Fifth Edition

Java™ Foundations

Introduction to Program Design & Data Structures

Fifth Edition

Java™ Foundations

Introduction to Program Design & Data Structures

John Lewis | Peter DePasquale | Joseph Chase
Virginia Tech | Apex Systems | Radford University

Pearson

Liudmila Habrus/123RF

Senior Vice President Courseware Portfolio Management:	Marcia J. Horton
Director, Portfolio Management: Engineering, Computer Science & Global Editions:	Julian Partridge
Portfolio Manager:	Matt Goldstein
Portfolio Management Assistant:	Meghan Jacoby
Managing Content Producer:	Scott Disanno
Content Producer:	Carole Snyder
Rights and Permissions Manager:	Ben Ferrini
Manufacturing Buyer, Higher Ed, Lake Side Communications, Inc. (LSC):	Maura Zaldivar-Garcia and Deidra Smith
Inventory Manager:	Bruce Boundy
Product Marketing Manager:	Yvonne Vannatta
Field Marketing Manager:	Demetrius Hall
Marketing Assistant:	Jon Bryant
Cover Designer:	Integra Software Services, Inc.
Cover Photo:	Liudmila Habrus/123RF
Printer/Binder:	LSC Communications, Inc.
Full-Service Project Management:	Ramya Radhakrishnan, Integra Software Services Pvt. Ltd.,

Library of Congress Cataloging-in-Publication Data on File

Names: Lewis, John, author. | DePasquale, Peter J. (Peter Joseph), author. | Chase, Joseph, author.

Title: Java foundations : introduction to program design & data structures / John Lewis, Virginia Tech, Peter DePasquale, Sailthru, Inc., Joseph Chase, Radford University.

Description: Fifth edition. | Boston : Pearson, [2020] | Includes index.

Identifiers: LCCN 2018046357 | ISBN 9780135205976 | ISBN 0135205972

Subjects: LCSH: Java (Computer program language)

Classification: LCC QA76.73.J38 L48845 2020 | DDC 005.13/3—dc23
LC record available at https://lccn.loc.gov/2018046357

1 19

Pearson

ISBN 10: 0-13-520597-2
ISBN 13: 978-0-13-520597-6

To my wife, Sharon, for everything.
– John

To my wonderful wife Susan, and our children, Grace, Anthony, Adam, Lily, EJ, and Peter IV.
Your continued love and support keep me going as always.
– 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 after the coverage in *Java Software Structures,* but has been reworked to flow cleanly from the introductory material. The result is a comprehensive, cohesive, and seamless exploration of programming concepts.

New in the Fifth Edition

We appreciate the feedback we've received about this book and are pleased that it continues to serve so well as an introductory text. The changes made in this edition build on the strong pedagogy established by previous editions while updating crucial areas.

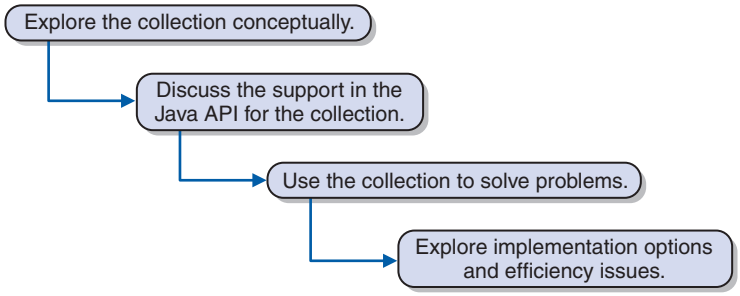
The biggest change in this edition is the overhaul of the graphical content to fully embrace the JavaFX platform, which has replaced Swing as the supported technology for graphics and Graphical User Interfaces (GUIs) in Java. The previous edition focused on Swing and had an introduction to JavaFX. The time has come to switch over completely to the new approach, which simplifies GUI development and provides better opportunities to discuss object-oriented programming.

The changes in this edition include:

- A brand new Chapter 6 on developing GUIs using JavaFX.
- A new Appendix F that discusses the rendering of graphics using JavaFX.
- A new Appendix G that explores the JavaFX Scene Builder, a drag-and-drop application for developing graphical front ends.

- Updated examples and discussions throughout the text.
- Updated end-of-chapter Programming Projects in several chapters.

In previous editions, we had established the following flow when discussing collections:



Your feedback has indicated that this approach is working well and we have continued and reinforced its use. It clarifies the distinction between the way the Java API supports a particular collection and the way it might be implemented from scratch. It makes it easier for instructors to point out limitations of the API implementations in a compare-and-contrast fashion. This approach also allows an instructor, on a case-by-case basis, to simply introduce a collection without exploring implementation details if desired.

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 an exploration of GUI processing using the JavaFX platform, focusing on controls, events, and event handlers. Several types of controls are discussed using numerous GUI examples. Mouse events, keyboard events, and layout panes are also explored.

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 (**Analysis of Algorithms**) lays the foundation for determining the efficiency of an algorithm and explains the important criteria that allow a developer to compare one algorithm to another in proper ways. Our emphasis in this chapter is understanding the important concepts more than getting mired in heavy math or formality.

Chapter 12 (**Introduction to Collections—Stacks**) establishes the concept of a collection, stressing the need to separate the interface from the implementation. It also conceptually introduces a stack, then explores an array-based implementation of a stack.

Chapter 13 (**Linked Structures—Stacks**) discusses the use of references to create linked data structures. It explores the basic issues regarding the management of linked lists, and then defines an alternative implementation of a stack (introduced in Chapter 12) using an underlying linked data structure.

Chapter 14 (**Queues**) explores the concept and implementation of a first-in, first-out queue. The Java API Queue interface is discussed, as are linked and circular array implementations with Queue in code font.

Chapter 15 (**Lists**) covers three types of lists: ordered, unordered, and indexed. These three types of lists are compared and contrasted, with discussion of the operations that they share and those that are unique to each type. Inheritance is used appropriately in the design of the various types of lists, which are implemented using both array-based and linked representations.

Chapter 16 (**Iterators**) is a new chapter that isolates the concepts and implementation of iterators, which are so important to collections. The expanded discussion drives home the need to separate the iterator functionality from the details of any particular collection.

Chapter 17 (**Recursion**) is a general introduction to the concept of recursion and how recursive solutions can be elegant. It explores the implementation details of recursion and discusses the basic idea of analyzing recursive algorithms.

Chapter 18 (**Searching and Sorting**) discusses the linear and binary search algorithms, as well as the algorithms for several sorts: selection sort, insertion sort, bubble sort, quick sort, and merge sort. Programming issues related to searching and sorting, such as using the `Comparable` interface as the basis of comparing objects, are stressed in this chapter. An application uses animation to demonstrate the efficiency of sorting algorithms. The `comparator` interface is examined and demonstrated as well.

Chapter 19 (**Trees**) provides an overview of trees, establishing key terminology and concepts. It discusses various implementation approaches and uses a binary tree to represent and evaluate an arithmetic expression.

Chapter 20 (**Binary Search Trees**) builds off of the basic concepts established in Chapter 10 to define a classic binary search tree. A linked implementation of a binary search tree is examined, followed by a discussion of how the balance in the tree nodes is key to its performance. That leads to exploring AVL and red/black implementations of binary search trees.

Chapter 21 (**Heaps and Priority Queues**) explores the concept, use, and implementations of heaps and specifically their relationship to priority queues. A heap sort is used as an example of its usefulness as well. Both linked and array-based implementations are explored.

Chapter 22 (**Sets and Maps**) explores these two types of collections and their importance to the Java Collections API.

Chapter 23 (**Multi-way Search Trees**) is a natural extension of the discussion of the previous chapters. The concepts of 2-3 trees, 2-4 trees, and general B-trees are examined and implementation options are discussed.

Chapter 24 (**Graphs**) explores the concept of undirected and directed graphs and establishes important terminology. It examines several common graph algorithms and discusses implementation options, including adjacency matrices.

Chapter 25 (**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.

Supplements

The following student resources are available for this book:

- **Source code** for all programs presented in the book
- **VideoNotes** that explore select topics from the book

Resources can be accessed at www.pearson.com/lewis

The following instructor resources can be found at Pearson Education's Instructor Resource Center:

- **Solutions** for select exercises and programming projects in the book
- **PowerPoint slides** for the presentation of the book content
- **Test bank**

To obtain access, please visit www.pearsonhighered.com/irc or contact your local Pearson Education sales representative.

Contents

Credits

Cover: Liudmila Habrus/123RF

Chapter 1 page 2: Reference: Java is a relatively new programming language compared to many others. It was developed in the early 1990s by James Gosling at Sun Microsystems. Java was released to the public in 1995 and has gained tremendous popularity since "The History of Java Technology" Oracle Corporation. 1995. Accessed at http://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html

Chapter 1 page 15: Excerpt: A research group at Auburn University has developed jGRASP, a free Java IDE that is included on the CD that accompanies this book. It can also be downloaded from www.jgrasp.org. "jGRASP" is developed by the Department of Computer Science and Software Engineering in the Samuel Ginn College of Engineering at Auburn University.

Chapter 1 page 20: Reference: The programming language Simula, developed in the 1960s, had many characteristics that define the modern object-oriented approach to software development. Nygaard, Kristen, Myhrhaug, Bjørn, and Dahl, Ole-Johan. "Simula. Common Base Language." Norwegian Computing Center. 1970. Accessed at http://www.nr.no/

Chapter 4: Excerpt: The Twelve Days of Christmas. "Twelve Days of Christmas." Mirth Without Mischief. 1780.

Chapter 11: Text: Another way of looking at the effect of algorithm complexity was proposed by Aho, Hopcroft, and Ullman. Aho, A.V., J.E. Hopcroft, and J.D. Ullman. "The Design and Analysis of Computer Algorithms." Addison-Wesley. 1974.

Chapter 20: Text: Adel'son-Vel'skii and Landis developed a method called AVL trees that is a variation on this theme. For each node in the tree, we will keep track of the height of the left and right subtrees. Adelson-Velskii, Georgii and Evengii Landis. "An Algorithm for the Organization of Information." 1962.

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

LOCATION OF VIDEONOTES IN THE TEXT