

ELLIOT B. KOFFMAN | PAUL A.T. WOLFGANG

OBJECTS, ABSTRACTION,
DATA STRUCTURES
AND DESIGN USING **J**AVA



WILEY
INTERNATIONAL
EDITION

RESTRICTED!
Not for Sale
in the
United States

Contents

Preface	vii
Chapter 1 Introduction to Software Design	1
1.1 The Software Challenge	2
1.2 The Software Life Cycle	3
Software Life Cycle Models	3
Software Life Cycle Activities	6
Requirements Specification	7
Analysis	7
Design	8
Exercises for Section 1.2	11
1.3 Using Abstraction to Manage Complexity	11
Procedural Abstraction	11
Data Abstraction	12
Information Hiding	12
Exercises for Section 1.3	13
1.4 Abstract Data Types, Interfaces, and Pre- and Postconditions	13
Interfaces	14
The implements Clause	16
Declaring a Variable of an Interface Type	17
An Interface for a Telephone Directory Class	17
Javadoc Comments	17
Contracts and Interfaces	18
Preconditions and Postconditions	19
Exercises for Section 1.4	20
1.5 Requirements Analysis, Use Cases, and Sequence Diagrams	21
<i>Case Study: Designing a Telephone Directory Program</i>	21
Exercises for Section 1.5	27
1.6 Design of an Array-Based Phone Directory	28
<i>Case Study: Designing a Telephone Directory Program (cont.)</i>	28
Exercises for Section 1.6	35
1.7 Implementing and Testing the Array-Based Phone Directory	35
<i>Case Study: Designing a Telephone Directory Program (cont.)</i>	35
Exercises for Section 1.7	42
1.8 Two Classes That Implement <code>PDUUserInterface</code>	42
<i>Case Study: Designing a Telephone Directory Program (cont.)</i>	43
Exercises for Section 1.8	54
Chapter Review, Exercises, and Programming Projects	54

Chapter 2 Program Correctness and Efficiency	59
2.1 Program Defects and “Bugs”	60
Syntax Errors	61
Run-time Errors or Exceptions	61
Logic Errors	65
Exercises for Section 2.1	66
2.2 The Exception Class Hierarchy	67
Exercises for Section 2.2	70
2.3 Catching and Handling Exceptions	70
Uncaught Exceptions	70
The <code>try-catch-finally</code> Sequence	71
Exercises for Section 2.3	77
2.4 Throwing Exceptions	77
The <code>throws</code> Clause	77
The <code>throw</code> Statement	79
Catching Exceptions in the Phone Directory Application	82
Exercises for Section 2.4	84
2.5 Testing Programs	85
Structured Walkthroughs	86
Levels and Types of Testing	86
Preparations for Testing	88
Testing Tips for Program Systems	89
Developing the Test Data	90
Testing Boundary Conditions	90
Who Does the Testing?	93
Stubs	93
Drivers	94
Testing a Class	94
Using a Test Framework	96
Regression Testing	97
Integration Testing	97
Exercises for Section 2.5	98
2.6 Debugging a Program	99
Using a Debugger	100
Exercises for Section 2.6	104
2.7 Reasoning about Programs: Assertions and Loop Invariants	104
Assertions	105
Loop Invariants	105
Enrichment Topic: The Java <code>assert</code> statement	107
Exercises for Section 2.7	108

2.8	Efficiency of Algorithms	108
	Big-O Notation	111
	Comparing Performance	115
	Algorithms with Exponential and Factorial Growth Rates	117
	Exercises for Section 2.8	117
	Chapter Review, Exercises, and Programming Projects	118
Chapter 3	Inheritance and Class Hierarchies	125
3.1	Introduction to Inheritance and Class Hierarchies	126
	<i>Is-a</i> Versus <i>Has-a</i> Relationships	128
	A Superclass and a Subclass	128
	Use of <code>this</code> .	129
	Initializing Data Fields in a Subclass	130
	The No-Parameter Constructor	131
	Protected Visibility for Superclass Data Fields	131
	Exercises for Section 3.1	132
3.2	Method Overriding, Method Overloading, and Polymorphism	133
	Method Overriding	133
	Method Overloading	135
	Polymorphism	136
	Exercises for Section 3.2	137
3.3	Abstract Classes, Assignment, and Casting in a Hierarchy	138
	Referencing Actual Objects	141
	Abstract Classes and Interfaces	141
	Abstract Class <code>Number</code> and the Java Wrapper Classes	141
	Summary of Features of Actual Classes, Abstract Classes, and Interfaces	142
	Exercises for Section 3.3	143
3.4	Class <code>Object</code> , Casting, and Cloning	143
	The Method <code>toString</code>	144
	Operations Determined by Type of Reference Variable	144
	Casting in a Class Hierarchy	145
	The Method <code>Object.equals</code>	148
	Cloning	149
	Exercises for Section 3.4	154
3.5	Multiple Inheritance, Multiple Interfaces, and Delegation	155
	Using Multiple Interfaces to Emulate Multiple Inheritance	155
	Implementing Reuse Through Delegation	157
	Exercises for Section 3.5	158
3.6	Packages and Visibility	159
	Packages	159
	The No-Package-Declared Environment	159
	Package Visibility	160
	Visibility Supports Encapsulation	160
	Exercises for Section 3.6	162

3.7	A Shape Class Hierarchy	162
	<i>Case Study: Drawing Geometric Figures</i>	163
	Exercises for Section 3.7	182
3.8	Object Factories	182
	<i>Case Study: Compute Areas and Perimeters of Geometric Shapes</i>	183
	Exercises for Section 3.8	185
	Chapter Review, Exercises, and Programming Projects	185
Chapter 4	Lists and the Collection Interface	193
4.1	The List Interface and ArrayList Class	194
	The ArrayList Class	195
	Specification of the ArrayList Class	197
	Exercises for Section 4.1	198
4.2	Application of ArrayList	198
	The Phone Directory Application Revisited	199
	Exercises for Section 4.2	201
4.3	Implementation of an ArrayList Class	201
	The add(Object obj) Method	202
	The add(int index, Object theValue) Method	203
	The remove Method	204
	The reallocate Method	204
	Performance of the KArrayList	205
	The Vector Class	205
	Exercises for Section 4.3	205
4.4	Single-Linked Lists and Double-Linked Lists	206
	A List Node	208
	Connecting Nodes	210
	Inserting a Node in a List	210
	Removing a Node	211
	Traversing a Linked List	211
	Double-Linked Lists	212
	Creating a Double-Linked List Object	215
	Circular Lists	215
	Exercises for Section 4.4	216
4.5	The LinkedList Class and the Iterator and ListIterator Interfaces	217
	The LinkedList Class	217
	The Iterator	217
	The Iterator Interface	219
	The ListIterator Interface	221
	Comparison of Iterator and ListIterator	224
	Conversion Between a ListIterator and an Index	224
	Exercises for Section 4.5	224

4.6	Implementation of a Double-Linked List Class	225
	Implementing the <code>KWLinkedList</code> Methods	225
	Implementing the <code>ListIterator</code> Interface	226
	Exercises for Section 4.6	233
4.7	Application of the <code>LinkedList</code> Class	234
	<i>Case Study</i> : Maintaining an Ordered list	234
	Exercises for Section 4.7	240
4.8	The Collection Hierarchy	240
	Common Features of Collections	241
	The <code>AbstractCollection</code> , <code>AbstractList</code> , and <code>AbstractSequentialList</code> Classes	242
	Exercises for Section 4.8	243
	Chapter Review, Exercises, and Programming Projects	243
Chapter 5 Stacks		249
<hr/>		
5.1	Stack Abstract Data Type	250
	Specification of the Stack Abstract Data Type	250
	Exercises for Section 5.1	252
5.2	Stack Applications	253
	<i>Case Study</i> : Finding Palindromes	253
	<i>Case Study</i> : Checking for Balanced Parentheses	256
	Exercises for Section 5.2	261
5.3	Implementing a Stack	262
	Implementing a Stack as an Extension of <code>Vector</code>	262
	Implementing a Stack with a List Component	263
	Implementing a Stack Using an Array	265
	Implementing a Stack as a Linked Data Structure	267
	Comparison of Stack Implementations	269
	Exercises for Section 5.3	269
5.4	Additional Stack Applications	270
	<i>Case Study</i> : Evaluating Postfix Expressions	271
	<i>Case Study</i> : Converting from Infix to Postfix	276
	<i>Case Study</i> : Part 2: Converting Expressions with Parentheses	285
	Tying Both Case Studies Together	288
	Exercises for Section 5.4	288
	Chapter Review, Exercises, and Programming Projects	288
Chapter 6 Queues		295
<hr/>		
6.1	Queue Abstract Data Type	296
	A Print Queue	296
	The Unsuitability of a “Print Stack”	297
	A Queue of Customers	297
	Using a Queue for Traversing a Multi-Branch Data Structure	297

Specification for a Queue Interface	298	
Exercises for Section 6.1	299	
6.2 Maintaining a Queue of Customers		300
<i>Case Study: Maintaining a Queue</i>	300	
Exercises for Section 6.2	304	
6.3 Implementing the Queue Interface		305
Using a Single-Linked List to Implement a Queue	305	
Implementing a Queue Using Java's <code>LinkedList</code>	307	
Implementing a Queue Using a Circular Array	309	
Comparing the Three Implementations	315	
Exercises for Section 6.3	316	
6.4 Simulating Waiting Lines Using Queues		316
<i>Case Study: Simulate a Strategy for Serving Airline Passengers</i>	317	
Exercises for Section 6.4	331	
Chapter Review, Exercises, and Programming Projects		332
Chapter 7 Recursion		337
<hr/>		
7.1 Recursive Thinking		338
Steps to Design a Recursive Algorithm	340	
Proving That a Recursive Method Is Correct	342	
Tracing a Recursive Method	343	
The Run-Time Stack and Activation Frames	343	
Exercises for Section 7.1	345	
7.2 Recursive Definitions of Mathematical Formulas		346
Recursion Versus Iteration	349	
Tail Recursion or Last-Line Recursion	350	
Efficiency of Recursion	350	
Exercises for Section 7.2	353	
7.3 Recursive Array Search		353
Design of a Recursive Linear Search Algorithm	354	
Implementation of Linear Search	354	
Design of Binary Search Algorithm	355	
Efficiency of Binary Search	357	
The Comparable Interface	357	
Implementation of Binary Search	358	
Testing Binary Search	359	
Method Arrays. <code>binarySearch</code>	359	
Exercises for Section 7.3	361	
7.4 Recursive Data Structures		361
Recursive Definition of a Linked List	361	
Class <code>LinkedListRec</code>	362	
Removing a List Node	365	
Exercises for Section 7.4	366	

7.5	Problem Solving with Recursion	367
	<i>Case Study: Towers of Hanoi</i>	367
	<i>Case Study: Counting Cells in a Blob</i>	372
	Exercises for Section 7.5	377
7.6	Backtracking	377
	<i>Case Study: Finding a Path Through a Maze</i>	378
	Exercises for Section 7.6	382
	Chapter Review, Exercises, and Programming Projects	383
Chapter 8 Trees		387
<hr/>		
8.1	Tree Terminology and Applications	389
	Tree Terminology	389
	Binary Trees	390
	Some Types of Binary Trees	390
	Fullness and Completeness	393
	General Trees	394
	Exercises for Section 8.1	395
8.2	Tree Traversals	396
	Visualizing Tree Traversals	396
	Traversals of Binary Search Trees and Expression Trees	397
	Exercises for Section 8.2	398
8.3	Implementing a BinaryTree Class	398
	The Node Class	398
	The BinaryTree Class	400
	Exercises for Section 8.3	406
8.4	Binary Search Trees	407
	Overview of a Binary Search Tree	407
	Performance	408
	Class TreeSet and Interface SearchTree	409
	The BinarySearchTree Class	409
	Insertion into a Binary Search Tree	412
	Removal from a Binary Search Tree	415
	<i>Case Study: Writing an Index for a Term Paper</i>	421
	Exercises for Section 8.4	423
8.5	Heaps and Priority Queues	424
	Inserting an Item into a Heap	425
	Removing an Item from a Heap	425
	Implementing a Heap	426
	Priority Queues	429
	The PriorityQueue Interface	430
	Using a Heap as the Basis of a Priority Queue	431
	Using a Comparator	434
	Exercises for Section 8.5	438

8.6	Huffman Trees	438
	<i>Case Study</i> : Building a Custom Huffman Tree	440
	Exercises for Section 8.6	446
	Chapter Review, Exercises, and Programming Projects	447
Chapter 9	Sets and Maps	453
<hr/>		
9.1	Sets and the Set Interface	454
	The Set Abstraction	455
	The Set Interface and Methods	456
	Comparison of Lists and Sets	458
	Exercises for Section 9.1	458
9.2	Maps and the Map Interface	459
	Exercises for Section 9.2	463
9.3	Hash Tables	464
	Hash Codes and Index Calculation	464
	Methods for Generating Hash Codes	465
	Open Addressing	466
	Table Wraparound and Search Termination	467
	Traversing a Hash Table	468
	Deleting an Item Using Open Addressing	469
	Reducing Collisions by Expanding the Table Size	469
	Reducing Collisions Using Quadratic Probing	470
	Problems with Quadratic Probing	470
	Chaining	471
	Performance of Hash Tables	472
	Exercises for Section 9.3	474
9.4	Implementing the Hash Table	476
	Interface <code>KWHashMap</code>	476
	Class <code>Entry</code>	476
	Class <code>HashTableOpen</code>	478
	Class <code>HashTableChain</code>	483
	Testing the Hash Table Implementations	486
	Exercises for Section 9.4	487
9.5	Implementation Considerations for Maps and Sets	488
	Methods <code>hashCode</code> and <code>equals</code>	488
	Implementing <code>HashSetOpen</code>	489
	Writing <code>HashSetOpen</code> as an Adapter Class	489
	Implementing the Java Map and Set Interfaces	490
	Nested Interface <code>Map.Entry</code>	490
	Creating a Set View of a Map	491
	Method <code>entrySet</code> and Classes <code>EntrySet</code> and <code>SetIterator</code>	491
	Classes <code>TreeMap</code> and <code>TreeSet</code>	492
	Exercises for Section 9.5	493

9.6	Additional Applications of Maps	493
	<i>Case Study: Implementing the Phone Directory Using a Map</i>	494
	<i>Case Study: Completing the Huffman Coding Problem</i>	496
	Exercises for Section 9.6	500
	Chapter Review, Exercises, and Programming Projects	501
Chapter 10	Sorting	505
<hr/>		
10.1	Using Java Sorting Methods	506
	Exercises for Section 10.1	509
10.2	Selection Sort	510
	Analysis of Selection Sort	511
	Code for Selection Sort	511
	Exercises for Section 10.2	512
10.3	Bubble Sort	513
	Analysis of Bubble Sort	514
	Code for Bubble Sort	515
	Exercises for Section 10.3	516
10.4	Insertion Sort	516
	Analysis of Insertion Sort	518
	Code for Insertion Sort	518
	Exercises for Section 10.4	519
10.5	Comparison of Quadratic Sorts	520
	Comparisons versus Exchanges	521
	Exercises for Section 10.5	521
10.6	Shell Sort: A Better Insertion Sort	521
	Analysis of Shell Sort	523
	Code for Shell Sort	524
	Exercises for Section 10.6	525
10.7	Merge Sort	525
	Analysis of Merge	526
	Code for Merge	527
	Algorithm for Merge Sort	528
	Trace of Merge Sort Algorithm	528
	Analysis of Merge Sort	529
	Code for Merge Sort	530
	Exercises for Section 10.7	531
10.8	Heapsort	531
	First Version of a Heapsort Algorithm	531
	Revising the Heapsort Algorithm	532
	Algorithm to Build a Heap	533
	Analysis of Revised Heapsort Algorithm	534
	Code for Heapsort	534
	Exercises for Section 10.8	536

10.9	Quicksort	536
	Algorithm for Quicksort	537
	Analysis of Quicksort	537
	Code for Quicksort	538
	Algorithm for Partitioning	539
	Code for partition	540
	A Revised Partition Algorithm	542
	Code for Revised partition Method	543
	Exercises for Section 10.9	544
10.10	Testing the Sort Algorithms	545
	Exercises for Section 10.10	546
10.11	The Dutch National Flag Problem (Optional Topic)	547
	<i>Case Study:</i> The Problem of the Dutch National Flag	547
	Exercises for Section 10.11	550
	Chapter Review, Exercises, and Programming Projects	551
Chapter 11	Self-Balancing Search Trees	555
<hr/>		
11.1	Tree Balance and Rotation	556
	Why Balance Is Important	556
	Rotation	557
	Algorithm for Rotation	557
	Implementing Rotation	559
	Exercises for Section 11.1	560
11.2	AVL Tree	561
	Balancing a Left-Left Tree	561
	Balancing a Left-Right Tree	562
	Four Kinds of Critically Unbalanced Trees	563
	Implementing an AVL Tree	565
	Inserting into an AVL Tree	567
	Removal from an AVL Tree	573
	Performance of the AVL Tree	573
	Exercises for Section 11.2	573
11.3	Red-Black Trees	574
	Insertion into a Red-Black Tree	575
	Removal from a Red-Black Tree	585
	Performance of a Red-Black Tree	586
	The <code>TreeMap</code> and <code>TreeSet</code> Classes	586
	Exercises for Section 11.3	586
11.4	2-3 Trees	587
	Searching a 2-3 Tree	587
	Inserting an Item into a 2-3 Tree	588
	Analysis of 2-3 Trees and Comparison with Balanced Binary Trees	592
	Removal from a 2-3 Tree	592
	Exercises for Section 11.4	594

11.5	2-3-4 and B-Trees	594
	2-3-4 Trees	594
	Implementation of TwoThreeFourTree Class	597
	Relating 2-3-4 Trees to Red-Black Trees	601
	B-Trees	602
	Exercises for Section 11.5	603
	Chapter Review, Exercises, and Programming Projects	604
Chapter 12 Graphs		613
12.1	Graph Terminology	614
	Visual Representation of Graphs	614
	Directed and Undirected Graphs	615
	Paths and Cycles	616
	Relationship Between Graphs and Trees	618
	Graph Applications	618
	Exercises for Section 12.1	619
12.2	The Graph ADT and Edge Class	619
	Exercises for Section 12.2	622
12.3	Implementing the Graph ADT	622
	Adjacency List	622
	Adjacency Matrix	623
	Overview of the Hierarchy	624
	Class AbstractGraph	625
	The ListGraph Class	627
	The MatrixGraph Class	630
	Comparing Implementations	630
	Exercises for Section 12.3	631
12.4	Traversals of Graphs	632
	Breadth-First Search	632
	Depth-First Search	637
	Exercises for Section 12.4	644
12.5	Applications of Graph Traversals	645
	<i>Case Study:</i> Shortest Path Through a Maze	645
	<i>Case Study:</i> Topological Sort of a Graph	649
	Exercises for Section 12.5	652
12.6	Algorithms Using Weighted Graphs	652
	Finding the Shortest Path from a Vertex to All Other Vertices	652
	Minimum Spanning Trees	656
	Exercises for Section 12.6	660
	Chapter Review, Exercises, and Programming Projects	661

Appendix A Introduction to Java**669**

A.1	The Java Environment and Classes	670
	The Java Virtual Machine 671	
	The Java Compiler 671	
	Classes and Objects 671	
	The Java API 672	
	The import Statement 672	
	Method main 672	
	Execution of a Java Program 673	
	Exercises for Section A.1 674	
A.2	Primitive Data Types and Reference Variables	674
	Primitive Data Types 674	
	Primitive-Type Variables 675	
	Primitive-Type Constants 676	
	Operators 676	
	Postfix and Prefix Increment 676	
	Type Compatibility and Conversion 678	
	Referencing Objects 679	
	Creating Objects 679	
	Exercises for Section A.2 680	
A.3	Java Control Statements	680
	Sequence and Compound Statements 680	
	Selection and Repetition Control 681	
	Nested if Statements 683	
	The switch Statement 684	
	Exercises for Section A.3 685	
A.4	Methods and Class Math	685
	The Instance Methods println and print 685	
	Call-by-Value Arguments 686	
	The Class Math 687	
	Escape Sequences 688	
	Exercises for Section A.4 689	
A.5	The String, StringBuffer, and StringTokenizer Classes	689
	The String Class 689	
	Strings Are Immutable 692	
	The Garbage Collector 692	
	Comparing Objects 693	
	The StringBuffer Class 694	
	The StringTokenizer Class 695	
	Exercises for Section A.5 697	
A.6	Wrapper Classes for Primitive Types	698
	Exercises for Section A.6 699	

A.7	Defining Your Own Classes	700
	Private Data Fields, Public Methods	704
	Constructors	704
	Modifier and Accessor Methods	705
	Use of <code>this.</code> in a Method	706
	The Method <code>toString</code>	706
	The Method <code>equals</code>	706
	Declaring Local Variables in Class <code>Person</code>	707
	An Application That Uses Class <code>Person</code>	708
	Objects as Arguments	709
	Classes as Components of Other Classes	710
	Java Documentation Style for Classes and Methods	710
	Exercises for Section A.7	713
A.8	Arrays	714
	Data Field <code>length</code>	716
	Method <code>System.arraycopy</code>	716
	Array Data Fields	717
	Array Results and Arguments	719
	Arrays of Arrays	719
	Exercises for Section A.8	722
A.9	Input/Output Using Class <code>JOptionPane</code>	723
	Converting Numeric Strings to Numbers	724
	GUI Menus Using Method <code>showOptionDialog</code>	725
	Exercises for Section A.9	725
A.10	Input/Output Using Streams	726
	Input Streams	726
	Console Input	726
	Output Streams	727
	Passing Arguments to Method <code>main</code>	727
	Closing Streams	727
	Exceptions	728
	A Complete File-Processing Application	728
	Tokenized Input	730
	Exercises for Section A.10	730
	Chapter Review, Exercises, and Programming Projects	731
Appendix B Overview of UML		737
<hr/>		
B.1	The Class Diagram	738
	Representing Classes and Interfaces	738
	Generalization	741
	Inner or Nested Classes	742
	Association	742
	Aggregation and Composition	743

B.2	Sequence Diagrams	744
	Time Axis	744
	Objects	744
	Life Lines	746
	Activation Bars	746
	Messages	746
	Use of Notes	746

Appendix C	Event-Oriented Programming	747
-------------------	-----------------------------------	------------

C.1	Elements of an Event-Oriented Application	748
	Components and Events	749
	Event Listeners	750
	The ActionListener Interface	750
	Registering an Event Listener	751
	Creating a User Interface	752
	Exercises for Section C.1	755
C.2	Overview of the AWT and Swing Hierarchy	756
	Example and Overview: Two Circles	757
	JFrame	761
	JPanel	762
	Graphics	763
	Graphics Coordinates	764
	Exercises for Section C.2	764
C.3	Layout Managers	764
	Border Layout	765
	Flow Layout	767
	Box Layout	768
	Grid Layout	769
	Combining Layouts	770
	Exercises for Section C.3	771
C.4	Components for Data Entry	771
	Check Box	772
	Radio Button	773
	Combo Box	775
	Text Field	776
	Label	778
	Text Area	778
	Exercises for Section C.4	778
C.5	Using Data Entry Components in a GUI	779
	<i>Case Study:</i> Liquid Volume Converter	779
	Limiting the Number of Significant Digits	784
	Formatting Currency for Different Locales	785
	Exercises for Section C.5	786

C.6	Menus and Toolbars	787
	The Classes JMenuItem, JMenu, and JMenuBar	787
	Icons	789
	Toolbars	789
	<i>Case Study: A Drawing Application</i>	791
	Exercises for Section C.6	799
C.7	Processing Mouse Events	799
	MouseListener and MouseMotionAdapter	800
	<i>Case Study: A Drawing Application (Continued)</i>	801
	Exercises for Section C.7	809
	Chapter Review, Exercises, and Programming Projects	809
	Glossary	815
	Index	825