CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

Data Structures and Algorithms in Java

Learn with confidence with this hands-on undergraduate textbook for CS2 courses. Active learning and real-world projects underpin each chapter, briefly reviewing programming fundamentals then progressing to core data structures and algorithms topics, including recursion, lists, stacks, trees, graphs, sorting, and complexity analysis. Creative projects and applications put theoretical concepts into practice, helping students master the fundamentals. Dedicated project chapters supply further programming practice using real-world, interdisciplinary problems which students can showcase in their own online portfolios. Example Interview Questions sections prepare students for job applications. The pedagogy supports self-directed and skills-based learning with over 250 “Try It Yourself” boxes, many with solutions provided, and over 500 progressively challenging end-of-chapter questions. Written in a clear and engaging style, this textbook is a complete resource for teaching the fundamental skills that today’s students need. Instructor resources are available online, including a test bank, solutions manual, and sample code.

Dan S. Myers is Associate Professor of Computer Science at Rollins College in Winter Park, Florida, where he has taught Data Structures and Algorithms for 10 years.

© in this web service Cambridge University Press & Assessment www.cambridge.org

"An intuitive and engaging introduction to foundational content; a terrific source for anyone looking to dive deeper into computer science!"

Remzi Arpaci-Dusseau, University of Wisconsin–Madison

"This book teaches data structures with an active learning approach. Students will first learn the constructs, then get hands-on experience by applying what they learned in real projects. In addition, the try-it-yourself sections are really helpful as simple brain teasers to ensure that students got the point. Students will be more engaged by learning this way."

Mohammed Farghally, Virginia Tech

"*Data Structures and Algorithms in Java* by Dan Myers is a carefully crafted book for second-level CS students, right after learning the rudiments of at least one programming language (be it Python, C/C++, or Java) and doing basic programming. To fill the gap from simple programming to serious CS, art, and natural science applications programming, a fundamental course is needed. Often named 'Data Structures and Algorithms,' this course helps students to learn beyond basic data structures such as numbers and arrays, and to move to the world of lists, stacks, hash tables, trees, queues, heaps, and graphs. There are also the fundamental algorithms, such as searching, sorting, hashing, and binary tree traversal algorithms, which are the building blocks of more complicated algorithms needed to develop applications programs. Here is an excellent book that skillfully brings students to serious programming, by providing meaningful explanations, beautiful figures, and short but to-the-point code fragments. The exercises at the end of every chapter are very helpful. Fascinating projects every few chapters make even an experienced programmer want to try, such as particle effects, generative art, fractals, logic puzzles, and specialized search engines. I strongly urge the CS instructor to consider this book before making commitment to their often-used textbook."

Çetin Kaya Koç, University of California, Santa Barbara

"With beautifully accessible prose, Professor Myers opens the door wide to a field known for its formidable jargon and subtle technical detail. This text offers scaffolded, incremental learning as well as innovative and engaging projects, all well designed and supported by a wealth of diverse exercises. His rich, interdisciplinary contexts also demonstrate the centrality of algorithms in many human endeavors."

Trevor Kearns, Greenfield Community College

"I wholeheartedly endorse *Data Structures and Algorithms in Java*. Its unique project-based approach effectively merges theory with practical application, making it an invaluable resource for undergraduate students. The book's emphasis on hands-on learning through Java programming projects elevates the understanding and deepens the knowledge of the applicability of data structures."

Tajmilur Rahman Md, Gannon University

"*Data Structures and Algorithms in Java* is a must-read for aspiring programmers. The book bridges theory and practice seamlessly using real-world projects illustrating key concepts. The book covers a wide array of data structures from the simple stack to the more complex graph. The author's beginner-friendly approach makes it an excellent companion for self-learners and computer science students alike."

Lilian Blot, University of York

© in this web service Cambridge University Press & Assessment www.cambridge.org

"The examples of projects offer a fun yet insightful way to boost students' creativity and enhance their problem-solving skills following the 'Try It Yourself' approach, ensuring the understanding of each concept."

Jeong Yang, Texas A&M University–San Antonio

"Dr. Myers carefully balances teaching students the Java they need after an introductory programming course with building more complex projects that students can use in their job portfolios. It covers the fundamental data structures and algorithms students need to know while immediately applying those ideas to longer-form projects in areas of student interest."

Brian Patterson, Oglethorpe University

"A stand-out feature of this book is its project-based approach, which bridges theoretical content with real-world applications. This method not only enhances students' understanding of data structures and algorithms, but also prepares them for technical roles through a variety of creative projects."

Bailin Deng, Cardiff University

"This book goes beyond the typical scope of a data structures and algorithms textbook. The carefully described projects are excellent examples of how to structure large programs."

Christian Trefftz, Grand Valley State University

"I have developed many techniques, tools, and examples to facilitate learning in my data structures classes. Dan Myers' textbook incorporates many similar strategies – it is conversational, establishes a foundation for new concepts and techniques then builds upon the foundation. It also incorporates numerous examples and problems, including historical references, making the material more interesting. This is a refreshing take on a potentially dry subject."

Dave Rosenberg, Wentworth Institute of Technology

"The project-based approach in the book highlights the real-world applications of topics presented, and facilitates greater understanding and clarity. The interview questions included underscore the importance of the concepts beyond the classroom. Several sections introduce new topics in simple terms along with relevant fun facts which draw the reader in and make the learning process an engaging one. Highly recommended!"

Omofolakunmi Olagbemi, Hope College

"Professor Myers' project-based approach not only motivates students but also offers a practical framework for applying complex concepts effectively. The projects in this text provide context for decision-making and enhance understanding through practical applications. Each project builds a solid foundation fostering thoughtful engagement with challenging material."

Michael Penta, Northern Essex Community College

© in this web service Cambridge University Press & Assessment

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

Data Structures and Algorithms in Java

A Project-Based Approach

Dan S. Myers

Rollins College, Florida

CAMBRIDGE
UNIVERSITY PRESS

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

CAMBRIDGE
UNIVERSITY PRESS

Shaftesbury Road, Cambridge CB2 8EA, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of Cambridge University Press & Assessment, a department of the University of Cambridge.

We share the University's mission to contribute to society through the pursuit of education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/highereducation/isbn/9781009260336

DOI: 10.1017/9781009260350

© Cambridge University Press & Assessment 2025

This publication is in copyright. Subject to statutory exception and to the provisions of relevant collective licensing agreements, no reproduction of any part may take place without the written permission of Cambridge University Press & Assessment.

When citing this work, please include a reference to the DOI 10.1017/9781009260350

First published 2025

Printed in Mexico by Litográfica Ingramex, S.A. de C.V, 2025

A catalogue record for this publication is available from the British Library

A Cataloging-in-Publication data record for this book is available from the Library of Congress

ISBN 978-1-009-26033-6 Hardback

Additional resources for this publication at www.cambridge.org/myers

Cambridge University Press & Assessment has no responsibility for the persistence or accuracy of URLs for external or third-party internet websites referred to in this publication and does not guarantee that any content on such websites is, or will remain, accurate or appropriate.

This publication is not sponsored, affiliated, or endorsed by Oracle or Java.

© in this web service Cambridge University Press & Assessment www.cambridge.org

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

For Chelsea, who helped make it possible

© in this web service Cambridge University Press & Assessment www.cambridge.org

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

Contents

Preface — *page* xxiii

Algorithms+Data Structures = Programs — 1

Overview — 1
Learning Objectives — 1
Goals — 1
A Brief History of Algorithms — 2
Why Do Data Structures and Algorithms Matter to Real-World Programmers? — 4
Learning from This Book — 5
Summary — 6
Exercises — 6
Notes and Further Reading — 7

1 Java Fundamentals — 8

Introduction — 8
Learning Objectives — 8
1.1 Hello, Java! — 9
1.1.1 The First Program — 9
1.1.2 How Java Programs Execute — 11
1.2 Variables and Data Types — 11
1.2.1 Declaring Variables — 12
1.2.2 Variable Names — 12
1.2.3 Primitive and Object Types — 12
1.2.4 Arithmetic — 14
1.2.5 Casting — 16

© in this web service Cambridge University Press & Assessment — www.cambridge.org

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

1.3 Working with Objects 16
1.3.1 Reading User Input with `Scanner` 16
1.3.2 Constants and Formatted Printing 19
1.3.3 Run-Time Exceptions 20
1.3.4 Calculations with `Math` 20
1.4 Conditional Execution 22
1.4.1 Relational and Logical Operators 22
1.4.2 The `if-else` Statement 23
1.4.3 Cho-han 25
1.4.4 Comparing Strings and Other Objects 28
1.5 Loops 30
1.5.1 The `while` Loop 30
1.5.2 The `for` Loop 31
1.5.3 Monte Carlo Simulation 33
1.6 Static Methods 35
1.6.1 Writing Methods 35
1.6.2 Automatic Documentation with JavaDoc 37
1.7 Project: Substitution Ciphers 38
1.7.1 The Caesar Cipher 39
1.7.2 The Vigenère Cipher 42
1.7.3 The Polybius Square 44
1.7.4 The Russian Nihilist Cipher 47
Summary 49
Exercises 50
Notes and Further Reading 55

2 Object-Oriented Programming 57

Introduction 57
Learning Objectives 57
2.1 The Concept of Object-Oriented Programming 57
2.1.1 Classes and Objects 58
2.1.2 Encapsulation 58
2.1.3 Advantages and Disadvantages 59
2.2 Developing a Class Step by Step 60
2.2.1 The Simplest Class: A Collection of Related Variables 60
2.2.2 Custom Constructors 62
2.2.3 The `toString` Method 64

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

2.2.4 Access Modifiers 65
2.2.5 Accessor and Mutator Methods 66
2.2.6 Throwing Exceptions 68
2.2.7 More on the `static` Keyword 69
2.2.8 Null Objects 71
2.3 The `ArrayList` Class 71
2.3.1 Creating an `ArrayList` 72
2.3.2 Useful Methods 72
2.3.3 Working with Primitive Types 74
2.4 Project: Deck of Cards 74
2.4.1 Enumerated Types 74
2.4.2 The `Card` Class 76
2.4.3 The `Deck` Class 78
2.4.4 Hi-Lo 82
Summary 84
Exercises 84
Notes and Further Reading 90

3 Project: Mindstorms 91

Introduction 91
Learning Objectives 91
3.1 Introduction to Java Graphics 91
3.1.1 Drawing Shapes 92
3.1.2 Graphics Methods 93
3.1.3 Colors 95
3.2 Project: Do-It-Yourself Turtle Graphics 96
3.2.1 Basic Commands 96
3.2.2 Setup 97
3.2.3 Instance Variables and Constructor 101
3.2.4 Moving and Angles 102
3.2.5 Drawing 104
3.2.6 Adding New Commands 105
3.2.7 Complex Shapes 105
Summary 109
Extensions 110
Notes and Further Reading 111

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

4 Arrays 112

Introduction 112
Learning Objectives 112
4.1 One-Dimensional Arrays 112
4.1.1 Creating and Accessing Arrays 113
4.1.2 Looping over Arrays 114
4.1.3 Modifying an Array in a Method 116
4.2 Multidimensional Arrays 118
4.2.1 Creating Multidimensional Arrays 119
4.2.2 The "Array of Arrays" Model and Ragged Arrays 120
4.2.3 Looping over Two-Dimensional Arrays 121
4.2.4 Ancient Algorithms: Magic Squares 123
4.3 Project: Cellular Automata 126
4.3.1 One-Dimensional Elementary Cellular Automata 126
4.3.2 Visualizing Elementary Cellular Automata 128
4.3.3 Conway's Life 132
4.3.4 Animating Life 135
4.4 Example Interview Questions Using Arrays 143
4.4.1 Removing Elements 143
4.4.2 Rotate an Array 144
4.4.3 Maximum Subarray Sum 145
Summary 146
Exercises 147
Notes and Further Reading 153

5 Searching and an Introduction to Algorithm Analysis 154

Introduction 154
Learning Objectives 154
5.1 Linear Search and Algorithm Performance 154
5.2 Binary Search 156
5.3 Project: Experimental Performance Analysis 159
5.3.1 Best, Worst, and Average Case Performance 159
5.3.2 Measuring Execution Time 159
5.4 The Growth of Functions 162
5.5 Big-O Notation 163
5.6 Applying Big-O Analysis to Programs 165

© in this web service Cambridge University Press & Assessment www.cambridge.org

5.7 Detailed Analysis of Binary Search 167
5.7.1 Correctness Using a Loop Invariant 167
5.7.2 Complexity 169
5.8 Common Complexities 171
5.9 Other Notations: Ω and Θ 171
5.9.1 Ω Notation for Lower Bounds 172
5.9.2 Θ Notation for Exact Bounds 172
5.10 Limitations of Asymptotic Analysis 173
Summary 173
Exercises 174
Notes and Further Reading 177

6 Lists 178

Introduction 178
Learning Objectives 178
6.1 Abstract Data Types 178
6.1.1 Separating Behaviors from Implementation 179
6.1.2 The List Abstract Data Type 179
6.2 Array Lists 180
6.2.1 Array List Structure 181
6.2.2 Implementing an `ArrayList` Class 182
6.2.3 Amortized Analysis of the Append Operation 185
6.2.4 Removing and Inserting 187
6.3 Linked Lists 189
6.3.1 Linked List Implementation 189
6.3.2 Adding to the Front 190
6.3.3 Getting an Item 191
6.3.4 Inserting at an Arbitrary Location 192
6.3.5 Removing 195
6.3.6 Doubly Linked and Circular Lists 197
6.4 Project: *Snake* 201
6.4.1 The `Snake` class 202
6.4.2 Class Members and Constructor 204
6.4.3 Keyboard Input with the `KeyListener` Interface 205
6.4.4 The `paint` Method 206
6.4.5 The Main Game Loop 207
6.5 Example Interview Questions Using Lists 210

© in this web service Cambridge University Press & Assessment

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

6.5.1 Comparing Linked and Array Lists 210
6.5.2 Reversing a List 210
6.5.3 The Tortoise and Hare Algorithm 211
Summary 212
Exercises 213
Notes and Further Reading 215

7 Project: Particle Effects 216

Introduction 216
Learning Objectives 216
7.1 Falling Sand 217
7.1.1 Listening for Mouse Input 219
7.1.2 Falling Physics 220
7.1.3 Better Performance with Double Buffering 223
7.1.4 Adding Colors 224
7.2 Boids 225
7.2.1 Flocking with Particles 226
7.2.2 Starting Code 227
7.2.3 The `Boid` Class 230
7.2.4 Implementing the Flocking Rules 231
Summary 233
Extensions 233
Notes and Further Reading 239

8 Recursion 240

Introduction 240
Learning Objectives 240
8.1 Writing Recursive Methods 240
8.1.1 Mathematical Functions 241
8.1.2 Recursive Binary Search 243
8.1.3 How Recursive Methods Execute 244
8.1.4 Stack Overflow Errors 247
8.1.5 Recursion vs. Iteration 247
8.2 Analyzing Recursive Algorithms 249
8.2.1 Recurrence Relations 249

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

8.2.2 Proving Correctness Using Mathematical Induction 250
8.3 Example: Drawing Recursive Trees 252
8.3.1 Symmetric Trees 252
8.3.2 Adjusting Branch Thickness 254
8.3.3 Asymmetric and Randomized Trees 255
8.4 Example Interview Questions Using Recursion 258
8.4.1 Reverse a List 258
8.4.2 Decimal to Binary Conversion 259
8.4.3 Moving on a Chessboard 260
Summary 261
Exercises 262
Notes and Further Reading 265

9 Project: Generative Art and Fractals 266

Introduction 266
Learning Objectives 266
9.1 Homage to *Homage to the Square* 267
9.2 In the Style of Piet Mondrian 271
9.3 Fractals 276
9.3.1 Sierpiński's Carpet 277
9.3.2 Vicsek's Cross 280
9.4 The Mandelbrot Set 283
9.4.1 Definition 283
9.4.2 Visualizing the Set 285
9.4.3 Zooming in on a Point 290
9.4.4 Adding Colors 292
Extensions 297
Notes and Further Reading 299

10 Sorting 300

Introduction 300
Learning Objectives 300
10.1 Sorting Preliminaries 300
10.1.1 Comparing and Ordering Items 300
10.1.2 Standard Libraries Exist, So Why Are You Reading this Book? 301

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

10.2	Quadratic Sorting Algorithms	302
	10.2.1 Selection Sort	302
	10.2.2 Insertion Sort	304
10.3	Quicksort	307
	10.3.1 Sorting by Divide-and-Conquer	307
	10.3.2 Partitioning	309
	10.3.3 Correctness of `partition`	312
	10.3.4 Intuitive Analysis	313
	10.3.5 Worst-Case Analysis	315
	10.3.6 Average-Case Analysis with Random Pivots	315
	10.3.7 Improving Quicksort	317
	10.3.8 Java's `Arrays.sort`	318
10.4	Merge Sort	318
	10.4.1 Implementation	318
	10.4.2 Analysis	323
	10.4.3 Merge Sort in Practice: Python's Timsort	323
10.5	Radix Sorting	324
10.6	Example Interview Questions Using Sorting	327
	10.6.1 Problems Where You Sort and then Do Something	327
	10.6.2 Finding the Median with Quickselect	328
	10.6.3 National Flag Problems	329
	Summary	330
	Exercises	331
	Notes and Further Reading	334

11 Stacks 336

	Introduction	336
	Learning Objectives	336
11.1	The Stack Abstract Data Type	336
	11.1.1 Stack Methods	337
	11.1.2 Example: Validating HTML	337
	11.1.3 Implementations	342
11.2	Stack-Based Arithmetic	343
	11.2.1 Prefix and Postfix Notations	344
	11.2.2 Evaluating Postfix Expressions with a Stack	345
	11.2.3 Converting Infix Expressions to Postfix	347
11.3	Project: Tiny Web Browser	349

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

11.3.1 Viewing an HTML Page 350
11.3.2 Listening for Link Events 354
11.3.3 Implementing the Address Bar and Back Button 354
11.4 Example Interview Questions Using Stacks 356
11.4.1 Removals to Make Matching Parentheses 356
11.4.2 Constant Time Minimum Operation 358
Summary 359
Exercises 359
Notes and Further Reading 361

12 Project: Logic Puzzles 362

Introduction 362
Learning Objectives 362
12.1 Solving Puzzles by Backtracking 362
12.1.1 Latin Squares 363
12.1.2 The Backtracking Search Strategy 364
12.1.3 Implementation 367
12.1.4 Sudoku 369
12.2 Who Owns the Fish? 370
12.2.1 Representing the Solution 371
12.2.2 Search Procedure 374
12.2.3 Coding the Constraints 375
Summary 377
Extensions 378
Notes and Further Reading 381

13 Queues and Buffers 382

Introduction 382
Learning Objectives 382
13.1 The Queue Abstract Data Type 382
13.1.1 Queue and Deque Methods 383
13.1.2 Java's `Queue` and `Deque` Interfaces 383
13.2 Queue and Deque Implementations 384
13.2.1 Using `LinkedList` 384
13.2.2 Array-Based Circular Buffers 384
13.2.3 A `CircularBuffer` Class 386

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

13.3 Application: Flood-Filling 388
13.3.1 The Breadth-First Flooding Algorithm 389
13.3.2 Flood-Filling in a Matrix Using `ArrayDeque` 390
13.4 Example Interview Questions Using Queues 392
13.4.1 Count the Number of Islands 392
13.4.2 Fairy Chess Pieces 393
Summary 395
Exercises 395
Notes and Further Reading 398

14 Hashing 399

Introduction 399
Learning Objectives 399
14.1 Hash Functions 399
14.1.1 Java's `hashCode` 400
14.1.2 A Real-World Hash Function: MurmurHash3 401
14.1.3 Properties of Good Hash Functions 403
14.1.4 Cryptographic Hash Functions 404
14.1.5 Application: Message Authentication 405
14.2 Project: Password Cracking 406
14.2.1 Shadow Password Files 406
14.2.2 Cryptographic Hashing with `MessageDigest` 407
14.2.3 Brute-Force Cracking 408
14.2.4 Dictionary Attacks 411
14.3 Application: Proof-of-Work and Bitcoin 413
14.3.1 Fighting Email Spam with Hashing 413
14.3.2 Proof-of-Work in the Bitcoin Blockchain 414
Summary 415
Exercises 415
Notes and Further Reading 417

15 Hash Tables 419

Introduction 419
Learning Objectives 419
15.1 The Map Abstract Data Type 419
15.1.1 Map Methods 420

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

15.1.2 Java's `HashMap` 421
15.2 Implementing Hash Tables 423
15.2.1 Direct Addressing 423
15.2.2 Chained Hashing 425
15.2.3 Analyzing Chaining 428
15.2.4 Open Addressing with Probing 430
15.2.5 Cuckoo Hashing 432
15.3 Example Interview Questions Using Hash Tables 433
15.3.1 Representing a Set 433
15.3.2 Find the Pairs with a Given Sum 434
15.3.3 Finding Anagrams 434
Summary 435
Exercises 436
Notes and Further Reading 438

16 Project: Ye Olde Shakespearean Search Engine 439

Introduction 439
Learning Objectives 439
16.1 Building a Search Index 439
16.2 Implementation 441
16.2.1 Getting the Text 442
16.2.2 The `Location` Class 443
16.2.3 The `Driver` Class 444
16.2.4 The `Index` Class 445
16.2.5 Adding a Script to the Index 448
16.2.6 Constructing the Index 450
16.2.7 Putting It All Together 451
16.3 Extending to Web Search 451
Summary 452
Extensions 452
Notes and Further Reading 454

17 Binary Trees 455

Introduction 455
Learning Objectives 455

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

17.1 Representing Hierarchical Data 455
17.1.1 Dendrology 455
17.1.2 Binary Trees 457
17.1.3 Implementation 458
17.2 Tree Traversals 460
17.3 Application: Syntax Trees 464
17.4 Binary Search Trees 468
17.4.1 Properties 468
17.4.2 Searching 469
17.4.3 Insertion 471
17.4.4 Deletion 473
17.4.5 Analysis 479
17.5 Example Interview Questions Using Trees 480
17.5.1 Mirror a Binary Tree 480
17.5.2 Checking If a Tree Is Height Balanced 482
Summary 483
Exercises 483
Notes and Further Reading 486

18 Self-Balancing Search Trees 488

Introduction 488
Learning Objectives 488
18.1 2-3-4 Trees 488
18.1.1 Nonbinary search trees 489
18.1.2 Insertion 490
18.1.3 Deletion 491
18.1.4 Going Wider: B-trees 495
18.2 Red–Black Trees 497
18.2.1 Red–Black Ordering Rules 497
18.2.2 Connection to 2-3-4 Trees 499
18.2.3 Bounds on the Height of the Tree 500
18.2.4 Insertion 500
18.2.5 Deletion 506
Summary 513
Exercises 513
Notes and Further Reading 516

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

19 Heaps and Priority Queues 517

Introduction 517
Learning Objectives 517
19.1 Tree-Based Heaps 517
19.1.1 Definition 518
19.1.2 Inserting into a Heap 518
19.1.3 Removing the Root Element 520
19.1.4 Initializing a Heap 521
19.1.5 Analysis of Bottom-Up Initialization 522
19.2 Implementing a Heap in an Array 523
19.2.1 Parent–Child Relationships 523
19.2.2 A `Heap` Class 524
19.3 Application: Heapsort 529
19.4 Example Interview Questions Using Heaps 532
19.4.1 Verify If a Binary Tree Is a Heap 532
19.4.2 Merging Sorted Lists 533
19.4.3 Find the Top-*k* Largest Elements in an Array 534
Summary 534
Exercises 535
Notes and Further Reading 536

20 Graph Algorithms 538

Introduction 538
Learning Objectives 538
20.1 Graph Basics 538
20.1.1 Some Important Graph Terms 539
20.1.2 Representing Graphs 541
20.1.3 Graphs (or the Lack Thereof) in Java 543
20.2 Traversals 545
20.2.1 Breadth-First Traversal 545
20.2.2 Depth-First Traversal 547
20.2.3 Application: Garbage Collection 549
20.3 Shortest Paths 552
20.3.1 Dijkstra's Algorithm 552
20.3.2 Analyzing Dijkstra's Algorithm 555
20.4 Example Interview Questions Using Graphs 559

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

20.4.1 Snakes and Ladders 559
20.4.2 Finding Cycles 560
Summary 562
Exercises 562
Notes and Further Reading 565

21 Project: Graph-Based Recommendation Engine 566

Introduction 566
Learning Objectives 566
21.1 Bipartite Graphs 566
21.1.1 Definition 567
21.1.2 Testing If a Graph Is Bipartite 568
21.2 Building the Recommendation System 570
21.2.1 The Object Graph and Random Walks 570
21.2.2 Implementation 572
Summary 577
Extensions 577
Notes and Further Reading 580

22 Project: Twisty Little Passages 581

Introduction 581
Learning Objectives 581
22.1 Minimum Spanning Trees 581
22.1.1 Kruskal's Algorithm 582
22.1.2 Prim's Algorithm 586
22.1.3 Correctness 590
22.2 Maze Generation 591
22.2.1 Perfect Mazes and Spanning Trees 591
22.2.2 Implementation 592
Summary 600
Extensions 600
Notes and Further Reading 602

References 604
Index 610

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

Preface

Teaching our students how to understand and use algorithms is the heart of the computer science curriculum. The relevance of algorithms to the modern world may seem obvious, but the unfortunate reality is that many of our students will finish their data structures and algorithms coursework with no idea how the material applies to actual software development. This book fills a gap in the data structures and algorithms literature by offering students and instructors an engaging and carefully curated text that presents all of the fundamental material of the data structures and algorithms course while also emphasizing meaningful applications in CS, the arts, and the natural sciences. Every content-based chapter is paired with at least one significant project or application illustrating how to use the chapter's theory in a creative context.

Motivation and Content

This book is aimed at the standard undergraduate "CS2" course that presents the core content of data structures and algorithms (sorting, lists, trees, complexity analysis, etc.) to students who have already completed introductory programming. It emerged as a response to real challenges I've encountered while teaching CS2 over the past 10 years, which I think will resonate with instructors at a broad range of institutions:

- **Review of CS1 material**. Many universities expect students to take CS2 as their second programming-focused course, typically after one semester covering the fundamentals of programming. Therefore, the CS2 course needs to begin with a review of fundamental concepts and a deeper presentation of object-oriented programming before introducing new material. This problem has become more acute in recent years as programs have shifted their introductory class to Python while keeping the data structures course in Java.
- **Developing students' project portfolios**. Job-market demand for CS graduates remains high, but competition for the best internships and full-time jobs is intense. Students are under pressure to develop a portfolio of significant projects that they can showcase in job and internship applications. Because students now seek internships earlier, there is often a need to include significant portfolio-ready projects earlier in the curriculum. The CS2 course is an ideal place to incorporate creative projects for early CS majors.
- **Relevance**. Many data structures resources, including the popular online practice sites, emphasize abstract puzzle-style problem solving. While solving algorithmic puzzles

© in this web service Cambridge University Press & Assessment www.cambridge.org

can certainly build students' skills, they don't do much to generate enthusiasm for the subject or show how data structures and algorithms connect to interesting applications. Professional developers often associate the subject solely with preparing for technical interviews.

The CS2 course, therefore, needs to do several things: build fundamental programming skills, teach theoretical content, prepare students for electives, and get students ready for job and internship applications, while also nurturing their growing engagement with the discipline. Over the coming years, we will likely see the current emphasis on student success and persistence in the CS1 course continue up to the rest of the computer science curriculum. In addition to a greater emphasis on active learning, we'll see a desire for relevance, interdisciplinary applications, and social awareness throughout the CS major. The content of this book was designed with those goals in mind. It features:

- coverage of fundamental data structures and algorithms topics, including recursion, lists, stacks, trees, graphs, sorting, and complexity analysis;
- project-based chapters that apply content in significant software projects, of the kind that could be major assignments or featured in a student's resume and portfolio;
- emphasis on interdisciplinary application areas;
- early review of object-oriented programming for recent CS1 students;
- active learning using reflective questions and solutions integrated throughout the text, informed by extensive undergraduate teaching experience; and
- an additional section in each theoretical chapter illustrating techniques that frequently show up in technical interview questions – working through these examples will help students engage with other technical interview resources.

In return, the book makes a careful trade-off to de-emphasize niche data structures, as well as limit coverage of advanced topics in algorithm design and analysis that are more likely to be featured in an upper-level course.

A Project-Based Approach

Every topic includes at least one significant Java programming project illustrating how the theoretical content can apply to interdisciplinary problems. Some of these examples are time-tested standards that will be familiar from previous books, but most are new. The goal of these projects is not only to give students programming practice, but to illustrate how data structures and algorithms are used to solve interesting problems. Examples include:

- a turtle graphics system using the Java Graphics API and object-oriented techniques (Chapter 3);
- particle-based graphic effects and artificial life simulations (Chapter 7);
- writing graphical applications to produce recursive generative art and fractal images, including the famous Mandelbrot set (Chapter 9);
- a small HTML viewer and web browser (Chapter 11);
- password cracking and hashing (Chapter 14);

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-26033-6 — Data Structures and Algorithms in Java
Dan S. Myers
Frontmatter
More Information

- a text-based search engine for a Shakespeare play (Chapter 16); and
- a recommendation system using random walks on a bipartite graph (Chapter 21).

The theoretical chapters feature smaller application sections that can be covered as part of a topical lecture. The larger projects are in stand-alone chapters and have been designed to be extensible, so that instructors can present the basic design of each project, work through the example code, and then assign modifications to the basic project as homework. The project chapters are written in a follow-along style that develops each application incrementally. Laboratory or small discussion sections are an ideal environment for presenting and coding the initial project solutions.

The projects and examples do not require any specific platform, development environment, or specialized software libraries. Any desktop or cloud-based editor that can compile and run Java is suitable for the course. All of the projects use straightforward object-oriented Java programming and the Java standard library; there are no projects that rely on custom libraries or elaborate prewritten code that exists outside of the text. After working through the examples and projects in this book, students will be comfortable writing moderate-sized Java programs and will be prepared for upper-level courses on object-oriented software design and algorithm analysis.

Student Background

This book is aimed at early-career computer science students with at least one semester of programming experience and some familiarity with object-oriented programming. It assumes that students have seen (but not yet mastered) the core concepts of structured programming – variables, data types, functions, control structures, I/O, etc. – typically covered in the standard "CS1" course, and that students are comfortable working with moderate-sized programs that combine those features into one application.

Students entering without previous Java experience should spend the first weeks of the class working through the material of Chapters 1 and 2, which introduce the basic elements of Java and object-oriented programming. Chapter 4 presents one-dimensional and two-dimensional arrays. After completing those chapters, students should be comfortable writing CS1-level programs in Java. Students who have already completed a Java course can move a little faster, but reviewing the core material on object-oriented programming and arrays is still recommended. This book focuses on using Java to implement data structures and algorithms topics, so it isn't intended to be a comprehensive guide to all of Java, or to object-oriented software design. For the most part, Java topics are introduced when they're required for a project.

Course Design

Instructors have a number of options when designing a class with this book. Covering every chapter is too much for a one-semester course, so you may prefer to focus on your favorite examples and projects. Additional resources for instructors, including full implementations of all projects and major examples, solutions to the end-of-chapter questions, a test bank, and teaching guide are available online. The text is organized as follows:

© in this web service Cambridge University Press & Assessment www.cambridge.org

- Chapters 1–4 present Java fundamentals. If students have had a previous course in Java, this can be covered efficiently. Chapter 3, the turtle graphics project, introduces the graphics framework that will be used throughout the first half of the book, so you should include it if you plan to use any of the projects from Chapters 4–9.
- Chapters 5–7 introduce algorithm analysis, Big-O notation, and lists. This is core content for any data structures course. Chapters 6 and 7 also demonstrate interfaces and graphical applications with user input.
- Chapter 8 presents recursion. If students have already seen standard recursive examples in an earlier course, you could cover this material quickly, in preparation for sorting algorithms in Chapter 10. Chapter 9 is a major project covering recursive art and the Mandelbrot fractal viewer.
- Chapters 10–17 cover the fundamental algorithms and data structures: sorting (10), stacks (11), backtracking search (12), queues (13), hashing (14), hash tables (15), and binary search trees (17). These will likely form the core of any CS2 course and you could realistically expect to spend 10 weeks on this material.
- Chapters 18–22 are additional material that may not fit into a one-semester course: self-balancing trees (18), heaps (19), and graphs (20–22).

The following table shows an example 14-week schedule for a course that starts with a review of Java and touches on all of the core content. This sequence covers roughly one chapter per week, with the vision that students will complete one guided project from the text (independently or in a weekly lab meeting), then complete assigned project extensions or a set of end-of-chapter questions for further practice.

Week	Topic	Chapter	Project
1	Java basics	1	Monte Carlo simulation
2	Java basics cont.	1	Substitution ciphers
3	Objects	2	Card games
4	Graphical applications	3	DIY turtle graphics
5	Arrays	4	Conway's Life
6	Algorithm analysis	5	Ch. 5 practice questions
7	Lists	6	*Snake* game or particle graphics (Ch. 7)
8	Recursion	8	Fractal art (Ch. 9)
9	Sorting	10	Ch. 10 practice questions
10	Stacks	11	Tiny web browser
11	Backtracking search	12	Logic puzzles
12	Hashing	14	Password cracker
13	Hash tables	15	Text search engine (Ch. 16)
14	Trees	17	Ch. 17 practice questions

This book has enough material for a two-course sequence. Many programs have made the choice to use a "bridge" course that sits between CS1 and CS2, with the goal of helping students transition between early programming and the full data structures course. With this model, the bridge course could cover Chapters 1–9, with their projects, at a moderate pace. The CS2 course would review analysis and recursion and then cover the remaining topics from Chapters 10–22.

© in this web service Cambridge University Press & Assessment www.cambridge.org

Self-Teaching

This book can also serve as a useful companion for working software developers who want to improve their data structures and algorithms knowledge. Many developers without a formal computer science background find it difficult to bridge the gap between early programming and upper-level theoretical texts on algorithm analysis intended for graduate students. This book was created to provide a smooth introduction to important theoretical topics and also serve as an antidote to the popular view that algorithms are only important for "whiteboard" technical interviews.

If you are using this book to teach yourself, simply start wherever you feel appropriate and work through the material at your own pace. If you have no or limited Java experience, you should definitely start with Chapters 1 and 2. Follow the directions for the example projects and, if time permits, add a few of the extensions. It isn't necessary, or even desirable, to complete every project or end-of-chapter question, so don't force yourself and allow yourself to move on if you become stuck. The sequence of topics and projects given above would also be appropriate for a self-taught developer. Completing that schedule will equip you with the core knowledge required for technical interviews and give a solid base for further study in more advanced books.

Responding to AI

ChatGPT was publicly released while I was halfway through writing this book. Although it remains to be seen how far large-language models will develop, it's fair to say that they're already powerful enough to have a disruptive effect on introductory CS courses. Our traditional model of assessment – leave class, write some small programs, and run them against test cases – fails in the face of automated tools that can solve those problems better than most undergraduates. As teachers, we have to strike a balance between remaining open to AI tools – which are going to be a key part of professional software development going forward – while still equipping our students with fundamental skills.

Although it wasn't originally planned with AI in mind, I believe the approach of this book is one part of that balance. The presentation here emphasizes *context*, rooting the discussion of data structures and algorithms in meaningful larger-scale projects and emphasizing connections across disciplines. Asking students to develop in a hands-on way, by building and extending the specific solutions given in the text, lessens their ability to rely only on an AI tool. Ultimately, though, nothing is safe. It seems likely that every piece of public code will eventually be absorbed into one language model or another, and clever students will always use every tool at their disposal. Now, at this moment, I see this book as one contribution to a conversation about how to continue evolving the curriculum and pedagogy of our discipline.

© in this web service Cambridge University Press & Assessment www.cambridge.org