


Preface to the Second Edition

A data structure is the logical or mathematical arrangement of data in memory. It considers not only the physical layout of the data items in the memory but also the relationships between these data items and the operations that can be performed on these items. The choice of appropriate data structures and algorithms forms the fundamental step in the design of an efficient program. Thus, a thorough understanding of data structure concepts is essential for students who wish to work in the design and implementation of software systems. C, a general-purpose programming language, having gained popularity in both academia and industry serves as an excellent choice for learning data structures.

This second edition of *Data Structures Using C* has been developed to provide a comprehensive and consistent coverage of both the abstract concepts of data structures as well as the implementation of these concepts using C language. The book utilizes a systematic approach wherein the design of each of the data structures is followed by algorithms of different operations that can be performed on them, and the analysis of these algorithms in terms of their running times.

New to the Second Edition

Based on the suggestions from students and faculty members, this edition has been updated and revised to increase the clarity of presentation where required. Some of the prominent changes are as follows:

- New sections on omega and theta notations,  multi-linked lists, forests, conversion of general trees into binary trees, 2-3 trees, binary heap implementation of priority queues, interpolation search, jump search, tree sort, bucket hashing, cylinder surface indexing
- Additional C programs on header linked lists, parentheses checking, evaluation of prefix expressions, priority queues, multiple queues, tree sort, file handling, address calculation sort
- New appendices on dynamic memory allocation, garbage collection, backtracking, Johnson's problem
- Stacks and queues and multi-way search trees are now covered in separate chapters with a more comprehensive explanation of concepts and applications

Extended Material

Chapter 1—This chapter has been completely restructured and reorganized so that it now provides a brief recapitulation of C constructs and syntax. Functions and pointers which were included as independent chapters in the first edition have now been jointly included in this chapter.

Chapter 2—New sections on primitive and non-primitive data structures, different approaches to designing algorithms, omega, theta, and little notations have been included. A number of new examples have also been added which show how to find the complexity of different functions.

Chapter 5—This chapter now includes brief sections on unions, a data type similar to structures.

Chapter 6—This chapter has been expanded to include topics on multi-linked lists, multi-linked list implementation of sparse matrices, and a C program on header linked lists.

Chapter 7—New C programs on parenthesis checking and evaluation of prefix expressions have been added. Recursion, which is one of the most common applications of stacks, has been moved to this chapter.

Chapter 8—New C programs on priority queues and multiple queues have been included.

Chapter 9—This chapter now includes sections on general trees, forests, conversion of general trees into binary trees, and constructing a binary tree from traversal results.

Chapter 10—An algorithm for in-order traversal of a threaded binary tree has been added.

Chapter 11—A table summarizing the differences between B and B+ trees and a section on 2-3 trees have been included.

Chapter 12—A brief section on how binary heaps can be used to implement priority queues has been added.

Chapter 13—This chapter now includes a section which shows the adjacency multi-list representation of graphs.

Chapter 14—As a result of organization, the sections on linear and binary search have been moved from Chapter 3 to this chapter. New search techniques such as interpolation search, jump search, and Fibonacci search have also been included. The chapter also extends the concept of sorting by including sections on practical considerations for internal sorting, sorting on multiple keys, and tree sort.

Chapter 15—New sections on bucket hashing and rehashing have been included.

Chapter 16—This chapter now includes a section on cylinder surface indexing which is one of the widely used indexing structures for files stored in hard disks.

Content and Coverage

This book is organized into 16 chapters.

Chapter 1, Introduction to C provides a review of basic C constructs which helps readers to familiarize themselves with basic C syntax and concepts that will be used to write programs in this book.

Chapter 2, Introduction to Data Structures and Algorithms introduces data structures and algorithms which serve as building blocks for creating efficient programs. The chapter explains how to calculate the time complexity which is a key concept for evaluating the performance of algorithms. From *Chapter 3* onwards, every chapter discusses individual data structures in detail.

Chapter 3, Arrays provides a detailed explanation of arrays that includes one-dimensional, two-dimensional, and multi-dimensional arrays. The operations that can be performed on such arrays are also explained.

Chapter 4, Strings discusses the concept of strings which are also known as character arrays. The chapter not only focuses on reading and writing strings but also explains various operations that can be used to manipulate the character arrays.

Chapter 5, Structures and Unions deals with structures and unions. A structure is a collection of related data items of different types which is used for implementing other data structures such as linked lists, trees, graphs, etc. We will also read about unions which is also a collection of variables of different data types, except that in case of unions, we can only store information in one field at any one time.

Chapter 6, Linked Lists discusses different types of linked lists such as singly linked lists, doubly linked lists, circular linked lists, doubly circular linked lists, header linked lists, and multi-linked lists. Linked list is a preferred data structure when it is required to allocate memory dynamically.

Chapter 7, Stacks focuses on the concept of last-in, first-out (LIFO) data structure called stacks. The chapter also shows the practical implementation of these data structures using arrays as well as linked lists. It also shows how stacks can be used for the evaluation of arithmetic expressions.

Chapter 8, Queues deals with the concept of first-in, first-out (FIFO) data structure called queues. The chapter also provides the real-world applications of queues.

Chapter 9, Trees focuses on binary trees, their traversal schemes and representation in memory. The chapter also discusses expression trees, tournament trees, and Huffman trees, all of which are variants of simple binary trees.

Chapter 10, Efficient Binary Trees broadens the discussion on trees taken up in *Chapter 9* by going one step ahead and discussing efficient binary trees. The chapter discusses binary search trees, threaded binary trees, AVL trees, red-black trees, and splay trees.

Chapter 11, Multi-way Search Trees explores trees which can have more than one key value in a single node, such as M-way search trees, B trees, B+ trees, tries, and 2-3 trees.

Chapter 12, Heaps discusses three types of heaps—binary heaps, binomial heaps, and Fibonacci heaps. The chapter not only explains the operations on these data structures but also makes a comparison, thereby highlighting the key features of each structure.

Chapter 13, Graphs contains a detailed explanation of non-linear data structure called graphs. It discusses the memory representation, traversal schemes, and applications of graphs in the real world.

Chapter 14, Searching and Sorting covers two of the most common operations in computer science, i.e. searching and sorting a list of values. It gives the technique, complexity, and program for different searching and sorting techniques.

Chapter 15, Hashing and Collision deals with different methods of hashing and techniques to resolve collisions.

Chapter 16, the last chapter of the book, *Files and Their Organization*, discusses the concept related to file organization. It explains the different ways in which files can be organized on the hard disk and the indexing techniques that can be used for fast retrieval of data.

The book also provides a set of seven appendices.

Appendix A introduces the concept of dynamic memory allocation in C programs.

Appendix B provides a brief discussion of garbage collection technique which is used for automatic memory management.

Appendix C explains **backtracking** which is a recursive algorithm that uses stacks.

Appendix D discusses Johnson's algorithm which is used in applications where an optimal order of execution of different activities has to be determined.

Appendix E includes two C programs which show how to read and write binary files.

Appendix F includes a C program which shows how to sort a list of numbers using address calculation sort.

Appendix G provides chapter-wise answers to all the objective questions.

Reema Thareja