国外著名高等院校
信息科学与技术优秀教材

数据结构的C++伪码实现

DATA STRUCTURES
A Pseudocode Approach with C++

Richard F.Gilberg
Behrouz A.Forouzan

英文版

人民邮电出版社
www.pptph.com.cn

汤姆森学习出版集团
THOMSON
BROOKS/COLE

封面设计：胡平利

国外著名高等院校信息科学与技术优秀教材

数据结构的 C++ 伪码实现

DATA STRUCTURES A Pseudocode Approach with C++

英文版

本书教您用实践的方法学习数据结构：

- 本书是专门针对计算机专业数据结构课程而编写的教材，作者在其教学实践中使用本书时获得很大成功。
- 清楚晓畅和易于理解的语言强调了结构化编程和软件工程学的概念。
- 先用伪码培养基本的编程技巧，然后详细准确地介绍 C++ 的语法，以此来增强编程初学者的信心。
- 丰富的插图和表格给学生更直观的印象。
- 标准模板库（Standard Template Library，STL）被作为可供选用的资源在附录中介绍。

ISBN 7-115-09766-6

9 787115 097668 >

THOMSON
BROOKS/COLE

ISBN7-115-09766-6/TP·2527
定价：55.00 元

人民邮电出版社
www.pptph.com.cn

THOMSON
BROOKS/COLE

国外著名高等院校信息科学与技术优秀教材

数据结构的C++伪码实现

DATA STRUCTURES A Pseudocode Approach with C++

英文版

人民邮电出版社

国外著名高等院校信息科学与技术优秀教材

数据结构的C++伪码实现

(英文版)

Data Structures A Pseudocode Approach with C++

Richard F. Gilberg

Behrouz A. Forouzan

人民邮电出版社　　汤姆森学习出版集团 THOMSON BROOKS/COLE

图书在版编目（CIP）数据

数据结构的C++伪码实现 /（美）吉尔伯格（Gilberg, R.F.），（美）福尔瓦赞（Forowzan, B.A.）编著. 北京：人民邮电出版社，2002.1
ISBN 7-115-09766-6

I. 数… II. ①吉…②福… III. ①数据结构—英文②C 语言—程序设计—英文 IV. TP311.12

中国版本图书馆 CIP 数据核字（2001）第 084700 号

版 权 声 明

国外著名高等院校信息科学与技术优秀教材

数据结构的 C++ 伪码实现

（英文版）

◆ 著 Richard F. Gilberg Behrouz A. Forouzan
责任编辑 陈冀康

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ pptph.com.cn
网址 http://www.pptph.com.cn
读者热线 010-67129212 010-67129211(传真)
北京汉魂图文设计有限公司制作
北京朝阳展望印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787 × 1092 1/16
印张：42.75
字数：1 047 千字 2002 年 1 月第 1 版
印数：1 – 3 000 册 2002 年 1 月北京第 1 次印刷

著作权合同登记 图字：01 – 2001 – 2487 号
ISBN 7-115-09766-6/TP·2527

定价：55.00 元

本书如有印装质量问题，请与本社联系 电话：(010)67129223

内 容 提 要

本书用 C++语言描述和学习数据结构。

全书分为 12 章，基本覆盖了数据结构的各方面的知识，包括查找、排序、链表、堆栈、队列、递归、树以及图等。书中提供了相应的算法和程序实现，还有许多针对性很强的练习题。附录部分给出了常用的 C++语言的知识，对读者进一步实现和应用本书知识提供帮助。全书的最后是部分习题的解答和术语表。

本书适合作为各高等院校计算机专业师生学习数据结构的教材，也可作为专业程序员学习数据结构的参考书籍。

出版说明

2001 年，教育部印发了《关于“十五”期间普通高等教育教材建设与改革的意见》。该文件明确指出，“九五”期间原国家教委在“抓好重点教材，全面提高质量”方针指导下，调动了各方面的积极性，产生了一大批具有改革特色的新教材。然而随着科学技术的飞速发展，目前高校教材建设工作仍滞后于教学改革的实践，一些教材内容陈旧，不能满足按新的专业目录修订的教学计划和课程设置的需要。为此该文件明确强调，要加强国外教材的引进工作。当前，引进的重点是信息科学与技术和生物科学与技术两大学科的教材。要根据专业（课程）建设的需要，通过深入调查、专家论证，引进国外优秀教材。要注意引进教材的系统配套，加强对引进教材的宣传，促进引进教材的使用和推广。

邓小平同志早在 1977 年就明确指出：“要引进外国教材，吸收外国教材中有益的东西。”随着我国加入 WTO，信息产业的国际竞争将日趋激烈，我们必须尽快培养出大批具有国际竞争能力的高水平信息技术人才。教材是一个很关键的问题，国外的一些优秀教材不但内容新，而且还提供了很多新的研究方法和思考方式。引进国外原版教材，可以促进我国教学水平的提高，提高学生的英语水平和学习能力，保证我们培养出的学生具有国际水准。

为了贯彻中央“科教兴国”的方针，配合国内高等教育教材建设的需要，人民邮电出版社约请有关专家反复论证，与国外知名的教材出版公司合作，陆续引进一些信息科学与技术优秀教材。第一批教材针对计算机专业的主干核心课程，是国外著名高等院校所采用的教材，教材的作者都是在相关领域享有盛名的专家教授。这些教材内容新，反映了计算机科学技术的最新发展，对全面提高我国信息科学与技术的教学水平必将起到巨大的推动作用。

出版国外著名高等院校信息科学与技术优秀教材的工作将是一个长期的、坚持不懈的过程，我社网站（www.pptph.com.cn）上介绍了我们首批陆续推出的图书的详细情况，后续教材的引进和出版情况我们会及时在网上发布，敬请关注。希望广大教师和学生将使用中的意见和建议及时反馈给我们，我们将根据您的反馈不断改进我们的工作，推出更多更好的引进版信息科学与技术教材。

人民邮电出版社

2001 年 12 月

序　言

使用计算机处理实际问题，必须解决的两个主要问题是如何在计算机中建立被处理对象的数据模型和如何在计算机中模拟实际问题的求解过程，最终通过计算机对实际问题的模拟处理，找到一个正确并且有效的解。前者侧重研究客观世界的信息在计算机中的表示（数据结构），后者侧重研究求解方法在计算机中的实现（算法）。显然，它们构成了计算机科学研究的主要对象，自然也成为计算机教育的核心内容。然而评价数据结构优劣的标准，主要在于使用的结构是否能够方便且有效地实现需要的算法；而讨论算法的具体实现和效率高低也必须依赖于一定的数据结构表示。因此在研究、学习数据结构和算法的过程中它们总是相互关联、相互穿插。

数据结构（或者称为算法与数据结构——其差别仅仅在于算法内容的多少而已）作为学习计算机专业的核心课程已经是国内外共识。主要目的是使学生较全面地理解算法和数据结构的概念、掌握各种数据结构与相关算法的实现方式，比较不同数据结构和算法的特点。通过学习数据结构，学生能够提高使用计算机解决实际问题的能力。在计算机技术飞速发展的今天，计算机的应用已经渗透到科学研究的各个领域，借助计算机进行问题求解已成为各理工科的共同需要。北京大学早在 1993 年就把算法与数据结构同英语、体育一样列为全校理科公共基础课，1997 年进一步明确为全校理科主干基础课。

本书是一本重点讲授数据结构的教材，全书共分 12 章。系统讲解了散列表（第 2 章）、链表（第 3 章）、栈（第 4 章）、队列（第 5 章）、树（第 7 章）、AVL 树（第 8 章）、堆（第 9 章）、B 树（第 10 章）和图（第 12 章）等各种典型的数据结构；介绍了这些数据结构在计算机内的不同表示形式；给出了与它们相关的主要算法的实现过程；分析了这些主要算法的时间空间代价。作者还在第 6 章对递归算法的工作原理、在第 11 章对解决排序问题的常用算法进行了比较详细的论述。

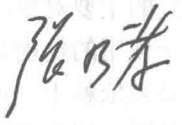
本书的主要特点是采用“伪代码”形式描述书中引入的各种算法。伪代码是一种形式上比较自由的描述语言，使用者可以根据需要插入自然语言的描述成分。使用伪代码的主要好处在于使初学者在设计算法初期不被特定程序设计语言的具体细节所束缚，而在采用自顶向下逐步求精方法把算法描述到足够精确以后，再改写成任何要求的程序设计语言。作者在设计本书的伪代码时根据软件工程和结构程序设计的思想，对过程的首部说明和过程的前/后置条件等方面都给出明确规定，有助于初学者从一开始就养成良好的程序设计习惯。当然学习数据结构必须与实际应用相结合，使用伪代码讲授可能增加了学生上机实践的难度，因此作者在每章的后面都提供了主要算法的 C++语言的代码，其目的是弥补伪代码的不足。虽然书中使用的 C++代码仅仅涉及一些面向对象语言的基本功能，并没有真正使用面向对象的方法刻画各种数据结构之间的内部联系，但是这种处理实际上要求读者对 C++语言已经有初步的了解。

本书作者多年从事计算机专业的教学工作，具有丰富的教学经验；曾经主讲过 Pascal 语

言、IBM 汇编语言、Cobol 语言和操作系统概论，近年来主要讲授 C 语言和数据结构。本书是作者在 1998 年出版的 *Data Structures: A Pseudocode Approach with C* 基础上做了进一步改进的版本。全书的结构组织合理，涵盖了国内计算机专业数据结构教学大纲的主要内容，讲解细腻，适合自学。每一章的后面都有详细的总结并配置了大量的课外作业，作者把作业分为复习思考题(exercise)、书面作业题（problem）和上机题（project）三类供读者选择使用。

本书是学习数据结构的很好的教学参考书，可供理工科各专业的老师和学生使用。希望本书英文版在国内的发行，对于吸取欧美先进的教学思想和方法、促进国内计算机的基础教育、探索原版计算机教材的引进和推广方面起到积极作用。

北京大学信息科学系教授



2001 年 11 月于北大

Preface

The study of data structures is both exciting and challenging. It is exciting because it presents a wide range of programming techniques that make it possible to solve larger and more complex problems. It is challenging because the complex nature of data structures brings with it many concepts that change the way we approach the design of programs.

Because the study of data structures encompasses an abundant amount of material, you will find that it is not possible to cover all of it in one term. In fact, data structures is such a pervasive subject that you will find it taught in lower-division, upper-division, and graduate programs.

Features of This Book

Our primary focus in this text is to present data structures as an introductory subject, taught in a lower-division course. With this focus in mind, we present the material in a simple, straightforward manner with many examples and figures. We also deemphasize the mathematical aspect of data structures, leaving the formal mathematical proofs of the algorithms for later courses.

Pseudocode

Pseudocode is an English-like presentation of the steps needed to solve a problem. It is written with a relaxed syntax that allows students to solve a problem at a level that hides the detail while they concentrate on the problem requirements. In other words, it allows students to concentrate on the big picture.

In addition to being an excellent design tool, pseudocode is also language independent. Consequently, students can use the same pseudocode design to implement an algorithm in several different languages. We developed our pseudocode syntax in our data structures classes over a 15-year period. During that time, our students have implemented the pseudocode algorithms in Pascal, C, and C++. In this text, we use C++ for all of our code implementations.

As we discuss the various data structures, we first present the general principles using diagrams to help the student visualize the concept. If the data structure is large and complex enough to require several algorithms, we use a structure chart to present a design solution. Once the design and structure are fully understood, we present a pseudocode algorithm, followed as appropriate by its C++ implementation.

Abstract Data Types

The second major feature of this text is its use of abstract data types (ADTs) implemented as C++ classes. To make ADTs data independent, we use template classes. All ADTs accept either one (data) or two (data

and key) arguments. In this way any data type, including derived types and structures, can be used with all ADTs. Conversely, each ADT can be used with any data type as long as the required operators are predefined for that type. We introduce the concept immediately in Chapter 1 and use it extensively throughout the text.

Not every data structure should be implemented as an ADT class. However, where appropriate, we develop a complete C++ implementation for the student's study and use. Specifically, students will find ADT class implementations for Lists (Chapter 3), Stacks (Chapter 4), Queues (Chapter 5), AVL Trees (Chapter 8), B-Trees (Chapter 10), and Graphs (Chapter 12). The code for all of the ADTs is available on the Instructor's Materials page at the Brooks/Cole Web site www.brookscole.com/compsci/gilberg/cs2pp.

Structure and Style

One of our basic educational tenets is that good habits are formed early. The corollary is that bad habits are hard to break. Therefore, we consistently emphasize the principles of structured programming and software engineering. Every algorithm and program in the book uses a consistent style. As the algorithms and programs are analyzed, style and standards are further explained. While we acknowledge that there are many good styles, our experience has shown that if students are exposed to a good style and implement it, they will be better able to adapt to other good styles. On the other hand, unlearning sloppy short-cut habits is very difficult.

Visual Approach

A brief scan of the book will demonstrate that our approach is primarily visual. There are over 345 figures, 35 tables, 140 algorithms, 180 programs, and numerous code examples. Although this amount of material tends to create a large book, these materials make it much easier for students to follow the concepts.

Pedagogical End Materials

End of chapter materials reenforce what the student has learned. The important topics in the chapter are summarized in bulleted lists. Following the summary are three practice sets.

Exercises are multiple choice and short answer questions covering the material in the chapter. The answers to the odd numbered questions are included in the back of the book.

Problems are short assignments that ask the student to develop a pseudocode algorithm or write a short program to be run on a computer. These problems can usually be developed in 2 to 3 hours. The instructor's manual contains complete solutions for all exercises and problems.

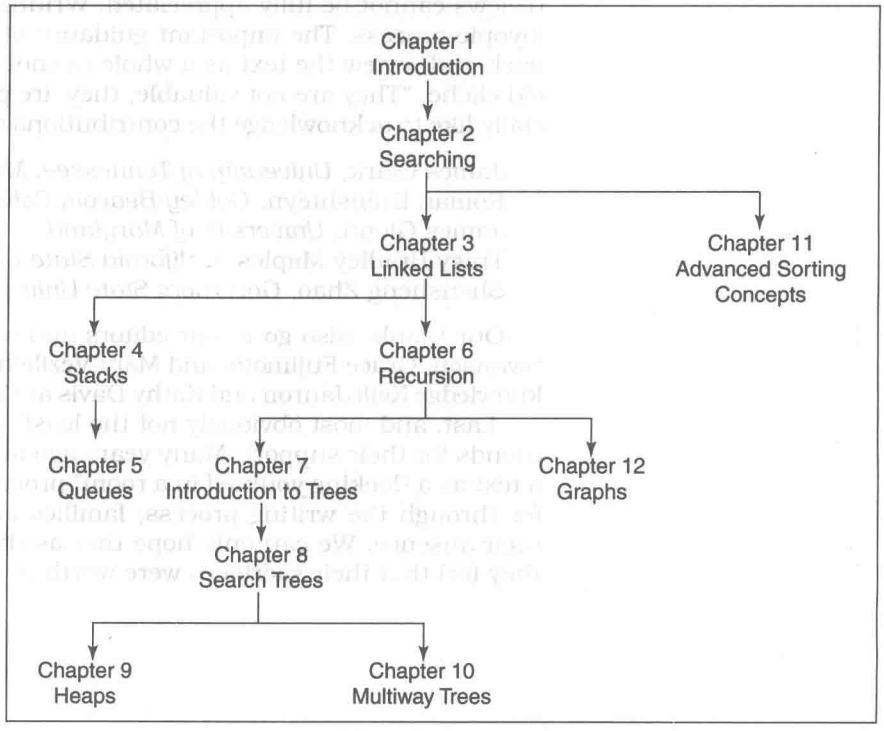
Projects are longer, major assignments that may take an average student 6 to 9 hours or more to develop.

Organization And Order Of Topics

We have tried to build flexibility into the text so that the material may be covered in the order that best suits the needs of a particular class. Although we use the materials in the order presented in the text, there are other possible sequences (shown in the figure on this page). We recommend that you assign Chapter 1 as general reading. It contains basic information on pseudocode, abstract data types, and algorithmics students will need for the rest of the text.

The first two sections of Chapter 2 review sequential and binary search concepts. The third section, hashed list searches, may be new material. If you have covered search algorithms in your programming class, you may save this chapter for later. On the other hand, if your students have not studied searching algorithms, then you will need to cover at least the first section. Many of the algorithms in the following chapters require an understanding of sequential and ordered list searching. In many texts, sorting is covered with searching. Because our sorting chapter includes the recursive implementation of quick sort and heap sort (which requires an understanding of trees and heaps), we place it at the end of the text. With the exception of these two sorts, however, it could be covered before Chapter 3.

Chapter 3 introduces linear lists and the basic linked list data structures. It also introduces the first complete ADT class. For these reasons, Chapter 3 should be covered before the remaining chapters in the text.



Possible subject sequences

The stack concept (Chapter 4) is basic to an understanding of recursion (Chapter 6), and recursion is in turn required to understand trees (Chapters 7, 8, and 10) and heaps (Chapter 9). Likewise, queues (Chapter 5) are used in breadth-first traversals in Chapters 7 and 12.

Chapter 9, Heaps, is a stand-alone chapter. Its only outside reference is the heap sort in Chapter 11.

We end the text with graphs in Chapter 12. Like many other data structure subjects, a complete course could be devoted to graphs. In this chapter, we review some basic graph concepts. Although this material could be covered anytime after Chapter 3, you will find that it contains some of the most difficult algorithms in the text. For this reason, we recommend that you present Chapter 12 at the end of the term,

when your students will be much better prepared to handle the material.

Acknowledgments

No text of this scope can be developed without the support of many people. This is especially true for this text. The basic algorithms were field-tested by our students at De Anza College. Our first acknowledgment, therefore, has to be to the hundreds of students who by using and commenting on the text made a vital contribution. We especially thank our student, Scott Demouthe, who not only proofed the text, but verified every exercise and problem at the ends of the chapters.

We would also like to acknowledge the support of the De Anza staff. Their encouragement helped us launch the project, and their comments contributed to its success. To name them all is impossible, but we especially thank John Perry, Delia Garbacea, and George Rice.

To anyone who has not been through the process, the value of peer reviews cannot be fully appreciated. Writing a text rapidly becomes a myopic process. The important guidance of reviewers who can stand back and review the text as a whole cannot be measured. To twist an old cliche, "They are not valuable, they are priceless." We would especially like to acknowledge the contributions of the following reviewers:

James Clark, *University of Tennessee, Martin*
Roman Erenshteyn, *Goldey-Beacom College*
James Glenn, *University of Maryland*
Tracy Bradley Maples, *California State University—Long Beach*
Shensheng Zhao, *Governors State University*

Our thanks also go to our editors and staff at Brooks/Cole, Kallie Swanson, Grace Fujimoto, and Mary Vezilich. We would also like to acknowledge Kelli Jauron and Kathy Davis at Carlisle Publishers Services.

Last, and most obviously not the least, we thank our families and friends for their support. Many years ago an author described writing a text as a "locking yourself in a room" process. While the authors suffer through the writing process, families and friends suffer through their absence. We can only hope that as they view the final product, they feel that their sacrifices were worth it.

Richard F. Gilberg
Behrouz A. Forouzan

Contents