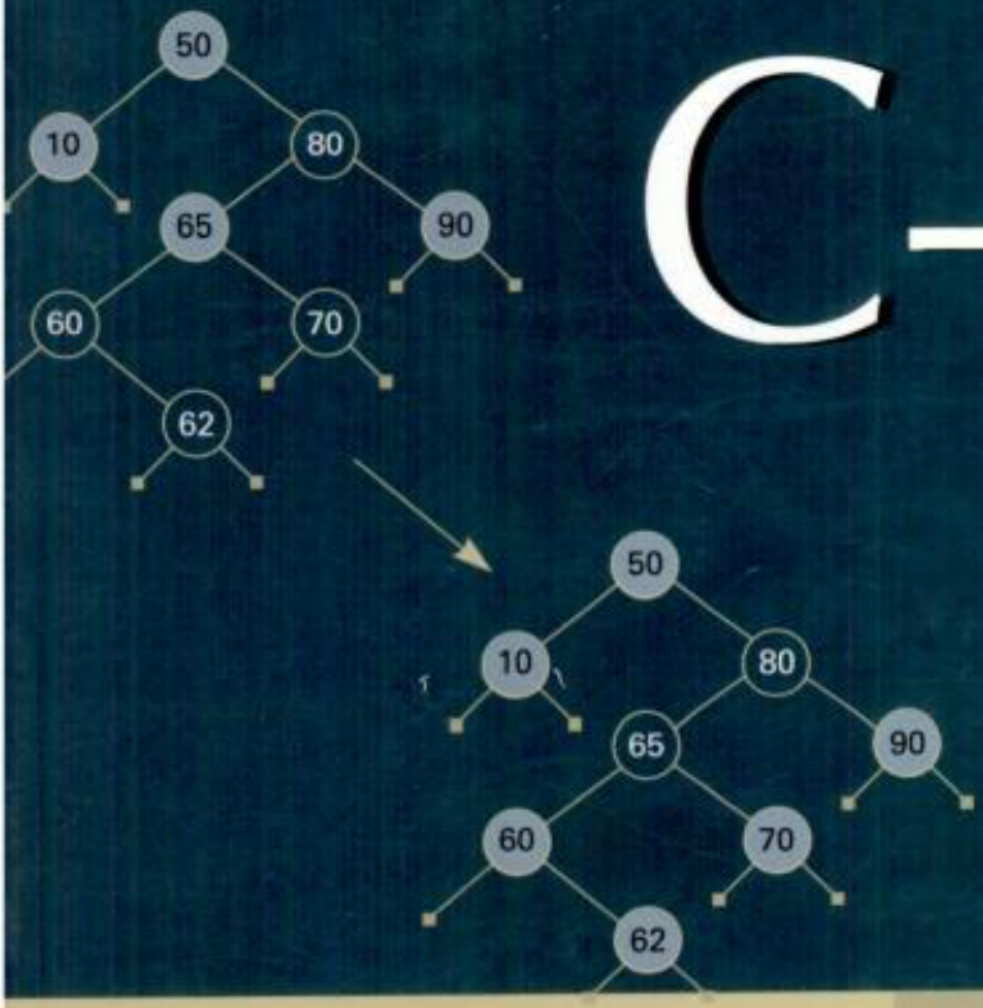
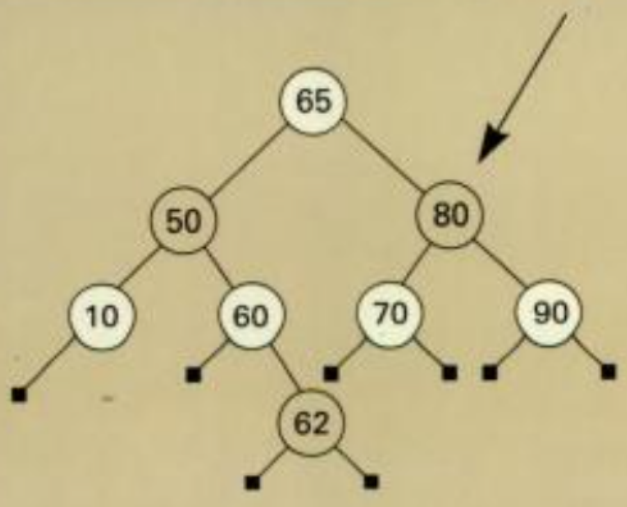
Universities Press

COMPUTER SCIENCE

Data Structures, Algorithms and Applications in C++





Second Edition

SARTAJ SAHNI

Data Structures, Algorithms and Applications in C++

Second Edition

SARTAJ SAHNI

Universities Press

Universities Press (India) Private Limited

Registered Office
3-5-819 Hyderguda, Hyderabad 500 029 (A.P.), India
Email: hyd2_upilco@sancharnet.in

Distributed by
Orient Longman Private Limited

Registered Office
3-6-752 Himayatnagar, Hyderabad 500 029 (A.P.), India

Other Offices
Bangalore, Bhopal, Bhubaneshwar, Chennai,
Ernakulam, Guwahati, Hyderabad, Jaipur, Kolkata,
Lucknow, Mumbai, New Delhi, Patna

First published in India by
Universities Press (India) Private Limited 2005
ISBN 81 7371 522 X

For sale in India, Pakistan, Nepal, Myanmar, Maldives, Bhutan, Bangladesh and Sri Lanka only.
Not for export to other countries.

Printed at
Orion Printers
Hyderabad 500 004

Published by
Universities Press (India) Private Limited
3-5-819 Hyderguda, Hyderabad 500 029 (A.P.), India

To my mother,

Santosh

My wife,

Neeta

and my children,

Agam, Neha, and Param

PREFACE

The study of data structures and algorithms is fundamental to computer science and engineering. A mastery of these areas is essential for us to develop computer programs that utilize computer resources in an effective manner. Consequently, all computer science and engineering curriculums include one or more courses devoted to these subjects. Typically, the first programming course introduces students to basic data structures (such as stacks and queues) and basic algorithms (such as those for sorting and matrix algebra). The second programming course covers more data structures and algorithms. The next one or two courses are usually dedicated to the study of data structures and algorithms.

The explosion of courses in the undergraduate computer science and engineering curriculums has forced many universities and colleges to consolidate material into fewer courses. At the University of Florida, for example, we offer a single one-semester undergraduate data structures and algorithms course. Students coming into this course have had a one-semester course in Java programming and another in discrete mathematics/structures.

Data Structures, Algorithms, and Applications in C++ has been developed for use in programs that cover this material in a unified course as well as in programs that spread out the study of data structures and algorithms over two or more courses. The book is divided into three parts. Part I, which consists of Chapters 1 through 4, is intended as a review of C++ programming concepts and of methods to analyze and measure the performance of programs. Students who are familiar with programming in C should be able to read Chapter 1 and bridge the gap between C and C++. Although Chapter 1 is not a primer on C++, it covers most of the C++ constructs with which students might have become rusty. These concepts include parameter passing, template functions, dynamic memory allocation, recursion, classes, inheritance, and throwing and catching exceptions. Chapters 2 and 3 are a review of methods to analyze the performance of a program—operation counts, step counts, and asymptotic notation (big oh, omega, theta, and little oh). Chapter 4 reviews methods to measure performance experimentally. This chapter also has a brief discussion of how cache affects measured run times. The applications considered in Chapter 2 explore fundamental problems typically studied in a beginning programming course—simple sort methods such as bubble, selection,

insertion, and rank (or count) sort; sequential search; polynomial evaluation using Horner's rule; and matrix operations such as matrix addition, transpose, and multiply. Chapter 3 examines binary search. Even though the primary purpose of Chapters 2 through 4 is to study performance analysis and measurement methods, these chapters also ensure that all students are familiar with a set of fundamental algorithms.

Chapters 5 through 16 form the second part of the book. These chapters provide an in-depth study of data structures. Chapters 5 and 6 form the backbone of this study by examining the array and pointer (or linked) methods of representing data. These two chapters develop C++ classes to represent the linear list data structure, using each representation method. We compare the different representation schemes with respect to their effectiveness in representing linear lists by presenting experimental data. The remaining chapters on data structures use the representation methods of Chapters 5 and 6 to arrive at representations for other data structures such as arrays and matrices (Chapter 7), stacks (Chapter 8), queues (Chapter 9), dictionaries (Chapters 10, 14, and 15), binary trees (Chapter 11), priority queues (Chapter 12), tournament trees (Chapter 15), and graphs (Chapter 16).

In our treatment of data structures, we have attempted to maintain compatibility with similar or identical structures that are available in the C++ Standard Templates Library (STL). For example, the linear list data structure that is the subject of Chapter 5 is modeled after the STL class `vector`. Throughout the book we make use of STL functions such as `copy`, `min`, and `max` so students becomes familiar with these functions.

The third part of this book, which comprises Chapters 17 through 21 (Chapters 20 and 21 are available from the Web site for this book), is a study of common algorithm-design methods. The methods we study are greedy (Chapter 17), divide and conquer (Chapter 18), dynamic programming (Chapter 19), backtracking (Chapter 20), and branch and bound (Chapter 21). Two lower-bound proofs (one for the minmax problem and the other for sorting) are provided in Section 18.4; approximation algorithms for machine scheduling (Section 12.6.2), bin packing (Section 13.5), and the 0/1 knapsack problem (Section 17.3.2) are also covered. NP-hard problems are introduced, informally, in Section 12.6.2.

A unique feature of this book is the emphasis on applications. Several real-world applications illustrate the use of each data structure and algorithm-design method developed in this book. Typically, the last section of each chapter is dedicated to applications of the data structure or design method studied earlier in the chapter. In many cases additional applications are also introduced early in the chapter. We have drawn applications from various areas—sorting (bubble, selection, insertion, rank, heap, merge, quick, bin, radix, and topological sort); matrix algebra (matrix addition, transpose, and multiplication); electronic design automation (finding the nets in a circuit, wire routing, component stack folding, switch-box routing, placement of signal boosters, crossing distribution, and backplane board ordering); compression and coding (LZW compression and Huffman coding); computational

geometry (convex hull and closest pair of points); simulation (machine shop simulation); image processing (component labeling); recreational mathematics (Towers of Hanoi, tiling a defective chessboard, and rat in a maze); scheduling (LPT schedules); optimization (bin packing, container loading, 0/1 knapsack, and matrix multiplication chains); statistics (histogramming, finding the minimum and maximum, and finding the kth smallest); and graph algorithms (spanning trees, components, shortest paths, max clique, bipartite graph covers, and traveling salesperson). Our treatment of these applications does not require prior knowledge of the application areas. The material covered in this book is self-contained and gives students a flavor for what these application areas entail.

By closely tying the applications to the more basic treatment of data structures and algorithm-design methods, we hope to give the student a greater appreciation of the subject. Further enrichment can be obtained by working through the more than 800 exercises in the book and from the associated Web site.

WEB SITE

The URL for the Web site for this book is

`http://www.cise.ufl.edu/~sahni/dsaac`

From this Web site you can obtain all the programs in the book together with sample data and generated output. The sample data are not intended to serve as a good test set for a given program; rather they are just something you can use to run the program and compare the output produced with the given output. Solutions to many of the exercises that appear in each chapter, codes for these solutions, sample tests and solutions to these tests, additional applications, and enhanced discussions of some of the material covered in the text also appear in the Web site.

HOW TO USE THIS BOOK

There are several ways in which this book may be used to teach the subject of data structures and/or algorithms. Instructors should make a decision based on the background of their students, the amount of emphasis instructors want to put on applications, and the number of semesters or quarters devoted to the subject. We give a few of the possible course outlines below. We recommend that the assignments require students to write and debug several programs, beginning with a collection of short programs and working up to larger programs as the course progresses. Students should read the text at a pace commensurate with classroom coverage of topics.

TWO-QUARTER SCHEDULE—QUARTER 1

One week of review. Data structures and algorithms sequence.

Week	Topic	Reading
1	Review of C++ and program performance.	Chapters 1–4. Assignment 1 given out.
2	Array-based representation.	Chapter 5. Assignment 1 due.
3	Linked representation.	Sections 6.1–6.4. Assignment 2 given out.
4	Bin sort and equivalence classes.	Sections 6.5.1 and 6.5.4. Assignment 2 due.
5	Arrays and matrices.	Chapter 7. Examination.
6	Stacks and queues.	Chapters 8 and 9. Assignment 3 given out.
7	Skip lists and hashing.	Chapter 10. Assignment 3 due.
8	Binary and other trees.	Sections 11.1–11.8. Assignment 4 given out.
9	Union-find application. Heaps and heap sort.	Sections 11.9.2, 12.1–12.4, and 12.6.1. Assignment 4 due.
10	Leftist trees, Huffman codes, and tournament trees.	Sections 12.5 and 12.6.3 and Chapter 13.