

世界著名计算机教材精选

PEARSON

数据结构与面向对象 程序设计 (C++版) (第4版)

Michael Main
Walter Savitch

金名 等

著
译



DATA STRUCTURES AND OTHER OBJECTS USING C++

Fourth Edition

清华大学出版社

世界著名计算机教材精选

数据结构与面向对象程序设计

(C++版)

(第4版)

Michael Main
Walter Savitch 著

金 名 等译

清华大学出版社
北 京

Simplified Chinese edition copyright © 2012 by PEARSON EDUCATION ASIA LIMITED and TSINGHUA UNIVERSITY PRESS.

Original English language title from Proprietor's edition of the Work.

Original English language title: Data Structures and Other Objects Using C++ , 4E by Michael Main, Walter Savitch © 2012

EISBN: 0-13-212948-5

All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Addison Wesley.

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macao).

本书中文简体翻译版由 Pearson Education(培生教育出版集团)授权给清华大学出版社在中国境内(不包括中国香港、澳门特别行政区)出版发行。

北京市版权局著作权合同登记号 图字:01-2011-3516 号

本书封面贴有 Pearson Education(培生教育出版集团)激光防伪标签,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

数据结构与面向对象程序设计: C++ 版: 第4版/(美)梅因(Main, M.), (美)萨维特奇(Savitch, W.) 著; 金名等译. --北京: 清华大学出版社, 2012.5

书名原文: Data Structures and Other Objects Using C++ , 4E

(世界著名计算机教材精选)

ISBN 978-7-302-27881-8

I. ①数… II. ①梅… ②萨… ③金… III. ①数据结构—教材 ②C语言—程序设计—教材 IV. ①TP311.12 ②TP312

中国版本图书馆 CIP 数据核字(2012)第 008292 号

责任编辑: 龙启铭

封面设计: 傅瑞学

责任校对: 焦丽丽

责任印制: 杨 艳

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 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 装 者: 北京鑫海金澳胶印有限公司

经 销: 全国新华书店

开 本: 185mm×260mm

印 张: 46.75

字 数: 1137 千字

版 次: 2012 年 5 月第 1 版

印 次: 2012 年 5 月第 1 次印刷

印 数: 1~3000

定 价: 89.00 元

产品编号: 042416-01

译 者 序

本书以数据结构为基础，以 C++ 为实现语言，从软件开发的角度的思想，系统阐述了指针和动态数组、链接、模板类、迭代器、栈、队列、递归实现、树和图等内容。为了让读者巩固所学知识，在每节后面都给出了大量练习题，并在每章后面给出了答案。为了让读者融会贯通所学知识，对同一个示例，给出了不同的实现，让读者体会到各种实现的优缺点，为了锻炼读者的动手能力，每章末给出了大量编程项目，在本书的配套网站 www.cs.colorado.edu/~main 还给出了一些更具挑战性的编程项目。

本书不仅非常适合于作为计算机及相关专业“数据结构”和“C++面向对象程序设计”的教材，也是计算机软件开发人员的常备参考书。

本书由金名主译，张长富、蔡建章、李匀、张建安、邓铁洪、徐君、李强、蒋恩俊、杨文保、李强、苏辛、周成兴、魏敬安、朱建波、徐志平、赵杰辉、傅祎、郭碧莲、郭洵、洪晓煜、黄宣达、江松波、柯渝、赖曲芳、廖阳、刘文红、李伟、郭涛、高磊、王振营、冯哲、韩毅、马以辉、李腾、邓卫、邓凡平、周云、董武、郑晓蕊、陈占军、倪泳智、黄虹、吕巧珍、裘蕾、金颖、陈河南、王嘉佳、吴建伟、宋雁、贺军等人也参与了本书的翻译工作，在此一并表示感谢！

由于水平有限，如有不妥之处，恳请读者指正

前 言

本书是为计算机科学专业的第二门课程编写的，在美国很多大学中称之为 CS2。本书的重点是规范说明、设计和实现，以及基本数据类型的使用，这些内容都将在第二学期的课程中介绍。而且，本书还介绍了很多重要的编程技术，提供了有关抽象技术、面向对象程序设计、算法的大 O 时间分析以及排序等内容。

本书假设学生已经完成了“计算机科学导论”和“程序设计”课程的学习，但本书也包括了在第一门课中没有完全涵盖的内容（如递归和指针）。本书使用的是 C++ 语言，但对 C++ 类的介绍是从零开始的，因此，对于那些以 C 而不是以 C++ 作为程序设计入门语言的学生来说，本书也是适用的。根据笔者的经验，这类学生需要对 C++ 输入和输出技术（参见附录 F），以及 C++ 参数类型（参见第 2 章）有所了解。当 C 程序员克服了有关输入/输出和参数的障碍后，就能领略 C++ 的类和面向对象特性。需要指出的是，根据学生不同的知识背景，本书有多种不同的学习方案。对于那些有较强应用背景的学生来说，可以有选择地学习本书的内容。

本版新增内容

C++ 标准模板库 (Standard Template Library, STL) 在我们的课程中起着更大的作用了，在本书中增加了精心挑选的素材给予介绍。对我们来说，重要的是让我们的学生既能理解如何在应用程序中使用 STL 类，也能掌握实现这些类（或类似类）的途径。基于这种思考，本版进行了以下修改：

- 新增了 2.6 节，利用 `pair` 类对标准模板库进行早期介绍。我们在学生还没有完全理解模板之前，就向他们介绍 STL。
- 在 3.4 节提前介绍 `multiset` 类和 STL 迭代器。这是介绍这些内容的好地方，因为学生刚刚明白了如何实现他们自己的第一个集合类（即 `bag` 类），该类是基于 `multiset` 类的。
- 在 4.5 节继续介绍 STL 的 `string` 类，在这里很适合学生用动态数组来实现自己的 `string` 类。
- 新增 5.6 节，对 3 个类似的 STL 类：`vector`、`list` 和 `deque` 进行比较。此时，学生有足够的知识来理解典型的 `vector` 和 `list` 类实现。
- 在 6.3 节首次介绍 STL 算法，在 11.2 节和 13.4 节进行了扩展介绍。
- 新增 8.4 节，介绍了联合使用动态数组和指针，实现 STL 的 `deque` 类的细节。
- 在 12.6 节讨论了 TR1 库中的散列表。

大多数章节增加了很多新编程项目，你还可以关注本书的配套网站了解我们开发的新项目：www.cs.colorado.edu/~main/dsoc.html。

每种数据类型的 5 个介绍步骤

总体来说,第4版继续介绍如下的经典数据结构:集合、包(或多集)、序列表、有序列表(利用“小于”操作符排序)、栈、队列、表和图。此外,还补充介绍了一些数据结构,比如优先队列。上述每种数据结构都将按照如下固定的模式来进行介绍。

第1步:抽象地理解数据结构。在这一层上,学生从概念和图形的层面上,得到对数据结构及其操作的理解。例如,学生可以可视化一个栈及其压入和弹出元素的操作。简单的应用程序更容易理解,而且可以手工完成,比如使用栈来颠倒一个单词的字母顺序。

第2步:把数据结构的规范说明编写成一个 C++ 类。在这一步中,学生明白和理解如何为实现数据类型的 C++ 类编写规范说明。该规范说明包括构造函数、公用成员函数的原型,以及其他公用属性(比如一个用于确定栈的最大尺寸的基本常量)。每个成员函数的原型都给出了前置条件和后置条件,完整地描述了该函数的行为。在这一层很重要,让学生明白规范说明与任何实现技术无关。事实上,相同的规范说明可以用于描述相同数据类型的多种不同实现。

第3步:使用数据类型。有了规范说明,学生就可以编写小型的应用程序或演示程序,来展示所使用的数据结构。这些应用程序只是基于数据类型的规范说明,还没有涉及实现。

第4步:选择适当的数据结构,并继续设计和实现数据类型。在对数据类型有了一个好的抽象理解之后,就可以选择一个恰当的数据结构,比如固定大小的数组、动态数组、节点链表或节点二叉树。对于很多数据类型来说,最初的设计和实现会选择简单的方式,比如固定大小的数组。随后,我们将用更复杂的基本结构来重新设计和重新实现相同的数据结构。

由于本书使用的是 C++ 类,对于某个数据类型的实现,将有多种可供选择的数据结构(比如数组、指针等)来作为类的私有成员变量。对于每个实现的类,我们强调应清楚理解私有成员变量与数据类型抽象表示之间的关联规则。我们要求每个学生应用简洁的语言写出这些规则(我们称之为抽象数据类型的不变式)。一旦不变式编写完毕后,就可以继续实现各种成员函数了。不变式有助于编写正确的函数,原因有两条:(1)当函数开始运行时,每个函数(构造函数除外)都知道不变式已为真;(2)当函数结束运行时,每个函数(析构函数除外)都要确保不变式为真。

第5步:分析实现。每个实现的分析包括正确性、灵活性(例如固定大小还是动态大小)以及各种操作的时间分析(使用大 O 表示法)。当同一种数据类型以多种不同的方式来实现时,学生就可以有很好的机会对它们进行分析。

本课程结束时,学生将学到什么

在本课程结束时,学生将可以彻底理解数据类型了。他们知道如何使用数据类型,如何以多种不同的方式来实现数据类型,知道不同实现方法的实际效果,可以使用大 O 表示法来分析出各种实现的效率,通过类的不变式来论证各种实现的正确性。

本课程对今后学习有持久重要影响的是,在本课程中学生亲身体验了各种规范说明、

设计和实现的过程。本课程对提高学生效率分析和正确性论证的能力也有很重要的帮助。但最重要的一点是，本课程向学生揭示了类可以在很多情况下很容易地使用。学生不再需要从零开始编写代码。我们告诉学生，将来某一天，当他们考虑某个问题时，会突然发现大量工作可以用 `bag` 类、`stack` 类、`queue` 类或其他类来完成。这些工作根本不需要他们自己去做什么，他们只需把在本课程中编写的 `bag` 类、`stack` 类、`queue` 类或其他类拿出来用就可以，无需任何修改。更多的情况是，他们可以使用标准数据类型库（比如 C++ 标准模板库）中各种熟悉的数据类型。事实上，本书中所介绍的数据类型是标准模板库中的精简版本，因此，当学生进一步学习真正的标准模板库时，他们就有了一个较为熟悉的基础知识。在这个基础上进行设计实现时，就知道哪种数据类型是他们最需要的了，他们也就成为了真正的程序员。

其他基础知识

在本课程中，除了介绍基本的数据结构知识外，还为进行“真正的程序设计”所需的其他知识打下基础，具体如下。

1. 面向对象程序设计

通过让学生深入理解 C++ 类，为面向对象程序设计（Object-Oriented Programming, OOP）打下基础。在课程早期会介绍有关类的重要内容：成员函数的表示法，私有成员与公用成员的分离，构造函数的作用，以及操作符重载等。这些内容足以让学生顺利而兴奋地开始类的学习。

当首次使用动态内存（第 4 章）时，将进一步介绍类的更多重要特性。此时，需要介绍另外 3 个方面的内容：复制构造函数、重载后的赋值操作符和析构函数。通过第一次使用动态内存时来介绍 OOP 的这些内容，可以给学生一个有关动态内存的具体蓝图。动态内存是一种可以拿来使用但随后必须收回的资源。

从概念上来说，OOP 最大的创新在于通过继承实现软件重用。当然，也可以在数据结构课程开始时就介绍继承（例如，把 `set` 类实现为 `bag` 类的一个子类）。但是，过早介绍会使得过多地关注太多新概念，影响对基本数据结构的理解。因此，在我们的课程中，将继承放在最后介绍。但对继承的概述（14.1 节和 14.2 节）可以在理解了复制构造函数之后就进行。基于这种想法，一些教师可能希望在栈与队列之前讲解第 14 章。

另一种方式是，对于那些已经了解类的基本知识的学生，让他们早些开始进行继承项目（比如 14.2 节的生态系统或 14.3 节的游戏引擎），而其他学生则先学习类。

2. 模板

模板函数和模板类是标准模板库的一个重要部分，使得程序员可以很容易地修改容器类中的基本数据项的类型。模板类还可以在一个程序中使用某个类的多个不同实例。同样，在栈（第 7 章）之前学习和使用模板（第 6 章）很重要，因为对使用了两种栈的表达式求值是一个很重要的应用。

3. 迭代器

迭代器是标准模板库的另一个重要部分,使得程序员可以很容易地遍历容器对象中的数据项(比如 `set` 类或 `bag` 类中的元素)。这种迭代器可以是内部的(实现为容器类的成员函数),也可以是外部的(由一个单独的类来实现,该类是容器类的友元)。我们在讲述第一个容器类(3.2 节的 `sequence` 类)时引入了内部迭代器。在第 6 章需要时,给 `bag` 类添加了一个内部迭代器。此时,还讨论了更复杂的外部迭代器,学生应该明白外部迭代器的优点。在本书中,迭代器为许多编程项目提供了一个很好的选择,例如,实现一个外部 `bag` 迭代器(第 6 章),或使用栈来实现二叉搜索树的内部迭代器(第 10 章)。

4. 递归

第一学期有时候会向学生介绍递归。但第一学期的很多示例是尾递归,其中,函数的最后一步是递归调用。这可能会给学生一个错误的印象,认为递归只不过是一个循环。鉴于此,在第二学期的课程中,尽量避免过早地使用尾递归。例如,链表中的遍历和其他操作都可以用尾递归来实现,但如果这样做,其结果是增强对递归的错误认识(当学生在操作含有成千上万个数据项的链表时,应忘记尾递归的链表操作,以免出现潜在的运行时栈溢出)。

因此,在第二学期的课程中,我们重点介绍的是使用除尾递归之外的其他递归解决方案。第 9 章就是遵循这种理念来介绍 3 个示例的。其中的两个示例(产生随机分形和穿越迷宫)对学生会有较大的触动。在我讲授的课堂上,先讲授递归(第 9 章),紧接着讲授树(第 10 章),因为在递归树算法中,递归是非常重用的。但是,对于那些希望更加强调递归的教师来说,可能会提前讲授递归,有的甚至提到第 2 章之前。

在学时比较充裕的课程中,可以学习高级树项目(第 11 章),分析递归树算法,解释一下保持树平衡的重要性。这两者都提高了最坏情况下的性能,避免出现潜在的运行时栈溢出。

5. 搜索和排序

第 12 章和第 13 章介绍了搜索和排序算法的基础知识。第 12 章回顾了有序数组的二叉搜索,许多学生之前已经见过这些内容了。第 13 章回顾了简单的二次排序方法,但该章的重点是快速算法:递归合并排序(最坏情况下的运行时间为 $O(n \log n)$)和 Tony Hoare 递归快速排序(平均运行时间为 $O(n \log n)$)以及基于树的堆排序(最坏情况下的运行时间为 $O(n \log n)$)。此外还对 C++ 标准模板库的排序函数做了新的介绍。

高级项目

本书提供了很多项目,可以用更高级的类来实现,或者让那些在设计大型类方面有扎实基础的学生来完成。部分高级项目如下:

- 使用动态内存的 `polynomial` 类(4.6 节)。
- 标准库迭代器,这是用于 `bag` 类的迭代器的最高级实现(6.3 节~6.5 节)。
- 用于二叉搜索树的迭代器(第 10 章的编程项目)。

- 用链表（第 8 章的项目）或堆（11.1 节）实现的优先队列。
- 用 B-树（11.3 节）实现的 set 类。我们对这个项目进行了全面介绍，提供了足够的信息，以便学生无需参考其他教材，就可以实现该类。在我们的课程中，我们已经成功指导一些优先的学生独立完成该项目。
- 使用继承的项目，如 14.2 节的生态系统项目。
- 使用抽象类实现继承项目，比如 14.3 节的 game 基类（利用该类可以很容易地实现两玩家的游戏，比如 Othello 或 connect 4）。
- 第 15 章的 graph 类以及相关的图算法。这是优先学生可能独立完成的另一个案例。

C++语言特性

C++是一种具有很多高级特性的复杂语言，在第二学期的课程中不会触及到这些内容。但本书将力求介绍所遇到的那些特性。在本书的第 1 版中，介绍了当年 C++的两个特性：新的布尔数据类型（参见图 2.1）和静态成员常量（参见 3.1 节）。关于静态成员变量的使用要求在 1998 年发布的标准中进行了修改，本书已经体现了这种修改（现在，常量的声明必须在类定义内部和外部同时进行）。1998 标准的另一个主要新特性是名称空间的使用，在这本书第 2 版中也已经体现了。一些老式的编译器可能不支持这两种新特性。为此，我们给出了一些解决这些问题的辅助办法（参见附录 E），以及一些有关下载和安装 GNU g++ 编译器的帮助（参见附录 K）。

章节安排的灵活性

本书在编写方式上给予了教师很大的选择空间，他们可以根据特定背景的学生调整教学内容，或者有选择地先讲授某些章节。本前言的最后给出了本书章节之间的相互关系。两个方框之间的连线表明上一个方框中的内容必须在下一个方框中的内容之前讲授。

这里给出了关于本书内容讲授次序的一些参考建议：

- **典型课程。**从第 1~10 章开始，如果学生已经具备 C++类的基础知识，可以跳过第 2 章的部分内容。这其中的大多数章节可以在一周内讲授完，但第 5 章、第 6 章、第 9 章和第 10 章则可能需要多一些的时间。通常，我们是用 13 周的时间来讲授这些内容，包括考试时间以及讲授链表和树的时间。剩余时间可以用来讲授第 11 章或 12.1 节和第 13 章。
- **侧重面向对象程序设计的课程。**如果学生在其他课程中已经学习了排序与搜索，那么就有时间来重点学习面向对象程序设计了。第 1~4 章可以详细讲授，然后介绍派生类（14.1 节）。此时，可以让学生做一些有趣的 OOP 项目，比如 14.2 节的生态系统或 14.3 节的游戏。然后讲授基本的数据结构（第 5~8 章），把队列实现为一个派生类（14.3 节），最后以第 9 章和第 10 章结束本课程，重点强调递归成员函数。
- **快速课程。**把第 1~3 章作为第一周的自学内容，从第 4 章开始讲授，这样在学期末会留出两三周的时间，于是学生可以在搜索、排序以及其他高级主题上投入更多

的时间。

我们也曾经以更快的速度来讲授本课程，不讲授栈和队列（但把这些章节作为自学内容）。

- **提前讲授递归或提前讲授排序。**前一到三周的时间用于讲授递归思想，然后阅读第1章和第9章的内容，可能还要补充递归项目的学习。

如果提前讲授递归，那么也就可以在介绍 C++ 类之前，讲授二叉搜索（12.1 节）和大部分的排序算法（第 13 章）。

通过 Internet 获取教辅内容

在 www.aw-bc.com/cssupport 网站中，所有读者都可以访问如下内容：

- 源代码。本书中出现过的所有 C++ 类、函数和程序。
- 勘误表。我们尽可能不犯错误，但有时候难免。在上述网站中给出了一个已经发现的错误列表，并随时按需更新。欢迎您贡献您所发现的错误。

下面内容只能提供给符合条件的教师：

- PPT
- 考试题
- 部分编程项目的参考答案
- 作业与实验练习
- 建议的教学大纲

致谢

当 Walter 教授在 Colorado 大学计算机系访问 Michael 教授时，开始了本书的写作。在 Walter 教授回到 California 大学工程与计算机科学系后，本作品才完成。我们非常感谢这些大学为我们提供条件，感谢那些优秀的学生，感谢那些志同道合的同事。

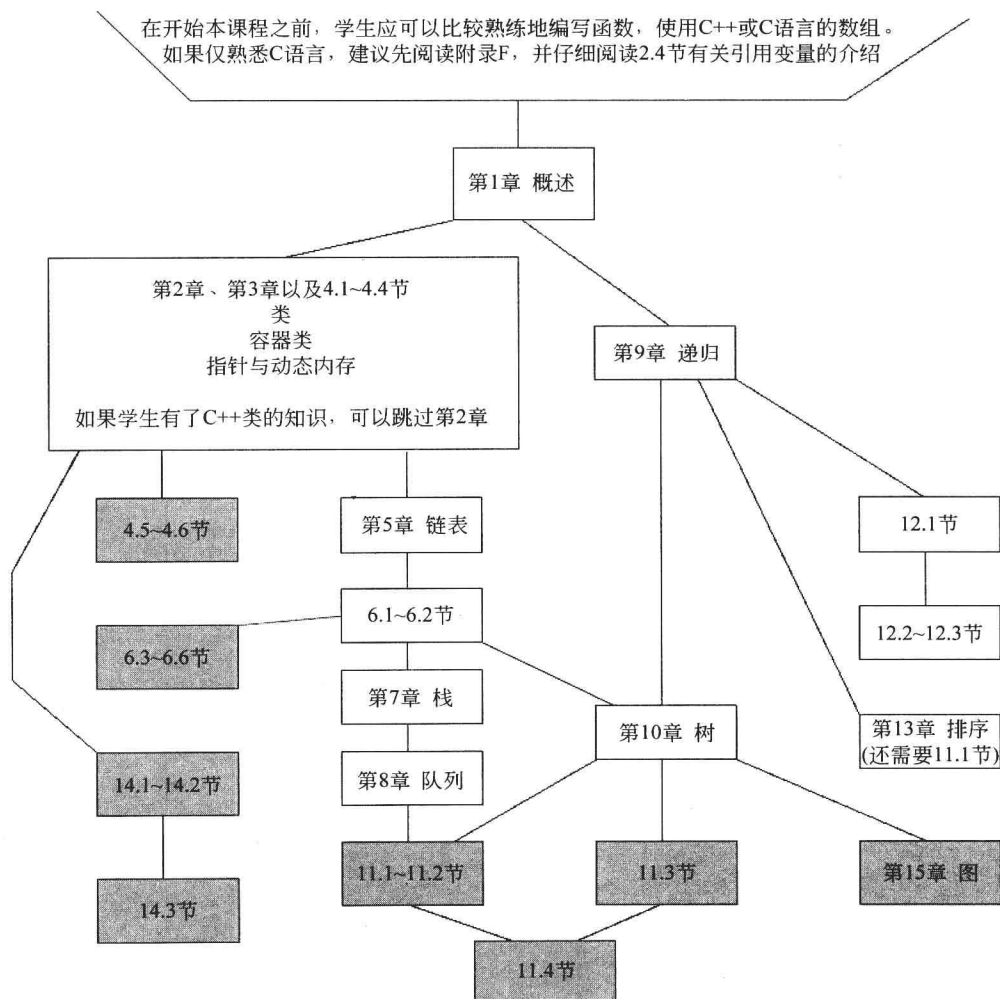
我们的学生给了我们特别重要的帮助，有将近 5000 个我们的学生阅读了本书的材料，提出建议，告诉我们他们是如何学习的。我们感谢那些在他们的课程中使用本书材料并提出反馈意见的审阅人和教师：Zachary Bergen、Cathy Bishop、Martin Burtscher、Gina Cherry、Courtney Comstock、Stephen Davies、Robert Fronhardt、John Gillett、Mike Hendricks、Ralph Hollingsworth、Yingdan Huang、Patrick Lynn、Ron McCarty、Shivakant Mishra、Evi Nemeth、Rick Osborne、Rachelle Reese 和 Nicholas Tran 等。本书还请以下人员进行了审阅：Wolfgang W. Bein、Bill Hankley、Michael Milligan、Paul Nagin、Jeff Parker、Andrew L. Wright、John R. Rose 和 Evan Zweifel 等。我们感谢这些同事的出色批评和鼓励。

感谢 Lesley McDowell 和 Chris Schenk，他们在 Colorado 大学计算机系的每一天都是快乐和热情的。我们还要感谢 AW 出版社的编辑和工作人员。Heather McNally 的工作激励着我们，与我们进行融洽的交流，使本作品出版的每一步都变得轻松惬意。在 Colorado 大学，Karin DeJamaer 和 Jessica Hector 给予了友善的鼓励，我们感谢他们。我们感谢和欣赏 Michael Hirsch 的编辑工作，他总是有着过人的精力、热情和鼓舞。最后，我们原来的编辑 Susan

Hartman 仍然继续给予我们支持、鼓励和指导，没有你，这本书不可能面世！

除了感谢以上这些人的工作和支持之外，我们还有感谢那些每天为我们提供快乐和鼓励的人。我们深切地感谢 Holly Arnold、Vanessa Crittenden、Meredith Boyles、Suzanne Church、Erika Civils、Lynne Conklin、Andrzej Ehrenfeucht、Paul Eisenbrey、Skip Ellis、John Kennedy、Rick Lowell、George Main、Mickey Main、Jesse Nuzzi、Ben Powell、Marga Powell、Megan Powell、Grzegorz Rozenberg、Hannah、Timothy 和 Janet 等。

章节相互关系图如下所示：



注：带底纹方框为高级主题

目 录

第 1 章 软件开发的阶段	1	2.3.4 使用名称空间里的数据项	46
1.1 规范说明、设计与实现	2	2.3.5 本节自测练习	48
1.1.1 概念设计：问题分解	3	2.4 类与参数	49
1.1.2 前置条件与后置条件	4	2.4.1 编程示例：point 类	49
1.1.3 使用由其他程序员提供的 函数	6	2.4.2 参数默认值	52
1.1.4 有关 ANSI/ISO C++ 标准的实现问题	8	2.4.3 参数	53
1.1.5 本节自测练习	11	2.4.4 当函数的返回值的 数据 类型为类时	58
1.2 运行时间分析	12	2.4.5 本节自测练习	59
1.2.1 台阶计数问题	12	2.5 操作符重载	60
1.2.2 大 O 表示法	16	2.5.1 二元比较操作符重载	60
1.2.3 C++ 函数的时间分析	17	2.5.2 二元算术操作符重载	61
1.2.4 最坏情况、平均情况以及 最好情况下的时间分析	19	2.5.3 输入输出操作符重载	62
1.2.5 本节自测练习	19	2.5.4 友元函数	64
1.3 测试与调试	20	2.5.5 point 类汇总	66
1.3.1 选择测试数据	20	2.5.6 操作符重载小结	68
1.3.2 边界值	20	2.5.7 本节自测练习	69
1.3.3 完全代码测试	21	2.6 标准模板库与 pair 类	69
1.3.4 调试	21	2.7 本章小结	70
1.3.5 本节自测练习	22	本章自测练习参考答案	71
1.4 本章小结	22	编程项目	73
本章自测练习参考答案	23	第 3 章 容器类	81
第 2 章 抽象数据类型与 C++ 类	25	3.1 bag 类	81
2.1 类与成员	25	3.1.1 bag 类的规范说明	81
2.1.1 编程示例：节流阀类 throttle	25	3.1.2 bag 类的文档说明	88
2.1.2 使用类	29	3.1.3 bag 类的演示程序	91
2.1.3 throttle 类的演示小程序	30	3.1.4 bag 类的设计	93
2.1.4 实现成员函数	31	3.1.5 类的不变式	94
2.1.5 可以调用其他成员的 成员函数	34	3.1.6 bag 类的实现	95
2.1.6 本节自测练习	34	3.1.7 bag 类的集成	101
2.2 构造函数	35	3.1.8 bag 类的测试	104
2.2.1 throttle 类的构造函数	35	3.1.9 bag 类的分析	105
2.2.2 修订 throttle 类的成员 函数	37	3.1.10 本节自测练习	106
2.2.3 内联成员函数	38	3.2 编程项目：sequence 类	107
2.2.4 本节自测练习	39	3.2.1 sequence 类的规范说明	107
2.3 使用名称空间、头文件与 实现文件	39	3.2.2 sequence 类的文档说明	111
2.3.1 创建名称空间	40	3.2.3 sequence 类的设计	113
2.3.2 头文件	40	3.2.4 sequence 类的伪代码实现	114
2.3.3 实现文件	44	3.2.5 本节自测练习	116
		3.3 交互式测试程序	116
		本节自测练习	121
		3.4 STL 中的 multiset 类及其迭代器	122
		3.4.1 multiset 模板类	122
		3.4.2 multiset 类的一些成员	123
		3.4.3 迭代器与[...]模式	123

3.4.4	测试迭代器的相等性	126	4.5.12	其他重载成员	181
3.4.5	multiset 类的其他操作符	126	4.5.13	string 类的其他操作	182
3.4.6	不合法的迭代器	126	4.5.14	string 类的设计	182
3.4.7	本节自测练习	128	4.5.15	string 类的实现	183
3.5	本章小结	128	4.5.16	string 类的演示程序	185
	本章自测练习参考答案	128	4.5.17	串联输出操作符	186
	编程项目	131	4.5.18	声明常量对象	187
第4章	指针与动态数组	138	4.5.19	由构造函数产生的类型 转换	187
4.1	指针与动态内存	138	4.5.20	在表达式中使用 已重载的操作符	187
4.1.1	指针变量	139	4.5.21	本章设计的 string 类与 C++ 库的 string 类	188
4.1.2	指针与赋值操作符一起 使用	140	4.5.22	本节自测练习	188
4.1.3	动态变量与 new 操作符	142	4.6	编程项目: polynomial 类	188
4.1.4	使用 new 操作符为 动态数组分配内存	143	4.7	本章小结	191
4.1.5	内存堆与 bad_alloc 异常	144		本章自测练习参考答案	192
4.1.6	delete 操作符	145		编程项目	194
4.1.7	本节自测练习	147	第5章	链表	196
4.2	把指针与数组作为参数	147	5.1	链表的基本节点类	196
4.2.1	以指针作为值参数	147	5.1.1	为节点声明类	196
4.2.2	数组参数	149	5.1.2	在链表节点中使用 typedef 语句	197
4.2.3	以指针或数组作为常量 参数	150	5.1.3	头指针和尾指针	197
4.2.4	以指针作为引用参数	152	5.1.4	空指针 NULL	198
4.2.5	本节自测练习	153	5.1.5	头指针或尾指针为 NULL 的含义	199
4.3	具有动态数组的 bag 类	156	5.1.6	节点类构造函数	199
4.3.1	指针成员变量	156	5.1.7	节点类成员函数	200
4.3.2	成员函数按需分配内存	157	5.1.8	成员选择操作符	201
4.3.3	值语义	161	5.1.9	本节自测练习	204
4.3.4	析构函数	163	5.2	链表工具包	205
4.3.5	修订后的 bag 类定义	164	5.2.1	链表工具包的头文件	205
4.3.6	修订后的 bag 类实现	165	5.2.2	计算链表的长度	206
4.3.7	修订后的 bag 类集成	170	5.2.3	链表的参数	209
4.3.8	本节自测练习	172	5.2.4	在链表头插入新节点	210
4.4	有关动态类的说明	172	5.2.5	在非链表头的其他位置 插入新节点	212
4.4.1	4 条规则	173	5.2.6	在链表中查找节点	215
4.4.2	复制构造函数的特殊 重要性	173	5.2.7	根据节点的位置在链表中 寻找节点	217
4.4.3	本节自测练习	174	5.2.8	链表复制	218
4.5	STL 的 string 类与编程项目	174	5.2.9	在链表头删除节点	221
4.5.1	以 null 结尾的字符串	174	5.2.10	在非链表头删除节点	222
4.5.2	初始化字符串变量	175	5.2.11	清空链表	223
4.5.3	空字符串	175	5.2.12	链表工具包的集成	224
4.5.4	读写字符串变量	175	5.2.13	使用链表工具包	228
4.5.5	strcpy 函数	176	5.2.14	本节自测练习	228
4.5.6	strcat 函数	177	5.3	用链表实现 bag 类	229
4.5.7	strlen 函数	177	5.3.1	第 3 个 bag 类的规范 说明	229
4.5.8	strcmp 函数	178	5.3.2	第 3 个 bag 类的类定义	230
4.5.9	string 类的规范说明	178			
4.5.10	string 类的构造函数	180			
4.5.11	重载 operator[]	181			

5.3.3	如何使 bag 类的 value_type 与节点类的 value_type 相匹配.....	233	6.4.1	返回引用类型的函数.....	287
5.3.4	在类中使用动态内存应遵循的规则.....	234	6.4.2	将引用返回值复制到别处时会发生什么.....	288
5.3.5	第 3 个 bag 类的实现.....	234	6.4.3	这里成员函数 data 需要两个版本.....	288
5.3.6	第 3 个 bag 类的集成.....	241	6.4.4	新节点类的头文件和实现文件.....	289
5.3.7	本节自测练习.....	244	6.4.5	本节自测练习.....	297
5.4	编程项目: 用链表实现 sequence 类.....	244	6.5	链表的迭代器.....	297
5.4.1	关于修订后的 sequence 类的设计建议.....	245	6.5.1	节点迭代器.....	297
5.4.2	关于修订后的 sequence 类的值语义.....	246	6.5.2	从 std::iterator 派生而来的节点迭代器.....	299
5.4.3	本节自测练习.....	246	6.5.3	节点迭代器的私有成员变量.....	300
5.5	动态数组、链表与双向链表.....	246	6.5.4	节点迭代器的构造函数.....	300
5.5.1	做出抉择.....	248	6.5.5	节点迭代器的*操作符.....	300
5.5.2	本节自测练习.....	249	6.5.6	节点迭代器两个版本的 ++ 操作符.....	300
5.6	标准模板库的 vector、list 和 deque 类.....	249	6.5.7	节点迭代器的相等和不相等比较.....	302
	本节测试练习.....	252	6.5.8	常量集合的迭代器.....	302
5.7	本章小结.....	252	6.5.9	本节自测练习.....	304
	本章自测练习参考答案.....	252	6.6	含迭代器的链表版 bag 模板类.....	305
	编程项目.....	257	6.6.1	如何为容器类提供迭代器.....	305
第 6 章	用模板、迭代器和 STL 进行软件开发.....	261	6.6.2	bag 类迭代器.....	306
6.1	模板函数.....	261	6.6.3	将迭代器定义在 bag 类中的原因.....	307
6.1.1	模板函数的语法.....	263	6.6.4	本节自测练习.....	314
6.1.2	使用模板函数.....	263	6.7	本章小结与 5 个 bag 类的小结.....	314
6.1.3	交换两个值的模板函数.....	265		本章自测练习答案.....	315
6.1.4	模板函数的参数匹配.....	266		编程项目.....	318
6.1.5	在数组中查找最大项的模板函数.....	267	第 7 章	栈.....	320
6.1.6	在已排序数组中插入一个数据项的模板函数.....	268	7.1	STL 的 stack 类.....	320
6.1.7	本节自测练习.....	270	7.1.1	标准库的 stack 类.....	321
6.2	模板类.....	270	7.1.2	编程示例: 翻转单词.....	322
6.2.1	模板类的语法.....	270	7.1.3	本节自测练习.....	323
6.2.2	进一步了解模板类实现文件.....	277	7.2	栈的应用.....	323
6.2.3	模板类成员函数的参数匹配.....	278	7.2.1	编程示例: 括号的匹配.....	323
6.2.4	使用模板类.....	278	7.2.2	编程示例: 算术表达式求值.....	325
6.2.5	详细讨论上一个演示程序.....	281	7.2.3	算术表达式求值的规范说明.....	326
6.2.6	本节自测练习.....	282	7.2.4	算术表达式求值的设计.....	326
6.3	STL 算法与迭代器的使用.....	282	7.2.5	算术表达式求值的实现.....	331
6.3.1	STL 算法.....	282	7.2.6	计算器程序使用的函数.....	332
6.3.2	标准迭代器的类型.....	283	7.2.7	算术表达式求值的测试与分析.....	332
6.3.3	数组的迭代器.....	285	7.2.8	算术表达式求值的改进.....	333
6.3.4	本节自测习题.....	285	7.2.9	本节自测练习.....	333
6.4	节点模板类.....	286	7.3	stack 类的实现.....	334
			7.3.1	栈的数组实现.....	334

7.3.2	栈的链表实现.....	337	9.1.2	跟踪递归调用.....	396
7.3.3	Koenig 查找.....	341	9.1.3	编程示例: write_vertical 的一个扩展.....	397
7.3.4	本节自测练习.....	341	9.1.4	深入分析递归.....	398
7.4	更复杂的栈应用.....	342	9.1.5	成功递归函数的一般 形式.....	401
7.4.1	后缀表达式求值.....	342	9.1.6	本节自测练习.....	402
7.4.2	将中缀表示法转换成后 缀表示法.....	344	9.2	递归的研究: 分形和迷宫.....	402
7.4.3	在中缀表达式中使用 优先级规则.....	346	9.2.1	编程示例: 产生随机 分形.....	403
7.4.4	中缀转换为后缀的 正确性.....	348	9.2.2	产生随机分形的函数及其 规范说明.....	404
7.4.5	本节自测练习.....	349	9.2.3	分形函数的设计和实现.....	405
7.5	本章小结.....	349	9.2.4	如何显示随机分形.....	407
	本章自测练习答案.....	350	9.2.5	编程示例: 穿越迷宫.....	407
	编程项目.....	351	9.2.6	穿越迷宫函数的规范 说明.....	408
第8章	队列.....	356	9.2.7	穿越迷宫函数的设计.....	409
8.1	STL 队列.....	356	9.2.8	穿越迷宫函数的实现.....	410
8.1.1	标准库的队列类.....	357	9.2.9	运用回溯穷举搜索的 递归模式.....	412
8.1.2	队列的使用.....	358	9.2.10	编程示例: 玩具熊游戏.....	413
8.1.3	本节自测练习.....	359	9.2.11	本节自测练习.....	414
8.2	队列的应用.....	359	9.3	推导递归.....	415
8.2.1	编程示例: 识别回文.....	359	9.3.1	如何确保没有无限递归.....	417
8.2.2	本节自测练习.....	361	9.3.2	归纳推导递归函数的 正确性.....	419
8.2.3	编程示例: 洗车模拟 程序.....	362	9.3.3	本节自测练习.....	419
8.2.4	洗车模拟程序的 规范说明.....	362	9.4	本章小结.....	420
8.2.5	洗车模拟程序的设计.....	362		本章自测练习答案.....	420
8.2.6	实现洗车类.....	365		编程项目.....	422
8.2.7	实现模拟函数.....	371	第10章	树.....	427
8.2.8	本节自测练习.....	372	10.1	树的简介.....	427
8.3	队列类的实现.....	373	10.1.1	二叉树.....	427
8.3.1	队列的数组实现.....	373	10.1.2	二叉分类树.....	430
8.3.2	有关队列中环形数组实现 的讨论.....	377	10.1.3	一般树.....	430
8.3.3	队列的链表实现.....	379	10.1.4	本节自测练习.....	431
8.3.4	实现细节.....	380	10.2	树的表示法.....	431
8.3.5	本节自测练习.....	384	10.2.1	完全二叉树的数组 表示法.....	432
8.4	实现 STL 的双端队列.....	384	10.2.2	使用节点类表示 二叉树.....	433
8.4.1	为双端队列的 value_type 项调用析构函数和构造 函数.....	387	10.2.3	本节自测练习.....	434
8.4.2	栈和队列的其他变体.....	388	10.3	二叉树节点类.....	435
8.4.3	本节自测练习.....	388	10.3.1	编程示例: 猜测动物 程序.....	439
8.5	栈、队列和优先队列类的引用 返回值.....	388	10.3.2	猜测动物程序的 设计与实现.....	440
8.6	本章小结.....	389	10.3.3	猜测动物程序的 改进.....	448
	本章自测练习答案.....	389	10.3.4	本节自测练习.....	448
	编程项目.....	390	10.4	树的遍历.....	449
第9章	递归思想.....	394			
9.1	递归函数.....	394			
9.1.1	递归思想的第一个例子.....	394			

10.4.1	二叉树的遍历	449	11.3.11	从 B 树中删除项	505
10.4.2	从树的节点中输出 数据	453	11.3.12	B 树的松散删除 操作	506
10.4.3	遍历中的问题	454	11.3.13	解决子节点项短缺 问题的私有成员 函数	508
10.4.4	函数作为参数	454	11.3.14	从 B 树中删除最大 的项	510
10.4.5	apply 函数的一个 模板版本	456	11.3.15	外部 B 树	512
10.4.6	使 apply 模板函数 更具有通用性	457	11.3.16	本节自测练习	512
10.4.7	树遍历的模板函数	458	11.4	树、日志和时间分析	513
10.4.8	本节自测练习	465	11.4.1	二叉查找树的时间 分析	513
10.5	二叉查找树	466	11.4.2	堆的时间分析	514
10.5.1	二叉查找树存储机制	466	11.4.3	对数运算	515
10.5.2	第 6 个 bag 类的定义	470	11.4.4	对数级算法	516
10.5.3	第 6 个 bag 类的某些 简单函数的实现	470	11.4.5	本节自测练习	516
10.5.4	计算某个元素在二叉 查找树中出现的次数	471	11.5	STL 的 map 类和 multimap 类	517
10.5.5	添加一个新元素到二叉 查找树中	472	11.6	本章小结	518
10.5.6	从二叉查找树中删除 某个元素	473		本章自测练习答案	518
10.5.7	二叉查找树的组合 操作符	475		编程项目	521
10.5.8	时间分析和迭代器	477	第 12 章	查找	523
10.5.9	本节自测练习	478	12.1	顺序查找和二叉查找	523
10.6	本章小结	478	12.1.1	顺序查找	523
	本章自测练习答案	479	12.1.2	顺序查找的分析	523
	编程项目	482	12.1.3	二叉查找	525
第 11 章	平衡树	487	12.1.4	二叉查找的设计	526
11.1	堆	487	12.1.5	二叉查找的分析	528
11.1.1	堆的存储规则	487	12.1.6	标准库的查找函数	531
11.1.2	使用堆结构实现的 优先队列	488	12.1.7	用于有序区间的函数	531
11.1.3	插入新项	488	12.1.8	用于无序区间的函数	532
11.1.4	删除项	489	12.1.9	STL 的 search 函数	533
11.2	STL 优先队列与堆算法	491	12.1.10	本节自测练习	534
	本节自测练习	492	12.2	开地址散列	534
11.3	B 树	492	12.2.1	散列简介	534
11.3.1	非平衡树的问题	493	12.2.2	表类的声明	536
11.3.2	B 树的规则	493	12.2.3	表类的设计	538
11.3.3	B 树的一个示例	495	12.2.4	表 ADT 的实现	541
11.3.4	用 B 树实现 set 类	495	12.2.5	选择散列函数来 减少冲突	547
11.3.5	在 B 树中查找项	499	12.2.6	再散列减少聚类	547
11.3.6	在 B 树中插入项	501	12.2.7	本节自测练习	548
11.3.7	B 树的松散插入操作	501	12.3	链式散列	549
11.3.8	修正子节点多余项 的私有成员函数	503		本节自测练习	551
11.3.9	回到成员函数 Insert	504	12.4	散列的时间分析	551
11.3.10	采用自顶向下方法 设计	505	12.4.1	散列表的装填因子	551
			12.4.2	本节自测练习	553
			12.5	程序设计: 使用 STL 向量的 表类	553
			12.5.1	新表类	553
			12.5.2	在新表类中使用向量	554

12.5.3	常量模板参数	554	14.1.1	如何声明派生类	612
12.5.4	函数模板参数	554	14.1.2	派生类的自动构造 函数	613
12.5.5	实现新表类	555	14.1.3	使用派生类	614
12.5.6	本节自测练习	556	14.1.4	派生类的自动赋值 操作符	615
12.6	TR1 库扩展中的散列表	556	14.1.5	派生类的自动析构 函数	616
12.7	本章小结	557	14.1.6	覆盖继承的成员函数	616
	本章自测练习答案	557	14.1.7	本节自测练习	617
	编程项目	561	14.2	仿真生态系统	618
第 13 章	排序	563	14.2.1	实现部分生物对象层次	618
13.1	二次排序算法	563	14.2.2	organism 类	618
13.1.1	选择排序的规范说明	563	14.2.3	animal 类: 具有新私有 成员变量的派生类	621
13.1.2	选择排序的设计	563	14.2.4	如何为派生类提供 新构造函数	622
13.1.3	选择排序的实现	565	14.2.5	animal 类的其他 成员函数	623
13.1.4	选择排序的效率分析	567	14.2.6	本节自测练习	627
13.1.5	插入排序	569	14.2.7	herbivore 类	628
13.1.6	插入排序的效率分析	571	14.2.8	池塘生态仿真程序	630
13.1.7	本节自测练习	573	14.2.9	池塘生态系统的实现 细节	634
13.2	递归排序算法	574	14.2.10	使用池塘模型	634
13.2.1	使用递归的分治法	574	14.2.11	动态内存的使用	635
13.2.2	归并排序	576	14.2.12	本节自测练习	636
13.2.3	归并函数	577	14.3	虚拟成员函数和 game 类	636
13.2.4	动态内存在归并排序中 的应用	581	14.3.1	game 类简介	636
13.2.5	归并排序的效率分析	582	14.3.2	受保护的成员	640
13.2.6	文件的归并排序	583	14.3.3	虚拟成员函数	640
13.2.7	快速排序	584	14.3.4	虚拟析构函数	641
13.2.8	partition 函数	585	14.3.5	game 类的受保护 虚拟成员函数	641
13.2.9	快速排序的效率分析	588	14.3.6	实现 Connect Four 游戏的派生类	642
13.2.10	为快速排序选择一个 好的基准元素	589	14.3.7	connect4 类的私有 成员变量	642
13.2.11	本节自测练习	589	14.3.8	connect4 类的构造 函数和 restart 函数	644
13.3	使用堆的 $O(n \log n)$ 算法	590	14.3.9	处理游戏状态的 三个函数	644
13.3.1	堆排序	590	14.3.10	处理移动的三个函数	645
13.3.2	构建堆	595	14.3.11	clone 函数	646
13.3.3	向下重排	596	14.3.12	编写派生自 game 类的游戏	646
13.3.4	堆排序的效率分析	597	14.3.13	game 类的 play 算法	647
13.3.5	本节自测练习	598	14.3.14	本节自测练习	650
13.4	STL 的排序与二叉查找	598	14.4	本章小结	650
13.4.1	原始 C 标准库函数 qsort	598		进阶阅读	650
13.4.2	STL 的排序函数	599		本章自测练习答案	651
13.4.3	STL 的堆排序	600		编程项目	655
13.4.4	STL 的二叉查找函数	600			
13.4.5	用于 STL 排序函数 的比较参数	601			
13.4.6	使用迭代器编写一个 自己的排序函数	601			
13.5	本章小结	602			
	本章自测练习答案	603			
	编程项目	605			
第 14 章	派生类与继承	610			
14.1	派生类	610			