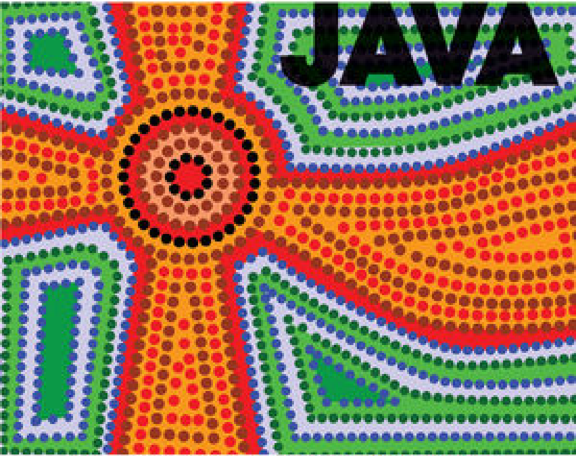
FIFTH EDITION

Data Structures & Algorithms in



MICHAEL T. GOODRICH | ROBERTO TAMASSIA

International Student Version

Data Structures and Algorithms in Java

Fifth Edition
International Student Version

Michael T. Goodrich
Department of Computer Science
University of California, Irvine

Roberto Tamassia
Department of Computer Science
Brown University

John Wiley & Sons, Inc.

Preface

This international student version of *Data Structures and Algorithms in Java* provides an introduction to data structures and algorithms, including their design, analysis, and implementation. In terms of curricula based on the ***IEEE/ACM 2001 Computing Curriculum***, this book is appropriate for use in the courses CS102 (I/O/B versions), CS103 (I/O/B versions), CS111 (A version), and CS112 (A/I/O/F/H versions). We discuss its use for such courses in more detail later in this preface.

The major changes in the fifth edition are the following:

- We added more examples and discussion of data structure and algorithm analysis.
- We enhanced consistency with the Java Collections Framework.
- We enhanced the discussion of algorithmic design techniques, like dynamic programming and the greedy method.
- We added new material on improved Java I/O methods.
- We created this ***international student version*** of the book, which contains content, such as Java internationalization and international units, more appropriate for readers outside of North America and Europe.
- We added a discussion of the difference between array variable-name assignment and array cloning.
- We included an expanded discussion of the Deque interface and LinkedList class in Java.
- We increased coverage of entry objects in the Java Collection Framework.
- We fully integrated all code fragment APIs to use generic types.
- We added discussions of the NavigatableMap interface, as well as their implementations in the Java Collections Framework using skip lists.
- We included a discussion of the Java TreeMap class.
- We provided discriptions of the sorting methods included in the Java library.
- We expanded and revised exercises, continuing our approach of dividing them into reinforcement, creativity, and project exercises.

This book is related to the following books:

- M.T. Goodrich, R. Tamassia, and D.M. Mount, *Data Structures and Algorithms in C++*, John Wiley & Sons, Inc. This book has a similar overall structure to the present book, but uses C++ as the underlying language (with some modest, but necessary pedagogical differences required by this approach).
- M.T. Goodrich and R. Tamassia, *Algorithm Design: Foundations, Analysis, and Internet Examples*, John Wiley & Sons, Inc. This is a textbook for a more advanced algorithms and data structures course, such as CS210 (T/W/C/S versions) in the IEEE/ACM 2001 curriculum.

Use as a Textbook

The design and analysis of efficient data structures has long been recognized as a vital subject in computing, for the study of data structures is part of the core of every collegiate computer science and computer engineering major program we are familiar with. Typically, the introductory courses are presented as a two- or three-course sequence. Elementary data structures are often briefly introduced in the first programming course or in an introduction to computer science course and this is followed by a more in-depth introduction to data structures in the courses that follow after this. Furthermore, this course sequence is typically followed at a later point in the curriculum by a more in-depth study of data structures and algorithms. We feel that the central role of data structure design and analysis in the curriculum is fully justified, given the importance of efficient data structures in most software systems, including the Web, operating systems, databases, compilers, and scientific simulation systems.

With the emergence of the object-oriented paradigm as the framework of choice for building robust and reusable software, we have tried to take a consistent object-oriented viewpoint throughout this text. One of the main ideas of the object-oriented approach is that data should be presented as being encapsulated with the methods that access and modify them. That is, rather than simply viewing data as a collection of bytes and addresses, we think of data objects as instances of an ***abstract data type*** (***ADT***), which includes a repertoire of methods for performing operations on data objects of this type. Likewise, object-oriented solutions are often organized utilizing common ***design patterns***, which facilitate software reuse and robustness. Thus, we present each data structure using ADTs and their respective implementations and we introduce important design patterns as means to organize those implementations into classes, methods, and objects.

For each ADT presented in this book, we provide an associated Java interface. Also, concrete data structures realizing the ADTs are discussed and we often give concrete Java classes implementing these interfaces. We also give Java implementations of fundamental algorithms, such as sorting and graph searching. Moreover, in addition to providing techniques for using data structures to implement ADTs, we also give sample applications of data structures, such as in HTML tag matching and a simple system to maintain a photo album. Due to space limitations, however, we sometimes show only code fragments of some implementations in this book and make additional source code available on the companion web site. The Java code implementing fundamental data structures in this book is organized into a single Java package, net.datastructures, which forms a coherent library of data structures and algorithms in Java specifically designed for educational purposes in a way that is complementary with the Java Collections Framework. The net.datastructures library is not required, however, to get full use from this book.

Online Resources

This book is accompanied by an extensive accompanying set of online resources, which can be found at the following web site:

www.wiley.com/go/global/goodrich

Students are encouraged to use this site along with the book, to help with exercises and increase understanding of the subject. Instructors are likewise welcome to use the site to help plan, organize, and present their course materials. Included on this Web site is a collection of educational aids that augment the topics of this book, for both students and instructors. Because of their added value, some of these online resources are password protected.

For the Student

For all readers, and especially for students, we include the following resources:

- All the Java source code presented in this book.
- PDF handouts of Powerpoint slides (four-per-page) provided to instructors.
- A database of hints to ***all*** exercises, indexed by problem number.
- An online study guide, which includes solutions to selected exercises.

The hints should be of considerable use to anyone needing a little help getting started on certain exercises, and the solutions should help anyone wishing to see completed exercises. Students who have purchased a new copy of this book will get password access to the hints and other password-protected online resources at no extra charge. Other readers can purchase password access for a nominal fee.

For the Instructor

For instructors using this book, we include the following additional teaching aids:

- Solutions to over two hundred of the book's exercises.
- A database of additional exercises, suitable for quizzes and exams.
- The complete net.datastructures package.
- Additional Java source code.
- Slides in Powerpoint and PDF (one-per-page) format.
- Self-contained special-topic supplements, including discussions on convex hulls, range trees, and orthogonal segment intersection.
- Ready-to-use, turn-key projects, complete with supporting Java code for graphical-user interfaces (GUIs), so that students can concentrate on data structure design, implementation, and usage, rather than GUI programming.

The slides are fully editable, so as to allow an instructor using this book full freedom in customizing his or her presentations. All the online resources are provided at no extra charge to any instructor adopting this book for his or her course.

A Resource for Teaching Data Structures and Algorithms

This book contains many Java-code and pseudo-code fragments, and hundreds of exercises, which are divided into roughly 40% reinforcement exercises, 40% creativity exercises, and 20% programming projects.

This book can be used for the CS2 course, as descirbed in the 1978 ACM Computer Science Curriculum, or in courses CS102 (I/O/B versions), CS103 (I/O/B versions), CS111 (A version), and/or CS112 (A/I/O/F/H versions), as described in the IEEE/ACM 2001 Computing Curriculum, with instructional units as outlined in Table 0.1.

Instructional Unit	**Relevant Material**
PL1. Overview of Programming Languages	Chapters 1 & 2
PL2. Virtual Machines	Sections 14.1.1, 14.1.2, & 14.1.3
PL3. Introduction to Language Translation	Section 1.9
PL4. Declarations and Types	Sections 1.1, 2.4, & 2.5
PL5. Abstraction Mechanisms	Sections 2.4, 5.1, 5.2, 5.3, 6.1.1, 6.2, 6.4, 6.3, 7.1, 7.3.1, 8.1, 9.1, 9.5, 11.4, & 13.1
PL6. Object-Oriented Programming	Chapters 1 & 2 and Sections 6.2.2, 6.3, 7.3.7, 8.1.2, & 13.3.1
PF1. Fundamental Programming Constructs	Chapters 1 & 2
PF2. Algorithms and Problem-Solving	Sections 1.9 & 4.2
PF3. Fundamental Data Structures	Sections 3.1, 5.1–3.2, 5.3, , 6.1–6.4, 7.1, 7.3, 8.1, 8.3, 9.1–9.4, 10.1, & 13.1
PF4. Recursion	Section 3.5
SE1. Software Design	Chapter 2 and Sections 6.2.2, 6.3, 7.3.7, 8.1.2, & 13.3.1
SE2. Using APIs	Sections 2.4, 5.1, 5.2, 5.3, 6.1.1, 6.2, 6.4, 6.3, 7.1, 7.3.1, 8.1, 9.1, 9.5, 11.4, & 13.1
AL1. Basic Algorithmic Analysis	Chapter 4
AL2. Algorithmic Strategies	Sections 11.1.1, 11.5.1, 12.3.1, 12.4.2, & 12.2
AL3. Fundamental Computing Algorithms	Sections 8.1.4, 8.2.2, 8.3.5, 9.2, & 9.3.1, and Chapters 11, 12, & 13
DS1. Functions, Relations, and Sets	Sections 4.1, 8.1, & 11.4
DS3. Proof Techniques	Sections 4.3, 6.1.4, 7.3.3, 8.3, 10.2, 10.3, 10.4, 10.5, 11.2.1, 11.3.1, 11.4.3, 13.1, 13.3.1, 13.4, & 13.5
DS4. Basics of Counting	Sections 2.2.3 & 11.1.5
DS5. Graphs and Trees	Chapters 7, 8, 10, & 13
DS6. Discrete Probability	Appendix A and Sections 9.2.2, 9.4.2, 11.2.1, & 11.5

Table 0.1: Material for Units in the IEEE/ACM 2001 Computing Curriculum.

Contents and Organization

The chapters for this course are organized to provide a pedagogical path that starts with the basics of Java programming and object-oriented design. We provide an early discussion of concrete structures, like arrays and linked lists, so as to provide a concrete footing to build upon when constructing other data structures. We then add foundational techniques like recursion and algorithm analysis, and, in the main portion of the book, we present fundamental data structures and algorithms, concluding with a discussion of memory management (that is, the architectural underpinnings of data structures). Specifically, the chapters for this book are organized as follows:

1. **Java Programming Basics**
2. **Object-Oriented Design**
3. **Arrays, Linked Lists, and Recursion**
4. **Mathematical Foundations**
5. **Stacks and Queues**
6. **List Abstractions**
7. **Tree Structures**
8. **Priority Queues**
9. **Maps and Dictionaries**
10. **Search Tree Structures**
11. **Sorting and Selection**
12. **Text Processing**
13. **Graphs**
14. **Memory**

A. **Useful Mathematical Facts**

A more detailed listing of the contents of this book can be found in the table of contents.

Prerequisites

We have written this book assuming that the reader comes to it with certain knowledge. We assume that the reader is at least vaguely familiar with a high-level programming language, such as C, C++, Python, or Java, and that he or she understands the main constructs from such a high-level language, including:

- Variables and expressions.
- Methods (also known as functions or procedures).
- Decision structures (such as if-statements and switch-statements).
- Iteration structures (for-loops and while-loops).

For readers who are familiar with these concepts, but not with how they are expressed in Java, we provide a primer on the Java language in Chapter 1. Still, this book is primarily a data structures book, not a Java book; hence, it does not provide a comprehensive treatment of Java. Nevertheless, we do not assume that the reader is necessarily familiar with object-oriented design or with linked structures, such as linked lists, for these topics are covered in the core chapters of this book.

In terms of mathematical background, we assume the reader is somewhat familiar with topics from high-school mathematics. Even so, in Chapter 4, we discuss the seven most-important functions for algorithm analysis. In fact, sections that use something other than one of these seven functions are considered optional, and are indicated with a star (★). We give a summary of other useful mathematical facts, including elementary probability, in Appendix A.

About the Authors

Professors Goodrich and Tamassia are well-recognized researchers in algorithms and data structures, having published many papers in this field, with applications to Internet computing, information visualization, computer security, and geometric computing. They have served as principal investigators in several joint projects sponsored by the National Science Foundation, the Army Research Office, the Office of Naval Research, and the Defense Advanced Research Projects Agency. They are also active in educational technology research.

Michael Goodrich received his Ph.D. in Computer Science from Purdue University in 1987. He is currently a Chancellor's Professor in the Department of Computer Science at University of California, Irvine. Previously, he was a professor at Johns Hopkins University. He is an editor for a number of journals in computer science theory, computational geometry, and graph algorithms. He is an ACM Distinguished Scientist, a Fellow of the American Association for the Advancement of Science (AAAS), a Fulbright Scholar, and a Fellow of the IEEE. He is a recipient of the IEEE Computer Society Technical Achievement Award, the ACM Recognition of Service Award, and the Pond Award for Excellence in Undergraduate Teaching.

Roberto Tamassia received his Ph.D. in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign in 1988. He is the Plastech Professor of Computer Science and the Chair of the Department of Computer Science at Brown University. He is also the Director of Brown's Center for Geometric Computing. His research interests include information security, cryptography, analysis, design, and implementation of algorithms, graph drawing and computational geometry. He is an IEEE Fellow and a recipient of the Technical Achievement Award from the IEEE Computer Society, for pioneering the field of graph drawing. He is an editor of several journals in geometric and graph algorithms. He previously served on the editorial board of *IEEE Transactions on Computers*.

In addition to their research accomplishments, the authors also have extensive experience in the classroom. For example, Dr. Goodrich has taught data structures and algorithms courses, including Data Structures as a freshman-sophomore level course and Introduction to Algorithms as an upper level course. He has earned several teaching awards in this capacity. His teaching style is to involve the students in lively interactive classroom sessions that bring out the intuition and insights behind data structuring and algorithmic techniques. Dr. Tamassia has taught Data Structures and Algorithms as an introductory freshman-level course since 1988. One thing that has set his teaching style apart is his effective use of interactive hypermedia presentations integrated with the Web.

Acknowledgments

There are a number of individuals who have made contributions to this book.

We are grateful to all our research collaborators and teaching assistants, who provided feedback on early drafts of chapters and have helped us in developing exercises, software, and algorithm animation systems. In particular, we would like to thank Jeff Achter, Vesselin Arnaudov, James Baker, Ryan Baker, Benjamin Boer, Mike Boilen, Devin Borland, Lubomir Bourdev, Stina Bridgeman, Bryan Cantrill, Yi-Jen Chiang, Robert Cohen, David Ellis, David Emory, Jody Fanto, Ben Finkel, Peter Fröhlich, Ashim Garg, Natasha Gelfand, Mark Handy, Michael Horn, Greg Howard, Benoît Hudson, Jovanna Ignatowicz, Seth Padowitz, Babis Papamanthou, James Piechota, Dan Polivy, Seth Proctor, Susannah Raub, Haru Sakai, Andy Schwerin, Michael Shapiro, Mike Shim, Michael Shin, Galina Shubina, Amy Simpson, Christian Straub, Ye Sun, Nikos Triandopoulos, Luca Vismara, Danfeng Yao, Jason Ye, and Eric Zamore. Lubomir Bourdev, Mike Demmer, Mark Handy, Michael Horn, and Scott Speigler developed a basic Java tutorial, which ultimately led to Chapter 1, Java Primer. Special thanks go to Eric Zamore, who contributed to the development of the Java code examples in this book and to the initial design, implementation, and testing of the net.datastructures library of data structures and algorithms in Java. We are also grateful to Vesselin Arnaudov and Mike Shim for