适配器和近容器。STL迭代器有着和指针类似的性质，程序用其来操纵STL容器元素。事实上，使用标准指针作为迭代器，标准的数组就可以作为STL容器来操纵。我们会发现，使用迭代器操纵容器是很方便的，并且在和STL算法结合的时候，可大大提高表达能力（在某些情况下，可以将多行代码精简为一条单语句）。STL算法是执行常见的数据操作［例如对元素（或者整个容器）进行搜索、排序和比较等］的函数。STL中实现的算法大约有70个，大部分使用迭代器来访问容器的元素。我们将了解每个首类容器都支持特定的迭代器类型，其中的一些比另一些具有更强大的功能。一个容器所支持的迭代器的类型决定了这个容器是否可以被特定的算法所使用。迭代器封装了用于访问容器元素的机制。这种封装使许多STL算法能够应用到多个容器，而不需要考虑底层容器的实现。只要一个容器的迭代器支持算法的最低要求，那么这个算法就可以对该容器的元素进行处理。这也使程序员能够创建可以处理多个不同容器类型的元素的算法。第20章讨论了如何使用指针、类和动态内存实现数据结构。基于指针的代码是非常复杂的，稍有遗漏和疏忽可能就会导致严重的内存访问侵犯和内存泄漏错误，但是却没有编译器警告提示。实现其他的数据结构，例如双端队列、优先队列、集合和映射（map）等，都要求大量的额外工作。此外，如果同一项目的许多程序员为不同的任务实现相似的容器和算法，那么代码就变得很难修改、维护和调试。STL的一个很大优点就是程序员可以重用STL的容器、迭代器和算法来实现公共的数据表示与操作。这种重用机制节约了大量的开发时间和资源。这一章易学易用，使大家认识到了STL的价值，并鼓励读者进行深入的研究学习。

第 22 章：使用 Orge 进行游戏编程，介绍了使用 Ogre（面向对象的图形绘制引擎）进行具有 3D 图形的游戏编程。Ogre 是主流的开源图形引擎软件，已用在包括一些计算机游戏在内的许多商业应用中。首先讨论 3D 游戏编程的基本概念，主要有图形、3D 模型、声音、用户输入、碰撞检测和游戏速度控制等。然后，提供了一

个完整的可运行的简单游戏的实例，该游戏与一款最初由Atari在1972年开发的经典视频游戏Pong相似。本章剖析了该例子，解释其中遇到的关键概念和功能。我们讨论了Ogre所用的各种各样的资源，以及如何用脚本去创建它们。学生将学习如何在3D环境中移动对象、定位对象和重新调整对象的大小，如何进行简单的碰撞检测，如何在游戏中显示文字和如何响应来自用户的键盘输入。还演示了如何使用OgreAL（是OpenAL音频库的包装器）来为游戏添加声音效果。

第23章：Boost库、技术报告1以及C++0x，重点讨论未来的C++。首先介绍了Boost库，它是免费的开源C++库的集合。经过精心设计的Boost库不与C++标准库相冲突。接着讨论技术报告1（TR1），它描述了对标准库提出的修改建议和补充。TR1中的许多库来自于当前在Boost中的库。这一章简要描述了TR1包括的所有库。对于两个最有用的库Boost.Regex和Boost.Smart_ptr，则给出了深入的代码示例。Boost.Regex库提供了对正则表达式的支持。我们示范了如何用Boost.Regex库去搜索一个匹配了正则表达式的字符串，还演示了如何用该库进行验证数据，替换部分字符串和将字符串拆分成一个个标记等。Boost.Smart_ptr库提供了智能指针，帮助管理动态分配的内存。我们讨论TR1中包括的两种智能指针shared_ptr和weak_ptr，通过示例来说明如何使用这两种指针以避免常见的内存管理错误。本章还讨论了即将发布的C++新标准，目前称作C++0x。介绍了这个新标准的目标，概述了最有可能被标准化的核心语言的变动。

第24章：其他主题，是各种C++主题的集合。这一章讨论另外一个强制类型转换运算符—const_cast。与C++从C继承而来的原始强制类型转换运算符（现在遭到抨击）相比，const_cast运算符与static_cast（第5章）、dynamic_cast（第13章）及reinterpret_cast（第17章）运算符一起，提供了更健壮的类型之间转化的机制。我们讨论了名字空间，这对于构建实际系统的软件开发者，特别是对那些由类库构建系统的开发者而言，是一个极其重要的特性。命名冲突会阻碍这样的大型软件的构建，但是名字空间可以防止命名冲突。有些编程者使用的键盘不支持运算符符号中的某些特殊字符（例如，!、&、^、~和|等），本章讨论的运算符关键字对他们非常有用。这些关键字也可以由不喜欢神秘运算符符号的编程者使用。我们讨论关键字mutable，它允许改变const对象的成员。以前，这种改变是通过“抛弃常量性”来实现的（但这被认为是一种危险的操作）。本章还讨论了成员指针运算符.*和->*、多重继承（包括“菱形继承”的问题）及virtual基类。

附录A：运算符的优先级和结合律表，完整地列出了C++所有的运算符符号集，其中的每一行对应一个运算符，同时列出它的运算符符号、名称和结合律。

附录B：ASCII字符集，本书中的所有程序都使用本附录提供的ASCII字符集。

附录C：基本数据类型，列出了C++的基本数据类型。

附录D：计数系统，讨论了二进制、八进制、十进制和十六进制计数系统。考虑了如何在不同计数系统之间进行数值的转换，并解释反码和补码的二进制表示。

附录E：C语言遗留代码问题，介绍一些补充的内容，其中包括若干通常在入门的课程中不会涉及的高级主题。我们展示如何重定向程序输入从文件中获得输入数据，重定向程序输出将输出数据放置到文件中，重定向一个程序的输出成为另一个程序的输入（好比管道连接），以及在一个已有的文件后面追加一个程序的输出等。我们开发了使用可变长度参数列表的函数，并说明命令行参数如何向main函数传递以及如何在程序中使用它们。讨论如何编译其组成部分分布在多个文件中的程序，如何用atexit注册在程序结束时执行的函数，以及如何用exit函数终止程序的运行等。还讨论了const和volatile类型限定符，用整数和浮点后缀指定数值常量的类型，使用信号处理库来阻止不可预料的事件，使用calloc和realloc创建和使用动态数组，以及将联合体作为一种节约空间的技术来使用，并在C++程序和遗留的C代码相连接时使用连接规范。正如标题所体现的那样，本附录主要是为那些在工作中遇到C遗留代码的C++编程者设计的。其实对于大多数C++编程者而言，在实际工作中多少会遇到一些这方面的问题。

附录F：预处理器，详细讨论预处理器的指令。本附录包括了关于#include指令和#define指令更完整的信息。#include指令在文件被编译前，将指定文件的一份副本包含进来，以代替该指令；#define指令创建符号常量和宏。本附录说明了条件编译，使编程者能够控制预处理器指令的执行及程序代码的编译。这里讨论了能把它的操作数转化为字符串的#运算符，以及能将两个记号连接在一起的##运算符。还介绍了各种预定义的预处理器符号常量（__LINE__、__FILE__、__DATE__、__STDC__、__TIME__和__TIMESTAMP__）。最后，本附录讨论了头文件<cassert>中的宏cassert，它在程序的测试、调试、核查和确认方面很有价值。

附录G：ATM实例研究代码，包含了我们用UML进行面向对象设计的实例研究的实现。这个附录是对这个实例研究的快速回顾与总结。

附录H：其他的UML 2示图类型，对OOD/UML实例研究中未涉及的UML 2的示图类型进行概述。

附录I：使用Visual Studio调试器，展示了Visual Studio调试器的主要特性。该调试器允许程序员监控应用程序的执行过程，从而定位和删除逻辑错误。本附录给出了一步步的操作指南，这种手把手的方式使学生很容易学会使用该调试器。

附录J：使用GNU C++调试器，展示了GNU C++调试器的主要特性。该调试器允许程序员监控应用程序的执行过程，从而定位和删除逻辑错误。本附录给出了一步步的操作指南，这种手把手的方式使学生很容易学会使用该调试器。

参考文献。该参考文献列出了大量的书籍和文章，鼓励学生进一步阅读这些关于C++和OOP的资料。

索引。这份内容全面的索引可以帮助读者，通过关键字在整本书中快速找到任何术语或概念的出处。

使用UML进行ATM的面向对象设计：可选学的软件工程实例研究的快速浏览

在这一小节中，我们将纵览一下本书采用UML进行面向对象设计的实例研究（尽管它是可选学的内容）。通过快速浏览，可以预先了解组成这个软件工程实例研究的9个部分（分布在第1～7章、第9章和第13章）。当读者学习完该实例研究后，会对一个重要的C++应用所进行的周密的面向对象设计和实现有彻底的认识。

ATM实例研究中的设计是由Deitel & Associates公司开发的，并经过了一个出色的、不断发展的、由工业界和学术界专家组成的审阅团队的仔细审查。为了满足入门性课程系列的要求，我们精心构思了这个设计。被银行及其遍布全球的客户所使用的实际的ATM系统，要经过更加完善的设计，并且要考虑不在本书范围内其他更多的待解决问题。这个设计过程的主要目标是要创建一个简单的设计，对OOD和UML初学者来说清晰易懂，同时又能阐明关键的OOD概念和相关的UML建模技术。我们尽最大努力使这个设计和代码的规模相对较小，更适合于入门性的课程系列。

1.21节：（必修章节）软件工程实例研究：对象技术和UML的介绍——介绍了用UML进行面向对象设计的实例研究。本节介绍对象技术的基本概念和术语，包括类、对象、封装、继承和多态性等。还讨论了UML的发展历史。这一节是该实例研究中唯一的必修章节。

2.8节：（选学章节）软件工程实例研究：分析ATM的需求文档——讨论了需求文档。该文档详细说明我们将设计和实现的系统［即一个简单的自动取款机（ATM）软件］的需求。我们大致研究了面向对象系统的结构和行为，讨论了UML如何通过它所提供的对系统进行建模的其他几种示图，大大方便了后续软件工程实例研究章节中的设计过程。还给出了一系列关于使用UML进行面向对象设计的URL和参考书目，讨论了由需求文档说明的ATM系统和其用户之间的交互。特别地，我们研究了可能出现在用户和系统自身之间的一个个场景，这些场景称为用例（use case）。我们使用UML的用例图对这些交互进行建模。

3.11节：（选学章节）软件工程实例研究：确定ATM需求文档中的类——开始设计ATM系统。我们通过从需求文档中提取名词和名词性短语来识别系统的类或者“构建块”。我们把这些类组织在一张类图中，该类图描述了我们所模拟的ATM系统的类结构。这张类图同时也描述了称为关联（association）的类和类之间的关系。

4.13节：（选修章节）软件工程实例研究：确定ATM系统中类的属性——集中考虑3.11节所讨论类的属性。一个类同时包含属性（数据）和操作（行为）。后面的章节中将看到，对象属性的改变往往会影响对象的行为。为了确定实例研究中类的属性，我们从需求文档提取描述名词和名词性短语的形容词，然后将这些属性放到3.11节创建的类图中。

5.11节：（选学章节）软件工程实例研究：确定ATM系统中对象的状态和活动——讨论对象在任何给定时刻如何达到一种特定的情况（称为一种状态）。当对象接收到一条改变状态的消息时，就发生一次状态转移。UML提供了状态机图（state machine diagram），它标识了一个对象可能拥有的状态集合，并对对象的状态转移进行建模。对象同样有活动（activity）——在对象的生命周期中所做的工作。UML还提供了活动图——对对象活动建模的流程图。在这一节，我们使用这两种类型的UML图开始对ATM系统特有行为的各方面进行建模，例如ATM系统如何完成一次取款交易，以及在用户确认身份时如何进行响应等。

6.22节：（选学章节）软件工程实例研究：确定ATM系统类的操作——确定类的操作或者服务。我们从需求文档中提取动词或者动词性短语，它们具体说明了每一个类操作。然后，修改3.11节中的类图，把这些操作添加到相关的类里面。此刻在这个实例研究中，我们已经从需求文档中收集到了可能获取的所有信息。可是，随着以后章节内容的介绍，例如介绍继承主题时，还将继续修改类和类图。

7.12节：（选学章节）软件工程实例研究：ATM系统中对象之间的协作——为ATM系统提供了“草图”模型。在这一小节可以看到它是如何对系统建模的。我们通过讨论协作（collaboration，对象之间互相发送的、用来通信的消息）来研究这个模拟程序的行为。在6.22节中所确定的类的操作变成了系统中对象之间的协作。我们确定这些协作，然后把它们集中到一张通信图（communication diagram，UML对协作建模的图）中。这张图揭示了有哪些对象及在什么时候进行协作。我们给出了执行ATM余额查询时对象之间协作的通信图。然后给出了对系统中的交互进行建模的UML顺序图，这种图强调消息发生的时间顺序。本节的一张顺序图对系统中的对象如何相互作用来执行取款和存款交易进行了建模。

9.11节：（选学章节）软件工程实例研究：开始对ATM系统的类进行编程——结束系统行为设计的介绍，开始进入实现的过程，这个过程着重用到第9章的知识。利用3.11节的UML类图、4.13节和6.22节中讨论的属性和操作，我们展示了如何由设计进行C++类的实现。在此，并没有实现所有的类——因为还没有完成所有的设计。根据ATM系统的UML图，我们为Withdrawal（取款）类创建了实现的代码。

13.10节：（选学章节）软件工程实例研究：在ATM系统中引入继承——继续面向对象编程的讨论。我们考虑继承，也就是说，共享共同特性的类可以从一个基类继承属性和操作。本节研究了ATM系统如何从使用继承中受益。我们将这些新内容表示在一张对继承关系建模的类图中，在UML中将这些关系称作泛化（generalization）。通过使用继承，我们修改了3.11节中的类图，使用相似的特性对类进行分组。这一节结束了ATM模拟系统建模部分的设计。在附录G中给出了完整实现这个模型的877行C++代码。

附录G：ATM实例研究代码——这个实例研究的多数内容涉及ATM系统的模型（即数据和逻辑）的设计。在这个附录中，我们用C++实现了该模型。利用创建的所有的UML图，给出了实现这个模型所必需的C++类。我们应用了本书所学的用UML进行面向对象设计和用C++进行面向对象编程的概念。在本附录的学习接近尾声时，学生应已完成了一个实际系统的设计和实现，并且有信心去开发较大的系统（例如，那些由专业软件工程师构建的系统）。

附录H：其他的UML 2示图类型——对OOD/UML实例研究中未涉及的UML 2的示图类型进行概述。

本书的学生资源

现在有许多C++开发工具可供大家使用。我们所著的《C++大学教程》（第六版）一书主要使用的是微软免费的Visual C++ Express版本（即Visual C++速成版，随书附送的CD中含有该软件）和免费的GNU C++。大多数Linux系统都已经预装了GNU C++，它同样可以安装在Mac OS X系统上。如果想更深入地了解Visual C++ Express版本，可以访问msdn.microsoft.com/vstudio/express/visualc。在gcc.gnu.org上有更多关于GNU C++的介绍。Apple在它们的Xcode开发工具包中已经包含了GNU C++，Mac OS X用户可以从developer.apple.com/tools/xcode上下载该开发工具包。

在我们的C++资源中心（www.deitel.com/cplusplus/）和本书的网站（http://int.prenhall.com/deitel）上，有其他的资源和软件可供大家下载。

此外，若要免费下载其他的编译器，可以访问下面的网站：

```
www.thefreecountry.com/developercity/ccompilers.shtml
www.compilers.net
```

关于早期C++编译器的警告和错误信息

本书中所设计的程序需要支持标准C++的编译器来编译。然而，不同的编译器难免会产生一些警告和错误。而且，尽管标准详细说明了错误产生的各种情形，但是它并没有指定编译器应该发出的消息。警告和错误消息随编译器的不同而不同，这是很正常的事情。

一些较早的C++编译器，例如，Microsoft Visual C++ 6、Borland C++ 5.5和各种GNU C++的早期版本，会在新版本不产生错误或警告信息的地方产生错误或警告信息。虽然本书中的大多数例子可以通过较早的编译器进行编译，但是还有少数例子需要经过轻微改动才能在其上工作。

关于using声明和C标准库函数的说明

C++标准库包括了C标准库的函数。根据C++标准文档，C标准库的头文件内容是std名字空间的一部分。对于C函数，一些旧的和新的编译器在遇到using声明时会产生错误消息。我们将在www.deitel.com/books/cpphtp6/index.html上公布这些问题。

在线的C++网络课堂

对于从Prentice Education购买C++ How to Program, 6/e（中译名为《C++大学基础教程（第六版）》，已由电子工业出版社出版）的读者（针对本书原版的销售市场）而言，可以同时使用基于Web的语音交互式多媒体辅助教学资料——C++ Multimedia Cyber Classroom, 6/e（C++多媒体网络课堂，第六版）。如果您使用的是一本二手书，也可以在MyPearsonStore.com上单独购买该“网络课堂”的访问权。我们的这个基于Web的“网络课堂”包括了教材中第1章至第14章所有的代码范例的有声讲解、约一半以上课后练习的解答、一份实验手册及其他内容。要想获知更多的基于Web的“网络课堂”的相关信息，请访问：

```
http://int.prenhall.com/deitel
```

使用过这个“网络课堂”的学生告诉我们，他们很喜欢其中的互动方式，它的确是一个强有力的教学参考工具。教师们也提及我们的“网络课堂”非常受他们学生的欢迎，因此，与仅仅只有教科书的课程相比，学生在这门课上所花费的时间更多，掌握的知识也更多。

本书的教师资源

《C++大学教程》（第六版）有大量配套的教师资源。Pearson Education的教师资源中心提供了以下的辅助教师教学的材料：

- 解答手册，含对书中各章后所附大部分练习题的解答
- 多项选择题的测验文件（对书中的每节大概有两题）
- PowerPoint的幻灯片，涵盖了书中的所有代码和图表，还包括了总结书中各个要点的摘要条目。教师可根据需要对这些PowerPoint的幻灯片进行定制。

如果您还不是一名注册的教师会员，请与所在地区的Pearson代理商联系以获得访问这些资源的资格[①]。

Deitel Buzz Online免费电子邮件时事通信

Deitel Buzz Online免费电子邮件时事通信每周都会发布我们最新的资源中心信息，以及最新行业发展趋势和发展的评论、与我们已出版和即将出版图书有关的免费论文与资源的链接、产品发布计划表、勘误表、挑战、奇闻轶事和Deitel公司教师引导式培训课程的信息等。此免费的电子邮件时事通信可以说是大家能及时了解到任何与本书相关信息的非常好的渠道。需要订阅Deitel Buzz Online免费电子邮件时事通信的读者，可直接登录：

```
www.deitel.com/newsletter/subscribe.html
```

Deitel在线资源中心

我们的网站www.deitel.com提供了许多资源中心（参见图2），涵盖了各种各样的主题，包括程序设计语言、软件、Web 2.0、互联网业务和开源项目等。我们在编写书籍和开展其他业务活动时都会致力于做出一定的研究，资源中心正是从这些研究工作中发掘出来的。我们已经发现了许多不错的资源，包括教程、文档、软件下载、文

① 国内教师申请教辅资源的方法，请参见本书末尾的“教学支持说明”页。