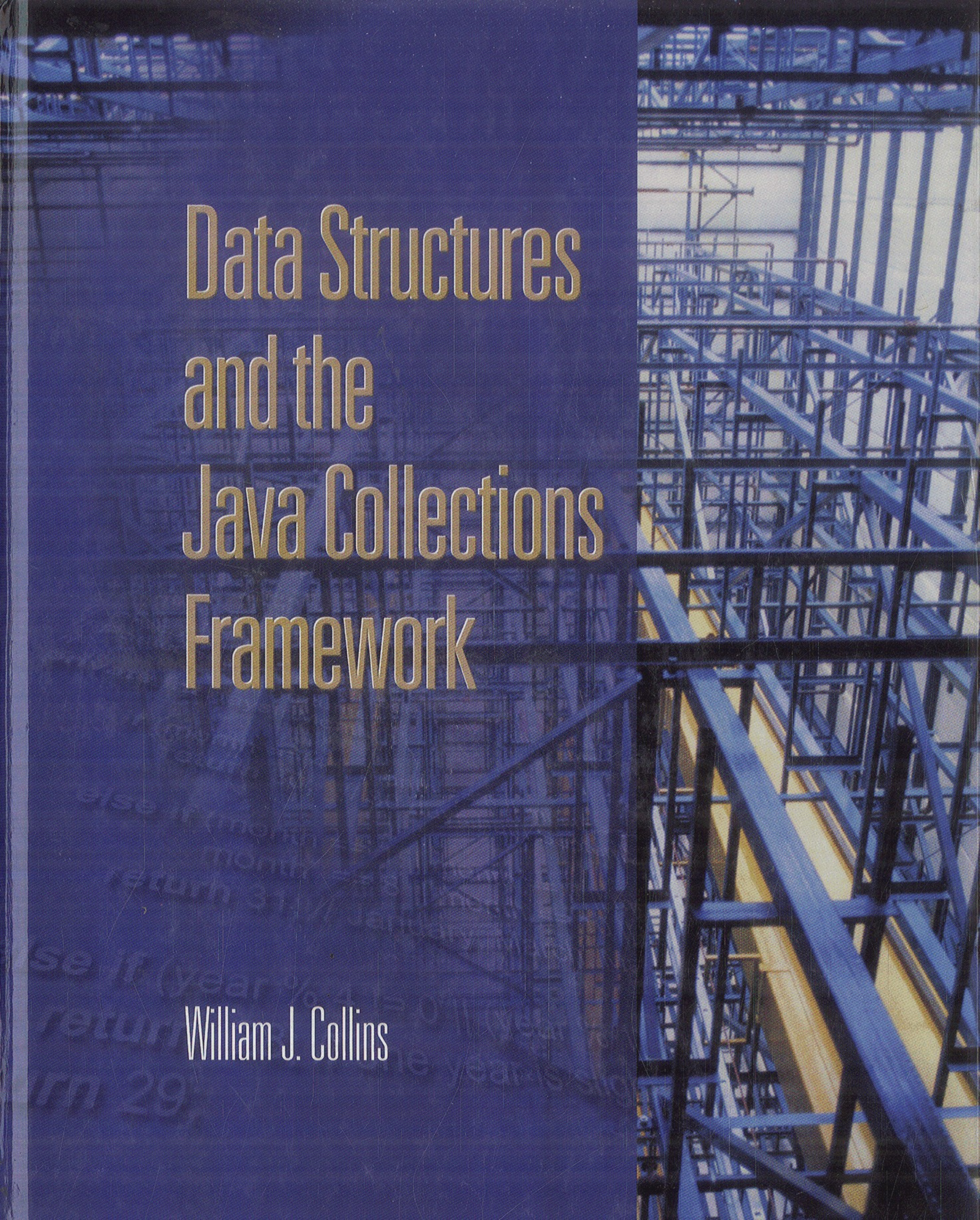

Data Structures and the Java Collections Framework

William J. Collins

CONTENTS

Preface xiv

CHAPTER 1 Important Features of Java 1

Chapter Objectives 1

1.1 Classes 2

1.1.1 Method Descriptions 2

1.1.2 Data Abstraction 5

1.1.3 An Employee *Class 7*

1.1.4 Local Variables and Fields 10

1.1.5 Constructors 10

1.1.6 Instance Variables and Static Variables 11

1.1.7 Visibility Modifiers 12

1.1.8 Graphical User Interfaces 13

1.1.9 The Company *Class 14*

Lab 1: *The* CompanyMain *Project 14*

1.1.10 Inheritance 14

1.1.11 The protected *Visibility Modifier 16*

1.1.12 Inheritance and Constructors 18

Lab 2: *The* SalariedEmployee *Class 22*

1.1.13 Polymorphism 23

1.1.14 Information Hiding 27

1.1.15 Exception Handling 27

1.1.16 Propagating Exceptions 30

Lab 3: *An Example of Exception Handling 31*

Summary 33

Exercises 33

Programming Project 1.1: Developing and Using a Sequence Class 37

CHAPTER 2 Interfaces and Collection Classes 39

Chapter Objectives 39

2.1 Abstract Methods and Abstract Classes 40

Lab 4: *A Class for Regular Polygons 41*

2.2 Interfaces 41

2.3 Arrays 45

2.4 Collection Classes 46

2.5 Storage Structures for Collection Classes 48

Lab 5: *The* ArrayCollection *Class's Implementation of the* Collection *Interface 49*

2.5.1 Linked Structures 49

2.5.2 The LinkedCollection *Class 49*

2.5.3 Fields and Method Definitions in the LinkedCollection *Class 51*

2.5.4 Iterators 55

Lab 6: *Expanding the* LinkedCollection *Class 59*

2.5.5 Data Structures and the Java Collections Framework 59

Summary 59

Exercises 60

Programming Project 2.1: Expanding the LinkedCollection Class 62

CHAPTER 3 Introduction to Software Engineering 65

Chapter Objectives 65

3.1 The Software Development Life Cycle 66

3.2 Problem Analysis 66

3.2.1 System Tests 68

3.3 Program Design 69

3.3.1 Method Descriptions and Fields 69

3.3.2 Dependency Diagrams 70

3.4 Program Implementation 73

3.4.1 Method Validation 73

Lab 7: *Drivers 74*

3.4.2 Is Correctness Feasible? 74

3.4.3 Estimating the Efficiency of Methods 75

3.4.4 Big-O Notation 76

3.4.5 Getting Big-O Estimates Quickly 78

3.4.6 Trade-Offs 82

3.4.7 Run-Time Analysis 84

3.4.8 Overview of the Random *Class 85*

***Lab 8:** Randomness and Timing 86*

3.5 Program Maintenance 86

Summary 87

Exercises 88

Programming Project 3.1: Further Expansion of the LinkedCollection Class 91

CHAPTER 4 Recursion 93

Chapter Objectives 93

4.1 Introduction 94

4.2 Factorials 94

4.2.1 Execution Frames 96

4.3 Decimal to Binary 98

***Lab 9:** Fibonacci Numbers 101*

4.4 Towers of Hanoi 102

4.4.1 A Recurrence Relation 110

4.5 Backtracking 111

4.5.1 An A-maze-ing Application 114

4.6 Binary Search 120

***Lab 10:** Iterative Binary Search 131*

***Lab 11:** Generating Permutations 131*

4.7 Indirect Recursion 131

4.8 The Cost of Recursion 132

Summary 133

Exercises 134

Programming Project 4.1: Iterative Version of Towers of Hanoi 142

Programming Project 4.2: Eight Queens 144

Programming Project 4.3: A Knight's Tour 146

CHAPTER 5 Array Lists 149

Chapter Objectives 149

5.1 The List Interface 150

5.2 The ArrayList Class 151

5.2.1 Method Descriptions for the ArrayList *Class 152*

5.2.2 ArrayList *Class Heading 158*

5.2.3 Fields in the ArrayList *Class 159*

5.2.4 ArrayList *Objects Are Serializable 160*

5.2.5 ArrayList *Objects Are Cloneable 161*

5.3 The ArrayList Implementation 162

5.3.1 Definition of the add *Method 163*

5.3.2 Amortized Time 165

5.3.3 The clone *Method and the Copy Constructor 166*

5.3.4 Fail-Fast Iterators 168

***Lab 12:** More Details on the* ArrayList *Class 169*

5.4 Application: High-Precision Arithmetic 169

5.4.1 Design of the VeryLongInt *Class 170*

5.4.2 Implementation of the VeryLongInt *Class 171*

***Lab 13:** Extending the* VeryLongInt *Class 175*

5.5 The Vector Class 175

Summary 175

Exercises 175

Programming Project 5.1: Extending the VeryLongInt Class 179

Programming Project 5.2: The Deque Class 180

CHAPTER 6 Linked Lists 185

Chapter Objectives 185

6.1 The LinkedList Class 186

6.1.1 The LinkedList *Class versus the* ArrayList *Class 187*

6.1.2 LinkedList *Iterators 190*

6.1.3 Fields and Implementation of the LinkedList *Class 195*

6.1.4 Fields and Implementation of ListItr *Class 203*

***Lab 14:** More Implementation Details of the* ListItr *Class 206*

***Lab 15:** Timing the* ArrayList *and* LinkedList *Classes 207*

6.1.5 Alternative Designs and Implementations of the LinkedList *Class 207*

6.1.6 Circular Linked Lists 210

6.2 Application: A Line Editor 211

6.2.1 Design of the Editor *Class 215*

6.2.2 Implementation of the Editor *Class 217*

6.2.3 Big-O Analysis of the Editor *Class Methods 221*

6.2.4 The EditorDriver *Class 221*

Summary 223

Exercises 224

Programming Project 6.1: Extending the Line Editor 226

Programming Project 6.2: Alternative Design and Implementation of the LinkedList Class 231

CHAPTER 7

Queues and Stacks 233

Chapter Objectives 233

7.1 Queues 234

7.1.1 Design and Implementation of the Queue *Class 235*

7.1.2 Alternative Designs and Implementation of the Queue *Class 237*

7.2 Computer Simulation 242

7.3 Application: A Simulated Car Wash 244

7.3.1 Design of the CarWash *Class 245*

7.3.2 Implementation of the CarWash *Class 246*

7.3.3 Analysis of the CarWash *Methods 249*

7.3.4 Randomizing the Arrival Times 250

***Lab 16:** Randomizing the Arrival Times 251*

7.4 Stacks 251

7.4.1 Design and Implementation of the Stack *Class 252*

7.4.2 The Stack *Class in the Java Collections Framework 253*

7.4.3 Alternative Designs and Implementations of the Stack *Class 253*

7.5 Application: How Compilers Implement Recursion 254

7.6 Application: Converting From Infix to Postfix 257

7.6.1 Postfix Notation 259

7.6.1 Transition Matrix 260

7.6.3 Tokens 261

***Lab 17:** Converting from Infix to Postfix 263*

7.6.4 Prefix Notation 263

Summary 267

Exercises 267

Programming Project 7.1: Extending Speedo's Car Wash 270

Programming Project 7.2: Run-Time Evaluation of a Condition 272

Programming Project 7.3: An Iterative Version of Maze-Search 276

CHAPTER 8

Binary Trees and Binary Search Trees 277

Chapter Objectives 277

8.1 Definition and Properties of Binary Trees 278

8.1.1 The Binary Tree Theorem 284

8.1.2 External Path Length 287

8.1.3 Traversals of a Binary Tree 288

8.2 Binary Search Trees 294

8.2.1 The BinSearchTree *Class 295*

8.2.2 Fields and Embedded Classes in the BinSearchTree *Class 298*

8.2.3 Implementation of the BinSearchTree *Class 298*

8.2.4 The remove *Method 304*

8.2.5 The TreeIterator *Class 313*

***Lab 18:** A Run-Time Estimate of the Average Height of a BinSearchTree Object 315*

Summary 316

Exercises 317

Programming Project 8.1: An Alternative Design and Implementation of the Binary-Search-Tree Data Structure 321

CHAPTER 9

Balanced Binary Search Trees 323

Chapter Objectives 323

9.1 A Problem with Binary Search Trees 324

9.2 Rotations 324

9.3 AVL Trees 329

9.3.1 The Height of an AVL Tree 330

9.3.2 The AVLTree *Class 332*

9.3.3 The fixAfterInsertion *Method 335*

9.3.4 Correctness of the add *Method 345*

9.4 Red-Black Trees 348

9.4.1 The Height of a Red-Black Tree 351

Summary 355

Exercises 356

Programming Project 9.1: Defining the remove Method in the AVLTree Class 360

CHAPTER 10 Tree Maps and Tree Sets 361

Chapter Objectives 361

10.1 The TreeMap Class 362

10.1.1 Method Descriptions of the TreeMap *Class 362*

10.1.2 The Fields in the TreeMap *Class 365*

10.1.3 The Comparator *and* Comparable *Classes 365*

10.1.4 The Entry *Class 367*

10.1.5 Implementation of the TreeMap *Class 367*

10.1.6 The fixAfterInsertion *Method 369*

10.1.7 Three Cases of Insertion 371

***Lab 19:** A Red-Black Tree Insertion with All Three Cases 375*

10.1.8 More TreeMap *Methods 375*

10.1.9 The fixAfterDeletion *Method 380*

***Lab 20:** A Red-Black Tree Removal with All Four Cases 388*

10.1.10 The entrySet *Method 389*

10.2 Application: TreeMap Objects: A Simple Thesaurus 389

10.2.1 Design and Implementation of the Thesaurus *Class 390*

10.2.2 Design and Implementation of the ThesaurusDriver *Class 392*

10.3 The TreeSet Class 395

10.3.1 Design and Implementation of the TreeSet *Class 398*

10.4 Application: A Simple Spell-Checker 399

10.4.1 Design and Implementation of the SpellChecker *Class 400*

10.4.2 Design and Implementation of the SpellCheckerDriver *Class 402*

Summary 405

Exercises 405

Programming Project 10.1: Enhancing the SpellChecker Project 408

Programming Project 10.2: Determining Word Frequencies 410

Programming Project 10.3: Building a Concordance 412

CHAPTER 11 Priority Queues 415

Chapter Objectives 415

11.1 Introduction 416

11.2 Definition of the PriorityQueue Interface 417

11.3 Implementations of the PriorityQueue Interface 417

11.3.1 The Heap *Class 419*

11.3.2 Fields in the Heap *Class 424*

11.3.3 Implementation of the Heap *Class 424*

11.3.4 The percolateUp *Method 425*

11.3.5 The percolateDown *Method 430*

***Lab 21:** Incorporating Fairness in Heaps 432*

11.4 Application: Huffman Codes 432

11.4.1 Huffman Trees 435

11.4.2 Greedy Algorithms 438

11.4.3 The Huffman *Class 439*

Summary 446

Exercises 447

Programming Project 11.1 Decoding a Huffman-Encoded Message 450

CHAPTER 12 Sorting 453

Chapter Objectives 453
12.1 Introduction 454
12.2 Insertion Sort 454
12.3 How Fast Can We Sort? 457
12.4 Fast Sorts 459
12.4.1 Merge Sort 459
12.4.2 Tree Sort 466
12.4.3 Heap Sort 467
12.4.4 Quick Sort 473
***Lab 22:** Run-times for Sort Methods 482*
Summary 482
Exercises 482
Programming Project 12.1: File Sorting 491

CHAPTER 13 Searching and the Hash Classes 495

Chapter Objectives 495
13.1 A Framework to Analyze Searching 496
13.2 Review of Searching 496
13.2.1 Sequential Search 496
13.2.2 Binary Search 497
13.2.3 Red-Black-Tree Search 498
13.3 The HashMap Class 499
13.3.1 Method Descriptions in the HashMap *Class 499*
13.3.2 Fields in the HashMap *Class 501*
13.3.3 Hashing 502
13.3.4 The hashCode *Method 505*
13.3.5 The Uniform Hashing Assumption 506
13.3.6 Chaining 507
13.3.7 Implementation of the HashMap *Class 509*
13.3.8 Analysis of Chained Hashing 513
13.3.9 The HashIterator *Class 515*
13.4 The HashSet Class 517
***Lab 23:** Timing the Hash Classes 517*
13.5 Open-Address Hashing 517
13.5.1 The remove *Method 519*
13.5.2 Primary Clustering 525
13.5.3 Double Hashing 526
13.5.4 Analysis of Open-Address Hashing 530
Summary 533
Exercises 534
Programming Project 13.1: Comparing Chained Hashing and Open-Address Hashing 538

CHAPTER 14 Graphs, Trees, and Networks 539

Chapter Objectives 539
14.1 Undirected Graphs 540
14.2 Directed Graphs 543
14.3 Trees 544
14.4 Networks 545
14.5 Graph Algorithms 547
14.5.1 Iterators 547
14.5.2 Connectedness 554
14.5.3 Generating a Minimum Spanning Tree 555
14.5.4 Finding the Shortest Path through a Network 560
14.6 Developing a Network Class 563
14.6.1 Method Descriptions in the Network *Class 565*
14.6.2 Fields in the Network *Class 567*
14.6.3 Implementation of the Network *Class 568*
***Lab 24:** The Traveling Salesperson Problem 579*
14.6.4 An Alternative Design and Implementation of the Network *Class 579*
14.7 Backtracking through a Network 582
Summary 584
Exercises 585

Programming Project 14.1: Completing the *Implementation of the Network Class under* the Adjacency-Matrix Design 589
Programming Project 14.2: A Network Search 590

APPENDIX 1 Mathematical Background 593

A1.1 Introduction 593
A1.2 Functions and Sequences 593
A1.3 Sums and Products 594
A1.4 Logarithms 595
A1.5 Mathematical Induction 597
A1.5.1 Induction and Recursion 604
Exercises 605

APPENDIX 2 The GUI and GUIListener Classes 607

A2.1 Introduction 607
A2.2 *Threads 608*
A2.3 Implementing the Process Interface 610
A2.4 The GUI Class 611
A2.4.1 The GUI *Constructor 613*
A2.4.2 The Other Methods in the GUI *Class 615*
A2.5 The GUIListener Class 615
A2.6 Putting It All Together 617

APPENDIX 3 The Java Collections Framework 619

A3.1 Introduction 619
A3.2 The Collection Interface 619
A3.3 The List Interface 621
A3.4 The ListIterator Interface 623
A3.5 The Set Interface 625
A3.6 The Map Interface 627
A3.6.1 The Entry *Interface within the* Map *Interface 630*
A3.7 The ArrayList Class 630
A3.8 The LinkedList Class 643
A3.8.1 The ListItr *Class within the* LinkedList *Class 657*
A3.9 The TreeSet Class 662
A3.10 The TreeMap Class 674
A3.10.1 The Iterator *Class within the* TreeMap *Class 688*
A3.11 The HashSet Class 689
A3.12 The HashMap Class 698
A3.12.1 The HashIterator *Class within the* HashMap *Class 707*

Bibliography 709
Index 711