FM_TOC C7068 45563 Page i 11/20/09 eeh

FUNDAMENTALS OF JAVA™

AP® Computer Science Essentials for the A Exam, Fourth Edition

Kenneth Lambert and Martin Osborne

COURSE TECHNOLOGY
CENGAGE Learning™

Australia • Brazil • Japan • Korea • Mexico • Singapore • Spain • United Kingdom • United States

COURSE TECHNOLOGY
CENGAGE Learning™

Fundamentals of Java™: AP® Computer Science Essentials for the A Exam, Fourth Edition
Kenneth Lambert, Martin Osborne

Executive Editor: Donna Gridley

Product Manager: Alyssa Pratt

Development Editor: Ann Shaffer

Associate Product Manager: Amanda Lyons

Editorial Assistant: Kim Klasner

Content Project Manager: Matthew Hutchinson

Marketing Associate: Julie Schuster

Director of Manufacturing: Denise Powers

Manuscript Quality Assurance Lead: Jeff Schwartz

Manuscript Quality Assurance Reviewers:
Serge Palladino, Danielle Shaw, John Freitas

Art Director: Faith Brosnan

Cover Designer: © Wing-ip Ngan, Ink design, Inc.

Cover Image: Indivision 07 Grow A/Getty Images

Compositor: GEX Publishing Services

For product information and technology assistance, contact us at
Cengage Learning Customer & Sales Support, 1-800-354-9706

For permission to use material from this text or product, submit all requests online at **www.cengage.com/permissions**
Further permissions questions can be emailed to
permissionrequest@cengage.com

ISBN-13: 978-0-538-74492-8

ISBN-10: 0-538-74492-8

Course Technology
20 Channel Center Street
Boston, MA 02210
USA

Cengage Learning is a leading provider of customized learning solutions with office locations around the globe, including Singapore, the United Kingdom, Australia, Mexico, Brazil, and Japan. Locate your local office at: **international.cengage.com/region**

Cengage Learning products are represented in Canada by Nelson Education, Ltd.

To learn more about Course Technology, visit **www.cengage.com/coursetechnology**

To learn more about Cengage Learning, visit **www.cengage.com**

Any fictional data related to persons or companies or URLs used throughout this book is intended for instructional purposes only. At the time this book was printed, any such data was fictional and not belonging to any real persons or companies.

Printed in the United States of America
1 2 3 4 5 6 16 15 14 13 12 11

Overview of This Book

Objectives—At the beginning of each chapter, objectives help you preview and review chapter topics.

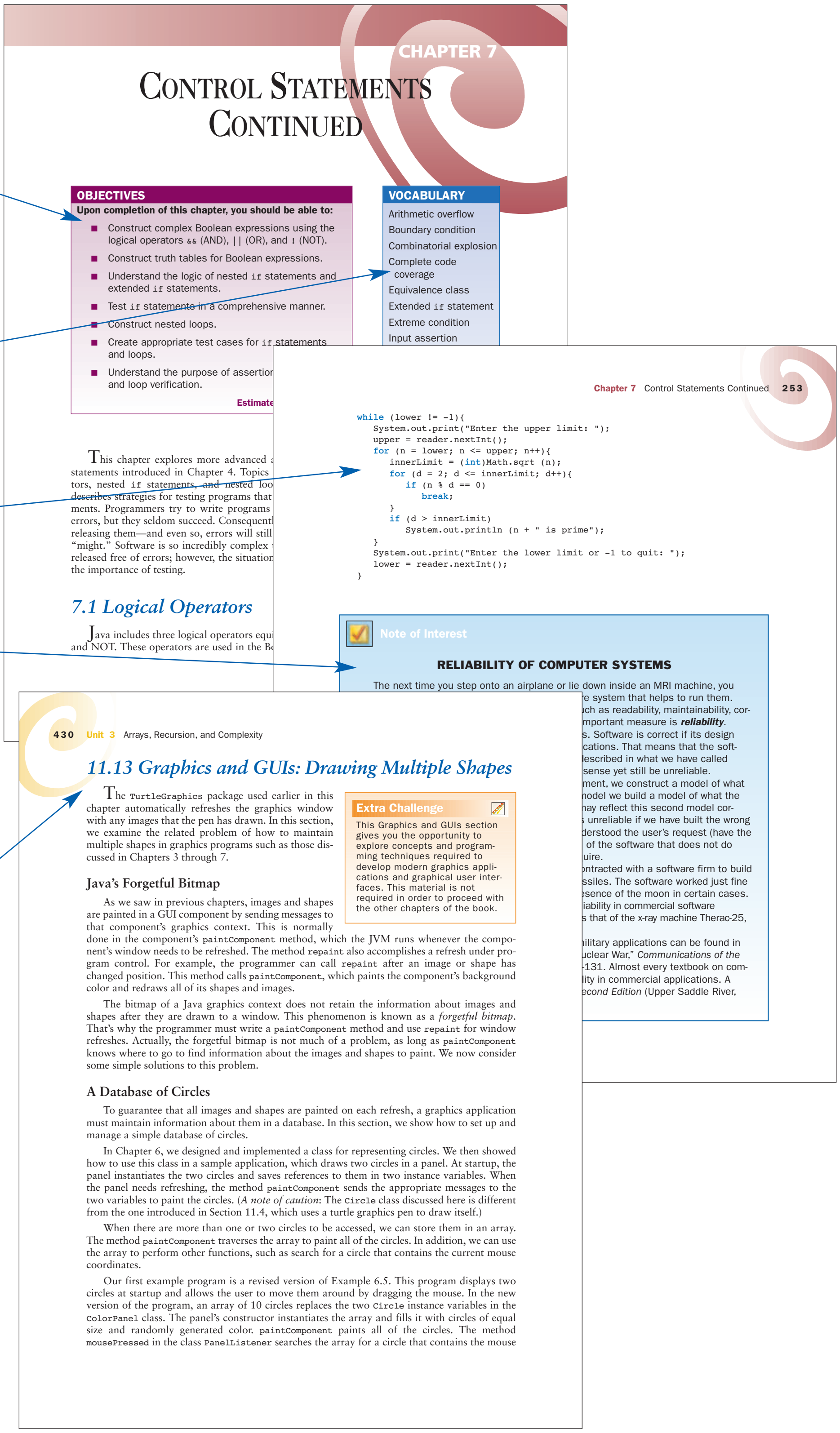
Vocabulary Terms—This chapter-opening list draws your attention to important terms.

Code Samples—Plentiful examples show how to write effective Java code.

Short Essays on Special Topics—These notes of interest throughout the book elaborate on important programming issues.

Graphics and GUIs—Optional sections give you the opportunity to focus on topics related to modern graphics applications.

Sample lesson pages



CHAPTER 7

CONTROL STATEMENTS CONTINUED

OBJECTIVES

Upon completion of this chapter, you should be able to:

- Construct complex Boolean expressions using the logical operators `&&` (AND), `||` (OR), and `!` (NOT).
- Construct truth tables for Boolean expressions.
- Understand the logic of nested `if` statements and extended `if` statements.
- Test `if` statements in a comprehensive manner.
- Construct nested loops.
- Create appropriate test cases for `if` statements and loops.
- Understand the purpose of assertion
and loop verification.

Estimate

VOCABULARY

Arithmetic overflow
Boundary condition
Combinatorial explosion
Complete code coverage
Equivalence class
Extended `if` statement
Extreme condition
Input assertion

This chapter explores more advanced
statements introduced in Chapter 4. Topics
tors, nested `if` statements, and nested loo
describes strategies for testing programs that
ments. Programmers try to write programs
errors, but they seldom succeed. Consequent
releasing them—and even so, errors will still
"might." Software is so incredibly complex
released free of errors; however, the situation
the importance of testing.

7.1 Logical Operators

Java includes three logical operators equ
and NOT. These operators are used in the B

Chapter 7 Control Statements Continued 253

```
while (lower != -1){
   System.out.print("Enter the upper limit: ");
   upper = reader.nextInt();
   for (n = lower; n <= upper; n++){
      innerLimit = (int)Math.sqrt (n);
      for (d = 2; d <= innerLimit; d++){
         if (n % d == 0)
            break;
      }
      if (d > innerLimit)
         System.out.println (n + " is prime");
   }
   System.out.print("Enter the lower limit or -1 to quit: ");
   lower = reader.nextInt();
}
```

Note of Interest

RELIABILITY OF COMPUTER SYSTEMS

The next time you step onto an airplane or lie down inside an MRI machine, you
e system that helps to run them.
ch as readability, maintainability, cor-
mportant measure is ***reliability***.
s. Software is correct if its design
cations. That means that the soft-
escribed in what we have called
sense yet still be unreliable.
ment, we construct a model of what
model we build a model of what the
ay reflect this second model cor-
unreliable if we have built the wrong
derstood the user's request (have the
of the software that does not do
uire.
ntracted with a software firm to build
ssiles. The software worked just fine
sence of the moon in certain cases.
ability in commercial software
s that of the x-ray machine Therac-25,

ilitary applications can be found in
uclear War," *Communications of the*
-131. Almost every textbook on com-
ity in commercial applications. A
econd Edition (Upper Saddle River,

430 Unit 3 Arrays, Recursion, and Complexity

11.13 Graphics and GUIs: Drawing Multiple Shapes

The `TurtleGraphics` package used earlier in this chapter automatically refreshes the graphics window with any images that the pen has drawn. In this section, we examine the related problem of how to maintain multiple shapes in graphics programs such as those discussed in Chapters 3 through 7.

Extra Challenge

This Graphics and GUIs section gives you the opportunity to explore concepts and programming techniques required to develop modern graphics applications and graphical user interfaces. This material is not required in order to proceed with the other chapters of the book.

Java's Forgetful Bitmap

As we saw in previous chapters, images and shapes are painted in a GUI component by sending messages to that component's graphics context. This is normally done in the component's `paintComponent` method, which the JVM runs whenever the component's window needs to be refreshed. The method `repaint` also accomplishes a refresh under program control. For example, the programmer can call `repaint` after an image or shape has changed position. This method calls `paintComponent`, which paints the component's background color and redraws all of its shapes and images.

The bitmap of a Java graphics context does not retain the information about images and shapes after they are drawn to a window. This phenomenon is known as a *forgetful bitmap*. That's why the programmer must write a `paintComponent` method and use `repaint` for window refreshes. Actually, the forgetful bitmap is not much of a problem, as long as `paintComponent` knows where to go to find information about the images and shapes to paint. We now consider some simple solutions to this problem.

A Database of Circles

To guarantee that all images and shapes are painted on each refresh, a graphics application must maintain information about them in a database. In this section, we show how to set up and manage a simple database of circles.

In Chapter 6, we designed and implemented a class for representing circles. We then showed how to use this class in a sample application, which draws two circles in a panel. At startup, the panel instantiates the two circles and saves references to them in two instance variables. When the panel needs refreshing, the method `paintComponent` sends the appropriate messages to the two variables to paint the circles. (*A note of caution*: The `Circle` class discussed here is different from the one introduced in Section 11.4, which uses a turtle graphics pen to draw itself.)

When there are more than one or two circles to be accessed, we can store them in an array. The method `paintComponent` traverses the array to paint all of the circles. In addition, we can use the array to perform other functions, such as search for a circle that contains the current mouse coordinates.

Our first example program is a revised version of Example 6.5. This program displays two circles at startup and allows the user to move them around by dragging the mouse. In the new version of the program, an array of 10 circles replaces the two `Circle` instance variables in the `ColorPanel` class. The panel's constructor instantiates the array and fills it with circles of equal size and randomly generated color. `paintComponent` paints all of the circles. The method `mousePressed` in the class `PanelListener` searches the array for a circle that contains the mouse

Overview of This Book

Sample end-of-lesson pages

Section Exercises—Review questions at the end of each section check your understanding of essential concepts.

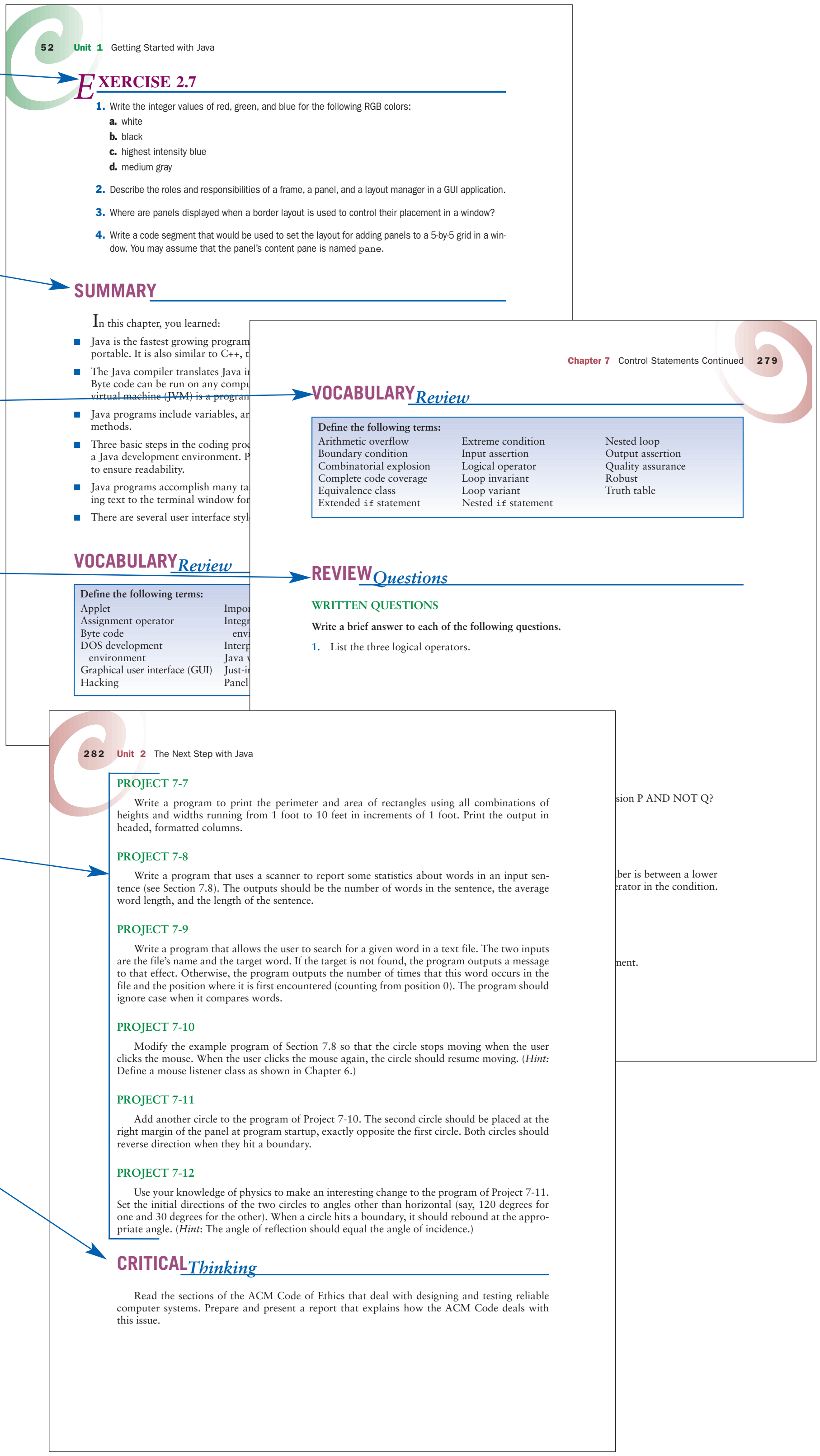
Summary—End-of-chapter recap summarizes what you learned.

Vocabulary Review—End-of-chapter list reminds you to check your knowledge of important terms.

Review Questions—End-of-chapter questions test your understanding of chapter concepts.

Programming Projects—Numerous end-of-chapter projects allow you to apply what you've learned.

Critical Thinking—Each chapter concludes with an exercise that asks you to use creative analysis to solve a problem.



52 Unit 1 Getting Started with Java

EXERCISE 2.7

1. Write the integer values of red, green, and blue for the following RGB colors:
   a. white
   b. black
   c. highest intensity blue
   d. medium gray
2. Describe the roles and responsibilities of a frame, a panel, and a layout manager in a GUI application.
3. Where are panels displayed when a border layout is used to control their placement in a window?
4. Write a code segment that would be used to set the layout for adding panels to a 5-by-5 grid in a window. You may assume that the panel's content pane is named `pane`.

SUMMARY

In this chapter, you learned:

- Java is the fastest growing program
portable. It is also similar to C++, t
- The Java compiler translates Java i
Byte code can be run on any compu
virtual machine (JVM) is a progran
- Java programs include variables, ar
methods.
- Three basic steps in the coding proc
a Java development environment. P
to ensure readability.
- Java programs accomplish many ta
ing text to the terminal window for
- There are several user interface styl

VOCABULARY *Review*

Define the following terms:
Applet
Assignment operator
Byte code
DOS development environment
Graphical user interface (GUI)
Hacking
Impo
Integr
envi
Interp
Java v
Just-i
Panel

Chapter 7 Control Statements Continued 279

VOCABULARY *Review*

Define the following terms:
Arithmetic overflow
Boundary condition
Combinatorial explosion
Complete code coverage
Equivalence class
Extended `if` statement
Extreme condition
Input assertion
Logical operator
Loop invariant
Loop variant
Nested `if` statement
Nested loop
Output assertion
Quality assurance
Robust
Truth table

REVIEW *Questions*

WRITTEN QUESTIONS

Write a brief answer to each of the following questions.

1. List the three logical operators.

sion P AND NOT Q?

ber is between a lower
erator in the condition.

ment.

282 Unit 2 The Next Step with Java

PROJECT 7-7

Write a program to print the perimeter and area of rectangles using all combinations of heights and widths running from 1 foot to 10 feet in increments of 1 foot. Print the output in headed, formatted columns.

PROJECT 7-8

Write a program that uses a scanner to report some statistics about words in an input sentence (see Section 7.8). The outputs should be the number of words in the sentence, the average word length, and the length of the sentence.

PROJECT 7-9

Write a program that allows the user to search for a given word in a text file. The two inputs are the file's name and the target word. If the target is not found, the program outputs a message to that effect. Otherwise, the program outputs the number of times that this word occurs in the file and the position where it is first encountered (counting from position 0). The program should ignore case when it compares words.

PROJECT 7-10

Modify the example program of Section 7.8 so that the circle stops moving when the user clicks the mouse. When the user clicks the mouse again, the circle should resume moving. (*Hint:* Define a mouse listener class as shown in Chapter 6.)

PROJECT 7-11

Add another circle to the program of Project 7-10. The second circle should be placed at the right margin of the panel at program startup, exactly opposite the first circle. Both circles should reverse direction when they hit a boundary.

PROJECT 7-12

Use your knowledge of physics to make an interesting change to the program of Project 7-11. Set the initial directions of the two circles to angles other than horizontal (say, 120 degrees for one and 30 degrees for the other). When a circle hits a boundary, it should rebound at the appropriate angle. (*Hint*: The angle of reflection should equal the angle of incidence.)

CRITICAL *Thinking*

Read the sections of the ACM Code of Ethics that deal with designing and testing reliable computer systems. Prepare and present a report that explains how the ACM Code deals with this issue.

PREFACE

This text is intended for a complete course in programming and problem solving. It covers the material of typical Computer Science 1 courses at the undergraduate level, but it is intended for the high school audience. It covers all the A-level Advanced Placement (AP) Java topics.

We present six major aspects of computing, some in standalone chapters and others spread across several chapters:

1. **Programming Basics.** This deals with the basic ideas related to solving problems with computers, including primitive data types, control structures, methods, algorithm development, and complexity analysis.
2. **Object-Oriented Programming.** OOP is today's dominant programming paradigm. All the essentials of this subject are covered.
3. **Data and Information Processing.** Fundamental data structures are discussed. These include strings, arrays, files, and lists. The general concept of abstract data type is introduced, and complexity analysis is used to evaluate the running times of different implementations of algorithms.
4. **Software Development Life Cycle.** Rather than isolate software development techniques in one or two chapters, the book deals with them throughout in the context of numerous case studies.
5. **Graphical User Interfaces and Event-Driven Programming.** Many books at this level restrict themselves to character-based terminal I/O. The reason is simple. Graphical user interfaces and event-driven programming usually are considered too complex for beginning students. In this book, we circumvent the complexity barrier and show how to develop programs with graphical user interfaces with almost the same ease as their terminal-based counterparts.
6. **Web Basics.** The programming of Web pages with HTML and applets is introduced.

Focus on Fundamental Computer Science Topics

There seem to be two types of introductory Java textbooks. The first emphasizes basic problem-solving and programming techniques, and the second emphasizes language features. This book takes the former approach and introduces Java features as they are needed to support programming concepts. In this way, we cover all the AP-required syntax without allowing the book to be syntax-driven. Some more advanced Java features, not part of the AP requirement, are covered in end-of-chapter sections and in the appendices.

Methods and Objects, Early or Late?

Occasionally, people argue about whether methods and objects should be introduced early or late in the first course. In Java, even the simplest program involves both methods and objects, so the problem really becomes one of how to introduce these concepts in a clear and meaningful manner from the outset. Starting with the first program, we show how to instantiate and send messages to objects. The book's early chapters (2 through 5) focus on the use of objects, arithmetic expressions, control constructs, and algorithms in the context of short, simple programs. As programs become more complex, it becomes advantageous to decompose them into cooperating components. With this end in mind, Chapter 6 shows how to develop systems of cooperating classes and methods. Thus, we take a pragmatic rather than an ideological approach to the question of when to introduce methods and objects, having complete confidence that students will master both by the end of the course.

Revisiting Control Structures, Classes, and Arrays

Years of teaching experience have demonstrated that beginning programming students struggle most with control structures, classes, and arrays. In this text, we have sought to soften the blow by introducing these ideas in two steps. First, a chapter gives an initial overview of a topic using the most basic features in simple but realistic applications. A follow-up chapter then revisits the topic to fill in and refine the details.

New in This Edition

AP computer science now has a single exam, which still covers the material typically offered in the first college course in programming and problem solving with Java. Starting in 2010, the exam will include the following items:

1. The Java constants `Integer.MAX_VALUE` and `Integer.MIN_VALUE`
2. Static variables and methods
3. Two-dimensional arrays
4. The `java.util.List` interface

The first three items were already covered in the third edition of this book. This new edition has been updated to offer substantial coverage of two-dimensional arrays and the `List` interface. Two-dimensional arrays are explored in a new chapter (Chapter 12) on advanced array operations. The `List` interface is examined in a new chapter (Chapter 14) on Java collections, which include the single collection class, `java.util.ArrayList`, which is covered by the AP exam. Chapter 14 also includes a non-required preview of other collections that would typically be covered in the next college-level course following this one. (Some of this material was formerly included in the old AP AB course.)

The new edition also includes two entirely new chapters that provide examples of programming and problem solving in two contemporary areas of computing, media processing (Chapter 5) and networked applications (Chapter 15). The chapter on media computing provides an early introduction to the use of objects and methods to represent and manipulate images and sound clips. The open-source Java toolkits, **images** and **sounds**, used in Chapter 5 are available at the author's Web site at *http://home.wlu.edu/~lambertk/hsjava*. Chapter 15 also introduces multithreading and sockets as means of managing communications between client and server programs on a network. Although none of the material in Chapters 5 or 15 is required for the AP exam, both chapters offer students exciting opportunities to learn the concepts and principles underlying the applications most users work with every day.

The text has been organized so that the optional topics covered in Chapter 5 (media computing), Chapter 13 (recursion and complexity), Chapter 14 (collections), and Chapter 15 (networked computing) can be skipped at the discretion of instructors. Whereas the third edition divided chapters into three units, this edition is divided into four, with the fourth and most advanced unit consisting of Chapters 13 through 15.

Finally, all of the code examples presented in this edition have been tested to be compliant with JDK 1.6.

Case Studies, the Software Life Cycle, and Comments

The book contains numerous case studies. These are complete Java programs ranging from the simple to the substantial. To emphasize the importance and usefulness of the software development life cycle, case studies are presented in the framework of a user request followed by analysis, design, and implementation, with well-defined tasks performed at each stage. Some case studies are carried through several chapters or extended in end-of-chapter programming projects.

Programming consists of more than just writing code, so we encourage students to submit an analysis and design as part of major programming assignments. We also believe that code should be properly commented, and for purposes of illustration, we include comments in selected examples of the code in the book.

Exercises

The book contains several different types of exercises. Most chapter sections end with exercise questions that reinforce the reading by asking basic questions about the material in the section. Each chapter ends with a set of review questions. All chapters except the first one include programming projects of varying degrees of difficulty. Each chapter concludes with a critical thinking activity that allows the student to reflect on a major topic covered in the chapter. Finally, each unit ends with a similar set of review questions, projects and a critical thinking activity.

Special Features

Scattered throughout the book are short essays. These present historical and social aspects of computing, including computer ethics and security.

We Appreciate Your Feedback

We have greatly appreciated all of the helpful suggestions and comments from the many instructors who have used the previous edition of this book. As always, we have tried to produce a high-quality text, but should you encounter any errors, please report them to *klambert@wlu.edu*. Information about the book, as well as a list of errata (should they exist), will be posted on the following Web site: *http://home.wlu.edu/~lambertk/hsjava*.

Acknowledgments

We are very grateful to the following reviewers, who offered helpful suggestions on this revision:

Proposal Reviewers:

Marilyn Carrell: Springdale High School
Virginia Cocanower: Bentonville High School
Jim Conrey: Kentucky County Day School
Robert Glen Martin: TAG Magnet High School
Steven Temple: York Suburban High School
Marilyn Turmelle: Booker T. Washington High School

Chapter Reviewers:

Marilyn Carrell: Springdale High School
Jim Conrey: Kentucky County Day School

We would like to thank several people whose work made this book possible:

Ann Shaffer
Developmental Editor

Victoria Legier
Associate Project Manager
GEX Publishing Services

Amy Jollymore
Acquisitions Editor
Cengage Learning

Alyssa Pratt
Senior Product Manager
Cengage Learning

Julie Schuster
Associate Marketing Manager
Cengage Learning

Amanda Lyons
Associate Product Manager
Cengage Learning

Kim Klasner
Editorial Assistant
Cengage Learning

Matthew Hutchinson
Content Project Manager
Cengage Learning

Jeff Schwartz
Manuscript Quality Assurance Lead
Cengage Learning

Julio Esperas
Manufacturing Coordinator
Cengage Learning

Faith Brosnan
Art Director
Cengage Learning

In addition, several individuals contributed material to the Instructor Resource Kit:

Jeannine Lawless
Editorial Assistant
Course Technology

Jan Clavey
Assistant Product Manager
Custom Editorial Production, Inc.

Supplemental Resources

The *Instructor Resources CD* or *DVD* contains the following teaching resources:

- The Data and Solution files for this course.
- ExamView® tests for each lesson. ExamView is a powerful testing software package that allows instructors to create and administer printed, computer (LAN-based), and Internet exams.
- Instructor's Manual that includes lecture notes for each chapter and references to the end-of-chapter and Unit Review activities.
- Answer keys that include solutions to the chapter and unit review questions.
- Copies of the figures that appear in the student text.
- Suggested syllabus with block, two quarter, and 18-week schedule.
- PowerPoint presentations for each lesson.

Instructor Resources Disk:
ISBN-13: 9781439078594
ISBN-10: 1439078599

Dedication

To Ann Shaffer

Kenneth A. Lambert
Lexington, Virginia

Martin Osborne
Bellingham, Washington