Second Edition

Java™ Foundations

Introduction to Program Design & Data Structures

Second Edition

Java™ Foundations

Introduction to Program Design & Data Structures

John Lewis

Virginia Tech

Peter DePasquale

The College of New Jersey

Joseph Chase

Radford University

Addison-Wesley

Boston Columbus Indianapolis New York San Francisco
Upper Saddle River Amsterdam Cape Town Dubai London Madrid
Milan Munich Paris Montreal Toronto Delhi Mexico City
Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

Editor-in-Chief:	Michael Hirsch
Editorial Assistant:	Stephanie Sellinger
Director of Marketing:	Margaret Whaples
Marketing Coordinator:	Kathryn Ferranti
Managing Editor:	Jeffrey Holcomb
Production Project Manager:	Heather McNally
Senior Manufacturing Buyer:	Carol Melville
Media Manufacturing Buyer:	Ginny Michaud
Art Director:	Linda Knowles
Cover Designer:	Susan Paradise
Cover Art:	Marc Romanelli/Getty Images
Media Project Manager:	Katelyn Boller
Full-Service Project Management:	Rose Kernan, Nesbitt Graphics, Inc.
Composition:	Nesbitt Graphics, Inc.

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on appropriate page within text.

The programs and applications presented in this book have been included for their instructional value. They have been tested with care, but are not guaranteed for any particular purpose. The publisher does not offer any warranties or representations, nor does it accept any liabilities with respect to the programs or applications.

Library of Congress Cataloging-in-Publication Data
Lewis, John, 1963-
Java foundations : introduction to program design & data structures / John
Lewis, Peter J. DePasquale, Joseph Chase. -- 2nd ed.
p. cm.
ISBN 0-13-212881-0
1. Java (Computer program language) I. DePasquale, Peter J. (Peter Joseph)
II. Chase, Joseph, 1964- III. Title.
QA76.73.J38L48845 2010
005.13'3--dc22
2009051134

10 9 8 7 6 5 4 3 2 1–EB–14 13 12 11 10

Addison-Wesley
is an imprint of

PEARSON

www.pearsonhighered.com

ISBN-10: 0-13-212881-0
ISBN-13: 978-0-13-212881-0

To my wife, Sharon, for everything.
– John

To my wife, Lisa, and our twins: Lily and Adam.
– Pete

To my loving wife, Melissa, for her support and encouragement.
– Joe

Preface

Welcome to *Java Foundations*. This book is designed to serve as the primary resource for a two- or three-term introductory course sequence, ranging from the most basic programming concepts to the design and implementation of complex data structures. This unified approach makes the important introductory sequence more cohesive and accessible for students.

We've borrowed the best elements from the industry-leading text *Java Software Solutions* for the introductory material, reworked to complement the design and vision of the overall text. For example, instead of having graphics sections spread throughout many chapters, the coverage of graphical user interfaces is accomplished in a well-organized chapter of its own.

In the later chapters, the exploration of collections and data structures is modeled somewhat after the coverage in *Java Software Structures*, but has been thoroughly retooled to flow cleanly from the introductory material. The result is a comprehensive, cohesive, and seamless exploration of programming concepts.

New in the Second Edition

We appreciate the feedback we've received about this book and are pleased that it served so well as an introductory text. The following modifications have been made to improve the presentation of particular topics and the overall flow:

- A stack is now used as the initial example of a collection so that the concept of a collection is more clearly established.
- The discussion of Generics has been expanded and clarified.
- The coverage of the Quick Sort and Merge Sort algorithms has been expanded.
- The coverage of Analysis of Algorithms has been separated into its own chapter and expanded.
- Material on Testing and Debugging has been incorporated into more appropriate locations of the text.
- The coverage of Search Trees and Heaps have been divided into separate chapters.

- Two new chapters covering Hashing and Databases have been added.
- End-of-chapter exercises and projects have been updated to reflect changes in the book.

Regarding Objects

Phrases like *objects-first*, *objects-early*, and *objects-late* continue to be bandied about by computing educators, despite the fact that the nuances of the pedagogy of the introductory sequence cannot be summed up so easily. We'll take this opportunity to discuss our approach.

First, this book is **purely object-oriented**, presented in a gradual, natural manner. Concepts that overlap with procedural programming, such as methods and their invocation, are discussed in terms of an object-oriented approach. Thus, no example is ever made up of a single class with multiple methods. In fact, in our examples the class that contains the `main` method never contains another.

We **use objects** right from the start, and discuss everything in object-oriented terms at all times. An overview of object-oriented concepts is given in Chapter 1, then reinforced and fleshed out throughout the book. Classes from the Java standard class library are introduced immediately, and objects from these classes are instantiated and used for the various services they provide. In the first four chapters, students explore and write programs made up of a single class with a single `main` method—but these programs actively *use* predefined classes and objects from the standard library in addition to exploring fundamental programming concepts such as expressions and conditionals.

We **never introduce third-party classes** simply as fodder to create examples. That approach can confuse students by blurring the distinction between classes that are part of the standard library (and thus always available) and "extras" thrown in by textbook authors as a convenience. Every non-library class used in an example is fully explored in this book. There's no "magic" behind the scenes.

The debate continues: should coverage of control structures come before the details of writing classes, or vice versa? The truth is there are advantages either way, and a knowledgeable instructor can capitalize on either approach. If class composition comes first, it exposes the underlying essence of objects earlier and demystifies their use. However, without the ability to use basic control structures, the examples at that point are often uninteresting and unrealistic. This book **explores control structures before writing classes**. Chapter 4 uses small, single-method examples to examine the details of conditionals and loops, providing a strong foundation for the multiclass examples in Chapter 5.

Chapter Breakdown

Chapter 1 (**Introduction**) introduces the Java programming language and the basics of program development. It contains an introduction to object-oriented development, including an overview of concepts and terminology. 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 `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. Packages, enumerated types, and wrapper classes are also discussed.

Chapter 4 (**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 `for` loop. The `Scanner` class is revisited for iterative input parsing and reading text files.

Chapter 5 (**Writing Classes**) explores the basic issues related to writing classes and methods. Topics include instance data, visibility, scope, method parameters, and return types. Constructors, method design, static data, and method overloading are covered as well. Testing and debugging are now covered in this chapter as well.

Chapter 6 (**Graphical User Interfaces**) is a thorough exploration of Java GUI processing, focusing on components, events, and listeners. Many types of components and events are discussed using numerous GUI examples. Additionally, layout mangers, containment hierarchies, borders, tooltips, and mnemonics are introduced.

Chapter 7 (**Arrays**) contains extensive coverage of arrays and array processing. Topics include bounds checking, initializer lists, command-line arguments, variable-length parameter lists, and multidimensional arrays.

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. Design issues related to polymorphism are examined as well.

Chapter 10 (**Exceptions**) covers exception handling and the effects of uncaught exceptions. The `try-catch` statement is examined, as well as a discussion of exception propagation. The chapter also explores the use of exceptions when dealing with input and output, and examines an example that writes a text file.

Chapter 11 (**Recursion**) covers the concept, implementation, and proper use of recursion. Several examples are used to elaborate on the discussion, including a maze traversal and the classic Towers of Hanoi problem.

Chapter 12 (**Analysis of Algorithms**) discusses the techniques for analyzing the complexity of algorithms, including recursive algorithms. Big Oh notation is introduced.

Chapter 13 (**Searching and Sorting**) explores the linear and binary searching algorithms, as well as five sorting algorithms. The sorts include both quadratic and O(N log N) algorithms. The efficiency of these algorithms is examined.

Chapter 14 (**Stacks**) introduces the concept of a collection and establishes the importance of separating the interface from the implementation. Stacks are used as the initial example of a collection, and both dynamic and fixed implementations of a stack are explored. Generic types are introduced in this chapter, detailing their use in supporting the collection classes.

Chapter 15 (**Queues**) introduces FIFO queues and discusses options for their implementation. As with stacks, a queue is explored first conceptually, then as tools to help us solve problems, and finally by examining their underlying data structures. Both array-based and dynamic link implementations are discussed.

Chapter 16 (**Trees**) introduces the terms and concepts behind trees. Various implementation strategies are discussed, and a recursive, linked approach is examined in detail. An example of a binary decision tree is explored as well.

Chapter 17 (**Binary Search Trees**) covers the concept of search trees and a linked implementation for a classic binary search tree. Tree rotation algorithms are also discussed.

Chapter 18 (**Heaps and Priority Queues**) discusses the concept of a heap and its relationship to trees. A full linked implementation of a heap is explored. Priority queues are used as an example of a collection in its own right, and the natural relationship between heaps and priority queues are explored.

Chapter 19 (**Graphs**) discusses both directed and undirected graphs. Additionally, weighted graphs are explored, and the differences between breadth-first and depth-first graph traversals are covered. Minimal spanning trees are introduced, and implementation strategies are discussed.

Chapter 20 (**Hashing**) covers the concept of creating a hash table to facilitate storage and retrieval of objects. Various classes that relate to hashing from the Java API are explored.

Chapter 21 (Databases) explores the concept of databases and their management, and discusses the basics of SQL queries. It then explores the techniques for establishing a connection between a Java program and a database, and the API used to interact with it.

Student CD

The CD included with each textbook contains:

- Source code for all of the programs in the text.
- The Java Software Development Kit (SDK).
- Various Java development environments, including NetBeans™, Eclipse™, DrJava, jGRASP™, and TextPad®.

Instructor Resources

The following supplements are available to qualified instructors only. Visit the Pearson Education Instructor Resource Center (www.pearsonhighered.com/irc) or send email to computing@aw.com for information on how to access these resources.

- Presentation Slides—lecture-ready presentations for each chapter in Microsoft PowerPoint® format.
- Solutions—full solutions to the exercises and programming projects.
- Test Bank with powerful test generator software—includes a wealth of free-response, multiple-choice, and true/false questions.

Acknowledgments

Educators and students from around the world have provided feedback on previous work that has allowed us to mold this book into a fresh, valuable resource. Your comments and questions are always welcome.

The talent and commitment of the team at Addison-Wesley continues to amaze us. We greatly appreciate the insight of Michael Hirsch, our editor, and the hard work of his assistant, Stephanie Sellinger. Rose Kernan at Nesbitt Graphics was a great help throughout the production process. We thank all of these people for ensuring that this book meets the highest quality standards.

We'd like to acknowledge the collective input from hundreds of professors and students around the world in the development of the material upon which this

book is based. There are too many of you to individually name, but your influence on *Java Software Solutions* and *Java Software Structures* is evident in *Java Foundations*.

Special thanks go to Ruth Dannenfelser, Cory Samaha, and Zach Zappala at the College of New Jersey for their help with solutions to the database projects. And our continued thanks go to Jason Snyder at Virginia Tech for his assistance testing code and many other contributions.

Groups like the ACM Special Interest Group on Computer Science Education (SIGCSE), the Consortium for Computing Sciences in Colleges (CCSC), and the Computer Science Teachers Association (CSTA) are phenomenal resources. Their conferences and online activities provide opportunities for educators from all levels and all types of schools to share ideas and materials. If you are a computing educator and are not involved with these groups, you're missing out.

Finally, we thank our families for their support and patience during the busy process of writing.

Contents

Second Edition

Java™ Foundations

Introduction to
Program Design & Data Structures