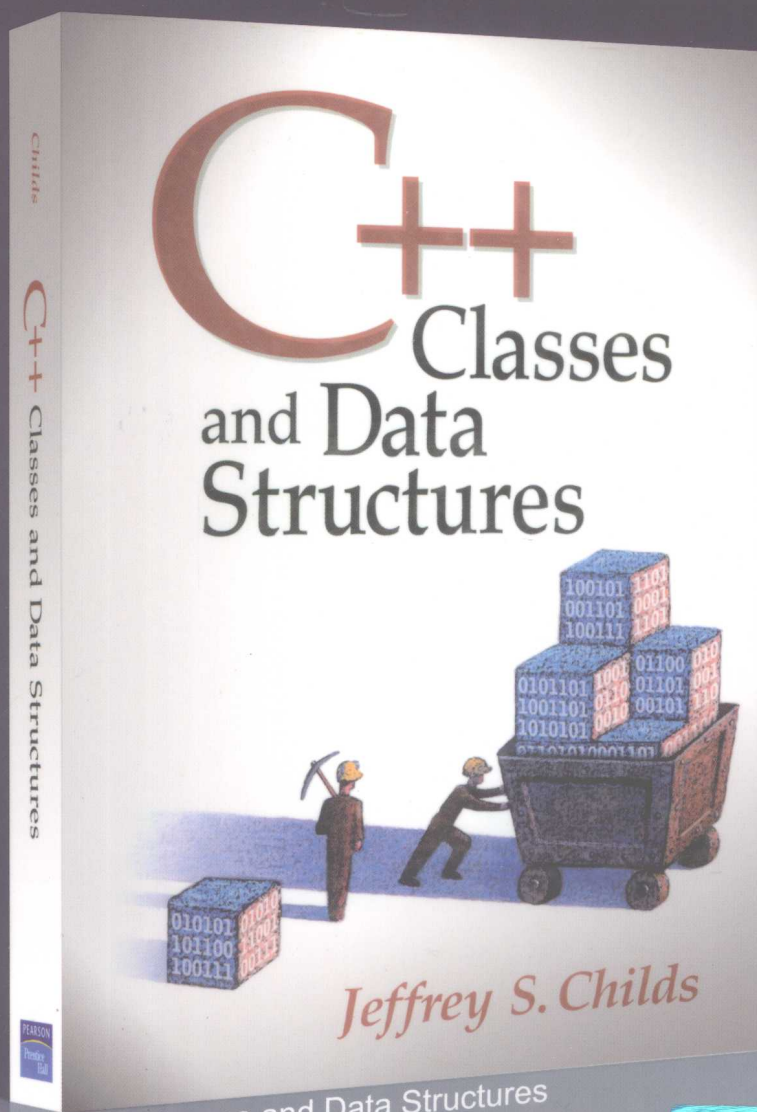
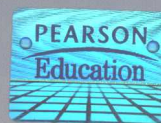


C++ 类和数据结构

(美) Jeffrey S. Childs 著 张杰良 译



C++ Classes and Data Structures



清华大学出版社

国外计算机科学经典教材

C++类和数据结构

(美) Jeffrey S. Childs 著
张杰良 译

清华大学出版社

北 京

Authorized translation from the English language edition, entitled C++ Classes and Data Structures, 978-0-13-158051-0 by Jeffrey S. Childs, published by Pearson Education, Inc, publishing as Prentice Hall PTR, Copyright © 2008.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION ASIA LTD., and TSINGHUA UNIVERSITY PRESS Copyright © 2008.

北京市版权局著作权合同登记号 图字: 01-2008-2454

本书封面贴有 Pearson Education(培生教育出版集团)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++类和数据结构/(美)查尔兹(Childs, J.S.) 著; 张杰良 译. —北京: 清华大学出版社, 2009.1
(国外计算机科学经典教材)

书名原文: C++ Classes and Data Structures

ISBN 978-7-302-19179-7

I. C… II. ①查…②张… III. ①C 语言—程序设计②数据结构 IV.TP312 TP311.12

中国版本图书馆 CIP 数据核字(2008)第 207516 号

责任编辑: 王 军 于 平

装帧设计: 孔祥丰

责任校对: 成凤进

责任印制: 杨 艳

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: 三河市兴旺装订有限公司

经 销: 全国新华书店

开 本: 185×260 印 张: 26.5 字 数: 550 千字

版 次: 2009 年 1 月第 1 版 印 次: 2009 年 1 月第 1 次印刷

印 数: 1~4000

定 价: 49.99 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: 027688-01

出版说明

近年来，我国的高等教育特别是计算机学科教育，进行了一系列大的调整 and 改革，亟需一批门类齐全、具有国际先进水平的计算机经典教材，以适应我国当前计算机科学的教學需要。通过使用国外优秀的计算机科学经典教材，可以了解并吸收国际先进的教学思想和教学方法，使我国的计算机科学教育能够跟上国际计算机教育发展的步伐，从而培养出更多具有国际水准的计算机专业人才，增强我国计算机产业的核心竞争力。为此，我们从国外多家知名的出版机构 Pearson、McGraw-Hill、John Wiley & Sons、Springer、Cengage Learning 等精选、引进了这套“国外计算机科学经典教材”。

作为世界级的图书出版机构，Pearson、McGraw-Hill、John Wiley & Sons、Springer、Cengage Learning 通过与世界级的计算机教育大师携手，每年都为全球的计算机高等教育奉献大量的优秀教材。清华大学出版社和这些世界知名的出版机构长期保持着紧密友好的合作关系，这次引进的“国外计算机科学经典教材”便全是出自上述这些出版机构。同时，为了组织该套教材的出版，我们在国内聘请了一批知名的专家和教授，成立了专门的教材编审委员会。

教材编审委员会的运作从教材的选题阶段即开始启动，各位委员根据国内外高等院校计算机科学及相关专业的现有课程体系，并结合各个专业的培养方向，从上述这些出版机构出版的计算机系列教材中精心挑选针对性强的题材，以保证该套教材的优秀性和领先性，避免出现“低质重复引进”或“高质消化不良”的现象。

为了保证出版质量，我们为这套教材配备了一批经验丰富的编辑、排版、校对人员，制定了更加严格的出版流程。本套教材的译者，全部由对应专业的高校教师或拥有相关经验的 IT 专家担任。每本教材的责编在翻译伊始，就定期不间断地与该书的译者进行交流与反馈。为了尽可能地保留与发扬教材原著的精华，在经过翻译、排版和传统的三审三校之后，我们还请编审委员或相关的专家教授对文稿进行审读，以最大程度地弥补和修正在前面一系列加工过程中对教材造成的误差和瑕疵。

由于时间紧迫和受全体制作人员自身能力所限，该套教材在出版过程中很可能还存在一些遗憾，欢迎广大师生来电来信批评指正。同时，也欢迎读者朋友积极向我们推荐各类优秀的国外计算机教材，共同为我国高等院校计算机教育事业贡献力量。

清华大学出版社

国外计算机科学经典教材

编审委员会

主任委员：

孙家广 清华大学教授

副主任委员：

周立柱 清华大学教授

委员（按姓氏笔画排序）：

王成山	天津大学教授
王 珊	中国人民大学教授
冯少荣	厦门大学教授
冯全源	西南交通大学教授
刘乐善	华中科技大学教授
刘腾红	中南财经政法大学教授
吉根林	南京师范大学教授
孙吉贵	吉林大学教授
阮秋琦	北京交通大学教授
何 晨	上海交通大学教授
吴百锋	复旦大学教授
李 彤	云南大学教授
沈钧毅	西安交通大学教授
邵志清	华东理工大学教授
陈 纯	浙江大学教授
陈 钟	北京大学教授
陈道蓄	南京大学教授
周伯生	北京航空航天大学教授
孟祥旭	山东大学教授
姚淑珍	北京航空航天大学教授
徐佩霞	中国科学技术大学教授
徐晓飞	哈尔滨工业大学教授
秦小麟	南京航空航天大学教授
钱培德	苏州大学教授
曹元大	北京理工大学教授
龚声蓉	苏州大学教授
谢希仁	中国人民解放军理工大学教授

译者序

数据结构是计算机专业的一门重要的基础课，在计算机科学各领域尤其是在软件设计和开发中发挥着举足轻重的作用。几乎所有的计算机软件系统，例如，操作系统、编辑工具和编译器等都要使用不同的数据结构。因此，数据结构是计算机专业的核心课程，是许多其他后续课程的重要基础。

目前，国内外有很多介绍数据结构方面的书籍，这些书籍都各具特色。但是大多数书籍都只注重于技术细节，缺乏详细的解释说明，没有数据结构基础知识的读者难以掌握理解。

考虑到上述事实，本书作者从自己教学的实际需求出发，根据自己对数据结构的理解，结合自己对数据结构的研究，考虑学生的学习需求，编写了本书。本书由易到难，不仅详细介绍了各种常见的数据结构，还提供了学习数据结构的基础知识。读者可以先阅读基础知识，再深入学习数据结构，通过这种安排方式，方便读者的学习，加强概念的理解。

本书采用当前流行的面向对象的 C++ 语言来描述数据结构和算法，因为 C++ 语言是程序员使用最广泛的语言。但是，本书还考虑到目前编程语言的多样性，在详细阐述数据结构概念时尽量避免使用与语言相关的术语解释，强调对概念的透彻理解，注重能力的培养，使读者能够使用其他编程语言编写出自己所需的数据结构。

通过本书的学习，读者不仅可以了解数据结构的基本概念和算法，还可以了解数据结构的应用场合；不仅可以使使用数据结构，还可以根据需求设计自己的数据结构；不仅可以选择高效的算法，还可以了解这样做的原因。确切地说，本书内容全面丰富，语言精练简洁，示例和练习的实践性、针对性强，是一本优秀的数据结构教材。

译者

2008 年 3 月

前言

编写本书的原因非常简单。当我在讲授数据结构课程时，通常无法在一本书中找到我喜欢讲授的所有主题。而且各个主题通常也不是按我喜欢的顺序排列，数据结构也不总是按我喜欢的方式建立。因此，我最终决定自己编写一本这方面的书，就是本书了。我相信这是一本优秀的书。

或许前言太简单了吧。如果我写的前言只有一小段，出版商会对我怒目而视的，因此我再多写一些内容，也好让大家确信这是一本优秀的图书。首先谈一下我本人讲授数据结构的方法。

方法

C++语言是一门非常适合于数据结构的语言，但是，作为教师，如何清晰地讲授那些C++中难以掌握的概念呢？实际上，这个困难是C++语言少有的缺点之一。这里我的方法就是首先激发学生学习概念。任何有过一年任教经验的人都知道，如果看上去所讲授的概念没有所指的话，那么学习它们的动机就会无力。本书自始至终都提供促人奋发学习的动机，而且我鼓励大家在自己的课堂中使用它们。

本书自始至终对那些许多人觉得难以讲授的概念提供了清晰的解释，我还知道您会对此感到满意；它们以一种适合于学生的风格书写。本书中介绍基本概念的方式，没有过分考虑它们在技术上是否100%恰当；它们大部分在技术上是恰当的，但是您知道我所谈论的恰当性指的是什么——它们虽然能够正确地表示所讲授的概念，但是却将那些适合于学生理解的表示方法转变成晦涩难懂的代表。学生已经不能理解概念了！本书的读者是学生，不是那些计算机科学方面的专家教授。本书还附有幻灯片，可以使这些概念更加清晰；我称这些幻灯片为“动作幻灯片(action slide)”。幻灯片都像旧电影书的页面一样，快速翻阅它们可以看到事物在运动，例如马在奔跑。只是这里翻阅幻灯片看到的是那些当前讨论的代码和数据结构。

除了更清晰讲授C++的方法外，我还有另一种方法，就是将关注点放在客户上——那些使用类或者数据结构的程序员。在他们第一门或者第二门编程方面的课程中，学生会满怀希望地问自己诸如下述类型的问题：“我怎么做才能使程序对用户更友好呢？”或者“我怎样做才能使程序的输出更整洁？”在这本书中，我希望学生问这样的问题：“我怎样才能使这个类(或者数据结构)让客户更方便使用呢？”或

者“我怎样做才能使这个类(或者数据结构)对客户尽可能灵活呢?”从计算机用户到客户程序员,关注点发生了一个转变。因此,在本书中,当考虑实现问题时都会考虑客户。

本书中所有的程序、类和数据结构都是在考虑客户的情况下从头开始编写的。代码使用 Microsoft Visual Studio 2005 C++编译和测试。在本书中,代码都编有行号,用来在正文中引用。建议学生这样利用代码编号,即首先分析代码,查看自己是否能够理解代码。如果有一两行代码不能完全掌握,就可以在正文中查找代码编号,阅读这些代码行的讨论。但是,整个讨论还是应该阅读的,以查看自己的理解正确与否。

在本书中,提供了一些帮助学生为应用程序选择合适数据结构,以及为数据结构选择合适实现的信息。当然,速度问题是必须考虑的,内存问题也是如此。实现开销使用的内存数量必须仔细权衡。确实今天的计算机拥有很多内存,但是纵观历史,当计算机的平均内存容量增加时,程序的平均规模也相应增加。内存问题非常重要,未来也一直重要。

本书从各方面努力,尽可能使各个练习有意义。下面举一个没有意义的练习例子,例如“将栈中的元素翻转,从而栈顶的元素位于栈底。”这会得到什么结果呢?它的目的何在呢?我试图确保各个练习都有一定的目的,从而不会使学生感到厌烦,认为数据结构无用。

本书自始至终,我都一直强调不管客户是否误用了函数,在编写类函数时,应该防止函数的溃崩。这是一个极高的目标,但这也是一个必须为之奋斗的目标,不得不面对它,因为客户调用函数时通常只需一行代码,这一行代码相当简单,通常不会发生错误,但是当程序溃崩时,客户在调试过程中就不知道问题出在哪里。

我的想法就是,通过示范如何设计一个类或者数据结构,从而使学生在设计自己的专用数据结构时知道考虑些什么问题。如果不希望学生设计自己的数据结构,那么就没有必要向他们示范数据结构的代码,只需使用它们即可。出于这方面的原因,在本书中,随数据结构一起介绍的问题包括客户、内存和速度。这些问题都是设计数据结构时需要考虑的重要因素。

我尽量缩减本书的篇幅,不让它成为一本难啃的大部头,我个人认为这是一件使学生和教师都满意的事情;教师可以讲授本书的所有内容,学生也能够消化吸收本书中的内容。本书和其他数据结构方面的书不同,其他书中有许多不同的主题,以满足各类人的需求,从而形成一本很厚的书。但是,实际上,从某种意义上讲,它没有满足任何人。

尽管本书使用 C++语言描述,但是我意识到在学生进入工作环境中以后,他们可能需要使用其他语言编写数据结构。因此,一个无法脱离 C++语言而单独思考数据结构的学生可能需要度过一段艰难的时间。因此,在讨论数据结构时,直至数据结构的实际实现,我都试图避免使用 C++术语来讨论数据结构。

我没有采用这样的方法，即从一个简单的(但是可能不切合实际)数据结构(例如栈)实现开始，然后再以它为基础，逐步添加更多的功能。我发现许多学生对这种方法已经不感兴趣；学生们努力去理解数据结构，在理解了之后他们意识到这个设计没有那么出色，需要改进，所以接下来给他们提供一个更出色的设计，学生又需要从头开始理解。从一个简单的例子入手，并不能帮助学生更好地理解所提出的思想；相反，他们可能会留下这样的印象，即他们在特定时刻学到的东西可能并不出色。尽管从某种意义上讲，我实际上采取了这种方法，但是我采用了不同的方式。例如，我向他们讲授为了理解一个复杂的栈需要知道的所有内容，只是在第 1 章中我使用的是一个普通的、常见的类。

如何使用本书

本书应该作为数据结构的第一门课程，根据课程安排，可以将本书用于第二学期或者第三学期的课程。第 1~6 章主要注重于对类的理解，从而为数据结构的理解打下一个坚实的基础，这一部分内容可能要比其他数据结构书本中更多。如果将本书用作第二学期的课程，那么学生应该对 C++ 或者 Java 已有所了解。在这种情况下，您需要通读第 1~10 章的内容，然后选读第 11~15 章中的主题或者基本知识。如果将本书用作第三学期的课程，那么您至少有了一定的基础，您可以浏览第 1~5 章的内容(或许是为了复习一下相关知识)，然后主要关注于第 6~15 章的内容。

如果将本书作为第二学期的课程，我想您会对下述事实心存感激，即在讲授数据结构之前我先讲授了类。也就是说，我不是以几页内容来讲授类，然后就转到数据结构上。如果同时将两个主题拼在一块，学生肯定会迷惑的，他们要努力同时掌握类的概念和数据结构的概念。同时，在他们的内心深处，他们可能还会奇怪栈为什么不使用类来生成呢；学生很容易拥有这种思想。然后，学生开始相信类只用于数据结构，这也不是一个好思想。因此，在介绍数据结构之前，本书花 5 章的内容来介绍类。在这 5 章中详细展现了类的优点，因此当使用类生成数据结构时，就不会再疑问为什么使用类生成数据结构。而且不管以何种方式讲授数据结构，这是需要讲授的主要内容，因此我们没有在区分这两个主题上浪费太多的时间。

如果将本书作为第三学期的课程，大家可能希望认真地阅读一下第 1~5 章的内容，以弥补那些在第二学期课程中还没有太深入的重要内容，因为后面章节的内容以前面章节的内容为基础。大家希望将注意力放在基于客户的设计决策上，因为，正如前面所述，本书自始至终一直以客户为中心。但是在如今，大多数书并不是这样做的。通常情况下，这些知识需要雇主来讲授。您也可以快速地浏览一遍前 5 章的内容。

或许您需要在自己的课程中讲授组合、继承和多态概念。有一种方法，就是在

面向对象编程的课程中先讲授这方面的内容。第6章包含继承和多态，但是这些内容是可选的。尽管不是全书使用这些主题，但是整本书自始至终都有练习要求使用继承。这些练习用一个星号标示。不管在数据结构中是否使用继承或多态，这种安排应该可以满足您的需求。

内容和组织方式

首先在第1章中以一种简单的方式介绍类，像介绍结构一样，但是在类中添加了函数。这里没有使用 `const` 限定符和构造函数，这就可以使学生能够深刻理解什么是类。然后，从第2章到第6章，以前面的概念为基础，逐步深入地介绍其他概念。

第2章尽可能早地介绍了类模板的概念(甚至比介绍构造函数还早)。这种表示顺序可能会使一些人感到奇怪，但是如果准备讲授使用类模板的数据结构，可以采用下述方式。

(1) 首先提供不好的数据结构设计(没有类模板)，然后提供较复杂的设计(使用类模板)，这是我在前面讲过要尽量避免使用的一种方式。

(2) 首先讲授一点类，接着是类模板，然后是数据结构。

(3) 要讲授类就直接讲授类模板，以众多的资料详细讲授它们，然后再讲授数据结构。

事实证明，方式(2)并不怎么出色，因为在学生准备好学习类模板时，他们已经开始学习数据结构了。但是，本书中的这种安排方式意味着在第2章中要花费一些时间。

本书中介绍的所有数据结构都是用类模板生成的。较早地介绍类模板可以使学生在事情仍然简单时就调整思路。然后，当在数据结构中使用类模板时，就不会迷惑。第2章中还介绍了重载运算符，因为它们与类模板密切地结合成一个整体。我个人觉得重载运算符和类模板之间的关系经常被忽略。例如，如果在链表中搜索一个元素，可能有一部分代码如下所示：

```
template <class DataType>
bool LinkedList<DataType>::retrieve( const DataType & element )
{
    // beginning code here is omitted
    while( ptr != NULL ) {
        if ( element == ptr->info ) {
            element = ptr->info;
            return true;
        }
    }
    // other code here is omitted
}
```

如果希望这段代码尽可能通用,那么 `element` 就有可能是某些简单的类型,例如整型,或者一些较复杂的类型,例如一个结构的对象,该对象的关键字值是我们希望搜索的值。在后一种情况中,需要为结构重载 `==` 运算符,从而可以比较关键字的值。然后再检索其他信息。因此,重载运算符和模板数据结构密切地结合成一个整体。

我讨论将对象(记录)作为元素存储在数据结构中的内容是不是太多了,情况确实如此。毕竟,数据结构的大多数实际应用都使用对象作为元素。学生应该习惯这一点。现在,要记在心里,本书前 6 章的内容只是有关类的讨论,因此,一旦我们习惯了将对象作为元素讨论,就会不再困惑,特别是由于在前 6 章中已经介绍了对象在类中的使用。

第 2 章中还介绍了 `Stack` 数据结构,但是,在该章中我们实际上并没有详细分析它的内容。我们只是在练习的程序中使用栈,这有助于讲授抽象的重要性。

第 3 章介绍 `const` 限定符和构造函数,还讨论了一些其他重要的内容:如何修改类。第 3 章中有两节是关于类的修改的(当然,也是以客户为中心)。

第 4 章致力于指针和动态数组的使用,假定学生没有这方面的背景知识。在第 4 章中,我们还讨论了如何调整动态数组的大小。当然,动态数组的大小实际上并没有改变,而是生成一个不同大小的新动态数组,然后将旧数组中的元素复制过去,并将旧数组释放,最后用作旧数组“名称”的指针被赋予新数组的地址。当我们进行这些操作时,实际上数组永远不会填满(除非耗尽了堆内存)。稍后,在数据结构的数组实现中使用了这一思想,以节省内存,并且保证数组不被填满。例如,在栈中,当向数组中入栈一个元素时,数组中已经没有空间了,因此调用一个函数,在入栈元素之前将数组的大小增加一倍。同理,在元素出栈之后,当栈数组只使用了 25% 空间时,就将数组的大小收缩一半。通过后面的章节可以证实,在最坏的条件下,采用这种扩展-收缩策略时,每次元素入栈或者出栈在数组之间复制的元素平均不会超过两个。

第 5 章介绍的是一个 `Array` 类,该类是其他与类相关的重要主题的基础,这些主题包括析构函数、重载赋值运算符和复制构造函数,以及当类的对象包含有动态分配的内存时为什么要使用它们的原因。通过使用这个简单的数组概念,这些重要的函数以及它们的工作方式就变得相当清晰。

第 6 章有一节介绍组合,这是必需的,因为后面的章节将使用这一思想。第 6 章还有两个选读节,包括继承和多态,那些喜欢的或者必须学习的人可以学习这两个主题。再强调一次,书中有一些练习标记有星号,这表明这些练习需要使用继承。

有了这些类的基础知识后,再介绍数据结构,学生就容易理解了。另外,还有可能提供更加漂亮的实现(不会产生混乱),例如使用可调数组,以及当乘数或除数是 2 的幂时,为提高速度使用的移位运算符。

第 7 章讲授生成数据结构的方法。就是在该章中,我们讨论数据结构中动态数

组的扩展和收缩策略。另外，该章还介绍链表。该章还讨论数组和链表在开销中浪费的内存量，从而使学生可以生成一个正式的决策，即在数据结构中使用数组实现还是链表实现。该章还示范了描述如何使用链表的代码段(最初没有使用类)。在这一讨论中，特别强调学生在编写这样的代码时，需要考虑地址，而不仅仅是箭头。实际上，我已尽最大的努力来示范在解决使用链表的问题时应该采用的思考过程。

第8章讨论栈和队列，分别讨论两者的数组和链表实现。固定长度的数组实现没有讨论，如果数组太小，数组就容易填满；如果数组太大，数组就会浪费大量的内存。因此，本章中基于数组的实现都使用动态数组，动态数组可以随着元素数量的变化调整大小。有时候，当介绍较复杂的数据结构时，会先介绍一个基于固定长度数组的实现。但是，正如前面所述，理解这些实现需要前面章节中(使用普通类)介绍的许多概念。

第9章整章讨论时间复杂度。听起来该章好像要深入讨论时间复杂度，但是我并没有这样做——本章讲授的内容只是确保能够深刻理解时间复杂度即可。本章分析影响数据结构选择的问题。本章采用了一种容易理解的方法来解释对数。并且还详细说明时间复杂度对计算速度的影响。

第10章分析作为数据结构的链表。在第11章中，在使用链式技术的散列表中，使用这个链表。HashTable类可以让客户编写最适合应用程序的散列函数。然后客户可以通过使用函数指针，将散列函数传递给HashTable构造函数，从而使HashTable函数可以调用客户编写的散列函数，而不用考虑客户在什么地方编写散列函数。第11章还讨论双向链表的实现，在双向链表中，任何元素只需花费 $\Theta(1)$ 时间就可以随意访问或者删除。这是通过使用散列表实现完成的。

第12章讨论优先级队列，并介绍树和堆。与堆函数速度提升相关的实现问题包括：在向上或者向下重堆化(reheaping)时使用“单赋值交换”和移位运算符。这一章中有两节可选内容是关于链堆的(为了保持操作的时间复杂度与基于数组的堆的操作一样，链堆其实嵌入到一个更复杂的数据结构中)。

递归在第13章中讨论。该章介绍一种理解递归的方式，即通过生成函数的副本来理解递归，通过这种方法可以极大地减少混淆。该章没有提供太多的递归示例，这是因为采用这种方式解释递归之后，就不需要太多的例子了。

第14章介绍一些重要的排序算法，以及一些最低限度的算法分析，从而可以帮助学生学习算法分析课程。这一章有一节专门讨论链表的排序。

二叉搜索树和图以一种开放式的方式在第15章中介绍。这里提醒学生，这些主题需要在一门更高级的课程中进一步地深入讨论。

相关资源

本书提供了一套完整的资源，可以从 <http://www.prenhall.com/childs> 上下载，这套资源包括下述内容：

- PowerPoint 幻灯片，包括动作幻灯片，演示代码如何影响数据结构。这些幻灯片可以作为学习资源提供给学生。
- 非编程练习的答案。
- 书中全部例子的完整程序代码和类代码。这些代码都是使用 Microsoft Visual Studio 2005 C++ 编译和测试的。这些代码可以从本书合作站点 www.tupwk.com.cn 下载。
- 一套测试题。

Jeffrey S.Childs

宾夕法尼亚州克莱瑞恩大学

目 录

第 1 章	结构和类	1
1.1	结构	1
1.2	类的基本概念	6
1.3	类的实现	9
1.4	类的测试	17
1.5	将函数定义放在类定义中	18
1.6	类的注释	20
1.7	结构和类之间的区别	21
1.8	小结	21
1.9	练习	22
第 2 章	重载运算符、类模板和抽象	25
2.1	重载运算符	25
2.2	在 Checkbook 类中使用 Check 结构	31
2.3	类模板	34
2.4	类和抽象	42
2.5	小结	45
2.6	练习	45
第 3 章	类的更多内容	51
3.1	const 限定符	51
3.2	构造函数	53
3.3	类的修改	57
3.4	修改 Checkbook 类保存支票历史记录	57
3.5	小结	67
3.6	练习	68
第 4 章	指针和动态数组	69
4.1	指针	69
4.2	[]运算符	75
4.3	动态分配内存	77

4.4	动态数组	79
4.5	delete 操作符	80
4.6	对象指针	83
4.7	堆内存耗尽	84
4.8	可调数组	86
4.9	小结	88
4.10	练习	89
第 5 章	Array 类	93
5.1	Array 类模板	93
5.2	使用 Array 类	103
5.3	析构函数	106
5.4	复制构造函数	107
5.5	重载赋值运算符函数	113
5.6	示例	118
5.7	Array 类的优缺点	121
5.8	标准模板库	122
5.9	小结	122
5.10	练习	123
第 6 章	面向对象编程简介	125
6.1	组合	125
6.2	继承	129
6.3	多态	137
6.4	小结	144
6.5	练习	144
第 7 章	生成数据结构的方法	147
7.1	在数据结构中使用数组	147
7.2	链式结构简介	151
7.3	链表编码	153
7.3.1	链表代码基础	153
7.3.2	在链表中搜索一个肯定存在的值	154
7.3.3	在链表中搜索可能不存在的值	157
7.3.4	在链表的表头插入一个结点	159
7.3.5	在链表中间插入一个结点	161
7.3.6	从链表中删除一个包含链表中某个值的结点	164

7.3.7	使用 header 结点简化代码	166
7.3.8	删除找到包含某值的结点	166
7.4	数组和链表的对比	167
7.4.1	数组和链表在速度上的比较	167
7.4.2	数组和链表在内存浪费上的比较	168
7.4.3	浪费内存分析	171
7.5	小结	174
7.6	练习	175
第 8 章	栈和队列	177
8.1	栈 ADT	177
8.2	栈的数组实现	178
8.3	栈的链表实现	184
8.4	队列 ADT	184
8.5	队列的链表实现	184
8.6	队列的其他链表实现	193
8.7	队列的数组实现	194
8.8	小结	202
8.9	练习	202
第 9 章	时间复杂度简介	211
9.1	时间复杂度基础	213
9.2	常量阶时间复杂度	222
9.3	大 O 表示法	223
9.4	对数阶时间复杂度	224
9.5	折半搜索算法	227
9.6	计算机速度：它来源于什么地方	231
9.7	数据结构函数的时间复杂度	232
9.8	数组扩展和收缩的平摊分析	232
9.9	小结	236
9.10	练习	237
第 10 章	链表作为数据结构	239
10.1	列表 ADT	239
10.2	在信息记录中使用关键码值	240
10.3	链表实现	240
10.3.1	链表说明文件	241

10.3.2 链表实现文件	244
10.4 其他实现	255
10.5 小结	259
10.6 练习	260
第 11 章 散列表	263
11.1 散列表 ADT	263
11.2 散列函数和散列表设计	263
11.3 散列表的实现问题	269
11.4 函数指针	271
11.5 散列表实现	272
11.6 使用散列表实现	277
11.7 双向链表的散列表实现	279
11.7.1 实现问题	283
11.7.2 DoublyLinkedList 类的说明文件	284
11.7.3 DoublyLinkedList 类的实现文件	287
11.8 小结	297
11.9 练习	297
第 12 章 优先级队列、树和堆	299
12.1 优先级队列 ADT	299
12.2 优先级队列设计	299
12.3 树	301
12.4 堆	306
12.5 使用单赋值交换	314
12.6 优先级队列的堆实现(基于数组)	316
12.7 链(内嵌)堆设计	324
12.8 优先级队列的链(内嵌)堆实现	329
12.9 小结	336
12.10 练习	337
第 13 章 递归	339
13.1 递归阶乘函数	339
13.2 递归函数编写原则	344
13.3 在链式结构上使用递归	345
13.4 递归函数的时间复杂度	348
13.5 小结	348