

GLOBAL
EDITION



Introduction to Java™ Programming and Data Structures

Comprehensive Version

ELEVENTH EDITION

Y. Daniel Liang



Digital Resources for Students

Your new textbook provides 12-month access to digital resources that may include VideoNotes (step-by-step video tutorials on programming concepts), source code, web chapters, quizzes, and more. Refer to the preface in the textbook for a detailed list of resources.

Follow the instructions below to register for the Companion Website for Daniel Liang's *Introduction to Java™ Programming and Data Structures, Comprehensive Version*, Eleventh Edition, Global Edition.

1. Go to www.pearsonglobaleditions.com/liang
2. Enter the title of your textbook or browse by author name.
3. Click Companion Website.
4. Click Register and follow the on-screen instructions to create a login name and password.

ISSLJC - SETUP - ALIEN - PAREU - BEGUN - LIKES

Use the login name and password you created during registration to start using the digital resources that accompany your textbook.

IMPORTANT:

This prepaid subscription does not include access to MyProgrammingLab, which is available at www.myprogramminglab.com for purchase.

This access code can only be used once. This subscription is valid for 12 months upon activation and is not transferable. If the access code has already been revealed it may no longer be valid.

For technical support go to <https://support.pearson.com/getsupport>

INTRODUCTION TO
JAVATM

**PROGRAMMING AND
DATA STRUCTURES**
COMPREHENSIVE VERSION

Eleventh Edition

Global Edition

Y. Daniel Liang

Armstrong State University



330 Hudson Street, NY NY 10013

To Samantha, Michael, and Michelle

Senior Vice President Courseware Portfolio
Management: Marcia J. Horton
Director, Portfolio Management: Engineering,
Computer Science & Global Editions: Julian
Partridge
Higher Ed Portfolio Management: Tracy Johnson
(Dunkelberger)
Portfolio Management Assistant: Kristy Alaura
Managing Content Producer: Scott Disanno
Content Producer: Robert Engelhardt
Web Developer: Steve Wright
Assistant Acquisitions Editor, Global Edition:
Aditee Agarwal
Assistant Project Editor, Global Edition: Shaoni
Mukherjee

Manager, Media Production, Global Edition:
Vikram Kumar
Senior Manufacturing Controller, Production,
Global Edition: Jerry Kataria
Rights and Permissions Manager: Ben Ferrini
Manufacturing Buyer, Higher Ed, Lake Side
Communications Inc (LSC): Maura Zaldivar-Garcia
Inventory Manager: Ann Lam
Marketing Manager: Demetrius Hall
Product Marketing Manager: Bram Van Kempen
Marketing Assistant: Jon Bryant
Cover Designer: Lumina Datamatics
Cover Image: Eduardo Rocha/ shutterstock.com
Full-Service Project Management: Shylaja Gattupalli,
SPi Global

Java™ and Netbeans™ screenshots ©2017 by Oracle Corporation, all rights reserved. Reprinted with permission. Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on the appropriate page within text. Microsoft and/or its respective suppliers make no representations about the suitability of the information contained in the documents and related graphics published as part of the services for any purpose. All such documents and related graphics are provided “as is” without warranty of any kind. Microsoft and/or its respective suppliers hereby disclaim all warranties and conditions with regard to this information, including all warranties and conditions of merchantability, whether express, implied or statutory, fitness for a particular purpose, title and non-infringement. In no event shall Microsoft and/or its respective suppliers be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of information available from the services. The documents and related graphics contained herein could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Microsoft and/or its respective suppliers may make improvements and/or changes in the product(s) and/or the program(s) described herein at any time. Partial screen shots may be viewed in full within the software version specified.

Pearson Education Limited

KAO Two
KAO Park
Harlow
CM17 9NA
United Kingdom

and Associated Companies throughout the world

Visit us on the World Wide Web at: www.pearsonglobaleditions.com

© Pearson Education Limited 2019

The rights of Y. Daniel Liang to be identified as the author of this work have been asserted by him in accordance with the Copyright, Designs and Patents Act 1988.

Authorized adaptation from the United States edition, entitled Introduction to Java Programming and Data Structures, Comprehensive Version, 11th Edition, ISBN 978-0-13-467094-2 by Y. Daniel Liang, published by Pearson Education © 2018.

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

Typeset by SPi Global

Printed and bound by Vivar in Malaysia

ISBN-10: 1-292-22187-9

ISBN-13: 978-1-292-22187-8

PREFACE

Dear Reader,

Many of you have provided feedback on earlier editions of this book, and your comments and suggestions have greatly improved the book. This edition has been substantially enhanced in presentation, organization, examples, exercises, and supplements.

The book is fundamentals first by introducing basic programming concepts and techniques before designing custom classes. The fundamental concepts and techniques of selection statements, loops, methods, and arrays are the foundation for programming. Building this strong foundation prepares students to learn object-oriented programming and advanced Java programming.

fundamentals-first

This book teaches programming in a problem-driven way that focuses on problem solving rather than syntax. We make introductory programming interesting by using thought-provoking problems in a broad context. The central thread of early chapters is on problem solving. Appropriate syntax and library are introduced to enable readers to write programs for solving the problems. To support the teaching of programming in a problem-driven way, the book provides a wide variety of problems at various levels of difficulty to motivate students. To appeal to students in all majors, the problems cover many application areas, including math, science, business, financial, gaming, animation, and multimedia.

problem-driven

The book seamlessly integrates programming, data structures, and algorithms into one text. It employs a practical approach to teach data structures. We first introduce how to use various data structures to develop efficient algorithms, and then show how to implement these data structures. Through implementation, students gain a deep understanding on the efficiency of data structures and on how and when to use certain data structures. Finally, we design and implement custom data structures for trees and graphs.

data structures

The book is widely used in the introductory programming, data structures, and algorithms courses in the universities around the world. This *comprehensive version* covers fundamentals of programming, object-oriented programming, GUI programming, data structures, algorithms, concurrency, networking, database, and Web programming. It is designed to prepare students to become proficient Java programmers. A *brief version* (*Introduction to Java Programming*, Brief Version, Eleventh Edition, Global Edition) is available for a first course on programming, commonly known as CS1. The brief version contains the first 18 chapters of the comprehensive version.

comprehensive version

brief version

The best way to teach programming is *by example*, and the only way to learn programming is *by doing*. Basic concepts are explained by example and a large number of exercises with various levels of difficulty are provided for students to practice. For our programming courses, we assign programming exercises after each lecture.

Our goal is to produce a text that teaches problem solving and programming in a broad context using a wide variety of interesting examples. If you have any comments on and suggestions for improving the book, please email me.

Sincerely,

Y. Daniel Liang

y.daniel.liang@gmail.com

www.pearsonglobaleditions.com/Liang

ACM/IEEE Curricular 2013 and ABET Course Assessment

The new ACM/IEEE Computer Science Curricular 2013 defines the Body of Knowledge organized into 18 Knowledge Areas. To help instructors design the courses based on this book, we provide sample syllabi to identify the Knowledge Areas and Knowledge Units. The sample syllabi are for a three semester course sequence and serve as an example for institutional customization. The sample syllabi are accessible from the Instructor Resource Center.

Many of our users are from the ABET-accredited programs. A key component of the ABET accreditation is to identify the weakness through continuous course assessment against the course outcomes. We provide sample course outcomes for the courses and sample exams for measuring course outcomes on the Instructor Resource Center.

What's New in This Edition?

This edition is completely revised in every detail to enhance clarity, presentation, content, examples, and exercises. The major improvements are as follows:

- The book's title is changed to Introduction to Java Programming and Data Structures with new enhancements on data structures. The book uses a practical approach to introduce design, implement, and use data structures and covers all topics in a typical data structures course. Additionally, it provides bonus chapters that cover advanced data structures such as 2-4 trees, B-trees, and red-black trees.
- Updated to the latest Java technology. Examples and exercises are improved and simplified by using the new features in Java 8.
- The default and static methods are introduced for interfaces in Chapter 13.
- The GUI chapters are updated to JavaFX 8. The examples are revised. The user interfaces in the examples and exercises are now resizable and displayed in the center of the window.
- Inner classes, anonymous inner classes, and lambda expressions are covered using practical examples in Chapter 15.
- More examples and exercises in the data structures chapters use lambda expressions to simplify coding. Method references are introduced along with the `Comparator` interface in Section 20.6.
- The `forEach` method is introduced in Chapter 20 as a simple alternative to the `foreach` loop for applying an action to each element in a collection.
- Use the default methods for interfaces in Java 8 to redesign and simplify `MyList`, `MyArrayList`, `MyLinkedList`, `Tree`, `BST`, `AVLTree`, `MyMap`, `MyHashMap`, `MySet`, `MyHashSet`, `Graph`, `UnweightedGraph`, and `WeightedGraph` in Chapters 24–29.
- Chapter 30 is brand new to introduce aggregate operations for collection streams.
- FXML and the Scene Builder visual tool are introduced in Chapter 31.
- The Companion Website has been redesigned with new interactive quiz, CheckPoint questions, animations, and live coding.
- More than 200 additional programming exercises with solutions are provided to the instructor on the Instructor Resource Center. These exercises are not printed in the text.

Pedagogical Features

The book uses the following elements to help students get the most from the material:

- The **Objectives** at the beginning of each chapter list what students should learn from the chapter. This will help them determine whether they have met the objectives after completing the chapter.
- The **Introduction** opens the discussion with a thought-provoking question to motivate the reader to delve into the chapter.
- **Key Points** highlight the important concepts covered in each section.
- **Check Points** provide review questions to help students track their progress as they read through the chapter and evaluate their learning.
- **Problems and Case Studies**, carefully chosen and presented in an easy-to-follow style, teach problem solving and programming concepts. The book uses many small, simple, and stimulating examples to demonstrate important ideas.
- The **Chapter Summary** reviews the important subjects that students should understand and remember. It helps them reinforce the key concepts they have learned in the chapter.
- **Quizzes** are accessible online, grouped by sections, for students to do self-test on programming concepts and techniques.
- **Programming Exercises** are grouped by sections to provide students with opportunities to apply the new skills they have learned on their own. The level of difficulty is rated as easy (no asterisk), moderate (*), hard (**), or challenging (***). The trick of learning programming is practice, practice, and practice. To that end, the book provides a great many exercises. Additionally, more than 200 programming exercises with solutions are provided to the instructors on the Instructor Resource Center. These exercises are not printed in the text.
- **Notes, Tips, Cautions, and Design Guides** are inserted throughout the text to offer valuable advice and insight on important aspects of program development.



Note

Provides additional information on the subject and reinforces important concepts.



Tip

Teaches good programming style and practice.



Caution

Helps students steer away from the pitfalls of programming errors.

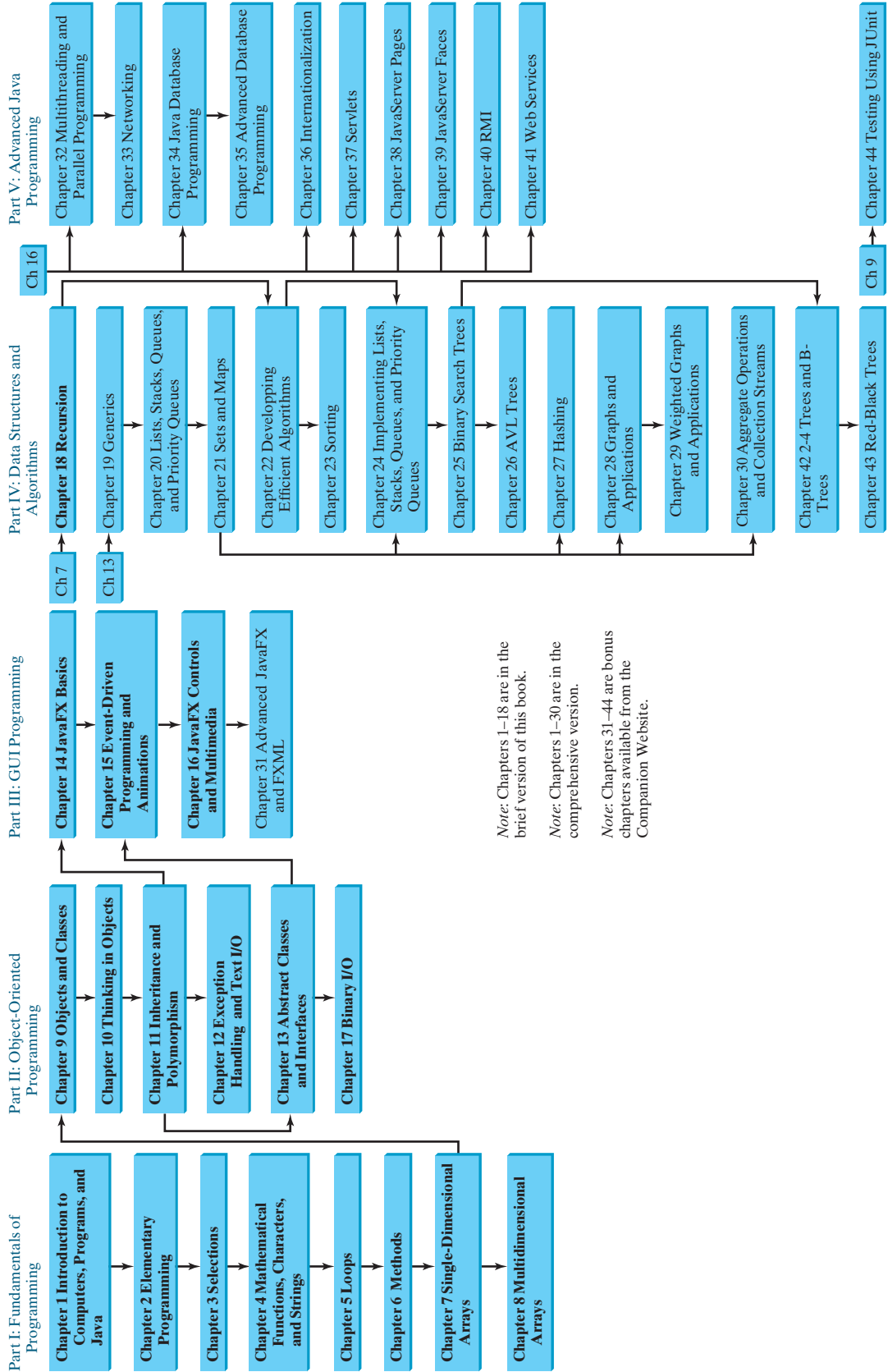


Design Guide

Provides guidelines for designing programs.

Flexible Chapter Orderings

The book is designed to provide flexible chapter orderings to enable GUI, exception handling, recursion, generics, and the Java Collections Framework to be covered earlier or later. The diagram on the next page shows the chapter dependencies.



Organization of the Book

The chapters can be grouped into five parts that, taken together, form a comprehensive introduction to Java programming, data structures and algorithms, and database and Web programming. Because knowledge is cumulative, the early chapters provide the conceptual basis for understanding programming and guide students through simple examples and exercises; subsequent chapters progressively present Java programming in detail, culminating with the development of comprehensive Java applications. The appendixes contain a mixed bag of topics, including an introduction to number systems, bitwise operations, regular expressions, and enumerated types.

Part I: Fundamentals of Programming (Chapters 1–8)

The first part of the book is a stepping stone, preparing you to embark on the journey of learning Java. You will begin to learn about Java (Chapter 1) and fundamental programming techniques with primitive data types, variables, constants, assignments, expressions, and operators (Chapter 2), selection statements (Chapter 3), mathematical functions, characters, and strings (Chapter 4), loops (Chapter 5), methods (Chapter 6), and arrays (Chapters 7–8). After Chapter 7, you can jump to Chapter 18 to learn how to write recursive methods for solving inherently recursive problems.

Part II: Object-Oriented Programming (Chapters 9–13, and 17)

This part introduces object-oriented programming. Java is an object-oriented programming language that uses abstraction, encapsulation, inheritance, and polymorphism to provide great flexibility, modularity, and reusability in developing software. You will learn programming with objects and classes (Chapters 9–10), class inheritance (Chapter 11), polymorphism (Chapter 11), exception handling (Chapter 12), abstract classes (Chapter 13), and interfaces (Chapter 13). Text I/O is introduced in Chapter 12 and binary I/O is discussed in Chapter 17.

Part III: GUI Programming (Chapters 14–16 and Bonus Chapter 31)

JavaFX is a new framework for developing Java GUI programs. It is not only useful for developing GUI programs, but also an excellent pedagogical tool for learning object-oriented programming. This part introduces Java GUI programming using JavaFX in Chapters 14–16. Major topics include GUI basics (Chapter 14), container panes (Chapter 14), drawing shapes (Chapter 14), event-driven programming (Chapter 15), animations (Chapter 15), and GUI controls (Chapter 16), and playing audio and video (Chapter 16). You will learn the architecture of JavaFX GUI programming and use the controls, shapes, panes, image, and video to develop useful applications. Chapter 31 covers advanced features in JavaFX.

Part IV: Data Structures and Algorithms (Chapters 18–30 and Bonus Chapters 42–43)

This part covers the main subjects in a typical data structures and algorithms course. Chapter 18 introduces recursion to write methods for solving inherently recursive problems. Chapter 19 presents how generics can improve software reliability. Chapters 20 and 21 introduce the Java Collection Framework, which defines a set of useful API for data structures. Chapter 22 discusses measuring algorithm efficiency in order to choose an appropriate algorithm for applications. Chapter 23 describes classic sorting algorithms. You will learn how to implement several classic data structures lists, queues, and priority queues in Chapter 24. Chapters 25 and 26 introduce binary search trees and AVL trees. Chapter 27 presents hashing and implementing maps and sets using hashing. Chapters 28 and 29 introduce graph applications. Chapter 30 introduces aggregate operations for collection streams. The 2-4 trees, B-trees, and red-black trees are covered in Bonus Chapters 42–43.

Part V: Advanced Java Programming (Chapters 32–41, 44)

This part of the book is devoted to advanced Java programming. Chapter 32 treats the use of multithreading to make programs more responsive and interactive and introduces parallel programming. Chapter 33 discusses how to write programs that talk with each other from different

hosts over the Internet. Chapter 34 introduces the use of Java to develop database projects. Chapter 35 delves into advanced Java database programming. Chapter 36 covers the use of internationalization support to develop projects for international audiences. Chapters 37 and 38 introduce how to use Java servlets and JavaServer Pages to generate dynamic content from Web servers. Chapter 39 introduces modern Web application development using JavaServer Faces. Chapter 40 introduces remote method invocation and Chapter 41 discusses Web services. Chapter 44 introduces testing Java programs using JUnit.

Appendixes

This part of the book covers a mixed bag of topics. Appendix A lists Java keywords. Appendix B gives tables of ASCII characters and their associated codes in decimal and in hex. Appendix C shows the operator precedence. Appendix D summarizes Java modifiers and their usage. Appendix E discusses special floating-point values. Appendix F introduces number systems and conversions among binary, decimal, and hex numbers. Finally, Appendix G introduces bitwise operations. Appendix H introduces regular expressions. Appendix I covers enumerated types.

Java Development Tools

You can use a text editor, such as the Windows Notepad or WordPad, to create Java programs and to compile and run the programs from the command window. You can also use a Java development tool, such as NetBeans or Eclipse. These tools support an integrated development environment (IDE) for developing Java programs quickly. Editing, compiling, building, executing, and debugging programs are integrated in one graphical user interface. Using these tools effectively can greatly increase your programming productivity. NetBeans and Eclipse are easy to use if you follow the tutorials. Tutorials on NetBeans and Eclipse can be found in the supplements on the Companion Website www.pearsonglobaleditions.com/Liang.

IDE tutorials

Student Resources

The Companion Website (www.pearsonglobaleditions.com/Liang) contains the following resources:

- Answers to CheckPoint questions
- Solutions to majority of even-numbered programming exercises
- Source code for the examples in the book
- Interactive quiz (organized by sections for each chapter)
- Supplements
- Debugging tips
- Video notes
- Algorithm animations

Supplements

The text covers the essential subjects. The supplements extend the text to introduce additional topics that might be of interest to readers. The supplements are available from the Companion Website.

Instructor Resources

The Companion Website, accessible from www.pearsonglobaleditions.com/Liang, contains the following resources:

- Microsoft PowerPoint slides with interactive buttons to view full-color, syntax-highlighted source code and to run programs without leaving the slides.
- Solutions to a majority of odd-numbered programming exercises.
- More than 200 additional programming exercises and 300 quizzes organized by chapters. These exercises and quizzes are available only to the instructors. Solutions to these exercises and quizzes are provided.
- Web-based quiz generator. (Instructors can choose chapters to generate quizzes from a large database of more than two thousand questions.)
- Sample exams. Most exams have four parts:
 - Multiple-choice questions or short-answer questions
 - Correct programming errors
 - Trace programs
 - Write programs
- Sample exams with ABET course assessment.
- Projects. In general, each project gives a description and asks students to analyze, design, and implement the project.

Some readers have requested the materials from the Instructor Resource Center. Please understand that these are for instructors only. Such requests will not be answered.

Online Practice and Assessment with MyProgrammingLab

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.

MyProgrammingLab is offered to users of this book in partnership with Turing's Craft, the makers of the CodeLab interactive programming exercise system. For a full demonstration, to see feedback from instructors and students, or to get started using MyProgrammingLab in your course, visit www.myprogramminglab.com.

Video Notes

We are excited about the new Video Notes feature that is found in this new edition. These videos provide additional help by presenting examples of key topics and showing how to solve problems completely from design through coding. Video Notes are available from www.pearsonglobaleditions.com/Liang.



VideoNote



Algorithm Animations

We have provided numerous animations for algorithms. These are valuable pedagogical tools to demonstrate how algorithms work. Algorithm animations can be accessed from the Companion Website.

Acknowledgments

I would like to thank Armstrong State University for enabling me to teach what I write and for supporting me in writing what I teach. Teaching is the source of inspiration for continuing to improve the book. I am grateful to the instructors and students who have offered comments, suggestions, corrections, and praise. My special thanks go to Stefan Andrei of Lamar University and William Bahn of University of Colorado Colorado Springs for their help to improve the data structures part of this book.

This book has been greatly enhanced thanks to outstanding reviews for this and previous editions. The reviewers are: Elizabeth Adams (James Madison University), Syed Ahmed (North Georgia College and State University), Omar Aldawud (Illinois Institute of Technology), Stefan Andrei (Lamar University), Yang Ang (University of Wollongong, Australia), Kevin Bierre (Rochester Institute of Technology), Aaron Braskin (Mira Costa High School), David Champion (DeVry Institute), James Chegwiddden (Tarrant County College), Anup Dargar (University of North Dakota), Daryl Detrick (Warren Hills Regional High School), Charles Dierbach (Towson University), Frank Ducrest (University of Louisiana at Lafayette), Erica Eddy (University of Wisconsin at Parkside), Summer Ehresman (Center Grove High School), Deena Engel (New York University), Henry A. Etlinger (Rochester Institute of Technology), James Ten Eyck (Marist College), Myers Foreman (Lamar University), Olac Fuentes (University of Texas at El Paso), Edward F. Gehringer (North Carolina State University), Harold Grossman (Clemson University), Barbara Guillot (Louisiana State University), Stuart Hansen (University of Wisconsin, Parkside), Dan Harvey (Southern Oregon University), Ron Hofman (Red River College, Canada), Stephen Hughes (Roanoke College), Vladan Jovanovic (Georgia Southern University), Deborah Kabura Kariuki (Stony Point High School), Edwin Kay (Lehigh University), Larry King (University of Texas at Dallas), Nana Kofi (Langara College, Canada), George Koutsogiannakis (Illinois Institute of Technology), Roger Kraft (Purdue University at Calumet), Norman Krumpe (Miami University), Hong Lin (DeVry Institute), Dan Lipsa (Armstrong State University), James Madison (Rensselaer Polytechnic Institute), Frank Malinowski (Darton College), Tim Margush (University of Akron), Debbie Masada (Sun Microsystems), Blayne Mayfield (Oklahoma State University), John McGrath (J.P. McGrath Consulting), Hugh McGuire (Grand Valley State), Shyamal Mitra (University of Texas at Austin), Michel Mitri (James Madison University), Kenrick Mock (University of Alaska Anchorage), Frank Murgolo (California State University, Long Beach), Jun Ni (University of Iowa), Benjamin Nystuen (University of Colorado at Colorado Springs), Maureen Opkins (CA State University, Long Beach), Gavin Osborne (University of Saskatchewan), Kevin Parker (Idaho State University), Dale Parson (Kutztown University), Mark Pendergast (Florida Gulf Coast University), Richard Povinelli (Marquette University), Roger Priebe (University of Texas at Austin), Mary Ann Pumphrey (De Anza Junior College), Pat Roth (Southern Polytechnic State University), Amr Sabry (Indiana University), Ben Setzer (Kennesaw State University), Carolyn Schauble (Colorado State University), David Scuse (University of Manitoba), Ashraf Shirani (San Jose State University), Daniel Spiegel (Kutztown University), Joslyn A. Smith (Florida Atlantic University), Lixin Tao (Pace University), Ronald F. Taylor (Wright State University), Russ Tront (Simon Fraser University), Deborah Trytten (University of Oklahoma), Michael Verdicchio (Citadel), Kent Vidrine (George Washington University), and Bahram Zartoshty (California State University at Northridge).

It is a great pleasure, honor, and privilege to work with Pearson. I would like to thank Tracy Johnson and her colleagues Marcia Horton, Demetrius Hall, Yvonne Vannatta, Kristy Alaura, Carole Snyder, Scott Disanno, Bob Engelhardt, Shylaja Gattupalli, and their colleagues for organizing, producing, and promoting this project.

As always, I am indebted to my wife, Samantha, for her love, support, and encouragement.

Acknowledgments for the Global Edition

Pearson would like to thank and acknowledge Yvan Maillot (Univresite Haute-Alsace) and Steven Yuwono (National University of Singapore) for contributing to this Global Edition, and Arif Ahmed (National Institute of Technology, Silchar), Annette Bieniusa (University of Kaiserslautern), Shaligram Prajapat (Devi Ahilya Vishwavidyalaya, Indore), and Ram Gopal Raj (University of Malaya) for reviewing this Global Edition.

CONTENTS

Chapter 1	Introduction to Computers, Programs, and Java™	23
1.1	Introduction	24
1.2	What Is a Computer?	24
1.3	Programming Languages	29
1.4	Operating Systems	31
1.5	Java, the World Wide Web, and Beyond	32
1.6	The Java Language Specification, API, JDK, JRE, and IDE	33
1.7	A Simple Java Program	34
1.8	Creating, Compiling, and Executing a Java Program	37
1.9	Programming Style and Documentation	40
1.10	Programming Errors	42
1.11	Developing Java Programs Using NetBeans	45
1.12	Developing Java Programs Using Eclipse	47
Chapter 2	Elementary Programming	55
2.1	Introduction	56
2.2	Writing a Simple Program	56
2.3	Reading Input from the Console	59
2.4	Identifiers	62
2.5	Variables	62
2.6	Assignment Statements and Assignment Expressions	64
2.7	Named Constants	65
2.8	Naming Conventions	66
2.9	Numeric Data Types and Operations	67
2.10	Numeric Literals	70
2.11	Evaluating Expressions and Operator Precedence	72
2.12	Case Study: Displaying the Current Time	74
2.13	Augmented Assignment Operators	76
2.14	Increment and Decrement Operators	77
2.15	Numeric Type Conversions	79
2.16	Software Development Process	81
2.17	Case Study: Counting Monetary Units	85
2.18	Common Errors and Pitfalls	87
Chapter 3	Selections	97
3.1	Introduction	98
3.2	boolean Data Type	98
3.3	if Statements	100
3.4	Two-Way if-else Statements	102
3.5	Nested if and Multi-Way if-else Statements	103
3.6	Common Errors and Pitfalls	105
3.7	Generating Random Numbers	109
3.8	Case Study: Computing Body Mass Index	111
3.9	Case Study: Computing Taxes	112
3.10	Logical Operators	115
3.11	Case Study: Determining Leap Year	119
3.12	Case Study: Lottery	120
3.13	switch Statements	122

3.14	Conditional Operators	125
3.15	Operator Precedence and Associativity	126
3.16	Debugging	128
Chapter 4	Mathematical Functions, Characters, and Strings	141
4.1	Introduction	142
4.2	Common Mathematical Functions	142
4.3	Character Data Type and Operations	147
4.4	The String Type	152
4.5	Case Studies	161
4.6	Formatting Console Output	167
Chapter 5	Loops	181
5.1	Introduction	182
5.2	The <code>while</code> Loop	182
5.3	Case Study: Guessing Numbers	185
5.4	Loop Design Strategies	188
5.5	Controlling a Loop with User Confirmation or a Sentinel Value	190
5.6	The <code>do-while</code> Loop	192
5.7	The <code>for</code> Loop	195
5.8	Which Loop to Use?	198
5.9	Nested Loops	200
5.10	Minimizing Numeric Errors	202
5.11	Case Studies	204
5.12	Keywords <i>break</i> and <i>continue</i>	208
5.13	Case Study: Checking Palindromes	211
5.14	Case Study: Displaying Prime Numbers	213
Chapter 6	Methods	227
6.1	Introduction	228
6.2	Defining a Method	228
6.3	Calling a Method	230
6.4	<code>void</code> vs. Value-Returning Methods	233
6.5	Passing Parameters by Values	236
6.6	Modularizing Code	239
6.7	Case Study: Converting Hexadecimals to Decimals	241
6.8	Overloading Methods	243
6.9	The Scope of Variables	246
6.10	Case Study: Generating Random Characters	247
6.11	Method Abstraction and Stepwise Refinement	249
Chapter 7	Single-Dimensional Arrays	269
7.1	Introduction	270
7.2	Array Basics	270
7.3	Case Study: Analyzing Numbers	277
7.4	Case Study: Deck of Cards	278
7.5	Copying Arrays	280
7.6	Passing Arrays to Methods	281
7.7	Returning an Array from a Method	284
7.8	Case Study: Counting the Occurrences of Each Letter	285
7.9	Variable-Length Argument Lists	288
7.10	Searching Arrays	289
7.11	Sorting Arrays	293

7.12	The Arrays Class	294
7.13	Command-Line Arguments	296
Chapter 8	Multidimensional Arrays	311
8.1	Introduction	312
8.2	Two-Dimensional Array Basics	312
8.3	Processing Two-Dimensional Arrays	315
8.4	Passing Two-Dimensional Arrays to Methods	317
8.5	Case Study: Grading a Multiple-Choice Test	318
8.6	Case Study: Finding the Closest Pair	320
8.7	Case Study: Sudoku	322
8.8	Multidimensional Arrays	325
Chapter 9	Objects and Classes	345
9.1	Introduction	346
9.2	Defining Classes for Objects	346
9.3	Example: Defining Classes and Creating Objects	348
9.4	Constructing Objects Using Constructors	353
9.5	Accessing Objects via Reference Variables	354
9.6	Using Classes from the Java Library	358
9.7	Static Variables, Constants, and Methods	361
9.8	Visibility Modifiers	366
9.9	Data Field Encapsulation	368
9.10	Passing Objects to Methods	371
9.11	Array of Objects	375
9.12	Immutable Objects and Classes	377
9.13	The Scope of Variables	379
9.14	The <code>this</code> Reference	380
Chapter 10	Object-Oriented Thinking	389
10.1	Introduction	390
10.2	Class Abstraction and Encapsulation	390
10.3	Thinking in Objects	394
10.4	Class Relationships	397
10.5	Case Study: Designing the Course Class	400
10.6	Case Study: Designing a Class for Stacks	402
10.7	Processing Primitive Data Type Values as Objects	404
10.8	Automatic Conversion between Primitive Types and Wrapper Class Types	407
10.9	The <code>BigInteger</code> and <code>BigDecimal</code> Classes	408
10.10	The <code>String</code> Class	410
10.11	The <code>StringBuilder</code> and <code>StringBuffer</code> Classes	416
Chapter 11	Inheritance and Polymorphism	433
11.1	Introduction	434
11.2	Superclasses and Subclasses	434
11.3	Using the <code>super</code> Keyword	440
11.4	Overriding Methods	443
11.5	Overriding vs. Overloading	444
11.6	The <code>Object</code> Class and Its <code>toString()</code> Method	446
11.7	Polymorphism	447
11.8	Dynamic Binding	447
11.9	Casting Objects and the <code>instanceof</code> Operator	451
11.10	The Object's <code>equals</code> Method	455

11.11	The ArrayList Class	456
11.12	Useful Methods for Lists	462
11.13	Case Study: A Custom Stack Class	463
11.14	The protected Data and Methods	464
11.15	Preventing Extending and Overriding	467
Chapter 12	Exception Handling and Text I/O	475
12.1	Introduction	476
12.2	Exception-Handling Overview	476
12.3	Exception Types	481
12.4	More on Exception Handling	484
12.5	The finally Clause	492
12.6	When to Use Exceptions	493
12.7	Rethrowing Exceptions	494
12.8	Chained Exceptions	495
12.9	Defining Custom Exception Classes	496
12.10	The File Class	499
12.11	File Input and Output	502
12.12	Reading Data from the Web	508
12.13	Case Study: Web Crawler	510
Chapter 13	Abstract Classes and Interfaces	521
13.1	Introduction	522
13.2	Abstract Classes	522
13.3	Case Study: the Abstract Number Class	527
13.4	Case Study: Calendar and GregorianCalendar	529
13.5	Interfaces	532
13.6	The Comparable Interface	535
13.7	The Cloneable Interface	540
13.8	Interfaces vs. Abstract Classes	545
13.9	Case Study: The Rational Class	548
13.10	Class-Design Guidelines	553
Chapter 14	JavaFX Basics	563
14.1	Introduction	564
14.2	JavaFX vs Swing and AWT	564
14.3	The Basic Structure of a JavaFX Program	564
14.4	Panes, Groups, UI Controls, and Shapes	567
14.5	Property Binding	570
14.6	Common Properties and Methods for Nodes	573
14.7	The Color Class	575
14.8	The Font Class	576
14.9	The Image and ImageView Classes	578
14.10	Layout Panes and Groups	580
14.11	Shapes	589
14.12	Case Study: The ClockPane Class	602
Chapter 15	Event-Driven Programming and Animations	615
15.1	Introduction	616
15.2	Events and Event Sources	618
15.3	Registering Handlers and Handling Events	619
15.4	Inner Classes	623
15.5	Anonymous Inner Class Handlers	624

15.6	Simplifying Event Handling Using Lambda Expressions	627
15.7	Case Study: Loan Calculator	631
15.8	Mouse Events	633
15.9	Key Events	635
15.10	Listeners for Observable Objects	638
15.11	Animation	640
15.12	Case Study: Bouncing Ball	648
15.13	Case Study: US Map	652
Chapter 16	JavaFX UI Controls and Multimedia	665
16.1	Introduction	666
16.2	Labeled and Label	666
16.3	Button	668
16.4	CheckBox	670
16.5	RadioButton	673
16.6	TextField	676
16.7	TextArea	677
16.8	ComboBox	681
16.9	ListView	684
16.10	ScrollBar	687
16.11	Slider	690
16.12	Case Study: Developing a Tic-Tac-Toe Game	693
16.13	Video and Audio	698
16.14	Case Study: National Flags and Anthems	701
Chapter 17	Binary I/O	713
17.1	Introduction	714
17.2	How Is Text I/O Handled in Java?	714
17.3	Text I/O vs. Binary I/O	715
17.4	Binary I/O Classes	716
17.5	Case Study: Copying Files	726
17.6	Object I/O	728
17.7	Random-Access Files	733
Chapter 18	Recursion	741
18.1	Introduction	742
18.2	Case Study: Computing Factorials	742
18.3	Case Study: Computing Fibonacci Numbers	745
18.4	Problem Solving Using Recursion	748
18.5	Recursive Helper Methods	750
18.6	Case Study: Finding the Directory Size	753
18.7	Case Study: Tower of Hanoi	755
18.8	Case Study: Fractals	758
18.9	Recursion vs. Iteration	762
18.10	Tail Recursion	762
Chapter 19	Generics	773
19.1	Introduction	774
19.2	Motivations and Benefits	774
19.3	Defining Generic Classes and Interfaces	776
19.4	Generic Methods	778
19.5	Case Study: Sorting an Array of Objects	780

19.6	Raw Types and Backward Compatibility	782
19.7	Wildcard Generic Types	783
19.8	Erasure and Restrictions on Generics	786
19.9	Case Study: Generic Matrix Class	788
Chapter 20	Lists, Stacks, Queues, and Priority Queues	797
20.1	Introduction	798
20.2	Collections	798
20.3	Iterators	802
20.4	Using the <code>forEach</code> Method	803
20.5	Lists	804
20.6	The <code>Comparator</code> Interface	809
20.7	Static Methods for Lists and Collections	813
20.8	Case Study: Bouncing Balls	816
20.9	Vector and Stack Classes	820
20.10	Queues and Priority Queues	821
20.11	Case Study: Evaluating Expressions	825
Chapter 21	Sets and Maps	837
21.1	Introduction	838
21.2	Sets	838
21.3	Comparing the Performance of Sets and Lists	846
21.4	Case Study: Counting Keywords	849
21.5	Maps	850
21.6	Case Study: Occurrences of Words	855
21.7	Singleton and Unmodifiable Collections and Maps	857
Chapter 22	Developing Efficient Algorithms	861
22.1	Introduction	862
22.2	Measuring Algorithm Efficiency Using Big <i>O</i> Notation	862
22.3	Examples: Determining Big <i>O</i>	864
22.4	Analyzing Algorithm Time Complexity	868
22.5	Finding Fibonacci Numbers Using Dynamic Programming	871
22.6	Finding Greatest Common Divisors Using Euclid's Algorithm	873
22.7	Efficient Algorithms for Finding Prime Numbers	877
22.8	Finding the Closest Pair of Points Using Divide-and-Conquer	883
22.9	Solving the Eight Queens Problem Using Backtracking	886
22.10	Computational Geometry: Finding a Convex Hull	889
Chapter 23	Sorting	903
23.1	Introduction	904
23.2	Insertion Sort	904
23.3	Bubble Sort	906
23.4	Merge Sort	909
23.5	Quick Sort	912
23.6	Heap Sort	916
23.7	Bucket and Radix Sorts	923
23.8	External Sort	925

Chapter 24	Implementing Lists, Stacks, Queues, and Priority Queues	939
24.1	Introduction	940
24.2	Common Operations for Lists	940
24.3	Array Lists	944
24.4	Linked Lists	951
24.5	Stacks and Queues	965
24.6	Priority Queues	969
Chapter 25	Binary Search Trees	975
25.1	Introduction	976
25.2	Binary Search Trees	976
25.3	Deleting Elements from a BST	989
25.4	Tree Visualization and MVC	995
25.5	Iterators	998
25.6	Case Study: Data Compression	1000
Chapter 26	AVL Trees	1011
26.1	Introduction	1012
26.2	Rebalancing Trees	1012
26.3	Designing Classes for AVL Trees	1015
26.4	Overriding the insert Method	1016
26.5	Implementing Rotations	1017
26.6	Implementing the delete Method	1018
26.7	The AVLTree Class	1018
26.8	Testing the AVLTree Class	1024
26.9	AVL Tree Time Complexity Analysis	1027
Chapter 27	Hashing	1031
27.1	Introduction	1032
27.2	What Is Hashing?	1032
27.3	Hash Functions and Hash Codes	1033
27.4	Handling Collisions Using Open Addressing	1035
27.5	Handling Collisions Using Separate Chaining	1039
27.6	Load Factor and Rehashing	1039
27.7	Implementing a Map Using Hashing	1041
27.8	Implementing Set Using Hashing	1050
Chapter 28	Graphs and Applications	1061
28.1	Introduction	1062
28.2	Basic Graph Terminologies	1063
28.3	Representing Graphs	1064
28.4	Modeling Graphs	1070
28.5	Graph Visualization	1080
28.6	Graph Traversals	1083
28.7	Depth-First Search (DFS)	1084
28.8	Case Study: The Connected Circles Problem	1088
28.9	Breadth-First Search (BFS)	1090
28.10	Case Study: The Nine Tails Problem	1093

Chapter 29	Weighted Graphs and Applications	1107
29.1	Introduction	1108
29.2	Representing Weighted Graphs	1109
29.3	The WeightedGraph Class	1111
29.4	Minimum Spanning Trees	1119
29.5	Finding Shortest Paths	1125
29.6	Case Study: The Weighted Nine Tails Problem	1134
Chapter 30	Aggregate Operations for Collection Streams	1145
30.1	Introduction	1146
30.2	Stream Pipelines	1146
30.3	IntStream, LongStream, and DoubleStream	1152
30.4	Parallel Streams	1155
30.5	Stream Reduction Using the reduce Method	1157
30.6	Stream Reduction Using the collect Method	1160
30.7	Grouping Elements Using the groupingby Collector	1163
30.8	Case Studies	1166

Chapter 31–44 are available from the Companion Website at www.pearsonglobaleditions.com/Liang

Chapter 31	Advanced JavaFX and FXML
Chapter 32	Multithreading and Parallel Programming
Chapter 33	Networking
Chapter 34	Java Database Programming
Chapter 35	Advanced Java Database Programming
Chapter 36	Internationalization
Chapter 37	Servlets
Chapter 38	JavaServer Pages
Chapter 39	JavaServer Faces
Chapter 40	Remote Method Invocation
Chapter 41	Web Services
Chapter 42	2-4 Trees and B-Trees
Chapter 43	Red-Black Trees
Chapter 44	Testing Using JUnit

APPENDIXES	1177
Appendix A Java Keywords	1179
Appendix B The ASCII Character Set	1180
Appendix C Operator Precedence Chart	1182
Appendix D Java Modifiers	1184
Appendix E Special Floating-Point Values	1186
Appendix F Number Systems	1187
Appendix G Bitwise Operations	1191
Appendix H Regular Expressions	1192
Appendix I Enumerated Types	1197
QUICK REFERENCE	1203
INDEX	1205

VideoNotes

Locations of VideoNotes

www.pearsonglobaleditions.com/Liang



VideoNote

Chapter 1	Introduction to Computers, Programs, and Java™	23	Coupon collector's problem	304	
	Your first Java program	34	Consecutive four	306	
	Compile and run a Java program	39	Chapter 8	Multidimensional Arrays	311
	NetBeans brief tutorial	45		Find the row with the largest sum	316
	Eclipse brief tutorial	47		Grade multiple-choice test	318
Chapter 2	Elementary Programming	55		Sudoku	322
	Obtain input	59		Multiply two matrices	331
	Use operators / and %	74		Even number of 1s	338
	Software development process	81	Chapter 9	Objects and Classes	345
	Compute loan payments	82		Define classes and objects	346
	Compute BMI	94		Use classes	358
Chapter 3	Selections	97		Static vs. instance	361
	Program addition quiz	99		Data field encapsulation	368
	Program subtraction quiz	109		The this keyword	380
	Use multi-way if-else statements	112		The Fan class	386
	Sort three integers	132	Chapter 10	Object-Oriented Thinking	389
	Check point location	134		The Loan class	391
Chapter 4	Mathematical Functions, Characters, and Strings	141		The BMI class	394
	Introduce Math functions	142		The StackOfIntegers class	402
	Introduce strings and objects	152		Process large numbers	408
	Convert hex to decimal	165		The String class	410
	Compute great circle distance	173		The MyPoint class	424
	Convert hex to binary	176	Chapter 11	Inheritance and Polymorphism	433
Chapter 5	Loops	181		Geometric class hierarchy	434
	Use while loop	182		Polymorphism and dynamic binding demo	448
	Guess a number	185		The ArrayList class	456
	Multiple subtraction quiz	188		The MyStack class	463
	Use do-while loop	192		New Account class	470
	Minimize numeric errors	202	Chapter 12	Exception Handling and Text I/O	475
	Display loan schedule	219		Exception-handling advantages	476
	Sum a series	220		Create custom exception classes	496
Chapter 6	Methods	227		Write and read data	502
	Define/invoke max method	230		HexFormatException	515
	Use void method	233	Chapter 13	Abstract Classes and Interfaces	521
	Modularize code	239		Abstract GeometricObject class	522
	Stepwise refinement	249		Calendar and Gregorian	
	Reverse an integer	258		Calendar classes	529
	Estimate π	261		The concept of interface	532
Chapter 7	Single-Dimensional Arrays	269		Redesign the Rectangle class	558
	Random shuffling	274	Chapter 14	JavaFX Basics	563
	Deck of cards	278		Getting started with JavaFX	564
	Selection sort	293		Understand property binding	570
	Command-line arguments	297		Use Image and ImageView	578
				Use layout panes	580
				Use shapes	589

22 VideoNotes

	Display a tic-tac-toe board	608		Tic-Tac-Toe	693
	Display a bar chart	610		Use Media, MediaPlayer, and MediaPlayerView	698
Chapter 15	Event-Driven Programming and Animations	615		Use radio buttons and text fields	705
	Handler and its registration	622		Set fonts	707
	Anonymous handler	625	Chapter 17	Binary I/O	713
	Move message using the mouse	634		Copy file	726
	Animate a rising flag	640		Object I/O	728
	Flashing text	646		Split a large file	738
	Simple calculator	656	Chapter 18	Recursion	741
	Check mouse-point location	658		Binary search	752
	Display a running fan	661		Directory size	753
Chapter 16	JavaFX UI Controls and Multimedia	665		Fractal (Sierpinski triangle)	758
	Use ListView	684		Search a string in a directory	769
	Use Slider	690		Recursive tree	772

Animations



Chapter 7	Single-Dimensional Arrays	269		radix sort on Companion Website	924
	linear search animation on Companion Website	290	Chapter 24	Implementing Lists, Stacks, Queues, and Priority Queues	939
	binary search animation on Companion Website	290		list animation on Companion Website	940
	selection sort animation on Companion Website	293		stack and queue animation on Companion Website	965
Chapter 8	Multidimensional Arrays	311	Chapter 25	Binary Search Trees	975
	closest-pair animation on the Companion Website	320		BST animation on Companion Website	976
Chapter 22	Developing Efficient Algorithms	861	Chapter 26	AVL Trees	1011
	binary search animation on the Companion Website	868		AVL tree animation on Companion Website	1012
	selection sort animation on the Companion Website	868	Chapter 27	Hashing	1031
	closest-pair animation on Companion Website	883		linear probing animation on Companion Website	1036
	Eight Queens animation on the Companion Website	886		quadratic probing animation on Companion Website	1037
	convex hull animation on the Companion Website	889		separate chaining animation on Companion Website	1040
Chapter 23	Sorting	903	Chapter 28	Graphs and Applications	1061
	insertion-sort animation on Companion Website	904		graph learning tool on Companion Website	1064
	bubble sort animation on the Companion Website	907		U.S. Map Search	1086
	merge animation on Companion Website	911	Chapter 29	Weighted Graphs and Applications	1107
	partition animation on Companion Website	915		weighted graph learning tool animation on Companion Website	1108
	heap animation on Companion Website	917			

CHAPTER

1

INTRODUCTION TO COMPUTERS, PROGRAMS, AND JAVA™

Objectives

- To understand computer basics, programs, and operating systems (§§1.2–1.4).
- To describe the relationship between Java and the World Wide Web (§1.5).
- To understand the meaning of Java language specification, API, JDK™, JRE™, and IDE (§1.6).
- To write a simple Java program (§1.7).
- To display output on the console (§1.7).
- To explain the basic syntax of a Java program (§1.7).
- To create, compile, and run Java programs (§1.8).
- To use sound Java programming style and document programs properly (§1.9).
- To explain the differences between syntax errors, runtime errors, and logic errors (§1.10).
- To develop Java programs using NetBeans™ (§1.11).
- To develop Java programs using Eclipse™ (§1.12).





what is programming?
programming
program

1.1 Introduction

The central theme of this book is to learn how to solve problems by writing a program.

This book is about programming. So, what is programming? The term *programming* means to create (or develop) software, which is also called a *program*. In basic terms, software contains instructions that tell a computer—or a computerized device—what to do.

Software is all around you, even in devices you might not think would need it. Of course, you expect to find and use software on a personal computer, but software also plays a role in running airplanes, cars, cell phones, and even toasters. On a personal computer, you use word processors to write documents, web browsers to explore the Internet, and e-mail programs to send and receive messages. These programs are all examples of software. Software developers create software with the help of powerful tools called *programming languages*.

This book teaches you how to create programs by using the Java programming language. There are many programming languages, some of which are decades old. Each language was invented for a specific purpose—to build on the strengths of a previous language, for example, or to give the programmer a new and unique set of tools. Knowing there are so many programming languages available, it would be natural for you to wonder which one is best. However, in truth, there is no “best” language. Each one has its own strengths and weaknesses. Experienced programmers know one language might work well in some situations, whereas a different language may be more appropriate in other situations. For this reason, seasoned programmers try to master as many different programming languages as they can, giving them access to a vast arsenal of software-development tools.

If you learn to program using one language, you should find it easy to pick up other languages. The key is to learn how to solve problems using a programming approach. That is the main theme of this book.

You are about to begin an exciting journey: learning how to program. At the outset, it is helpful to review computer basics, programs, and operating systems (OSs). If you are already familiar with such terms as central processing unit (CPU), memory, disks, operating systems, and programming languages, you may skip Sections 1.2–1.4.

1.2 What Is a Computer?

A computer is an electronic device that stores and processes data.

A computer includes both *hardware* and *software*. In general, hardware comprises the visible, physical elements of the computer, and software provides the invisible instructions that control the hardware and make it perform specific tasks. Knowing computer hardware isn’t essential to learning a programming language, but it can help you better understand the effects that a program’s instructions have on the computer and its components. This section introduces computer hardware components and their functions.

A computer consists of the following major hardware components (see Figure 1.1):

- A central processing unit (CPU)
- Memory (main memory)
- Storage devices (such as disks and CDs)
- Input devices (such as the mouse and the keyboard)
- Output devices (such as monitors and printers)
- Communication devices (such as modems and network interface cards (NIC))

A computer’s components are interconnected by a subsystem called a *bus*. You can think of a bus as a sort of system of roads running among the computer’s components; data and power travel along the bus from one part of the computer to another. In personal computers,



hardware
software

bus

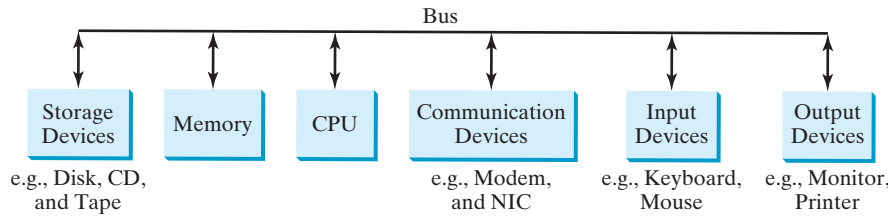


FIGURE I.1 A computer consists of a CPU, memory, storage devices, input devices, output devices, and communication devices.

the bus is built into the computer’s *motherboard*, which is a circuit case that connects all of the parts of a computer together. motherboard

1.2.1 Central Processing Unit

The *central processing unit (CPU)* is the computer’s brain. It retrieves instructions from the memory and executes them. The CPU usually has two components: a *control unit* and an *arithmetic/logic unit*. The control unit controls and coordinates the actions of the other components. The arithmetic/logic unit performs numeric operations (addition, subtraction, multiplication, and division) and logical operations (comparisons). CPU

Today’s CPUs are built on small silicon semiconductor chips that contain millions of tiny electric switches, called *transistors*, for processing information.

Every computer has an internal clock that emits electronic pulses at a constant rate. These pulses are used to control and synchronize the pace of operations. A higher clock *speed* enables more instructions to be executed in a given period of time. The unit of measurement of clock speed is the *hertz (Hz)*, with 1 Hz equaling 1 pulse per second. In the 1990s, computers measured clock speed in *megahertz (MHz)*, but CPU speed has been improving continuously; the clock speed of a computer is now usually stated in *gigahertz (GHz)*. Intel’s newest processors run at about 3 GHz. speed hertz megahertz gigahertz

CPUs were originally developed with only one core. The *core* is the part of the processor that performs the reading and executing of instructions. In order to increase the CPU processing power, chip manufacturers are now producing CPUs that contain multiple cores. A multicore CPU is a single component with two or more independent cores. Today’s consumer computers typically have two, three, and even four separate cores. Soon, CPUs with dozens or even hundreds of cores will be affordable. core

1.2.2 Bits and Bytes

Before we discuss memory, let’s look at how information (data and programs) are stored in a computer.

A computer is really nothing more than a series of switches. Each switch exists in two states: on or off. Storing information in a computer is simply a matter of setting a sequence of switches on or off. If the switch is on, its value is 1. If the switch is off, its value is 0. These 0s and 1s are interpreted as digits in the binary number system and are called *bits* (binary digits). bits

The minimum storage unit in a computer is a *byte*. A byte is composed of eight bits. A small number such as 3 can be stored as a single byte. To store a number that cannot fit into a single byte, the computer uses several bytes. byte

Data of various kinds, such as numbers and characters, are encoded as a series of bytes. As a programmer, you don’t need to worry about the encoding and decoding of data, which the computer system performs automatically, based on the encoding scheme. An *encoding scheme* is a set of rules that govern how a computer translates characters and numbers into data with which the computer can actually work. Most schemes translate each character into a encoding scheme