

GLOBAL  
EDITION



# Absolute JAVA

SIXTH EDITION

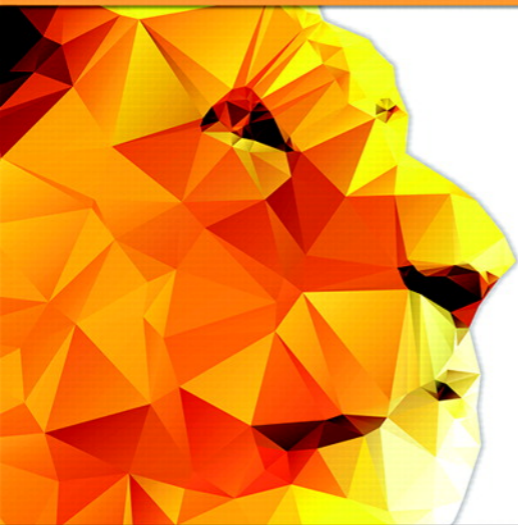
Walter Savitch

ALWAYS LEARNING

PEARSON

# ABSOLUTE JAVA™

SIXTH EDITION



Walter Savitch

**ABSOLUTE**

**JAVA™**

**6th Edition  
Global Edition**

This page intentionally left blank

# ABSOLUTE

# JAVA™

**6th Edition**  
**Global Edition**

## Walter Savitch

*University of California, San Diego*

Contributor

## Kenrick Mock

*University of Alaska Anchorage*

**PEARSON**

Boston Columbus Indianapolis New York San Francisco Hoboken  
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montréal Toronto  
Delhi Mexico City São Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

*Vice President and Editorial Director, ECS:* Marcia J. Horton  
*Acquisitions Editor:* Matt Goldstein  
*Assistant Acquisitions Editor, Global Edition:* Aditee Agarwal  
*Editorial Assistant:* Kelsey Loanes  
*Product Marketing Manager:* Bram Van Kempen  
*Marketing Assistant:* Jon Bryant  
*Senior Managing Editor:* Scott Disanno  
*Production Project Manager:* Rose Kernan  
*Project Editor, Global Edition:* Radhika Raheja  
*Program Manager:* Carole Snyder  
*Global HE Director of Vendor Sourcing and Procurement:* Diane Hynes

*Director of Operations:* Nick Sklitsis  
*Operations Specialist:* Maura Zaldivar-Garcia  
*Cover Designer:* Lumina Datamatics  
*Manager, Rights and Permissions:* Rachel Youdelman  
*Associate Project Manager, Rights and Permissions:* Timothy Nicholls  
*Senior Manufacturing Controller, Production, Global Edition:* Trudy Kimber  
*Media Production Manager, Global Edition:* Vikram Kumar  
*Full-Service Project Management:* Niraj Bhatt, iEnergizer Aptara®, Ltd.  
*Composition:* iEnergizer Aptara®, Ltd.  
*Cover Image:* © LeicherOliver/Shutterstock

Pearson Education Limited  
Edinburgh Gate  
Harlow  
Essex CM20 2JE  
England

and Associated Companies throughout the world

Visit us on the World Wide Web at:  
[www.pearsonglobaleditions.com](http://www.pearsonglobaleditions.com)

© Pearson Education Limited 2016

The right of Walter Savitch and Kenrick Mock to be identified as the author of this work has been asserted by him in accordance with the Copyright, Designs and Patents Act 1988.

*Authorized adaptation from the United States edition, entitled Absolute JAVA, 6th Edition, 9781292109220 9780134041674 by Walter Savitch and Kenrick Mock published by Pearson Education © 2016.*

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without either the prior written permission of the publisher or a license permitting restricted copying in the United Kingdom issued by the Copyright Licensing Agency Ltd, Saffron House, 6–10 Kirby Street, London EC1N 8TS.

All trademarks used herein are the property of their respective owners. The use of any trademark in this text does not vest in the author or publisher any trademark ownership rights in such trademarks, nor does the use of such trademarks imply any affiliation with or endorsement of this book by such owners.

British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library

10 9 8 7 6 5 4 3 2 1

ISBN 10: 129210922X

ISBN 13: 9781292109220

Typeset in Adobe Garamond 10.5/12 by iEnergizer Aptara®, Ltd.

Printed and bound by Courier Westford in Malaysia

# Preface

This book is designed to serve as a textbook and reference for programming in the Java language. Although it does include programming techniques, it is organized around the features of the Java language rather than any particular curriculum of programming techniques. The main audience I had in mind when writing this book was undergraduate students who have not had extensive programming experience with the Java language. As such, it would be a suitable Java text or reference for either a first programming course or a later computer science course that uses Java. This book is designed to accommodate a wide range of users. The introductory chapters are written at a level that is accessible to beginners, while the boxed sections of those chapters serve to quickly introduce more experienced programmers to basic Java syntax. Later chapters are still designed to be accessible, but are written at a level suitable for students who have progressed to these more advanced topics.

## CHANGES IN THIS EDITION

This sixth edition presents the same programming philosophy as the fifth edition. For instructors, you can teach the same course, presenting the same topics in the same order with no changes in the material covered or the chapters assigned. The changes to this edition consist almost exclusively of supplementary material added to the chapters of the previous edition, namely:

- An introduction to functional programming with Java 8's lambda expressions.
- Additional content and examples on looping, networking, and exception handling.
- Introduction to building GUIs using JavaFX.
- Fifteen new programming projects.
- Five new video notes for a total of 51 video notes. These videos cover specific topics and offer solutions to selected programming projects. The videos walk students through the process of problem solving and coding to reinforce key programming concepts. An icon appears in the margin of the book when a video is available about the corresponding topic in the text.

## NO NONSTANDARD SOFTWARE

Only classes in the standard Java libraries are used. No nonstandard software is used anywhere in the book.

## JAVA COVERAGE

All programs have been tested with Java 8. Oracle is not proposing any changes to future versions of Java that would affect the approach in this book.

## OBJECT-ORIENTED PROGRAMMING

This book gives extensive coverage of encapsulation, inheritance, and polymorphism as realized in the Java language. The chapters on Swing GUIs provide coverage of and extensive practice with event driven programming.

## FLEXIBILITY IN TOPIC ORDERING

This book allows instructors wide latitude in reordering the material. This is important if a book is to serve as a reference. It is also in keeping with my philosophy of writing books that accommodate themselves to an instructor's style rather than tying the instructor to an author's personal preference of topic ordering. With this in mind, each chapter has a prerequisite section at the beginning; this section explains what material must be covered before doing each section of the chapter. Starred sections, which are explained next, further add to flexibility.

## STARRED SECTIONS

Each chapter has a number of starred (★) sections, which can be considered optional. These sections contain material that beginners might find difficult and that can be omitted or delayed without hurting the continuity of the text. It is hoped that eventually the reader would return and cover this material. For more advanced students, the starred sections should not be viewed as optional.

## ACCESSIBLE TO STUDENTS

It is not enough for a book to present the right topics in the right order. It is not even enough for it to be clear and correct when read by an instructor or other expert. The material needs to be presented in a way that is accessible to the person who does not yet know the content. Like my other textbooks that have proven to be very popular, this book was written to be friendly and accessible to the student.

## SUMMARY BOXES

Each major point is summarized in a short boxed section. These boxed sections are spread throughout each chapter. They serve as summaries of the material, as a quick reference source, and as a way to quickly learn the Java syntax for features the reader knows about in general but for which he or she needs to know the Java particulars.

## SELF-TEST EXERCISES

Each chapter contains numerous Self-Test Exercises at strategic points in the chapter. Complete answers for all the Self-Test Exercises are given at the end of each chapter.





## VIDEO NOTES

VideoNotes are step-by-step videos that guide readers through the solution to an end-of-chapter problem or further illuminate a concept presented in the text. Icons in the text indicate where a VideoNote enhances a topic. Fully navigable problems allow for self-paced instruction. VideoNotes are located at [www.pearsonglobaleditions.com/savitch](http://www.pearsonglobaleditions.com/savitch).

## OTHER FEATURES

Pitfall sections, programming tip sections, and examples of complete programs with sample I/O are given throughout each chapter. Each chapter ends with a summary section and a collection of programming projects suitable to assign to students.

## HOW TO ACCESS INSTRUCTOR AND STUDENT RESOURCE MATERIALS

**Online Practice and Assessment with MyProgrammingLab™.** MyProgrammingLab helps students fully grasp the logic, semantics, and syntax of programming. Through practice exercises and immediate, personalized feedback, MyProgrammingLab improves the programming competence of beginning students who often struggle with the basic concepts and paradigms of popular high-level programming languages.

A self-study and homework tool, a MyProgrammingLab course consists of hundreds of small practice problems organized around the structure of this textbook. For students, the system automatically detects errors in the logic and syntax of their code submissions and offers targeted hints that enable students to figure out what went wrong—and why. For instructors, a comprehensive gradebook tracks correct and incorrect answers and stores the code inputted by students for review.

For a full demonstration, to see feedback from instructors and students, or to get started using MyProgrammingLab in your course, visit [www.myprogramminglab.com](http://www.myprogramminglab.com).

## SUPPORT MATERIAL

The following support materials are available to all users of this Global Editions book at [www.pearsonglobaleditions.com/savitch](http://www.pearsonglobaleditions.com/savitch):

- Source code from the book

The following resources are available to qualified instructors only at [www.pearsonglobaleditions.com/savitch](http://www.pearsonglobaleditions.com/savitch). Please contact your local sales representative for access information:

- Instructor's Manual with Solutions
- PowerPoint® slides

## ACKNOWLEDGMENTS

Numerous individuals have contributed invaluable help and support in making this book happen: My former editor, Susan Hartman at Addison-Wesley, first conceived of the idea for this book and worked with me on the first editions; My current editor, Matt

Goldstein, provided support and inspiration for getting subsequent editions reviewed, revised, and out the door; Kelsey Loanes, Rose Kernan, Demetrius Hall, and the other fine people at Pearson also provided valuable assistance and encouragement.

The following reviewers provided corrections and suggestions for this book. Their contributions were a great help. I thank them all. In alphabetical order they are:

Jim Adams	Chandler-Gilbert Community College
Gerald W. Adkins	Georgia College & State University
Dr. Bay Arinze	Drexel University
Tamara Babaian	Bentley University
James Baldo	George Mason University
Prof. Richard G. Baldwin	Austin Community College
Kevin Bierre	Rochester Institute of Technology
Jon Bjornstad	Gavilan College
Janet Brown-Sederberg	Massasoit Community College
Tom Brown	Texas A&M University, Commerce
Charlotte Busch	Texas A&M University, Corpus Christi
Stephen Chandler	NW Shoals Community College
Hong Cheng	Southern Arkansas University
KY Daisy Fan	Cornell University
Adrienne Decker	University of Buffalo
Brian Downs	Century College
Jeffrey Edgington	University of Denver
Keith Frikken	Miami University
Ahmad Ghafarian	North Georgia College & State University
Arthur Geis	College of DuPage
Massoud Ghyam	University of Southern California
Susan G. Glenn	Gordon College
Nigel Gwee	Louisiana State University
Judy Hankins	Middle Tennessee State University
May Hou	Norfolk State University
Sterling Hough	NHTI
Chris Howard	DeVry University
Eliot Jacobson	University of California, Santa Barbara
Balaji Janamanchi	Texas Tech University
Suresh Kalathur	Boston University
Edwin Kay	Lehigh University
Dr. Clifford R. Kетtemborough	IT Consultant and Professor

Frank Levey	Manatee Community College
Xia Lin	Drexel University
Mark M. Meysenburg	Doane College
Sridhar P. Nerur	The University of Texas at Arlington
Hoang M. Nguyen	Deanza College
Rick Ord	University of California, San Diego
Prof. Bryson R. Payne	North Georgia College & State University
David Primeaux	Virginia Commonwealth University
Neil Rhodes	University of California, San Diego
W. Brent Seales	University of Kentucky
Lili Shashaani	Duquesne University
Riyaz Sikora	The University of Texas at Arlington
Christopher Simpkins	Georgia Tech
Jeff Six	University of Delaware
Donald J Smith	Community College of Allegheny County
Tom Smith	Skidmore College
William Smith	Tulsa Community College
Xueqing (Clare) Tang	Governors State University
Ronald F. Taylor	Wright State University
Thomas VanDrunen	Wheaton College
Shon Vick	University of Maryland, Baltimore County
Natalie S. Wear	University of South Florida
Dale Welch	University of West Florida
David A. Wheeler	
Wook-Sung Yoo	Gannon University

Special thanks goes to Kenrick Mock (University of Alaska Anchorage) who executed the updating of this edition. He once again had the difficult job of satisfying me, the editor, and himself. I thank him for a truly excellent job.

Walter Savitch

Pearson wishes to thank and acknowledge the following people for their work on the Global Edition:

#### **Contributors**

Vikas Deep Dhiman	Amity University
Madhurima Hooda	Amity University

#### **Reviewers**

Manasa Rengarar	NMAM Institute of Technology
S.H. Chung	Wawasan Open University
Issam A. El-Moughrabi	Gulf University of Science and Technology

## LOCATION OF VIDEONOTES IN THE TEXT

[www.pearsonglobaleditions.com/savitch](http://www.pearsonglobaleditions.com/savitch)



VideoNote

<b>Chapter 1</b>	Compiling a Java Program, page 42 Solution to Programming Project 1.7, page 88
<b>Chapter 2</b>	Using <code>printf</code> , page 94 Pitfalls Involving <code>nextLine()</code> , page 115 Solution to Programming Project 2.11, page 129 Solution to Programming Project 2.12, page 130
<b>Chapter 3</b>	Nested Loop Example, page 177 Debugging Walkthrough, page 184 Generating Random Numbers, page 191 Solution to Programming Project 3.9, page 202 Solution to Programming Project 3.13, page 203
<b>Chapter 4</b>	Information Hiding Example, page 239 Example Using the <code>StringTokenizer</code> Class on a CSV File, page 279 Solution to Programming Project 4.9, page 287
<b>Chapter 5</b>	Deep Copy vs. Shallow Copy Example, page 353 Solution to Programming Project 5.9, page 376
<b>Chapter 6</b>	Arrays of Objects, page 390 Solution to Programming Project 6.8, page 454 Solution to Programming Project 6.15, page 456
<b>Chapter 7</b>	Inheritance Walkthrough, page 464 Solution to Programming Project 7.3, page 509 Solution to Programming Project 7.5, page 511
<b>Chapter 8</b>	Late Binding Example, page 518 Solution to Programming Project 8.1, page 550 Solution to Programming Project 8.9, page 553
<b>Chapter 9</b>	Solution to Programming Project 9.1, page 609 Solution to Programming Project 9.7, page 611
<b>Chapter 10</b>	Reading a Text File, page 625 Solution to Programming Project 10.1, page 679 Solution to Programming Project 10.9, page 681
<b>Chapter 11</b>	Recursion and the Stack, page 696 Solution to Programming Project 11.3, page 720
<b>Chapter 12</b>	Solution to Programming Project 12.9, page 746
<b>Chapter 13</b>	Solution to Programming Project 13.1, page 790 Solution to Programming Project 13.11, page 793
<b>Chapter 14</b>	Solution to Programming Project 14.7, page 836
<b>Chapter 15</b>	Walkthrough of the Hash Table Class, page 906 Solution to Programming Project 15.1, page 931

<b>Chapter 16</b>	Using HashMap with a Custom Class, page 948 Solution to Programming Project 16.3, page 975 Solution to Programming Project 16.5, page 976
<b>Chapter 17</b>	GUI Layout Using an IDE, page 1009 Solution to Programming Project 17.1, page 1053
<b>Chapter 18</b>	Walkthrough of a Simple Drawing Program, page 1082 Solution to Programming Project 18.7, page 1117
<b>Chapter 19</b>	Walkthrough of a Program with Race Conditions, page 1134 Networking with Streams, page 1138 Functional Programming Example, page 1172 Solution to Programming Project 19.3, page 1196 Solution to Programming Project 19.11, page 1197
<b>Chapter 20</b>	No video notes (Chapter on website)

This page intentionally left blank

# Brief Contents

Chapter 1	GETTING STARTED	33
Chapter 2	CONSOLE INPUT AND OUTPUT	89
Chapter 3	FLOW OF CONTROL	131
Chapter 4	DEFINING CLASSES I	205
Chapter 5	DEFINING CLASSES II	291
Chapter 6	ARRAYS	377
Chapter 7	INHERITANCE	459
Chapter 8	POLYMORPHISM AND ABSTRACT CLASSES	515
Chapter 9	EXCEPTION HANDLING	555
Chapter 10	FILE I/O	613
Chapter 11	RECURSION	683
Chapter 12	UML AND PATTERNS	725
Chapter 13	INTERFACES AND INNER CLASSES	747
Chapter 14	GENERICS AND THE <code>ArrayList</code> CLASS	795
Chapter 15	LINKED DATA STRUCTURES	839
Chapter 16	COLLECTIONS, MAPS AND ITERATORS	935
Chapter 17	SWING I	981
Chapter 18	SWING II	1057
Chapter 19	JAVA NEVER ENDS	1119
Chapter 20	APPLETS AND HTML (online at <a href="http://www.pearsonglobaleditions.com/savitch">www.pearsonglobaleditions.com/savitch</a> )	
Appendix 1	KEYWORDS	1199
Appendix 2	PRECEDENCE AND ASSOCIATIVITY RULES	1201
Appendix 3	ASCII CHARACTER SET	1203
Appendix 4	FORMAT SPECIFICATIONS FOR <code>printf</code>	1205
Appendix 5	SUMMARY OF CLASSES AND INTERFACES	1207
	INDEX	1275

This page intentionally left blank



# Contents

## Chapter 1 Getting Started 33

### 1.1 INTRODUCTION TO JAVA 34

- Origins of the Java Language ★ 34
- Objects and Methods 35
- Applets ★ 36
- A Sample Java Application Program 37
- Byte-Code and the Java Virtual Machine 40
- Class Loader ★ 42
- Compiling a Java Program or Class 42
- Running a Java Program 43
- TIP: Error Messages 44

### 1.2 EXPRESSIONS AND ASSIGNMENT STATEMENTS 45

- Identifiers 45
- Variables 47
- Assignment Statements 48
- TIP: Initialize Variables 50
- More Assignment Statements ★ 51
- Assignment Compatibility 52
- Constants 53
- Arithmetic Operators and Expressions 55
- Parentheses and Precedence Rules ★ 56
- Integer and Floating-Point Division 58
- PITFALL: Round-Off Errors in Floating-Point Numbers 59
- PITFALL: Division with Whole Numbers 60
- Type Casting 61
- Increment and Decrement Operators 62

### 1.3 THE CLASS `String` 65

- String Constants and Variables 65
- Concatenation of Strings 66
- Classes 67
- String Methods 69
- Escape Sequences 74
- String Processing 75
- The Unicode Character Set ★ 75

- 1.4 PROGRAM STYLE 78**
  - Naming Constants 78
  - Java Spelling Conventions 80
  - Comments 81
  - Indenting 82
  - Chapter Summary 83
  - Answers to Self-Test Exercises 84
  - Programming Projects 86

## Chapter 2 Console Input and Output 89

- 2.1 SCREEN OUTPUT 90**
  - `System.out.println` 90
  - TIP: Different Approaches to Formatting Output 93
  - Formatting Output with `printf` 93
  - TIP: Formatting Monetary Amounts with `printf` 97
  - TIP: Legacy Code 98
  - Money Formats Using `NumberFormat` ★ 99
  - Importing Packages and Classes 102
  - The `DecimalFormat` Class ★ 104
- 2.2 CONSOLE INPUT USING THE `Scanner` CLASS 108**
  - The `Scanner` Class 108
  - PITFALL: Dealing with the Line Terminator, `'\n'` 115
  - The Empty String 116
  - TIP: Prompt for Input 116
  - TIP: Echo Input 116
  - EXAMPLE: Self-Service Checkout 118
  - Other Input Delimiters 119
- 2.3 INTRODUCTION TO FILE INPUT 121**
  - The `Scanner` Class for Text File Input 121
  - Chapter Summary 124
  - Answers to Self-Test Exercises 124
  - Programming Projects 127

## Chapter 3 Flow of Control 131

- 3.1 BRANCHING MECHANISM 132**
  - `if-else` Statements 132
  - Omitting the `else` 133
  - Compound Statements 134
  - TIP: Placing of Braces 135
  - Nested Statements 136

Multiway `if-else` Statement 136  
EXAMPLE: State Income Tax 137  
The `switch` Statement 139  
PITFALL: Forgetting a `break` in a `switch` Statement 143  
The Conditional Operator ★ 144

### 3.2 BOOLEAN EXPRESSIONS 145

Simple Boolean Expressions 145  
PITFALL: Using `=` in Place of `==` 146  
PITFALL: Using `==` with Strings 147  
Lexicographic and Alphabetic Order 148  
Building Boolean Expressions 151  
PITFALL: Strings of Inequalities 152  
Evaluating Boolean Expressions 152  
TIP: Naming Boolean Variables 155  
Short-Circuit and Complete Evaluation 156  
Precedence and Associativity Rules 157

### 3.3 LOOPS 164

`while` Statement and `do-while` Statement 164  
Algorithms and Pseudocode 166  
EXAMPLE: Averaging a List of Scores 169  
The `for` Statement 170  
The Comma in `for` Statements 173  
TIP: Repeat *N* Times Loops 175  
PITFALL: Extra Semicolon in a `for` Statement 175  
PITFALL: Infinite Loops 176  
Nested Loops 177  
The `break` and `continue` Statements ★ 180  
The `exit` Statement 181

### 3.4 DEBUGGING 182

Loop Bugs 182  
Tracing Variables 182  
General Debugging Techniques 183  
EXAMPLE: Debugging an Input Validation Loop 184  
Preventive Coding 188  
Assertion Checks ★ 189

### 3.5 RANDOM NUMBER GENERATION ★ 191

The `Random` Object 191  
The `Math.random()` Method 193  
Chapter Summary 194  
Answers to Self-Test Exercises 194  
Programming Projects 200

**Chapter 4 Defining Classes I 205**

**4.1 CLASS DEFINITIONS 206**

- Instance Variables and Methods 209
- More about Methods 212
- TIP: Any Method Can Be Used as a `void` Method 216
- Local Variables 218
- Blocks 219
- TIP: Declaring Variables in a `for` Statement 220
- Parameters of a Primitive Type 220
- PITFALL: Use of the Terms “Parameter” and “Argument” 227
- Simple Cases with Class Parameters 229
- The `this` Parameter 229
- Methods That Return a Boolean Value 231
- The Methods `equals` and `toString` 234
- Recursive Methods 237
- TIP: Testing Methods 237

**4.2 INFORMATION HIDING AND ENCAPSULATION 239**

- `public` and `private` Modifiers 240
- EXAMPLE: Yet Another Date Class 241
- Accessor and Mutator Methods 242
- TIP: A Class Has Access to Private Members of All Objects of the Class 247
- TIP: Mutator Methods Can Return a Boolean Value ★ 248
- Preconditions and Postconditions 249

**4.3 OVERLOADING 250**

- Rules for Overloading 250
- PITFALL: Overloading and Automatic Type Conversion 254
- PITFALL: You Cannot Overload Based on the Type Returned 256

**4.4 CONSTRUCTORS 258**

- Constructor Definitions 258
- TIP: You Can Invoke Another Method in a Constructor 266
- TIP: A Constructor Has a `this` Parameter 266
- TIP: Include a No-Argument Constructor 267
- EXAMPLE: The Final Date Class 268
- Default Variable Initializations 269
- An Alternative Way to Initialize Instance Variables 269
- EXAMPLE: A Pet Record Class 270
- The `StringTokenizer` Class ★ 274
- Chapter Summary 279
- Answers to Self-Test Exercises 280
- Programming Projects 285

## Chapter 5 Defining Classes II 291

### 5.1 STATIC METHODS AND STATIC VARIABLES 293

Static Methods 293

PITFALL: Invoking a Nonstatic Method Within a Static Method 295

TIP: You Can Put a `main` in Any Class 296

Static Variables 300

The `Math` Class 305

Wrapper Classes 309

Automatic Boxing and Unboxing 310

Static Methods in Wrapper Classes 312

PITFALL: A Wrapper Class Does Not Have a No-Argument Constructor 315

### 5.2 REFERENCES AND CLASS PARAMETERS 316

Variables and Memory 317

References 318

Class Parameters 323

PITFALL: Use of `=` and `==` with Variables of a Class Type 327

The Constant `null` 329

PITFALL: Null Pointer Exception 330

The `new` Operator and Anonymous Objects 330

EXAMPLE: Another Approach to Keyboard Input ★ 331

TIP: Use Static Imports ★ 333

### 5.3 USING AND MISUSING REFERENCES 335

EXAMPLE: A Person Class 336

PITFALL: `null` Can Be an Argument to a Method 341

Copy Constructors 345

PITFALL: Privacy Leaks 347

Mutable and Immutable Classes 351

TIP: Deep Copy versus Shallow Copy 353

TIP: Assume Your Coworkers Are Malicious 354

### 5.4 PACKAGES AND `javadoc` 354

Packages and `import` Statements 355

The Package `java.lang` 356

Package Names and Directories 356

PITFALL: Subdirectories Are Not Automatically Imported 359

The Default Package 359

PITFALL: Not Including the Current Directory in Your Class Path 360

Specifying a Class Path When You Compile ★ 360

Name Clashes ★ 361

Introduction to `javadoc` ★ 362

Commenting Classes for `javadoc` ★ 362

Running `javadoc` ★ 364

Chapter Summary 366  
Answers to Self-Test Exercises 367  
Programming Projects 371

## Chapter 6 Arrays 377

### 6.1 INTRODUCTION TO ARRAYS 378

Creating and Accessing Arrays 379  
The `length` Instance Variable 382  
TIP: Use `for` Loops with Arrays 384  
PITFALL: Array Indices Always Start with Zero 384  
PITFALL: Array Index Out of Bounds 384  
Initializing Arrays 385  
PITFALL: An Array of Characters Is Not a String 387

### 6.2 ARRAYS AND REFERENCES 388

Arrays Are Objects 388  
PITFALL: Arrays with a Class Base Type 390  
Array Parameters 390  
PITFALL: Use of `=` and `==` with Arrays 392  
Arguments for the Method `main` ★ 397  
Methods that Return an Array 399

### 6.3 PROGRAMMING WITH ARRAYS 400

Partially Filled Arrays 401  
EXAMPLE: A Class for Partially Filled Arrays 404  
TIP: Accessor Methods Need Not Simply Return Instance Variables 408  
The “for-each” Loop ★ 408  
Methods with a Variable Number of Parameters ★ 412  
EXAMPLE: A String Processing Example ★ 415  
Privacy Leaks with Array Instance Variables 416  
EXAMPLE: Sorting an Array 420  
Enumerated Types ★ 424  
TIP: Enumerated Types in `switch` Statements ★ 429

### 6.4 MULTIDIMENSIONAL ARRAYS 431

Multidimensional Array Basics 431  
Using the `length` Instance Variable 434  
Ragged Arrays ★ 435  
Multidimensional Array Parameters and Returned Values 435  
EXAMPLE: A Grade Book Class 436  
  
Chapter Summary 442  
Answers to Self-Test Exercises 443  
Programming Projects 450

## Chapter 7 Inheritance 459

### 7.1 INHERITANCE BASICS 460

- Derived Classes 461
- Overriding a Method Definition 471
- Changing the Return Type of an Overridden Method 471
- Changing the Access Permission of an Overridden Method 472
- PITFALL: Overriding versus Overloading 473
- The `super` Constructor 474
- The `this` Constructor 476
- TIP: An Object of a Derived Class Has More than One Type 477
- PITFALL: The Terms *Subclass* and *Superclass* 480
- EXAMPLE: An Enhanced `StringTokenizer` Class ★ 481

### 7.2 ENCAPSULATION AND INHERITANCE 484

- PITFALL: Use of Private Instance Variables from the Base Class 485
- PITFALL: Private Methods Are Effectively Not Inherited 486
- Protected and Package Access 487
- PITFALL: Forgetting about the Default Package 490
- PITFALL: A Restriction on Protected Access ★ 490

### 7.3 PROGRAMMING WITH INHERITANCE 493

- TIP: Static Variables Are Inherited 493
- TIP: “is a” versus “has a” 493
- Access to a Redefined Base Method 493
- PITFALL: You Cannot Use Multiple `super`s 495
- The Class Object 496
- The Right Way to Define `equals` 497
- TIP: `getClass` versus `instanceof` ★ 499
- Chapter Summary 504
- Answers to Self-Test Exercises 505
- Programming Projects 508

## Chapter 8 Polymorphism and Abstract Classes 515

### 8.1 POLYMORPHISM 516

- Late Binding 517
- The `final` Modifier 519
- EXAMPLE: Sales Records 520
- Late Binding with `toString` 527
- PITFALL: No Late Binding for Static Methods 528
- Downcasting and Upcasting 529
- PITFALL: Downcasting 533

- TIP: Checking to See Whether Downcasting Is Legitimate ★ 533
- A First Look at the `clone` Method 536
- PITFALL: Sometimes the `clone` Method Return Type Is `Object` 537
- PITFALL: Limitations of Copy Constructors ★ 538

## 8.2 ABSTRACT CLASSES 541

- Abstract Classes 542
- PITFALL: You Cannot Create Instances of an Abstract Class 546
- TIP: An Abstract Class Is a Type 547
  
- Chapter Summary 548
- Answers to Self-Test Exercises 548
- Programming Projects 550

## Chapter 9 Exception Handling 555

### 9.1 EXCEPTION HANDLING BASICS 557

- `try-catch` Mechanism 557
- Exception Handling with the `Scanner` Class 559
- TIP: Exception Controlled Loops 560
- Throwing Exceptions 562
- EXAMPLE: A Toy Example of Exception Handling 564
- Exception Classes 569
- Exception Classes from Standard Packages 570
- Defining Exception Classes 572
- TIP: Preserve `getMessage` 576
- TIP: An Exception Class Can Carry a Message of Any Type 578
- Multiple `catch` Blocks 583
- PITFALL: Catch the More Specific Exception First 585

### 9.2 THROWING EXCEPTIONS IN METHODS 588

- Throwing an Exception in a Method 588
- Declaring Exceptions in a `throws` Clause 590
- Exceptions to the Catch or Declare Rule 593
- `throws` Clause in Derived Classes 594
- When to Use Exceptions 595
- Example: Retrieving a High Score 596
- Event-Driven Programming ★ 599

### 9.3 MORE PROGRAMMING TECHNIQUES FOR EXCEPTION HANDLING 601

- PITFALL: Nested `try-catch` Blocks 601
- The `finally` Block ★ 601
- Rethrowing an Exception ★ 603
- The `AssertionError` Class ★ 603



`ArrayIndexOutOfBoundsException` 604

Chapter Summary 604

Answers to Self-Test Exercises 605

Programming Projects 609

## Chapter 10 File I/O 613

### 10.1 INTRODUCTION TO FILE I/O 614

Streams 614

Text Files and Binary Files 615

### 10.2 TEXT FILES 616

Writing to a Text File 616

PITFALL: A `try` Block Is a Block 622

PITFALL: Overwriting an Output File 622

Appending to a Text File 623

TIP: `toString` Helps with Text File Output 624

Reading from a Text File 625

Reading a Text File Using `Scanner` 625

Testing for the End of a Text File with `Scanner` 628

Reading a Text File Using `BufferedReader` 635

TIP: Reading Numbers with `BufferedReader` 639

Testing for the End of a Text File with `BufferedReader` 639

Path Names 641

Nested Constructor Invocations 642

`System.in`, `System.out`, and `System.err` 643

### 10.3 THE `File` CLASS 645

Programming with the `File` Class 645

### 10.4 BINARY FILES ★ 649

Writing Simple Data to a Binary File 650

UTF and `writeUTF` 654

Reading Simple Data from a Binary File 655

Checking for the End of a Binary File 660

PITFALL: Checking for the End of a File in the Wrong Way 661

Binary I/O of Objects 662

The `Serializable` Interface 663

PITFALL: Mixing Class Types in the Same File 666

Array Objects in Binary Files 666

### 10.5 RANDOM ACCESS TO BINARY FILES ★ 668

Reading and Writing to the Same File 668

PITFALL: `RandomAccessFile` Need Not Start Empty 674

Chapter Summary 674  
Answers to Self-Test Exercises 675  
Programming Projects 679

## Chapter 11 Recursion 683

### 11.1 RECURSIVE void METHODS 685

EXAMPLE: Vertical Numbers 685  
Tracing a Recursive Call 688  
A Closer Look at Recursion 691  
PITFALL: Infinite Recursion 693  
Stacks for Recursion ★ 694  
PITFALL: Stack Overflow ★ 696  
Recursion versus Iteration 696

### 11.2 RECURSIVE METHODS THAT RETURN A VALUE 697

General Form for a Recursive Method That Returns a Value 698  
EXAMPLE: Another Powers Method 698

### 11.3 THINKING RECURSIVELY 703

Recursive Design Techniques 703  
Binary Search ★ 704  
Efficiency of Binary Search ★ 710  
EXAMPLE: Finding a File 712  
  
Chapter Summary 715  
Answers to Self-Test Exercises 715  
Programming Projects 720

## Chapter 12 UML and Patterns 725

### 12.1 UML 726

History of UML 727  
UML Class Diagrams 727  
Class Interactions 728  
Inheritance Diagrams 728  
More UML 730

### 12.2 PATTERNS ★ 731

Adaptor Pattern ★ 731  
The Model-View-Controller Pattern ★ 732  
EXAMPLE: A Sorting Pattern 733  
Restrictions on the Sorting Pattern 739  
Efficiency of the Sorting Pattern ★ 739

TIP: Pragmatics and Patterns	740
Pattern Formalism	740
Chapter Summary	741
Answers to Self-Test Exercises	741
Programming Projects	743

## Chapter 13 Interfaces and Inner Classes 747

### 13.1 INTERFACES 749

Interfaces	749
Abstract Classes Implementing Interfaces	751
Derived Interfaces	751
PITFALL: Interface Semantics Are Not Enforced	753
The Comparable Interface	755
EXAMPLE: Using the Comparable Interface	756
Defined Constants in Interfaces	761
PITFALL: Inconsistent Interfaces	762
The Serializable Interface ★	765
The Cloneable Interface	765

### 13.2 SIMPLE USES OF INNER CLASSES 770

Helping Classes	770
TIP: Inner and Outer Classes Have Access to Each Other's Private Members	771
EXAMPLE: A Bank Account Class	771
The .class File for an Inner Class	775
PITFALL: Other Uses of Inner Classes	776

### 13.3 MORE ABOUT INNER CLASSES 776

Static Inner Classes	776
Public Inner Classes	777
TIP: Referring to a Method of the Outer Class	779
Nesting Inner Classes	781
Inner Classes and Inheritance	781
Anonymous Classes	782
TIP: Why Use Inner Classes?	784
Chapter Summary	785
Answers to Self-Test Exercises	785
Programming Projects	790

## Chapter 14 Generics and the ArrayList Class 795

### 14.1 THE ArrayList CLASS 797

Using the ArrayList Class	798
TIP: Summary of Adding to an ArrayList	802

Methods in the Class `ArrayList` 803  
 The “for-each” Loop 806  
 EXAMPLE: Golf Scores 809  
 TIP: Use `trimToSize` to Save Memory 812  
 PITFALL: The `clone` Method Makes a Shallow Copy ★ 812  
 The `Vector` Class 813  
 Parameterized Classes and Generics 814  
 PITFALL: Nonparameterized `ArrayList` and `Vector` Classes 814

## 14.2 GENERICS 814

Generic Basics 815  
 TIP: Compile with the `-Xlint` Option 817  
 EXAMPLE: A Generic Class for Ordered Pairs 817  
 PITFALL: A Generic Constructor Name Has No Type Parameter 820  
 PITFALL: You Cannot Plug in a Primitive Type for a Type Parameter 821  
 PITFALL: A Type Parameter Cannot Be Used Everywhere a Type Name  
 Can Be Used 821  
 PITFALL: An Instantiation of a Generic Class Cannot be an  
 Array Base Type 822  
 TIP: A Class Definition Can Have More Than One Type Parameter 823  
 PITFALL: A Generic Class Cannot Be an Exception Class 824  
 Bounds for Type Parameters 825  
 TIP: Generic Interfaces 828  
 Generic Methods ★ 828  
 Inheritance with Generic Classes ★ 830  
  
 Chapter Summary 832  
 Answers to Self-Test Exercises 832  
 Programming Projects 835

## Chapter 15 Linked Data Structures 839

### 15.1 JAVA LINKED LISTS 842

EXAMPLE: A Simple Linked List Class 842  
 Working with Linked Lists 846  
 PITFALL: Privacy Leaks 851  
 Node Inner Classes 852  
 EXAMPLE: A Generic Linked List 855  
 PITFALL: Using `Node` Instead of `Node<T>` 860  
 The `equals` Method for Linked Lists 860

### 15.2 COPY CONSTRUCTORS AND THE `clone` METHOD ★ 862

Simple Copy Constructors and `clone` Methods ★ 862  
 Exceptions ★ 863

PITFALL: The `clone` Method Is Protected in `object` ★ 865  
TIP: Use a Type Parameter Bound for a Better `clone` ★ 866  
EXAMPLE: A Linked List with a Deep Copy `clone` Method ★ 870  
TIP: Cloning Is an “All or Nothing” Affair 873

### 15.3 ITERATORS 873

Defining an Iterator Class 874  
Adding and Deleting Nodes 879

### 15.4 VARIATIONS ON A LINKED LIST 884

Doubly Linked List 884  
The Stack Data Structure 893  
The Queue Data Structure 895  
Running Times and Big- $O$  Notation 898  
Efficiency of Linked Lists 903

### 15.5 HASH TABLES WITH CHAINING 904

A Hash Function for Strings 905  
Efficiency of Hash Tables 908

### 15.6 SETS 909

Fundamental Set Operations 910  
Efficiency of Sets Using Linked Lists 915

### 15.7 TREES 916

Tree Properties 916  
EXAMPLE: A Binary Search Tree Class ★ 919  
Efficiency of Binary Search Trees ★ 924

Chapter Summary 925  
Answers to Self-Test Exercises 926  
Programming Projects 931

## Chapter 16 Collections, Maps and Iterators 935

### 16.1 COLLECTIONS 936

Wildcards 938  
The Collection Framework 938  
PITFALL: Optional Operations 944  
TIP: Dealing with All Those Exceptions 945  
Concrete Collection Classes 946  
Differences between `ArrayList<T>` and `Vector<T>` 956  
Nonparameterized Version of the Collection Framework ★ 956  
PITFALL: Omitting the `<T>` 957

**16.2 MAPS 957**

Concrete Map Classes 960

**16.3 ITERATORS 964**

The Iterator Concept 964

The `Iterator<T>` Interface 964

TIP: For-Each Loops as Iterators 967

List Iterators 968

PITFALL: `next` Can Return a Reference 970

TIP: Defining Your Own Iterator Classes 972

Chapter Summary 973

Answers to Self-Test Exercises 973

Programming Projects 974

**Chapter 17 Swing I 981****17.1 EVENT-DRIVEN PROGRAMMING 983**

Events and Listeners 983

**17.2 BUTTONS, EVENTS, AND OTHER SWING BASICS 984**

EXAMPLE: A Simple Window 985

PITFALL: Forgetting to Program the Close-Window Button 990

Buttons 991

Action Listeners and Action Events 992

PITFALL: Changing the Heading for `actionPerformed` 994

TIP: Ending a Swing Program 994

EXAMPLE: A Better Version of Our First Swing GUI 995

Labels 998

Color 999

EXAMPLE: A GUI with a Label and Color 1000

**17.3 CONTAINERS AND LAYOUT MANAGERS 1002**

Border Layout Managers 1003

Flow Layout Managers 1006

Grid Layout Managers 1007

Panels 1011

EXAMPLE: A Tricolor Built with Panels 1012

The `Container` Class 1016

TIP: Code a GUI's Look and Actions Separately 1019

The Model-View-Controller Pattern ★ 1020

**17.4 MENUS AND BUTTONS 1021**

- EXAMPLE: A GUI with a Menu 1021
- Menus, Menu Items, and Menu Bars 1021
- Nested Menus ★ 1026
- The `AbstractButton` Class 1026
- The `setActionCommand` Method 1029
- Listeners as Inner Classes ★ 1030

**17.5 TEXT FIELDS AND TEXT AREAS 1033**

- Text Areas and Text Fields 1034
- TIP: Labeling a Text Field 1040
- TIP: Inputting and Outputting Numbers 1040
- A Swing Calculator 1041
  
- Chapter Summary 1046
- Answers to Self-Test Exercises 1047
- Programming Projects 1053

**Chapter 18 Swing II 1057****18.1 WINDOW LISTENERS 1058**

- EXAMPLE: A Window Listener Inner Class 1060
- The `dispose` Method 1063
- PITFALL: Forgetting to Invoke `setDefaultCloseOperation` 1064
- The `WindowAdapter` Class 1064

**18.2 ICONS AND SCROLL BARS 1066**

- Icons 1066
- Scroll Bars 1072
- EXAMPLE: Components with Changing Visibility 1077

**18.3 THE `Graphics` CLASS 1081**

- Coordinate System for Graphics Objects 1081
- The Method `paint` and the Class `Graphics` 1082
- Drawing Ovals 1087
- Drawing Arcs 1087
- Rounded Rectangles ★ 1091
- `paintComponent` for Panels 1092
- Action Drawings and `repaint` 1092
- Some More Details on Updating a GUI ★ 1098