Second Edition

Java™
Foundations

Introduction to
Program Design & Data Structures
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, jGRASPTM, 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—including 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

**Preface**

**Chapter 1 Introduction**

1.1 The Java Programming Language 2  
A Java Program 3  
Comments 5  
Identifiers and Reserved Words 6  
White Space 9  

1.2 Program Development 11  
Programming Language Levels 11  
Editors, Compilers, and Interpreters 13  
Development Environments 15  
Syntax and Semantics 16  
Errors 17  

1.3 Problem Solving 18  

1.4 Software Development Activities 19  

1.5 Object-Oriented Programming 21  
Object-Oriented Software Principles 21  

**Chapter 2 Data and Expressions**

2.1 Character Strings 32  
The `print` and `println` Methods 32  
String Concatenation 34  
Escape Sequences 37  

2.2 Variables and Assignment 38  
Variables 38  
The Assignment Statement 40  
Constants 42  

2.3 Primitive Data Types 43  
Integers and Floating Points 43
Characters 45
Booleans 47

2.4 Expressions 47
Arithmetic Operators 48
Operator Precedence 48
Increment and Decrement Operators 51
Assignment Operators 53

2.5 Data Conversion 54
Conversion Techniques 56

2.6 Reading Input Data 57
The Scanner Class 57

Chapter 3 Using Classes and Objects 71

3.1 Creating Objects 72
Aliases 74

3.2 The String Class 76

3.3 Packages 79
The import Declaration 80

3.4 The Random Class 82

3.5 The Math Class 85

3.6 Formatting Output 88
The NumberFormat Class 88
The DecimalFormat Class 90
The printf Method 92

3.7 Enumerated Types 92

3.8 Wrapper Classes 95
Autoboxing 97

Chapter 4 Conditionals and Loops 105

4.1 Boolean Expressions 106
Equality and Relational Operators 107
Logical Operators 108
4.2 The if Statement  
  The if-else Statement  
  Using Block Statements  
  The Conditional Operator  
  Nested if Statements  

4.3 Comparing Data  
  Comparing Floats  
  Comparing Characters  
  Comparing Objects  

4.4 The switch Statement  

4.5 The while Statement  
  Infinite Loops  
  Nested Loops  
  Other Loop Controls  

4.6 Iterators  
  Reading Text Files  

4.7 The do Statement  

4.8 The for Statement  
  Iterators and for Loops  
  Comparing Loops  

Chapter 5 Writing Classes  

5.1 Classes and Objects Revisited  
  Identifying Classes and Objects  
  Assigning Responsibilities  

5.2 Anatomy of a Class  
  Instance Data  
  UML Class Diagrams  

5.3 Encapsulation  
  Visibility Modifiers  
  Accessors and Mutators  

5.4 Anatomy of a Method  
  The return Statement  
  Parameters  

## CONTENTS

### Sliders
- 255

### Combo Boxes
- 260

### Timers
- 265

### 6.3 Layout Managers
- Flow Layout
  - 271
- Border Layout
  - 274
- Grid Layout
  - 278
- Box Layout
  - 280
- Containment Hierarchies
  - 284

### 6.4 Mouse and Key Events
- Mouse Events
  - 284
- Key Events
  - 292
- Extending Adapter Classes
  - 297

### 6.5 Dialog Boxes
- File Choosers
  - 301
- Color Choosers
  - 304

### 6.6 Some Important Details
- Borders
  - 305
- Tool Tips and Mnemonics
  - 309

### 6.7 GUI Design
- 315

## Chapter 7 Arrays

### 7.1 Array Elements
- 326

### 7.2 Declaring and Using Arrays
- Bounds Checking
  - 329
- Alternate Array Syntax
  - 335
-Initializer Lists
  - 335
- Arrays as Parameters
  - 336

### 7.3 Arrays of Objects
- 337

### 7.4 Command-Line Arguments
- 346

### 7.5 Variable-Length Parameter Lists
- 348

### 7.6 Two-Dimensional Arrays
- Multidimensional Arrays
  - 355
### Chapter 8 Inheritance

8.1 Creating Subclasses 364
   - The `protected` Modifier 368
   - The `super` Reference 370
   - Multiple Inheritance 371

8.2 Overriding Methods 374
   - Shadowing Variables 375

8.3 Class Hierarchies 376
   - The `object` Class 378
   - Abstract Classes 380

8.4 Visibility 382

8.5 Designing for Inheritance 385
   - Restricting Inheritance 385

### Chapter 9 Polymorphism

9.1 Late Binding 394

9.2 Polymorphism via Inheritance 394

9.3 Interfaces 407
   - Interface Hierarchies 412
   - The `Comparable` Interface 413
   - The `Iterator` Interface 413

9.4 Polymorphism via Interfaces 414
   - Event Processing 416

### Chapter 10 Exceptions

10.1 Exception Handling 424

10.2 Uncaught Exceptions 425

10.3 The `try-catch` Statement 425
   - The `finally` Clause 429

10.4 Exception Propagation 430
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.5</td>
<td>The Exception Class Hierarchy</td>
<td>433</td>
</tr>
<tr>
<td></td>
<td>Checked and Unchecked Exceptions</td>
<td>436</td>
</tr>
<tr>
<td>10.6</td>
<td>I/O Exceptions</td>
<td>437</td>
</tr>
<tr>
<td></td>
<td><strong>Chapter 11 Recursion</strong></td>
<td>447</td>
</tr>
<tr>
<td>11.1</td>
<td>Recursive Thinking</td>
<td>448</td>
</tr>
<tr>
<td></td>
<td>Infinite Recursion</td>
<td>449</td>
</tr>
<tr>
<td></td>
<td>Recursion in Math</td>
<td>449</td>
</tr>
<tr>
<td>11.2</td>
<td>Recursive Programming</td>
<td>450</td>
</tr>
<tr>
<td></td>
<td>Recursion vs. Iteration</td>
<td>453</td>
</tr>
<tr>
<td></td>
<td>Direct vs. Indirect Recursion</td>
<td>453</td>
</tr>
<tr>
<td>11.3</td>
<td>Using Recursion</td>
<td>454</td>
</tr>
<tr>
<td></td>
<td>Traversing a Maze</td>
<td>454</td>
</tr>
<tr>
<td></td>
<td>The Towers of Hanoi</td>
<td>459</td>
</tr>
<tr>
<td></td>
<td><strong>Chapter 12 Analysis of Algorithms</strong></td>
<td>469</td>
</tr>
<tr>
<td>12.1</td>
<td>Algorithm Efficiency</td>
<td>470</td>
</tr>
<tr>
<td>12.2</td>
<td>Growth Functions and Big-Oh Notation</td>
<td>471</td>
</tr>
<tr>
<td>12.3</td>
<td>Comparing Growth Functions</td>
<td>473</td>
</tr>
<tr>
<td></td>
<td>Method Calls</td>
<td>475</td>
</tr>
<tr>
<td></td>
<td>Analyzing Recursive Algorithms</td>
<td>477</td>
</tr>
<tr>
<td></td>
<td><strong>Chapter 13 Searching and Sorting</strong></td>
<td>481</td>
</tr>
<tr>
<td>13.1</td>
<td>Searching</td>
<td>482</td>
</tr>
<tr>
<td></td>
<td>Linear Search</td>
<td>485</td>
</tr>
<tr>
<td></td>
<td>Binary Search</td>
<td>487</td>
</tr>
<tr>
<td>13.2</td>
<td>Sorting</td>
<td>489</td>
</tr>
<tr>
<td></td>
<td>Selection Sort</td>
<td>490</td>
</tr>
<tr>
<td></td>
<td>Insertion Sort</td>
<td>497</td>
</tr>
<tr>
<td></td>
<td>Bubble Sort</td>
<td>498</td>
</tr>
<tr>
<td></td>
<td>Quick Sort</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Merge Sort</td>
<td>501</td>
</tr>
</tbody>
</table>
13.3 Analyzing Searching and Sorting Algorithms 503
   Comparing Search Algorithms 504
   Comparing Sort Algorithms 504

Chapter 14 Stacks 513
14.1 Introduction to Collections 514
   Abstract Data Types 514
   The Java Collections API 516
14.2 A Stack Collection 517
14.3 Inheritance, Polymorphism, and Generics 519
   Generics 520
14.4 A Stack ADT 521
14.5 Using Stacks: Evaluating Postfix Expressions 523
14.6 Exceptions 530
14.7 Implementing a Stack: with Arrays 532
   Managing Capacity 532
14.8 The ArrayStack Class 533
   The push Operation 537
   The pop Operation 537
   The peek Operation 538
   Other Operations 538
14.9 References as Links 538
14.10 Managing Linked Lists 541
   Accessing Elements 541
   Inserting Nodes 542
   Deleting Nodes 543
   Sentinel Nodes 544
14.11 Elements Without Links 544
   Doubly Linked Lists 545
14.12 Implementing a Stack: With Links 546
   The LinkedStack Class 546
   The push Operation 550
   The pop Operation 552
   Other Operations 555

14.13 Implementing Stacks:
   The Java.Util.Stack Class 553
   Unique Operations 553
   Inheritance and Implementation 554

14.14 Packages 554
   Organizing Packages 555
   Using CLASSPATH 555

Chapter 15 Queues 567

15.1 A Queue ADT 568
15.2 Using Queues: Code Keys 570
15.3 Using Queues: Ticket Counter Simulation 574
15.4 Implementing Queues: With Links 577
   The enqueue Operation 582
   The dequeue Operation 583
   Other Operations 583
15.5 Implementing Queues: With Arrays 584
   The enqueue Operation 587
   The dequeue Operation 590
   Other Operations 590

Chapter 16 Trees 595

16.1 Tree Terminology 596
   Tree Classifications 597
16.2 Tree Traversals 598
   Preorder Traversal 600
   Inorder Traversal 601
   Postorder Traversal 601
   Level-Order Traversal 602
16.3 Strategies for Implementing Trees 602
    Computed Links in an Array 603
    Stored Links in an Array 603
    Linked Nodes 604

16.4 A Binary Tree Implementation 605

16.5 Decision Trees 606

Chapter 17 Binary Search Trees 625

17.1 Binary Search Trees 626
    Adding an Element to a Binary Search Tree 627
    Removing an Element from a Binary Search Tree 629

17.2 Binary Search Tree Implementation 630

17.3 Balanced Binary Search Trees 640
    Right Rotation 641
    Left Rotation 642
    Right-Left Rotation 642
    Left-Right Rotation 643

Chapter 18 Heaps and Priority Queues 649

18.1 Heaps 650
    Adding an Element to a Heap 650
    Removing the Largest Element from a Heap 651

18.2 Heap Implementation 652

18.3 Heap Sort 660

18.4 Priority Queues 660

Chapter 19 Graphs 669

19.1 Undirected Graphs 670

19.2 Directed Graphs 671
19.3 Weighted Graphs 673

19.4 Common Graph Algorithms 674
   Traversals 674
   Testing for Connectivity 678
   Minimum Spanning Trees 680
   Determining the Shortest Path 682

19.5 Strategies for Implementing Graphs 683
   Adjacency Lists 684
   Adjacency Matrices 684

Chapter 20 Hashing 689

20.1 Hashing 690

20.2 Hashing Functions 692
   The Division Method 692
   The Folding Method 693
   The Mid-square Method 693
   The Radix Transformation Method 694
   The Digit Analysis Method 694
   The Length-Dependent Method 694
   Hashing Functions in the Java Language 695

20.3 Resolving Collisions 695
   Chaining 695
   Open Addressing 698

20.4 Deleting Elements from A Hash Table 701
   Deleting from a Chained Implementation 701
   Deleting from an Open Addressing Implementation 702

20.5 Hash Tables in the Java Collections API 703
   The HashTable Class 704
   The HashSet Class 704
   The HashMap Class 706
   The IdentityHashMap Class 707
   The WeakHashMap Class 709
   LinkedHashSet and LinkedHashMap 710
Chapter 21 Databases 717

21.1 Introduction to Databases 718

21.2 Establishing a Connection to a Database 720
   Obtaining A Database Driver 720

21.3 Creating and Altering Database Tables 722
   Create Table 723
   Alter Table 724
   Drop Column 725

21.4 Querying The Database 725
   Show Columns 726

21.5 Inserting, Viewing, and Updating (Modifying) Data 728
   Insert 729
   Select . . . from 729
   Update 734

21.6 Deleting Data and Database Tables 735
   Deleting Data 735
   Deleting Database Tables 736

Appendix A Glossary 741

Appendix B Number Systems 775
   Place Value 776
   Bases Higher Than 10 778
   Conversions 779
   Shortcut Conversions 781

Appendix C The Unicode Character Set 787

Appendix D Java Bitwise Operators 791
Second Edition

Java™ Foundations

Introduction to
Program Design & Data Structures