A Practical Guide to Data Structures and Algorithms Using Java

Sally and Ken Goldman
Preface

This handbook of data structures and algorithms is designed as a comprehensive resource for computer science students and practitioners. The book is, quite literally, the product of a marriage of theory and practice. As an alternative to the survey approach taken by traditional data structures and algorithms textbooks, this book builds on a theoretical foundation to offer a top-down application-centered approach and a systematic treatment of data structure design and their practical implementation.

The book serves three major purposes: guidance, implementation, and insight. Charts, decision trees, and text provide guidance through the large body of material presented. Unlike a textbook, it is not necessary to read the entire book to fully benefit from its contents. Our intention is that readers with a specific problem will follow the provided guidance and organizational tools to quickly identify the most appropriate data structure or algorithm for their problem. For example, readers seeking a data structure for an application are first guided to a suitable abstract data type (ADT), and then to the most appropriate implementation of that ADT. Trade-offs between competing data types and implementations motivate each decision in the context of the problem at hand.

Traditional textbooks generally gloss over the different possible variations of a given data structure type. For example, a typical textbook has a chapter on “hashing” that treats all of the various uses of hashing uniformly as one idea (for example, hash-based implementations of a set or mapping). However, in reality, implementing them all in terms of a single ADT would lead to inefficiencies for alternate uses. Consider an application that requires a mapping from each word in a text document to the positions at which it occurs. One could use Java’s HashMap to associate each word with a linked list of line numbers. However, each insertion to associate a new word with a line number would require using get (to discover that the word is not yet in the mapping), and then put (that duplicates most of the work performed by get). In this book, we explicitly include the BucketMapping interface to provide efficient support for such an application. By explicitly introducing separate interfaces and ADTs for important variations in usage, differences can be highlighted and understood.

The book includes complete implementations for a wide variety of important data structures and algorithms. Unlike most textbooks that sweep details under the rug to simplify the implementation for “ease of explanation,” we have taken the approach of providing complete object-oriented implementations within an extensible class hierarchy. Yet we have not done so at the expense of clarity. Because of the completeness of implementation, chapters on some topics are longer than one might see in a textbook covering a similar topic. However, the organization of the chapters simplifies navigation, and the detailed implementations provide design insights useful to practitioners. Our implementations follow standard Java programming conventions.

Parts II and III of the book cover a large number of data structures and algorithms. We include many abstract data types not provided in the standard Java libraries, but for those data types that are also present in the Java Collections classes, we have tried to remain consistent with the Java interfaces and semantics wherever possible. However, we have diverged in places where our design goals differ. One important departure from the Java Collections is our separation of the iterator concept into two types: markers and trackers. Unlike Java’s provided iterator implementations, markers and trackers support concurrent modification of data structures. In addition, the introduction of a tracker, which maintains the location of a particular object even if its location changes within the structure, is crucial for efficient implementations of even some standard algorithms, such as the use of a priority queue to implement Dijkstra’s shortest path algorithm. However, care must be taken in
many data structure implementations to efficiently support tracking, and our presentation includes a discussion of such design choices.

We integrate the presentation of algorithms with the ADTs that support them. In many cases the algorithms are implemented in terms of the ADT interface and included in an abstract implementation of the ADT. The advantage of such an approach is that the algorithm (in both its presentation and instantiation) is decoupled from the particular ADT implementation.

As thorough as we have tried to be, it would not be possible to cover all possible variations of each data structure. Therefore, explanations of each implementation are designed not only to assist readers in understanding the given implementations of data structures and algorithms, but also to support readers in customizing implementations to suit the requirements of particular applications. Making such modifications while preserving correctness and efficiency requires an understanding of not only how the code operates, but why the code is correct and what aspects of the implementation contribute to its efficiency. To this end, we have provided clearly identified explanations of correctness properties for each implementation, as well as correctness highlights that explain how each method depends upon and preserves these properties. For data structures, these properties often relate to an abstraction function that captures, without undue formalism, how the organization of the data structure maps to the user view of the abstraction. This aids understanding at the intuitive level and serves as a foundation for the methodology we use to reason about program correctness. In this way, if readers choose to modify the provided code, they will be able to check that their change preserves the correctness of the implementation. Similarly, we provide a clearly identified section in which time complexity analysis is provided for each data structure and algorithm. Readers interested in modifying a particular method can look in that section to understand how that method (and consequently their proposed change) influences the overall performance of the implementation. Space complexity issues are also discussed.

The format of the book is designed for easy reference. In Parts II and III, each major data type and its implementations are presented in a sequence of chapters beginning with the semantics of that data type, and followed by each implementation. Within each chapter, standardized section headings help the reader quickly locate the required information. The stylized format is designed to help readers with different needs find what they want to read, as well as what they want to skip. A reader planning to use, but not modify, a data structure implementation may decide to read the introductory explanations and then skim through the implementation while omitting the correctness highlights, and finally read the time complexity analysis at the end of the chapter.

The case studies presented throughout the book provide further examples of how various data structures and algorithms presented in Parts II and III can be applied. They also exemplify the process by which those particular data structures and algorithms were selected for the application. One appendix provides a brief overview of the major features of the Java programming language, another appendix reviews asymptotic notation and complexity analysis, and a third appendix briefly discusses the design patterns we illustrate in this book. Source code for the data types, including interfaces, implementations, algorithms, and sample test cases, is included on the accompanying CD. Complete documentation, in Javadoc format, is also provided.

Note to instructors: An introductory data structures and algorithms course could begin with Part I, with an emphasis on selecting abstract data types and implementations appropriate for applications. Then, based on the interests of the instructor and students, a selected subset of the ADTs could be covered in detail. It is not necessary to present every data structure for each ADT, but instead the comparison tables can be used to highlight the differences, and then students can concentrate on one or two representative implementations of each. For courses with a more applied focus, homework and projects might concentrate on empirical comparisons of the provided implementations, modifications based on the optimizations suggested in the chapters, and projects based on the case studies. A more theoretical course might cover the complexity analysis material in the appendix early in the course, and focus more on theoretical analysis of the algorithms and correctness proofs. Online educational materials will be made available at http://goldman.cse.wustl.edu in December 2007.
# Contents

<table>
<thead>
<tr>
<th>Preface</th>
<th>xxiii</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgments</td>
<td>xxv</td>
</tr>
<tr>
<td>Authors</td>
<td>xxi</td>
</tr>
</tbody>
</table>

## I INTRODUCTION

1 Design Principles  
1.1 Object-Oriented Design and This Book .......................... 4  
1.2 Encapsulation ............................................. 5  
1.3 Invariants and Representation Properties ...................... 6  
1.4 Interfaces and Data Abstraction .......................... 7  
1.5 Case Study on Conceptual Design: Historical Event Collection  
1.6 Case Study on Structural Design: Trees ...................... 10  
1.7 Further Reading ........................................... 13

2 Selecting an Abstract Data Type  
2.1 An Illustrative Example ........................................ 16  
2.2 Broad ADT groups ........................................... 19  
2.3 Partition of a Set ........................................... 21  
2.4 A Collection of Elements .................................... 21  
2.5 Markers and Trackers ....................................... 23  
2.6 Positioning and Finding Elements  
2.6.1 Manually Positioned Collections  
2.6.2 Algorithmically Positioned Collections .................. 26  
2.7 Graphs ....................................................... 28

3 How to Use This Book  
3.1 Conventions ................................................... 35  
3.2 Parts II and III Presentation Structure ...................... 36  
3.2.1 ADT Chapters ............................................ 36  
3.2.2 Data Structures ......................................... 38  
3.2.3 Algorithms .............................................. 40  
3.3 Appendices and CD ............................................ 41

## II COLLECTION DATA STRUCTURES AND ALGORITHMS  

4 Part II Organization ............................................. 45

5 Foundations  
5.1 Wrappers for Delegation ....................................... 50  
5.2 Objects Abstract Class ....................................... 51  
5.2.1 Singleton Classes: Empty and Deleted ...................... 51  
5.2.2 Object Equivalence ....................................... 51  
5.2.3 Object Comparison ....................................... 53

vii
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>Digitizer Interface</td>
<td>54</td>
</tr>
<tr>
<td>5.4</td>
<td>Bucketizer Interface</td>
<td>56</td>
</tr>
<tr>
<td>5.5</td>
<td>Object Pool to Reduce Garbage Collection Overhead</td>
<td>59</td>
</tr>
<tr>
<td>5.6</td>
<td>Concurrency</td>
<td>60</td>
</tr>
<tr>
<td>5.7</td>
<td>Iterators for Traversing Data Structures</td>
<td>62</td>
</tr>
<tr>
<td>5.8</td>
<td>Locator Interface</td>
<td>63</td>
</tr>
<tr>
<td>5.8.1</td>
<td>Case Study: Maintaining Request Quorums for Byzantine Agreement</td>
<td>63</td>
</tr>
<tr>
<td>5.8.2</td>
<td>Interface</td>
<td>64</td>
</tr>
<tr>
<td>5.8.3</td>
<td>Markers and Trackers</td>
<td>65</td>
</tr>
<tr>
<td>5.8.4</td>
<td>Iteration Using Locators</td>
<td>67</td>
</tr>
<tr>
<td>5.8.5</td>
<td>Iteration Order and Concurrent Modifications</td>
<td>68</td>
</tr>
<tr>
<td>5.9</td>
<td>Version Class</td>
<td>70</td>
</tr>
<tr>
<td>5.10</td>
<td>Visitors for Traversing Data Structures</td>
<td>71</td>
</tr>
<tr>
<td>6</td>
<td>Partition ADT and the Union-Find Data Structure</td>
<td>73</td>
</tr>
<tr>
<td>6.1</td>
<td>Partition Element Interface</td>
<td>73</td>
</tr>
<tr>
<td>6.2</td>
<td>Selecting a Data Structure</td>
<td>74</td>
</tr>
<tr>
<td>6.3</td>
<td>Union-Find Data Structure</td>
<td>74</td>
</tr>
<tr>
<td>6.4</td>
<td>Internal Representation</td>
<td>75</td>
</tr>
<tr>
<td>6.5</td>
<td>Representation Properties</td>
<td>76</td>
</tr>
<tr>
<td>6.6</td>
<td>Methods</td>
<td>77</td>
</tr>
<tr>
<td>6.7</td>
<td>Performance Analysis</td>
<td>80</td>
</tr>
<tr>
<td>6.8</td>
<td>Case Study: Preserving Locators When Merging Data Structures</td>
<td>81</td>
</tr>
<tr>
<td>6.9</td>
<td>Further Reading</td>
<td>83</td>
</tr>
<tr>
<td>6.10</td>
<td>Quick Method Reference</td>
<td>87</td>
</tr>
<tr>
<td>7</td>
<td>Collection of Elements</td>
<td>89</td>
</tr>
<tr>
<td>7.1</td>
<td>Collection Interface</td>
<td>90</td>
</tr>
<tr>
<td>7.2</td>
<td>Tracked Collection Interface</td>
<td>92</td>
</tr>
<tr>
<td>7.3</td>
<td>ADTs Implementing the Collection Interface</td>
<td>92</td>
</tr>
<tr>
<td>7.3.1</td>
<td>Manually Positioned Collections</td>
<td>92</td>
</tr>
<tr>
<td>7.3.2</td>
<td>Algorithmically Positioned Untagged Collections</td>
<td>92</td>
</tr>
<tr>
<td>7.3.3</td>
<td>Algorithmically Positioned Tagged Ungrouped Collections</td>
<td>93</td>
</tr>
<tr>
<td>7.3.4</td>
<td>Algorithmically Positioned Tagged Grouped Collections</td>
<td>94</td>
</tr>
<tr>
<td>8</td>
<td>Abstract Collection</td>
<td>95</td>
</tr>
<tr>
<td>8.1</td>
<td>Internal Representation</td>
<td>95</td>
</tr>
<tr>
<td>8.2</td>
<td>Representation Properties</td>
<td>96</td>
</tr>
<tr>
<td>8.3</td>
<td>Methods</td>
<td>96</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Trivial Accessors</td>
<td>96</td>
</tr>
<tr>
<td>8.3.2</td>
<td>Algorithmic Accessors</td>
<td>97</td>
</tr>
<tr>
<td>8.3.3</td>
<td>Representation Mutators</td>
<td>99</td>
</tr>
<tr>
<td>8.3.4</td>
<td>Content Mutators</td>
<td>100</td>
</tr>
<tr>
<td>8.4</td>
<td>Abstract Locator Inner Class</td>
<td>100</td>
</tr>
<tr>
<td>8.5</td>
<td>Visiting Iterator</td>
<td>102</td>
</tr>
<tr>
<td>8.6</td>
<td>Performance Analysis</td>
<td>105</td>
</tr>
<tr>
<td>8.7</td>
<td>Quick Method Reference</td>
<td>105</td>
</tr>
</tbody>
</table>
# Positional Collection ADT

9 Positional Collection ADT 107

9.1 Interface ................................................. 107
9.2 Positional Collection Locator Interface .......... 109
9.3 Terminology .............................................. 110
9.4 Competing ADTs ........................................ 110
9.5 Selecting a Data Structure ............................. 111
  9.5.1 Tradeoffs among Array-Based Data Structures . 114
  9.5.2 Tradeoffs among List-Based Data Structures .. 115
9.6 Summary of Positional Collection Data Structures . 116
9.7 Further Reading ......................................... 119

# Abstract Positional Collection

10 Abstract Positional Collection 121

10.1 Abstract Positional Collection ........................ 121
10.2 Internal Representation ................................. 121
10.3 Quick Method Reference ................................. 122

# Array Data Structure

11 Array Data Structure 125

11.1 Internal Representation ................................. 126
11.2 Representation Properties .............................. 128
11.3 Methods .................................................. 128
  11.3.1 Constructors ...................................... 128
  11.3.2 Trivial Accessors .................................. 129
  11.3.3 Representation Accessors ......................... 130
  11.3.4 Algorithmic Accessors ............................. 131
  11.3.5 Representation Mutators ......................... 132
  11.3.6 Content Mutators .................................. 133
  11.3.7 Locator Initializers ............................... 139
11.4 Sorting Algorithms ...................................... 140
  11.4.1 Insertion Sort ...................................... 143
  11.4.2 Mergesort ........................................... 145
  11.4.3 Heap Sort .......................................... 147
  11.4.4 Tree Sort ............................................ 149
  11.4.5 Quicksort .......................................... 149
  11.4.6 Radix Sort ......................................... 155
  11.4.7 Bucket Sort ....................................... 157
11.5 Selection and Median Finding ......................... 158
11.6 Basic Marker Inner Class .............................. 160
11.7 Marker Inner Class ..................................... 162
11.8 Performance Analysis .................................. 163
11.9 Further Reading ........................................ 166
11.10 Quick Method Reference ............................... 167

# Circular Array Data Structure

12 Circular Array Data Structure 171

12.1 Internal Representation ............................... 171
12.2 Representation Properties ............................ 173
12.3 Methods ................................................ 173
  12.3.1 Constructors ...................................... 173
  12.3.2 Representation Accessors ......................... 174
  12.3.3 Representation Mutators ......................... 175
  12.3.4 Content Mutators .................................. 176
12.4 Performance Analysis .................................. 181
12.5 Quick Method Reference ............................... 183


<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td><strong>Dynamic Array and Dynamic Circular Array Data Structures</strong></td>
<td>185</td>
</tr>
<tr>
<td>13.1</td>
<td>Dynamic Array</td>
<td>185</td>
</tr>
<tr>
<td>13.2</td>
<td>Internal Representation</td>
<td>186</td>
</tr>
<tr>
<td>13.3</td>
<td>Representation Properties</td>
<td>187</td>
</tr>
<tr>
<td>13.4</td>
<td>Methods</td>
<td>187</td>
</tr>
<tr>
<td>13.4.1</td>
<td>Constructors</td>
<td>187</td>
</tr>
<tr>
<td>13.4.2</td>
<td>Representation Mutators</td>
<td>188</td>
</tr>
<tr>
<td>13.4.3</td>
<td>Content Mutators</td>
<td>189</td>
</tr>
<tr>
<td>13.5</td>
<td>Performance Analysis</td>
<td>190</td>
</tr>
<tr>
<td>13.6</td>
<td>Dynamic Circular Array</td>
<td>191</td>
</tr>
</tbody>
</table>

| 14      | **Tracked Array Data Structure**           | 193  |
| 14.1    | Internal Representation                     | 194  |
| 14.2    | Representation Properties                   | 197  |
| 14.3    | Node Inner Class                           | 198  |
| 14.4    | Tracked Array Methods                       | 198  |
| 14.4.1  | Constructors                               | 198  |
| 14.4.2  | Representation Accessors                    | 199  |
| 14.4.3  | Algorithmic Accessors                       | 199  |
| 14.4.4  | Representation Mutators                     | 200  |
| 14.4.5  | Content Mutators                            | 201  |
| 14.4.6  | Locator Initializers                        | 204  |
| 14.5    | Wrappers for Sorting Tracked Arrays         | 205  |
| 14.6    | Tracker Inner Class                         | 207  |
| 14.7    | Performance Analysis                        | 213  |
| 14.8    | Quick Method Reference                      | 214  |

| 15      | **Singly Linked List Data Structure**       | 217  |
| 15.1    | Internal Representation                      | 217  |
| 15.2    | Representation Properties                    | 220  |
| 15.3    | List Item Inner Class                        | 220  |
| 15.4    | Singly Linked List Methods                   | 221  |
| 15.4.1  | Constructors and Factory Methods            | 221  |
| 15.4.2  | Representation Accessors                     | 222  |
| 15.4.3  | Algorithmic Accessors                        | 222  |
| 15.4.4  | Representation Mutators                      | 224  |
| 15.4.5  | Content Mutators                             | 224  |
| 15.4.6  | Locator Initializers                         | 231  |
| 15.5    | Sorting Algorithms Revisited                 | 232  |
| 15.5.1  | Insertion Sort                              | 232  |
| 15.5.2  | Mergesort                                   | 233  |
| 15.5.3  | Heap Sort                                   | 235  |
| 15.5.4  | Tree Sort                                   | 236  |
| 15.5.5  | Quicksort                                   | 237  |
| 15.5.6  | Radix Sort                                  | 242  |
| 15.5.7  | Bucket Sort                                 | 244  |
| 15.6    | Selection and Median Finding                 | 244  |
| 15.7    | Tracker Inner Class                          | 246  |
| 15.8    | Performance Analysis                         | 251  |
| 15.9    | Quick Method Reference                       | 254  |
16 Doubly Linked List Data Structure 257
16.1 Internal Representation .................................................. 257
16.2 Representation Properties ................................................. 258
16.3 Doubly Linked List Item Inner Class ................................. 259
16.4 Doubly Linked List Methods .............................................. 259
  16.4.1 Constructors and Factory Methods ................................ 259
  16.4.2 Representation Accessors ........................................... 260
  16.4.3 Algorithmic Accessors .............................................. 260
  16.4.4 Representation Mutators ........................................... 261
16.5 Performance Analysis .................................................... 261
16.6 Quick Method Reference ................................................ 263

17 Buffer ADT and Its Implementation 265
17.1 Internal Representation .................................................. 265
17.2 Representation Properties ................................................. 266
17.3 Methods ........................................................................ 266
  17.3.1 Constructors ............................................................ 266
  17.3.2 Trivial Accessors ....................................................... 267
  17.3.3 Algorithmic Accessors .............................................. 268
  17.3.4 Content Mutators ...................................................... 268
  17.3.5 Locator Initializers ................................................... 269
17.4 Performance Analysis ..................................................... 270
17.5 Quick Method Reference ................................................ 270

18 Queue ADT and Implementation 271
18.1 Internal Representation .................................................. 271
18.2 Methods ........................................................................ 271
18.3 Performance Analysis ..................................................... 272
18.4 Quick Method Reference ................................................ 273

19 Stack ADT and Implementation 275
19.1 Internal Representation .................................................. 275
19.2 Methods ........................................................................ 276
19.3 Performance Analysis ..................................................... 278
19.4 Quick Method Reference ................................................ 278

20 Set ADT 279
20.1 Case Study: Airline Travel Agent ....................................... 279
20.2 Interface ........................................................................ 280
20.3 Terminology .................................................................... 280
20.4 Hasher Interface .............................................................. 281
20.5 Competing ADTs ............................................................. 282
20.6 Selecting a Data Structure ............................................... 283
20.7 Summary of Set Data Structures ....................................... 286
20.8 Further Reading .............................................................. 289

21 Direct Addressing Data Structure 291
21.1 Internal Representation .................................................. 291
21.2 Representation Properties ................................................. 293
21.3 Default Direct Addressing Hasher Class ............................ 293
21.4 Methods ........................................................................ 293
22 Open Addressing Data Structure 305
  22.1 Internal Representation 305
  22.2 Representation Properties 308
  22.3 Default Open Addressing Hasher Class 309
  22.4 Open Addressing Methods 310
    22.4.1 Constructors 310
    22.4.2 Trivial Accessors 311
    22.4.3 Representation Accessors 311
    22.4.4 Selecting a Hash Function 312
    22.4.5 Algorithmic Accessors 313
    22.4.6 Representation Mutators 314
    22.4.7 Content Mutators 316
  22.5 Performance Analysis 317
  22.6 Quick Method Reference 318

23 Separate Chaining Data Structure 321
  23.1 Internal Representation 322
  23.2 Representation Properties 324
  23.3 Chain Item Inner Class 325
  23.4 Default Separate Chaining Hasher Class 326
  23.5 Separate Chaining Methods 326
    23.5.1 Constructors 326
    23.5.2 Trivial Accessors 328
    23.5.3 Algorithmic Accessors 328
    23.5.4 Representation Mutators 329
    23.5.5 Content Mutators 331
    23.5.6 Locator Initializers 333
  23.6 Marker Inner Class 334
  23.7 Performance Analysis 339
  23.8 Quick Method Reference 340

24 Priority Queue ADT 343
  24.1 Case Study: Huffman Compression 343
  24.2 Interface 345
  24.3 Priority Queue Locator Interface 345
  24.4 Selecting a Data Structure 346
  24.5 Terminology 347
  24.6 Competing ADTs 347
  24.7 Summary of Priority Queue Data Structures 348
  24.8 Further Reading 351
25 Binary Heap Data Structure 353
25.1 Internal Representation ........................................... 353
25.2 Representation Properties ........................................ 356
25.3 Methods .............................................................. 356
  25.3.1 Constructors .................................................... 356
  25.3.2 Trivial Accessors ............................................... 357
  25.3.3 Representation Accessors .................................... 357
  25.3.4 Algorithmic Accessors ...................................... 357
  25.3.5 Representation Mutators .................................... 358
  25.3.6 Content Mutators ............................................ 361
  25.3.7 Locator Initializers .......................................... 366
25.4 Locator Inner Class .............................................. 367
25.5 Performance Analysis ............................................ 368
25.6 Quick Method Reference ......................................... 370

26 Leftist Heap Data Structure 373
26.1 Internal Representation ........................................... 373
26.2 Representation Properties ........................................ 375
26.3 Leftist Heap Node Inner Class ................................. 376
26.4 Leftist Heap Methods ............................................. 378
  26.4.1 Constructors .................................................... 378
  26.4.2 Algorithmic Accessors ...................................... 379
  26.4.3 Content Mutators ............................................ 382
  26.4.4 Locator Initializers .......................................... 391
26.5 Tracker Inner Class .............................................. 391
26.6 Performance Analysis ............................................ 393
26.7 Quick Method Reference ......................................... 396

27 Pairing Heap Data Structure 399
27.1 Internal Representation ........................................... 399
27.2 Representation Properties ........................................ 402
27.3 Heap Node Inner Class ........................................... 402
27.4 Pairing Heap Methods ............................................. 405
  27.4.1 Constructors and Factory Methods ....................... 406
  27.4.2 Algorithmic Accessors ...................................... 406
  27.4.3 Representation Mutators .................................... 407
  27.4.4 Content Mutators ............................................ 408
  27.4.5 Locator Initializers .......................................... 414
27.5 Tracker Inner Class .............................................. 415
27.6 Performance Analysis ............................................ 418
27.7 Quick Method Reference ......................................... 419

28 Fibonacci Heap Data Structure 423
28.1 Internal Representation ........................................... 424
28.2 Representation Properties ........................................ 426
28.3 Fibonacci Heap Node Inner Class .............................. 426
28.4 Fibonacci Heap Methods ......................................... 428
  28.4.1 Constructors and Factory Methods ....................... 428
  28.4.2 Representation Mutators .................................... 429
  28.4.3 Content Mutators ............................................ 433
28.5 Performance Analysis ............................................ 437
28.6 Quick Method Reference ......................................... 441
29 Ordered Collection ADT 443
29.1 Case Study: Historical Event Collection (Range Queries) 443
29.2 Case Study: Linux Virtual Memory Map 444
29.3 Interface 445
29.4 Terminology 446
29.5 Competing ADTs 447
29.6 Selecting a Data Structure 447
29.7 Summary of Ordered Collection Data Structures 449
29.8 Further Reading 452

30 Sorted Array Data Structure 455
30.1 Internal Representation 455
30.2 Representation Properties 456
30.3 Methods 457
30.3.1 Constructors 457
30.3.2 Trivial Accessors 457
30.3.3 Binary Search Algorithm 458
30.3.4 Algorithmic Accessors 460
30.3.5 Content Mutators 464
30.3.6 Utilities for the B-Tree and B+~Tree Classes 465
30.3.7 Locator Initializers 467
30.4 Performance Analysis 467
30.5 Quick Method Reference 469

31 Abstract Search Tree Class 471
31.1 Internal Representation 471
31.2 Representation Properties 473
31.3 Abstract Tree Node Inner Class 474
31.4 Abstract Search Tree Class 475
31.5 Abstract Search Tree Methods 475
31.5.1 Constructors 475
31.5.2 Algorithmic Accessors 475
31.5.3 Content Mutators 479
31.6 Quick Method Reference 480

32 Binary Search Tree Data Structure 481
32.1 Internal Representation 481
32.2 Representation Properties 485
32.3 BSTNode Inner Class 485
32.4 Binary Search Tree Methods 489
32.4.1 Constructors and Factory Methods 490
32.4.2 Algorithmic Accessors 490
32.4.3 Content Mutators 496
32.4.4 Locator Initializers 500
32.5 Tracker Inner Class 501
32.6 Performance Analysis 504
32.7 Quick Method Reference 506

33 Balanced Binary Search Trees 509
33.1 Methods 510
33.2 Quick Method Reference 512
### 34 Red-Black Tree Data Structure

- **34.1 Internal Representation** ........................................ 514
- **34.2 Representation Properties** .................................... 515
- **34.3 RBNode Inner Class** ........................................... 515
- **34.4 Methods** .......................................................... 517
  - **34.4.1 Constructors and Factory Methods** ..................... 517
  - **34.4.2 Content Mutators** ........................................ 517
- **34.5 Performance Analysis** .......................................... 528
- **34.6 Quick Method Reference** ...................................... 528

### 35 Splay Tree Data Structure

- **35.1 Internal Representation** ........................................ 531
- **35.2 Methods** .......................................................... 532
  - **35.2.1 Constructors** .............................................. 532
  - **35.2.2 Representation Mutators** ................................ 533
  - **35.2.3 Algorithmic Accessors** .................................. 533
  - **35.2.4 Content Mutators** ........................................ 537
  - **35.2.5 Locator Initializers** ..................................... 541
- **35.3 Performance Analysis** .......................................... 541
- **35.4 Quick Method Reference** ...................................... 543

### 36 B-Tree Data Structure

- **36.1 Internal Representation** ........................................ 545
- **36.2 Representation Properties** .................................... 546
- **36.3 B-Tree Node Inner Class** .................................... 549
  - **36.3.1 B-Tree Node Methods** .................................. 550
  - **36.3.2 B-Tree Node Representation Mutators** ............... 551
  - **36.3.3 B-Tree Node Content Mutators** ....................... 555
- **36.4 B-Tree Methods** ................................................ 557
  - **36.4.1 Constructors and Factory Methods** .................... 557
  - **36.4.2 Algorithmic Accessors** ................................ 558
  - **36.4.3 Representation Mutators** ................................ 562
  - **36.4.4 Content Mutators** ........................................ 563
  - **36.4.5 Locator Initializers** ..................................... 567
- **36.5 Marker Inner Class** ............................................ 568
- **36.6 Performance Analysis** .......................................... 570
- **36.7 Quick Method Reference** ...................................... 573

### 37 B+-Tree Data Structure

- **37.1 Case Study: A Web Search Engine** ............................ 575
- **37.2 Internal Representation** ........................................ 576
- **37.3 Representation Properties** .................................... 577
- **37.4 Leaf Node Inner Class** ...................................... 578
- **37.5 B+-Tree Methods** .............................................. 579
  - **37.5.1 Constructors and Factory Methods** .................... 583
  - **37.5.2 Representation Accessors** ................................ 584
  - **37.5.3 Algorithmic Accessors** ................................ 584
  - **37.5.4 Content Mutators** ....................................... 585
- **37.6 Performance Analysis** .......................................... 588
- **37.7 Quick Method Reference** ...................................... 591
38 Skip List Data Structure 593
38.1 Internal Representation ........................................ 593
38.2 Representation Properties ..................................... 596
38.3 Tower Inner Class .................................................. 597
38.4 Skip List Methods .................................................. 598
  38.4.1 Constructors ................................................... 598
  38.4.2 Algorithmic Accessors .................................... 600
  38.4.3 Representation Mutators .................................. 604
  38.4.4 Content Mutators .......................................... 605
  38.4.5 Locator Initializers ......................................... 610
38.5 Tracker Inner Class ............................................... 610
38.6 Performance Analysis ............................................ 613
38.7 Quick Method Reference ........................................ 617

39 Digitized Ordered Collection ADT 619
39.1 Case Study: Packet Routing ................................... 619
39.2 Case Study: Inverted Index for Text Retrieval .......... 620
39.3 Digitized Ordered Collection Interface ................. 621
39.4 Selecting a Data Structure ................................... 622
39.5 Terminology ....................................................... 623
39.6 Competing ADTs .................................................. 624
39.7 Summary of Digitized Ordered Collection Data Structures ............... 625
39.8 Trie Variations ................................................... 630
39.9 Suffix Trees ....................................................... 630
39.10 Indexing Tries ..................................................... 631
39.11 Further Reading .................................................. 633

40 Trie Node Types 635
40.1 Trie Node Interface ............................................... 635
40.2 Abstract Trie Node Class ....................................... 635
40.3 Trie Leaf Node Interface ...................................... 636
40.4 Abstract Trie Leaf Node Class ............................... 637

41 Trie Data Structure 639
41.1 Internal Representation ....................................... 639
41.2 Representation Properties ................................... 642
41.3 Internal Node Inner Class ................................... 642
41.4 Leaf Node Inner Class ......................................... 644
41.5 Search Data Inner Class ...................................... 644
41.6 FindResult Enumerated Type ................................ 649
41.7 Trie Methods ....................................................... 650
  41.7.1 Constructors and Factory Methods .................. 650
  41.7.2 Algorithmic Accessors ................................ 651
  41.7.3 Content Mutators ......................................... 657
  41.7.4 Locator Initializers .................................... 661
41.8 Trie Tracker Inner Class ....................................... 662
41.9 Performance Analysis ......................................... 664
41.10 Quick Method Reference ..................................... 667
# 42 Compact Trie Data Structure 671

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>42.1 Internal Representation</td>
<td>671</td>
</tr>
<tr>
<td>42.2 Representation Properties</td>
<td>672</td>
</tr>
<tr>
<td>42.3 Compact Trie Methods</td>
<td>672</td>
</tr>
<tr>
<td>42.3.1 Constructors and Factory Methods</td>
<td>673</td>
</tr>
<tr>
<td>42.3.2 Algorithmic Accessors</td>
<td>673</td>
</tr>
<tr>
<td>42.3.3 Content Mutators</td>
<td>675</td>
</tr>
<tr>
<td>42.4 Performance Analysis</td>
<td>679</td>
</tr>
<tr>
<td>42.5 Quick Method Reference</td>
<td>680</td>
</tr>
</tbody>
</table>

# 43 Compressed Trie Data Structure 683

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>43.1 Internal Representation</td>
<td>683</td>
</tr>
<tr>
<td>43.2 Representation Properties</td>
<td>685</td>
</tr>
<tr>
<td>43.3 Compressed Trie Node Interface</td>
<td>686</td>
</tr>
<tr>
<td>43.4 Internal Node Inner Class</td>
<td>686</td>
</tr>
<tr>
<td>43.5 Compressed Trie Leaf Node Inner Class</td>
<td>687</td>
</tr>
<tr>
<td>43.6 Compressed Trie Search Data Inner Class</td>
<td>687</td>
</tr>
<tr>
<td>43.7 Compressed Trie Methods</td>
<td>688</td>
</tr>
<tr>
<td>43.7.1 Constructors and Factory Methods</td>
<td>688</td>
</tr>
<tr>
<td>43.7.2 Algorithmic Accessors</td>
<td>689</td>
</tr>
<tr>
<td>43.7.3 Content Mutators</td>
<td>690</td>
</tr>
<tr>
<td>43.8 Performance Analysis</td>
<td>693</td>
</tr>
<tr>
<td>43.9 Quick Method Reference</td>
<td>694</td>
</tr>
</tbody>
</table>

# 44 Patricia Trie Data Structure 697

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>44.1 Internal Representation</td>
<td>697</td>
</tr>
<tr>
<td>44.2 Representation Properties</td>
<td>699</td>
</tr>
<tr>
<td>44.3 Patricia Trie Node Inner Class</td>
<td>700</td>
</tr>
<tr>
<td>44.4 Patricia Trie Search Data Inner Class</td>
<td>702</td>
</tr>
<tr>
<td>44.5 Patricia Trie Methods</td>
<td>705</td>
</tr>
<tr>
<td>44.5.1 Constructors</td>
<td>705</td>
</tr>
<tr>
<td>44.5.2 Content Mutators</td>
<td>706</td>
</tr>
<tr>
<td>44.6 Performance Analysis</td>
<td>717</td>
</tr>
<tr>
<td>44.7 Quick Method Reference</td>
<td>717</td>
</tr>
</tbody>
</table>

# 45 Ternary Search Trie Data Structure 719

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>45.1 Internal Representation</td>
<td>719</td>
</tr>
<tr>
<td>45.2 Representation Properties</td>
<td>721</td>
</tr>
<tr>
<td>45.3 Ternary Search Trie Internal Node Inner Class</td>
<td>722</td>
</tr>
<tr>
<td>45.4 Ternary Search Trie Search Data Inner Class</td>
<td>722</td>
</tr>
<tr>
<td>45.5 Ternary Search Trie Methods</td>
<td>723</td>
</tr>
<tr>
<td>45.5.1 Constructors and Factory Methods</td>
<td>723</td>
</tr>
<tr>
<td>45.5.2 Algorithmic Accessors</td>
<td>724</td>
</tr>
<tr>
<td>45.6 Performance Analysis</td>
<td>725</td>
</tr>
<tr>
<td>45.7 Quick Method Reference</td>
<td>725</td>
</tr>
</tbody>
</table>

# 46 Spatial Collection ADT 729

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>46.1 Case Study: Collision Detection in Video Games</td>
<td>729</td>
</tr>
<tr>
<td>46.2 Interface</td>
<td>730</td>
</tr>
<tr>
<td>46.3 Competing ADTs</td>
<td>731</td>
</tr>
<tr>
<td>46.4 Summary of Spatial Collection Data Structures</td>
<td>732</td>
</tr>
<tr>
<td>46.5 Further Reading</td>
<td>732</td>
</tr>
</tbody>
</table>
47 KD-Tree Data Structure 735
47.1 Internal Representation .................................................. 736
47.2 Representation Properties ............................................. 739
47.3 Alternating Comparator .................................................. 739
47.4 KDNode Inner Class ....................................................... 742
47.5 KDTreeImpl Class ......................................................... 747
47.6 KD-Tree Methods ........................................................ 750
47.7 Performance Analysis .................................................... 752
47.8 Quick Method Reference ................................................. 754

48 Quad Tree Data Structure 757
48.1 Internal Representation .................................................. 757
48.2 Representation Properties ............................................. 760
48.3 Partitioning a Two-Dimensional Space ............................... 761
48.4 QTNode Inner Class ....................................................... 762
48.5 Box Inner Class .......................................................... 765
48.6 Quad Tree Methods ....................................................... 767
  48.6.1 Constructors and Factory Methods .............................. 767
  48.6.2 Representation Accessors ......................................... 768
  48.6.3 Algorithmic Accessors ............................................. 768
  48.6.4 Content Mutators .................................................... 772
  48.6.5 Locator Initializers ................................................. 781
48.7 Performance Analysis .................................................... 782
48.8 Quick Method Reference ................................................. 782

49 Tagged Collection ADTs 785
49.1 Tagged Element .......................................................... 786
  49.1.1 Mutable Tagged Element ......................................... 787
  49.1.2 Tagged Element Comparator .................................... 787
  49.1.3 Tagged Element Digitizer ........................................ 788
  49.1.4 Tagged Element XY Comparator ................................ 788
49.2 Tagged Collection Interface .......................................... 789
49.3 Tracked Tagged Interface .............................................. 791
49.4 Competing ADTs ........................................................ 791
49.5 Selecting a Tagged Collection ADT .................................. 792
49.6 Tagged Collection Wrapper ............................................ 793
49.7 Mapping ADT ............................................................. 797
  49.7.1 Direct Addressing Mapping ...................................... 799
  49.7.2 Open Addressing Mapping ....................................... 799
  49.7.3 Separate Chaining Mapping ..................................... 801
49.8 Tagged Priority Queue ADT ............................................. 801
  49.8.1 Tagged Priority Queue Wrapper ................................ 803
  49.8.2 Tagged Binary Heap .............................................. 806
  49.8.3 Tagged Leftist Heap .............................................. 806
  49.8.4 Tagged Pairing Heap ............................................. 807
  49.8.5 Tagged Fibonacci Heap .......................................... 807
49.9 Tagged Ordered Collection ADT ...................................... 807
  49.9.1 Tagged Ordered Collection Wrapper ........................... 811
  49.9.2 Tagged Sorted Array .............................................. 812
  49.9.3 Tagged Binary Search Tree ..................................... 813
  49.9.4 Tagged Splay Tree ................................................ 813
50 Tagged Bucket Collection ADTs

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>50.1</td>
<td>Case Study: Historical Event Collection (Indexing)</td>
<td>826</td>
</tr>
<tr>
<td>50.2</td>
<td>Case Study: Biosequence Comparison</td>
<td>828</td>
</tr>
<tr>
<td>50.3</td>
<td>Bucket Factory Interface</td>
<td>829</td>
</tr>
<tr>
<td>50.4</td>
<td>Tagged Bucket Collection Interface</td>
<td>829</td>
</tr>
<tr>
<td>50.5</td>
<td>Selecting a Tagged Bucket Collection ADT</td>
<td>831</td>
</tr>
<tr>
<td>50.6</td>
<td>Selecting a Data Structure</td>
<td>832</td>
</tr>
<tr>
<td>50.7</td>
<td>Tagged Bucket Collection Wrapper</td>
<td>832</td>
</tr>
</tbody>
</table>

III GRAPH DATA STRUCTURES AND ALGORITHMS

51 Part III Organization

52 Graph ADT

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>52.1</td>
<td>Case Study: Task Scheduler</td>
<td>844</td>
</tr>
<tr>
<td>52.2</td>
<td>Terminology</td>
<td>845</td>
</tr>
<tr>
<td>52.3</td>
<td>Edge Interface</td>
<td>848</td>
</tr>
<tr>
<td>52.4</td>
<td>Graph Interface</td>
<td>848</td>
</tr>
<tr>
<td>52.5</td>
<td>Graph Representation Interface</td>
<td>850</td>
</tr>
<tr>
<td>52.6</td>
<td>Selecting a Data Structure</td>
<td>851</td>
</tr>
<tr>
<td>52.7</td>
<td>Summary of Graph Data Structures</td>
<td>854</td>
</tr>
</tbody>
</table>

53 Abstract Graph and Graph Algorithms

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>53.1</td>
<td>Representation Properties</td>
<td>858</td>
</tr>
<tr>
<td>53.2</td>
<td>Methods</td>
<td>858</td>
</tr>
<tr>
<td>53.3</td>
<td>In-Tree Representation</td>
<td>861</td>
</tr>
<tr>
<td>53.4</td>
<td>Finding Shortest Paths with Breadth-First Search</td>
<td>865</td>
</tr>
<tr>
<td>53.5</td>
<td>Finding Cycles and Connected Components with Depth-First Search</td>
<td>867</td>
</tr>
<tr>
<td>53.6</td>
<td>Topological Sort</td>
<td>873</td>
</tr>
<tr>
<td>53.7</td>
<td>Strongly Connected Components</td>
<td>875</td>
</tr>
<tr>
<td>53.8</td>
<td>Performance Analysis</td>
<td>878</td>
</tr>
<tr>
<td>53.9</td>
<td>Case Study: Garbage Collection</td>
<td>878</td>
</tr>
<tr>
<td>53.10</td>
<td>Further Reading</td>
<td>880</td>
</tr>
<tr>
<td>53.11</td>
<td>Quick Method Reference</td>
<td>881</td>
</tr>
</tbody>
</table>
## 54 Adjacency Matrix Data Structure

54.1 Internal Representation ........................................... 883
54.2 Representation Properties .................................. 886
54.3 Methods ......................................................... 887
  54.3.1 Constructors .............................................. 887
  54.3.2 Trivial Accessors ........................................ 888
  54.3.3 Algorithmic Accessors ................................. 888
  54.3.4 Representation Mutators ............................ 889
  54.3.5 Content Mutators .................................... 890
54.4 Edge Iterators .................................................. 893
54.5 Incident Edge Iterator Inner Class ....................... 893
54.6 Adjacency Matrix Class .................................. 896
54.7 Performance Analysis .................................... 897
54.8 Quick Method Reference ................................. 898

## 55 Adjacency List Data Structure

55.1 Internal Representation ........................................... 902
55.2 Representation Properties .................................. 905
55.3 Methods ......................................................... 906
  55.3.1 Constructors .............................................. 906
  55.3.2 Trivial Accessors ........................................ 906
  55.3.3 Algorithmic Accessors ................................. 906
  55.3.4 Content Mutators .................................... 908
55.4 Edge Iterators .................................................. 910
55.5 Edge Iterator Inner Class .................................. 910
55.6 Adjacency List Class .................................. 912
55.7 Performance Analysis .................................... 912
55.8 Quick Method Reference ................................. 915

## 56 Weighted Graph ADT

56.1 Case Study: Airline Travel Agent (Revisited) .......... 917
56.2 Case Study: Image Segmentation .......................... 919
56.3 Terminology .................................................... 921
56.4 Weighted Edge Interface .................................. 922
56.5 Simple Weighted Edge Class ............................... 922
56.6 Weighted Graph Interface .................................. 923
56.7 Selecting a Data Structure .................................. 923
56.8 Weighted Adjacency Matrix Data Structure .......... 924
56.9 Weighted Adjacency List Data Structure .............. 924

## 57 Abstract Weighted Graph and Weighted Graph Algorithms

57.1 Greedy Tree Builder ............................................ 926
57.2 Dijkstra’s Single-Source Shortest Path Algorithm .... 931
57.3 Prim’s Minimum Spanning Tree Algorithm .............. 935
57.4 Kruskal’s Minimum Spanning Tree Algorithm .......... 938
57.5 Bellman-Ford’s Single-Source Shortest Path Algorithm .... 941
57.6 Shortest Path Matrix ....................................... 943
57.7 Floyd-Warshall’s All-Pairs Shortest Path Algorithm ......... 945
57.8 Edmonds-Karp Maximum Flow Algorithm .............. 948
57.9 Further Reading ............................................. 954
57.10 Quick Method Reference ................................. 955
IV APPENDICES 957

A Java Fundamentals 959
A.1 Types and Operators ........................................... 959
A.2 Packages .......................................................... 961
A.3 Classes and Objects ............................................. 961
  A.3.1 Instance Variables ........................................... 962
  A.3.2 Methods ....................................................... 962
  A.3.3 Constructors ................................................ 965
A.4 The Heap ......................................................... 965
A.5 Exception Handling ............................................ 967
A.6 Assertions ....................................................... 969
A.7 Inner Classes .................................................... 969
A.8 Static Members .................................................. 970
A.9 Interfaces ........................................................ 971
A.10 The Class Hierarchy ........................................... 973
  A.10.1 Extending Classes ......................................... 973
  A.10.2 Type Checking and Casting .............................. 973
  A.10.3 Superconstructors ......................................... 974
  A.10.4 Overriding Methods ...................................... 974
  A.10.5 Polymorphism ............................................... 974
  A.10.6 Iterators ..................................................... 975
A.11 Generics ........................................................ 976

B Complexity Analysis 979
B.1 Time Complexity ................................................ 980
B.2 Asymptotic Notation ............................................ 981
B.3 Space Complexity ............................................... 984
B.4 Expected Time Complexity ................................... 985
B.5 Amortized Analysis ............................................. 985
B.6 Solving Recurrence Equations with the Master Method ... 987

C Design Patterns Illustrated in This Book 991
C.1 Abstract Factory ............................................... 991
C.2 Adaptor ............................................................ 991
C.3 Bridge ............................................................. 992
C.4 Builder ............................................................. 992
C.5 Command ........................................................ 992
C.6 Composite ......................................................... 993
C.7 Decorator ........................................................ 993
C.8 Facade .............................................................. 993
C.9 Factory Method .................................................. 994
C.10 Flyweight ......................................................... 994
C.11 Iterator ........................................................... 994
C.12 Leasing ............................................................ 995
C.13 Proxy ............................................................... 995
C.14 Singleton ........................................................ 995
C.15 Strategy .......................................................... 996
C.16 Template Method .............................................. 996
C.17 Visitor ............................................................. 996

References 997
Index 1005