Welcome to the Fourth Edition of *Java Software Solutions, Foundations of Program Design*. We are pleased that this book has served the needs of so many students and faculty over the years. This edition is designed to further enhance the pedagogy of introductory computing.

**What’s New in the Fourth Edition**

The overall vision of the book has not changed significantly from that of the third edition. The emphasis remains on presenting underlying core concepts. The Graphics Track sections in each chapter still segregate the coverage of graphics and graphical user interfaces. The casual writing style and entertaining examples still rule the day.

The changes made for this edition are designed to permeate and reinforce the pedagogy of the material. These changes also address some of the trends that are happening in many Java programming courses. Specifically:

- The discussions and examples fully embrace the Java 2 Standard Edition Version 1.5, including coverage of:
  - The standard `Scanner` class, which simplifies keyboard input and parsing.
  - Enumerated types.
  - Autoboxing, which automatically converts primitives to wrapper objects and vice versa.
  - Variable length parameter lists.
  - The enhanced `for` loop, which facilitates iterating over a collection.
  - Generic types, which facilitate the definition of collection classes.

We provide a smoother evolution of object-oriented concepts, including a brief introduction to the most essential elements of writing classes prior to covering conditionals and loops. We incorporate class examples into all remaining chapters, adding key aspects of class design as appropriate.
We place a stronger emphasis on object-oriented design, including techniques for identifying classes, objects, and their relationships.

We have reorganized the flow of graphical user interface (GUI) topics into a more logical order throughout the Graphics Track sections of each chapter. The topics from the third edition’s “catch all” GUI chapter have been integrated throughout the text.

We present most example programs dealing with GUIs and graphics as frame-based Java applications instead of applets. Details of applets are now provided in an appendix.

The reference material in the appendices has been augmented with new topics, including an overview of GUI events and a summary of regular expressions for parsing using the `Scanner` class.

A whole suite of new and improved supplements is available with this edition. The supplements are discussed in detail later in the preface.

**Cornerstones of the Text**

This text is based on the following basic ideas that we believe make for a sound introductory text:

> **True object-orientation.** A text that really teaches a solid object-oriented approach must use what we call object-speak. That is, all processing should be discussed in object-oriented terms. That does not mean, however, that the first program a student sees must discuss the writing of multiple classes and methods. A student should learn to use objects before learning to write them. This text uses a natural progression that culminates in the ability to design real object-oriented solutions.

> **Sound programming practices.** Students should not be taught how to program; they should be taught how to write good software. There’s a difference. Writing software is not a set of cookbook actions, and a good program is more than a collection of statements. This text integrates practices that serve as the foundation of good programming skills. These practices are used in all examples and are reinforced in the discussions. Students learn how to solve problems as well as how to implement solutions. We introduce and integrate basic software engineering techniques throughout the text.

> **Examples.** Students learn by example. This text is filled with fully implemented examples that demonstrate specific concepts. We have intertwined
small, readily understandable examples with larger, more realistic ones. There is a balance between graphics and nongraphics programs.

> **Graphics and GUIs.** Graphics can be a great motivator for students, and their use can serve as excellent examples of object-orientation. As such, we use them throughout the text in a well-defined set of sections that we call the Graphics Track. This coverage includes the use of event processing and GUIs. Students learn to build GUIs in the appropriate way by using a natural progression of topics. The Graphics Track can be avoided entirely for those who do not choose to use graphics.

**Paths Through the Text**
This book is designed to be flexible, so that instructors can tailor its presentation to the needs of their students. Instructors can take a variety of paths through the text. Graphics can be emphasized or deemphasized as desired.

**Chapter Breakdown**
Chapter 1 (Introduction) introduces computer systems in general, including basic architecture and hardware, networking, programming, and language translation. Java is introduced in this chapter, and the basics of general program development, as well as object-oriented programming, are discussed. This chapter contains broad introductory material that can be covered while students become familiar with their development environment.

Chapter 2 (Data and Expressions) explores some of the basic types of data used in a Java program and the use of expressions to perform calculations. It discusses the conversion of data from one type to another, and how to read input interactively from the user with the help of the standard **Scanner** class.

Chapter 3 (Using Classes and Objects) explores the use of predefined classes and the objects that can be created from them. Classes and objects are used to manipulate character strings, produce random numbers, perform complex calculations, and format output. Enumerated types are also discussed.

Chapter 4 (Writing Classes) explores the basic issues related to writing classes and methods. Topics include instance data, visibility, scope, method parameters, and return types. Encapsulation and constructors are covered as well. Some of the more involved topics are deferred to or revisited in Chapter 6.
Chapter 5 (Conditionals and Loops) covers the use of boolean expressions to make decisions. All related statements for conditionals and loops are discussed, including the enhanced version of the \texttt{for} loop. The \texttt{Scanner} class is revisited for iterative input parsing and reading text files.

Chapter 6 (Object-Oriented Design) reinforces and extends the coverage of issues related to the design of classes. Techniques for identifying the classes and objects needed for a problem and the relationships among them are discussed. This chapter also covers static class members, interfaces, and the design of enumerated type classes. Method design issues and method overloading are also discussed.

Chapter 7 (Arrays) contains extensive coverage of arrays and array processing. Topics include command-line arguments, variable length parameter lists, and multidimensional arrays. The \texttt{ArrayList} class and its use as a generic type is explored as well.

Chapter 8 (Inheritance) covers class derivations and associated concepts such as class hierarchies, overriding, and visibility. Strong emphasis is put on the proper use of inheritance and its role in software design.

Chapter 9 (Polymorphism) explores the concept of binding and how it relates to polymorphism. Then we examine how polymorphic references can be accomplished using either inheritance or interfaces. Sorting is used as an example of polymorphism. Design issues related to polymorphism are examined as well.

Chapter 10 (Exceptions) explores the class hierarchy from the Java standard library used to define exceptions, as well as the ability to define our own exception objects. We also discuss the use of exceptions when dealing with input and output, and examine an example that writes a text file.

Chapter 11 (Recursion) covers the concept, implementation, and proper use of recursion. Several examples from various domains are used to demonstrate how recursive techniques make certain types of processing elegant.

Chapter 12 (Data Structures) introduces the idea of a collection and its underlying data structure. Abstraction is revisited in this context and the classic data structures are explored. Generic types are introduced as well. This chapter serves as an introduction to a CS2 course.
Supplements

A variety of supplemental materials is available for this text. The following resources are available for all students at www.aw-bc.com/cssupport:

- Source code for all the programs in the text.
- Various Java development environments—these are also on the CD in the back of this book.
- The Keyboard class, for those who want access to the author-provided input class that was used in previous editions of the book. In this edition, it has been supplanted by the new standard Scanner class.
- Lab manual—lab exercises are designed to accompany the topic progression in the text.
- Addison-Wesley's CodeMate—an online environment that brings the text to life. A free subscription, included on the card in the front of this text, allows you to work with many of the text's Code Listings and Programming Projects to improve your programming skills.

In addition, the following supplements are available to qualified instructors. Please contact your Addison-Wesley sales representative, or send e-mail to aw.cse@aw.com, for information on how to access them:

- Presentation Slides—in PowerPoint, Keynote, and PDF formats.
- Instructor's Manual—includes chapter notes and teaching tips.
- Solutions—includes solutions to all exercises and programming projects.
- Test Bank in powerful test generator software—includes a wealth of free response, multiple choice, and true/false type questions.
- Lab Manual—lab exercises are designed to accompany the topic progression in the text. A printed version of this manual is also available.

Acknowledgments

We are most grateful to the faculty and students from around the world who have provided their feedback on previous editions of this book. We are pleased to see the depth of the faculty’s concern for their students and the students’ thirst for knowledge. Your comments and questions are always welcome.

We stand in awe of the talent and heroic effort demonstrated by the team at Addison-Wesley. Michael Hirsch, our editor, has amazing insight and commitment. Susan Hartman Sullivan has been a brilliant guiding force since this book’s
inception. We thank Marketing Manager Michelle Brown and Marketing Coordinator Lesly Hershman for making sure that instructors understand the pedagogical advantages of the text. The cover and interior design were designed by the skilled talents of Joyce Wells. Jeff Holcomb led the amazing production effort given completely unreasonable time constraints. The Addison-Wesley folks are supported by the phenomenal team at Argosy Publishing, including Daniel Rausch and Edalin Michael. We thank all of these people for ensuring that this book met the highest quality standards.

Special thanks go to the following people who provided valuable advice to us about this edition via their participation in focus groups, interviews, and reviews: Robert Burton—Brigham Young University; Robert Cohen—University of Massachusetts, Boston; Christopher Eliot—University of Massachusetts, Amherst; Matt Evett—Eastern Michigan University; James Heliotis—Rochester Institute of Technology; Blayne Mayfield—Oklahoma State University; Carolyn Schauble—Colorado State University; and Faye Navabi-Tadayon—Arizona State University.

The reviewers of previous editions of this text, as well as many other instructors and friends, have provided valuable feedback. They include:

Lewis Barnett University of Richmond
Tom Bennet Mississippi College
Gian Mario Besana DePaul University
Hans-Peter Bischof Rochester Institute of Technology
Robert Burton Brigham Young University
James Cross Auburn University
Eman El-Sheikh University of West Florida
John Gauch University of Kansas
Chris Haynes Indiana University
Laurie Hendren McGill University
Mike Higgs Austin College
Karen Kluge Dartmouth College
Jason Levy University of Hawaii
Peter MacKenzie McGill University
Blayne Mayfield Oklahoma State University
Lawrence Osborne Lamar University
Barry Pollack City College of San Francisco
Thanks also go to my colleagues at Villanova University who have provided so much wonderful feedback. They include Bob Beck, Cathy Helwig, Dan Joyce, Anany Levitin, Najib Nadi, Beth Taddei, and Barbara Zimmerman.

Special thanks go to Pete DePasquale of The College of New Jersey for the design and evolution of the PaintBox project, as well as the original Java Class Library appendix.

Many other people have helped in various ways. They include Ken Arnold, Kevin Henry, John Loftus, Sebastian Niezgoda, and Sammy Perugini. Our apologies to anyone we may have forgotten.

The ACM Special Interest Group on Computer Science Education (SIGCSE) is a tremendous resource. Their conferences provide an opportunity for educators from all levels and all types of schools to share ideas and materials. If you are an educator in any area of computing and are not involved with SIGCSE, you’re missing out.