

DATA STRUCTURES USING JAVA™



Yedidyah Langsam • Moshe Augenstein
Aaron M. Tenenbaum

Contents

Preface	xv
CHAPTER 1 Introduction to Data Structures	1
1.1 Information and Meaning	1
<i>Binary and Decimal Integers</i>	3
<i>Real Numbers</i>	4
<i>Character Strings</i>	5
<i>Hardware and Software</i>	6
<i>Concept of Implementation</i>	7
<i>Example</i>	8
<i>Abstract Data Types</i>	13
<i>Sequences as Value Definitions</i>	16
<i>ADT for Varying-Length Character Strings</i>	18
<i>Data Types in Java</i>	19
<i>Objects and Java</i>	20
<i>Value and Reference Semantics</i>	21
<i>Data Structures and Java</i>	22
Exercises	24
1.2 Arrays, Strings, and Vectors in Java	24
<i>The Array as an ADT</i>	26
<i>Using One-Dimensional Arrays</i>	27
<i>Implementing One-Dimensional Arrays</i>	29
<i>Arrays as Parameters</i>	32
<i>Two-Dimensional Arrays</i>	32
<i>Multidimensional Arrays</i>	35
<i>Character Strings in Java</i>	38
<i>Character String Operations</i>	40
<i>Vectors in Java</i>	40
Exercises	42

1.3	Classes and Objects in Java	44
	<i>Constructors</i>	46
	<i>Representing Other Data Structures</i>	47
	<i>Rational Numbers</i>	47
	<i>Using the class Rational</i>	53
	<i>Allocation of Storage and Scope of Classes</i>	57
	<i>Vectors in Java</i>	64
	Exercises	66

CHAPTER 2 The Stack 68

2.1	Definition and Examples	68
	<i>Primitive Operations</i>	69
	<i>Example</i>	72
	<i>The Stack as an Abstract Data Type</i>	75
	Exercises	75
2.2	Representing Stacks in Java	76
	<i>Implementing the pop Operation</i>	79
	<i>Testing for Exceptional Conditions</i>	81
	<i>Implementing the Push Operation</i>	82
	Exercises	85
2.3	Example: Infix, Postfix, and Prefix	85
	<i>Basic Definitions and Examples</i>	85
	<i>Evaluating a Postfix Expression</i>	88
	<i>Program to Evaluate a Postfix Expression</i>	89
	<i>Limitations of the Program</i>	92
	<i>Converting an Expression from Infix to Postfix</i>	92
	<i>Program to Convert an Expression from Infix to Postfix</i>	97
	Exercises	99
2.4	Stack of Objects of Varying Types	101
	<i>Stacks in Java Using Vectors</i>	104
	<i>Dealing with Heterogeneous Data</i>	106
	<i>The Class Class and the java.lang.reflect Package</i>	111
	<i>The Predefined Stack Class</i>	113
	Exercises	117

CHAPTER 3 Recursion 119

3.1	Recursive Definition and Processes	119
	<i>Factorial Function</i>	119
	<i>Multiplication of Natural Numbers</i>	122
	<i>Fibonacci Sequence</i>	123
	<i>Binary Search</i>	124
	<i>Properties of Recursive Definitions or Algorithms</i>	127
	Exercises	127

3.2	Recursion in Java	128
	<i>Factorial in Java</i>	128
	<i>The Fibonacci Numbers in Java</i>	132
	<i>The Binary Search in Java</i>	134
	<i>Recursive Chains</i>	135
	<i>Recursive Definition of Algebraic Expressions</i>	136
	Exercises	139
3.3	Writing Recursive Programs	141
	<i>The Towers of Hanoi Problem</i>	143
	<i>Translation from Prefix to Postfix Using Recursion</i>	148
	Exercises	152
3.4	Simulating Recursion	155
	<i>Return from a Method</i>	157
	<i>Implementing Recursive Methods</i>	158
	<i>Simulation of Factorial</i>	159
	<i>Improving the Simulated Routine</i>	162
	<i>Eliminating gotos</i>	164
	<i>Simulating the Towers of Hanoi</i>	169
	Exercises	175
3.5	Efficiency of Recursion	177
	Exercises	178

CHAPTER 4 Queues and Lists 179

4.1	The Queue and Its Sequential Representation	179
	<i>The Queue as an Abstract Data Type</i>	180
	<i>Java Implementation of Queues</i>	181
	<i>Priority Queue</i>	186
	<i>Array Implementation of a Priority Queue</i>	187
	Exercises	189
4.2	Linked Lists	190
	<i>Inserting and Removing Nodes from a List</i>	192
	<i>Linked Implementation of Stacks</i>	196
	<i>getnode and freenode Operations</i>	196
	<i>Linked Implementation of Queues</i>	198
	<i>The Linked List as a Data Structure</i>	199
	<i>Examples of List Operations</i>	201
	<i>List Implementation of Priority Queues</i>	203
	<i>Header Nodes</i>	204
	Exercises	205
4.3	Lists in Java	206
	<i>Array Implementation of Lists</i>	206
	<i>Limitations of the Array Implementation</i>	211

	<i>Linked Lists Using Dynamic Variables</i>	212
	<i>Queues as Lists in Java</i>	214
	<i>Examples of List Operations in Java</i>	216
	<i>Comparing the Dynamic and Array Implementations of Lists</i>	219
	<i>Implementing Header Nodes</i>	219
	Exercises	220
4.4	An Example: Simulation Using Linked Lists	221
	<i>Simulation Process</i>	222
	<i>Data Structures</i>	223
	<i>Simulation Program</i>	226
	Exercises	229
4.5	Other List Structures	231
	<i>Circular Lists</i>	231
	<i>Stack as a Circular List</i>	232
	<i>Queue as a Circular List</i>	233
	<i>Primitive Operations on Circular Lists</i>	233
	<i>The Josephus Problem</i>	235
	<i>Addition of Long Positive Integers Using Circular Lists</i>	241
	<i>Doubly Linked Lists</i>	245
	<i>Addition of Long Integers Using Doubly Linked Lists</i>	248
	Exercises	255

CHAPTER 5 Trees **258**

5.1	Binary Trees	258
	<i>Operations on Binary Trees</i>	263
	<i>Applications of Binary Trees</i>	263
	Exercises	269
5.2	Binary Tree Representations	270
	<i>Node Representation of Binary Trees</i>	270
	<i>Internal and External Nodes</i>	274
	<i>Implicit Array Representation of Binary Trees</i>	275
	<i>Choosing a Binary Tree Representation</i>	281
	<i>Binary Tree Traversals in Java</i>	281
	<i>Threaded Binary Trees</i>	284
	<i>Traversal Using a father Field</i>	288
	<i>Heterogeneous Binary Trees</i>	291
	Exercises	292
5.3	An Example: The Huffman Algorithm	295
	<i>The Huffman Algorithm</i>	298
	<i>Java Application</i>	299
	Exercises	304

5.4	Representing Lists as Binary Trees	305
	<i>Finding the kth Element</i>	306
	<i>Deleting an Element</i>	309
	<i>Implementing Tree-Represented Lists in Java</i>	312
	<i>Constructing a Tree-Represented List</i>	315
	<i>The Josephus Problem Revisited</i>	316
	Exercises	318
5.5	Trees and their Applications	318
	<i>Java Representations of Trees</i>	320
	<i>Tree Traversals</i>	325
	<i>General Expressions as Trees</i>	325
	<i>Evaluating an Expression Tree</i>	328
	<i>Constructing a Tree</i>	330
	Exercises	332
5.6	Example: Game Trees	333
	Exercises	339
CHAPTER 6	Sorting	341
6.1	General Background	341
	<i>Efficiency Considerations</i>	343
	<i>O Notation</i>	345
	<i>Efficiency of Sorting</i>	347
	Exercises	349
6.2	Exchange Sorts	351
	<i>Bubble Sort</i>	351
	<i>Quicksort</i>	353
	<i>Efficiency of Quicksort</i>	359
	Exercises	362
6.3	Selection and Tree Sorting	363
	<i>Straight Selection Sort</i>	364
	<i>Binary Tree Sorts</i>	365
	<i>Heapsort</i>	367
	<i>Heap as a Priority Queue</i>	368
	<i>Sorting Using a Heap</i>	370
	<i>Heapsort Method</i>	373
	Exercises	375
6.4	Insertion Sorts	376
	<i>Simple Insertion</i>	376
	<i>Shell Sort</i>	377
	<i>Address Calculation Sort</i>	380
	Exercises	383

6.5	Merge and Radix Sorts	385
	<i>Merge Sorts</i>	385
	<i>The Cook-Kim Algorithm</i>	388
	<i>Radix Sort</i>	388
	Exercises	392

CHAPTER 7 Searching **395**

7.1	Basic Search Techniques	395
	<i>Dictionary as an Abstract Data Type</i>	396
	<i>Algorithmic Notation</i>	397
	<i>Sequential Searching</i>	398
	<i>Efficiency of Sequential Searching</i>	399
	<i>Reordering a List for Maximum Search Efficiency</i>	400
	<i>Searching an Ordered Table</i>	402
	<i>Indexed Sequential Search</i>	403
	<i>Binary Search</i>	404
	<i>Interpolation Search</i>	407
	Exercises	408
7.2	Tree Searching	411
	<i>Inserting into a Binary Search Tree</i>	414
	<i>Deleting from a Binary Search Tree</i>	414
	<i>Efficiency of Binary Search Tree Operations</i>	416
	<i>Efficiency of Nonuniform Binary Search Trees</i>	419
	<i>Optimum Search Trees</i>	421
	<i>Balanced Trees</i>	422
	Exercises	431
7.3	General Search Trees	433
	<i>Multiway Search Trees</i>	433
	<i>Searching a Multiway Tree</i>	435
	<i>Implementing a Multiway Tree</i>	436
	<i>Traversing a Multiway Tree</i>	438
	<i>Insertion in a Multiway Search Tree</i>	440
	<i>B-Trees</i>	444
	<i>Algorithms for B-Tree Insertion</i>	450
	<i>Computing father and index</i>	454
	<i>Deletion in Multiway Search Trees</i>	457
	<i>Efficiency of Multiway Search Trees</i>	461
	<i>Improving the B-Tree</i>	464
	<i>B⁺-Trees</i>	468
	<i>Digital Search Trees</i>	469
	<i>Tries</i>	472
	Exercises	474
7.4	Hashing	475
	<i>Resolving Hash Clashes by Open Addressing</i>	477
	<i>Deleting Items from a Hash Table</i>	480

<i>Efficiency of Rehashing Methods</i>	480
<i>Hash Table Reordering</i>	483
<i>Brent's Method</i>	484
<i>Binary Tree Hashing</i>	486
<i>Improvements with Additional Memory</i>	489
<i>Coalesced Hashing</i>	492
<i>Separate Chaining</i>	494
<i>Hashing in External Storage</i>	498
<i>Separator Method</i>	500
<i>Dynamic Hashing and Extendible Hashing</i>	501
<i>Linear Hashing</i>	506
<i>Choosing a Hash Function</i>	511
<i>Perfect Hash Functions</i>	514
<i>Universal Classes of Hash Functions</i>	518
Exercises	519

CHAPTER 8 Graphs and their Applications 521

8.1 Graphs	521
<i>An Application of Graphs</i>	524
<i>Java Representation of Graphs</i>	526
<i>Transitive Closure</i>	528
<i>Warshall's Algorithm</i>	531
<i>Shortest-Path Algorithm</i>	532
Exercises	535
8.2 Flow Problem	536
<i>Improving a Flow Function</i>	538
<i>Example</i>	541
<i>Algorithm and Program</i>	543
Exercises	547
8.3 Linked Representation of Graphs	548
<i>Dijkstra's Algorithm Revisited</i>	556
<i>Organizing the Set of Graph Nodes</i>	558
<i>Application to Scheduling</i>	559
<i>Java Program</i>	563
Exercises	566
8.4 Graph Traversal and Spanning Forests	569
<i>Traversal Methods for Graphs</i>	569
<i>Spanning Forests</i>	573
<i>Undirected Graphs and Their Traversals</i>	575
<i>Depth-First Traversal</i>	577
<i>Applications of Depth-First Traversal</i>	580
<i>Efficiency of Depth-First Traversal</i>	581
<i>Breadth-First Traversal</i>	582
<i>Minimum Spanning Trees</i>	583

<i>Kruskal's Algorithm</i>	585
<i>Round-Robin Algorithm</i>	586
Exercises	586

CHAPTER 9 Storage Management 588

9.1 General Lists	588
<i>Operations That Modify a List</i>	590
<i>Examples</i>	591
<i>Linked List Representation of a List</i>	592
<i>Representation of Lists</i>	595
<i>crList Operation</i>	596
<i>Use of List Headers</i>	598
<i>Freeing List Nodes</i>	600
<i>General Lists in Java</i>	601
<i>Programming Languages and Lists</i>	603
Exercises	605
9.2 Automatic List Management	605
<i>Reference Count Method</i>	606
<i>Garbage Collection</i>	611
<i>Algorithms for Garbage Collection</i>	612
<i>Collection and Compaction</i>	618
<i>Variations of Garbage Collection</i>	624
Exercises	625
9.3 Dynamic Memory Management	626
<i>Compaction of Blocks of Storage</i>	627
<i>First-Fit, Best-Fit, and Worst-Fit</i>	629
<i>Improvements in the First-Fit Method</i>	633
<i>Freeing Storage Blocks</i>	634
<i>Boundary Tag Method</i>	636
<i>Buddy System</i>	638
<i>Other Buddy Systems</i>	645
Exercises	647