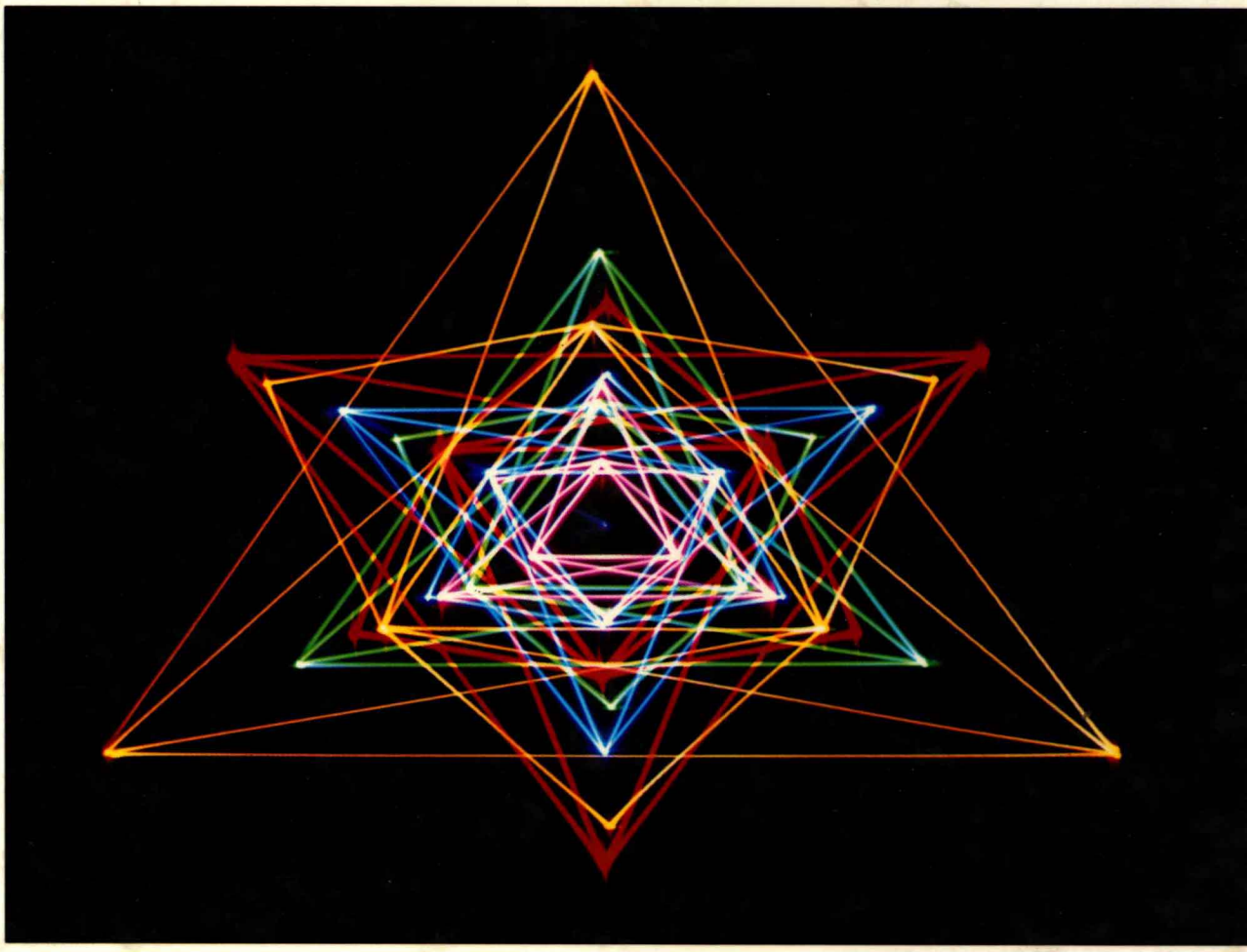
Introduction to Data Structures with Pascal



Thomas L. Naps
Bhagat Singh

Introduction to Data Structures with Pascal

Thomas L. Naps
Lawrence University

Bhagat Singh
University of Wisconsin Center System

WEST PUBLISHING COMPANY
St. Paul New York San Francisco Los Angeles

Copyediting: Martha S. Knutson
Design: Lucy Lesiak
Typesetting: Rolin Graphics
Cover Design: Paul Kemp
Cover Photograph: Floyd Rollefstad, Laser Fantasy Productions

Library of Congress Cataloging-in-Publication Data

Naps, Thomas L.
Introduction to data structures with PASCAL.

Includes index.
1. PASCAL (Computer program language) 2. Data structures (Computer science) I. Singh, Bhagat, 1940- . II. Title.
QA76.73.P2N37 1986 005.7'3 85-22582
ISBN 0-314-93207-0

Introduction to Data Structures with Pascal

To

Joyce and Rosemary

Preface

With the rapid evolution of computer science, the study of data structures has found its way into the undergraduate curriculum as early as the sophomore year. Moreover, an increasingly diverse collection of students is studying this subject. Once solely the domain of hard-core computer science majors, data structures is now taken by business, economics, engineering, and mathematics majors who wish to enhance their chosen discipline with a strong background in computer science. Texts that emphasize a highly rigorous mathematical approach to data structures no longer are appropriate for this growing number of heterogeneous students. In *Introduction to Data Structures with Pascal*, we attempt to satisfy the needs of this new group of students by providing a text that emphasizes implementing and evaluating data structures in practical situations and avoids relying on an overly theoretical approach. The text is designed for anyone who has had a solid programming background in Pascal. Such a background should include a detailed grasp of procedures, functions, parameter passing, and arrays. To a lesser extent, some prior work with records and files is desirable although these topics are developed as needed in the text itself.

Introduction to Data Structures with Pascal represents an alternative to our earlier pseudocode-based *Introduction to Data Structures*. The development of such an alternative was motivated by a near 50-50 split which we found among the reviewers of our efforts. Though these reviewers were essentially in agreement as to topics covered and the order in which they were presented, there was a pronounced and irresolvable split between those who believed in a pseudocode approach and those who felt that seeing algorithms in the familiar Pascal language would make concepts easier for students. It is to this latter group that this version of the text is addressed.

In addition to providing a Pascal-specific orientation, we have also expanded our coverage of important software engineering concepts to help students as they begin using data structures in larger programming projects. Sections on Program Design Considerations appear in each chapter. Some of these sections (notably those in Chapter 3, Chapter 7, and Chapter 11) deal with the development of complete large-scale programs which use the data structures studied up to that point. Others touch upon design issues which are then further expanded in the exercises and programming problems.

Three new appendices have also been added for those students of Pascal who may need a bit more general experience as well as a thorough treatment of data structures. Appendix C presents a description of those testing, verification, and debugging strategies that are so essential for students developing large-scale programs. Appendix B on formal analysis of algorithms presents the basics of this topic. We have chosen to treat this issue in a separate appendix because we want the book to retain its intuitive, implementation-oriented approach. Appendix A discusses how various versions of Pascal implement direct access files—a topic which we consider essential despite its absence from the Pascal standard. A "real world" proof of the essential nature of this topic is that virtually all widely used implementations of Pascal have included procedures for direct access files in their repetoires. Appropriate references to all of these appendices are made throughout the text, making it easy for the student to merge this additional material with specific data structures topics. Depending upon the curriculum at your institution, this increased emphasis upon software engineering issues and the coverage of advanced topics in a Pascal-specific fashion may in fact make the text suitable for an advanced course in programming, as described in the C.S. II Course Guidelines recommended by the recent ACM Curriculum Task Force for C.S. II.

Further changes from the pseudocode version are less oriented toward Pascal but instead induced by feedback we have been receiving from the field. Exercises have been added in each chapter. The thematic Wing-and-a-Prayer Airlines problems proved popular, so we have included two other thematic problem lines running throughout the text—the Fly-By-Night Credit Company and the Bay Area Brawlers Professional Football Club. In Chapter 5, we have expanded our discussion of recursion, offering a more gradual introduction to this difficult topic. Discussions of algorithm efficiency in the body of the text itself have been isolated in separate sections for greater emphasis and easier reference. Finally, new "In the World of Applications" boxes have been developed for Chapters 4, 7, and 9 to replace old ones which some readers felt lacked a "real world" flavor.

However, despite these changes, the two major goals of this text remain unchanged from the pseudocode version:

1. The student must acquire an understanding of the algorithms that manipulate various data structures.
2. The student must learn to select from among data structures for a given application.

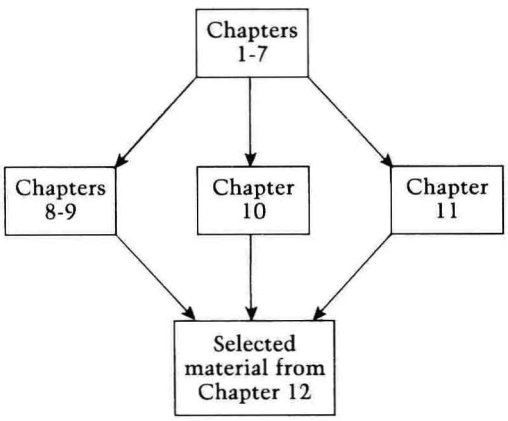
Relative to this second goal, data structures could be considered the toolbox of the computer scientist or data processor. Many jobs can be accomplished in more than one way; the student who understands data structures thoroughly is able to match the tool to the job most effectively.

Understanding of data structures is enhanced by pictures. For this reason, we have made ours a graphically oriented book. Throughout the text, algorithms written in Pascal are enhanced by "graphic documentation" to clarify the responsibilities of various segments of code. Our experience as instructors of data structures has so often indicated that a student's question about how to do something can be answered by the simple dictum to "draw a picture." Data structures are, after all, the programmer's way of implementing certain types of mental images within the computer. Every segment of a program is in some way responsible for maintaining or altering that structural image. Can there be a better way to document that segment than to picture what it is intended to do? In a certain sense, this graphic orientation embodies the essence of data abstraction—that the student understand the data structure itself rather than memorize a particular algorithm or the details of a particular implementation.

The order of topics presented in the book is relatively flexible, although the first seven chapters represent a core of fundamental material that should be covered before moving on to any of the topics in Chapters 8 through 12. Chapter 1 lays the groundwork for the rest of the book by specifying a rationale for data structures and developing those advanced features of Pascal (including records and pointer variables) that will be used throughout. The first data structure discussed in detail is the linked list, presented in Chapter 2. Initial implementation of linked lists in this chapter is achieved through the use of arrays instead of Pascal's NEW and DISPOSE procedures to allow the student to actually trace through the implementation. In Chapter 3, we consider strings in detail as an example of an application in which pointers, indices, and linked lists can be used effectively. Chapter 4 presents two special types of lists: queues and stacks, and Chapter 5 follows up on this theme by looking at two critically important applications of stacks: the parsing of arithmetic expressions and recursion. Chapters 6 and 7 cover in detail what we consider to be the most versatile of all data structures—the tree. We discuss both binary and general trees, along with such variations as threading and height-balancing.

Although the first seven chapters of this book should be studied consecutively, there is considerable leeway in the order in which the remaining topics may be covered. Some instructors may find it possible to cover only selected topics from Chapters 8 through 12, and these chapters are connected loosely enough to allow picking and choosing. Chapter 8 fully describes multidimensional and sparse matrices, which serves as a convenient introduction to the adjacency matrix representation of graphs and networks, the topic of Chapter 9. Chapters 8 and 9 thus form a cohesive unit. However, it is possible to proceed directly from Chapter 7 to either Chapter 10 (Sorting) or Chapter 11 (Search Methods). Chapter 12, Data Structures and Data Management, presents a nice wrapup for a data structures course by describing how a variety of data structures can be applied in the areas of memory management and database management. Because not all instructors will find time in a semester to cover the material in Chapter 12, we have written it such that it lends itself to independent study. For those students going on into systems programming, the material in Chapter 12 on memory management algorithms used by operating systems should prove particularly valuable. For the more business-oriented student, the database management section is more suitable. The following flowchart summarizes the possible sequences in which chapters can be covered.

Additional features of the book include "In the World of Applications . . ." boxes, which emphasize how data structures are used in a variety of settings; chapter exercises and programming problems; a glossary; and an appendix of solutions to odd-numbered exercises. The Instructor's Manual provides useful hints on how to present the material in each chapter, solutions to even-numbered exercises, and additional ideas for programming problems and projects. Transparency masters that can be used to illustrate key concepts are available to adopters from the publisher. A list of figures from the text available as transparency masters is in the Instructor's Manual.



Acknowledgements

Although work on this text was greatly smoothed by the previous work we had done for the pseudocode version, a large number of people have nonetheless played key roles in its development. Of course, our students never cease to amaze us with the clever ideas and insights they continually offer. Jerry Westby, our editor, always had another (and another) innovative idea to try. Our work with production editor Peggy Adams and the staff at West Publishing proved to be a pleasure. Thanks go also to the large group of skilled reviewers who offered constructive criticism on our efforts:

John Rezac, Johnson County Community College
Andrew Bernat, University of Texas at El Paso
Michael Henry, West Virginia University
Charles Williams, Georgia State University
W. Bruce Croft, University of Massachusetts at Amherst
Robert Holloway, University of Wisconsin at Madison
Tamisra Sanyal, Monroe Community College
Martha Hansard, Georgia State University
Barent Johnson, University of Wisconsin at Oshkosh

Finally, our most sincere gratitude is directed to our wives and families. The warmth and freshness they provide keep our lives nicely unstructured.

Index

Contents