

Contents

Preface to First Edition	xi
Preface to the Second Edition	xiii
Preface to the “Root 7” Edition	xv
0 Introduction	1
0.1 Read Me	1
0.2 He Can’t Say That, Can He?	2
1 The Object-Oriented Method	5
1.1 Data Abstraction and Encapsulation	6
1.2 The Object Model	7
1.3 Object-Oriented Terminology	8
1.4 A Special-Purpose Class: A Bank Account	11
1.5 A General-Purpose Class: An Association	14
1.6 Sketching an Example: A Word List	18
1.7 Sketching an Example: A Rectangle Class	20
1.8 Interfaces	22
1.9 Who Is the User?	24
1.10 Conclusions	25
1.11 Laboratory: The Day of the Week Calculator	29
2 Comments, Conditions, and Assertions	33
2.1 Pre- and Postconditions	34
2.2 Assertions	34
2.3 Craftsmanship	36
2.4 Conclusions	37
2.5 Laboratory: Using Javadoc Commenting	39
3 Vectors	43
3.1 The Interface	45
3.2 Example: The Word List Revisited	47
3.3 Example: Word Frequency	48
3.4 The Implementation	50
3.5 Extensibility: A Feature	53
3.6 Example: L-Systems	56
3.7 Example: Vector-Based Sets	57
3.8 Example: The Matrix Class	60
3.9 Conclusions	64

3.10 Laboratory: The Silver Dollar Game	67
4 Generics	69
4.1 Motivation (in case we need some)	70
4.1.1 Possible Solution: Specialization	71
4.2 Implementing Generic Container Classes	72
4.2.1 Generic Associations	72
4.2.2 Parameterizing the Vector Class	74
4.2.3 Restricting Parameters	79
4.3 Conclusions	80
5 Design Fundamentals	81
5.1 Asymptotic Analysis Tools	81
5.1.1 Time and Space Complexity	82
5.1.2 Examples	85
5.1.3 The Trading of Time and Space	91
5.1.4 Back-of-the-Envelope Estimations	92
5.2 Self-Reference	94
5.2.1 Recursion	94
5.2.2 Mathematical Induction	101
5.3 Properties of Design	108
5.3.1 Symmetry	108
5.3.2 Friction	110
5.4 Conclusions	110
5.5 Laboratory: How Fast Is Java?	115
6 Sorting	119
6.1 Approaching the Problem	119
6.2 Selection Sort	122
6.3 Insertion Sort	125
6.4 Mergesort	127
6.5 Quicksort	131
6.6 Radix Sort	134
6.7 Sorting Objects	138
6.8 Ordering Objects Using Comparators	140
6.9 Vector-Based Sorting	143
6.10 Conclusions	144
6.11 Laboratory: Sorting with Comparators	147
7 A Design Method	149
7.1 The Interface-Based Approach	149
7.1.1 Design of the Interface	150
7.1.2 Development of an Abstract Implementation	151
7.1.3 Implementation	152
7.2 Example: Development of Generators	152
7.3 Example: Playing Cards	155

7.4	Conclusions	160
8	Iterators	161
8.1	Java's Enumeration Interface	161
8.2	The Iterator Interface	163
8.3	Example: Vector Iterators	165
8.4	Example: Rethinking Generators	167
8.5	Example: Filtering Iterators	170
8.6	Conclusions	172
8.7	Laboratory: The Two-Towers Problem	175
9	Lists	179
9.1	Example: A Unique Program	182
9.2	Example: Free Lists	183
9.3	Partial Implementation: Abstract Lists	186
9.4	Implementation: Singly Linked Lists	188
9.5	Implementation: Doubly Linked Lists	201
9.6	Implementation: Circularly Linked Lists	206
9.7	Implementation: Vectors	209
9.8	List Iterators	209
9.9	Conclusions	211
9.10	Laboratory: Lists with Dummy Nodes	215
10	Linear Structures	219
10.1	Stacks	221
10.1.1	Example: Simulating Recursion	222
10.1.2	Vector-Based Stacks	225
10.1.3	List-Based Stacks	227
10.1.4	Comparisons	228
10.2	Queues	229
10.2.1	Example: Solving a Coin Puzzle	231
10.2.2	List-Based Queues	234
10.2.3	Vector-Based Queues	235
10.2.4	Array-Based Queues	238
10.3	Example: Solving Mazes	242
10.4	Conclusions	244
10.5	Laboratory: A Stack-Based Language	247
10.6	Laboratory: The Web Crawler	251
11	Ordered Structures	253
11.1	Comparable Objects Revisited	253
11.1.1	Example: Comparable Ratios	254
11.1.2	Example: Comparable Associations	256
11.2	Keeping Structures Ordered	258
11.2.1	The OrderedStructure Interface	258
11.2.2	The Ordered Vector and Binary Search	259

11.2.3 Example: Sorting Revisited	264
11.2.4 A Comparator-based Approach	265
11.2.5 The Ordered List	267
11.2.6 Example: The Modified Parking Lot	270
11.3 Conclusions	272
11.4 Laboratory: Computing the “Best Of”	275
12 Binary Trees	277
12.1 Terminology	277
12.2 Example: Pedigree Charts	280
12.3 Example: Expression Trees	281
12.4 Implementation	282
12.4.1 The BinaryTree Implementation	284
12.5 Example: An Expert System	287
12.6 Traversals of Binary Trees	290
12.6.1 Preorder Traversal	291
12.6.2 In-order Traversal	293
12.6.3 Postorder Traversal	295
12.6.4 Level-order Traversal	296
12.6.5 Recursion in Iterators	297
12.7 Property-Based Methods	299
12.8 Example: Huffman Compression	303
12.9 Example Implementation: Ahnentafel	307
12.10 Conclusions	309
12.11 Laboratory: Playing Gardner’s Hex-a-Pawn	313
13 Priority Queues	315
13.1 The Interface	315
13.2 Example: Improving the Huffman Code	317
13.3 A Vector-Based Implementation	318
13.4 A Heap Implementation	319
13.4.1 Vector-Based Heaps	320
13.4.2 Example: Heapsort	326
13.4.3 Skew Heaps	329
13.5 Example: Circuit Simulation	333
13.6 Conclusions	337
13.7 Laboratory: Simulating Business	341
14 Search Trees	343
14.1 Binary Search Trees	343
14.2 Example: Tree Sort	345
14.3 Example: Associative Structures	345
14.4 Implementation	348
14.5 Splay Trees	354
14.6 Splay Tree Implementation	357
14.7 An Alternative: Red-Black Trees	361

14.8	Conclusions	363
14.9	Laboratory: Improving the BinarySearchTree	367
15	Maps	369
15.1	Example Revisited: The Symbol Table	369
15.2	The Interface	370
15.3	Simple Implementation: MapList	372
15.4	Constant Time Maps: Hash Tables	374
15.4.1	Open Addressing	375
15.4.2	External Chaining	383
15.4.3	Generation of Hash Codes	385
15.4.4	Hash Codes for Collection Classes	391
15.4.5	Performance Analysis	392
15.5	Ordered Maps and Tables	392
15.6	Example: Document Indexing	395
15.7	Conclusions	398
15.8	Laboratory: The Soundex Name Lookup System	401
16	Graphs	403
16.1	Terminology	403
16.2	The Graph Interface	404
16.3	Implementations	408
16.3.1	Abstract Classes Reemphasized	408
16.3.2	Adjacency Matrices	410
16.3.3	Adjacency Lists	416
16.4	Examples: Common Graph Algorithms	422
16.4.1	Reachability	422
16.4.2	Topological Sorting	424
16.4.3	Transitive Closure	427
16.4.4	All Pairs Minimum Distance	428
16.4.5	Greedy Algorithms	429
16.5	Conclusions	434
16.6	Laboratory: Converting Between Units	439
A	Answers	441
A.1	Solutions to Self Check Problems	441
A.2	Solutions to Odd-Numbered Problems	451
B	Beginning with Java	489
B.1	A First Program	489
B.2	Declarations	491
B.2.1	Primitive Types	491
B.2.2	Reference Types	493
B.3	Important Classes	494
B.3.1	The structure.InputStream Class	494
B.3.2	The java.util.Scanner Class	495

B.3.3	The PrintStream Class	496
B.3.4	Strings	497
B.4	Control Constructs	498
B.4.1	Conditional Statements	498
B.4.2	Loops	499
B.5	Methods	502
B.6	Inheritance and Subtyping	502
B.6.1	Inheritance	502
B.6.2	Subtyping	503
B.6.3	Interfaces and Abstract Classes	504
B.7	Use of the Assert Command	506
B.8	Use of the Keyword Protected	507
C	Collections	511
C.1	Collection Class Features	511
C.2	Parallel Features	511
C.3	Conversion	512
D	Documentation	513
D.1	Structure Package Hierarchy	513
D.2	Principles	515
	Index	517