

LEWIS • LOFTUS

JAVA™

Software Solutions

FOUNDATIONS OF PROGRAM DESIGN

Tenth Edition



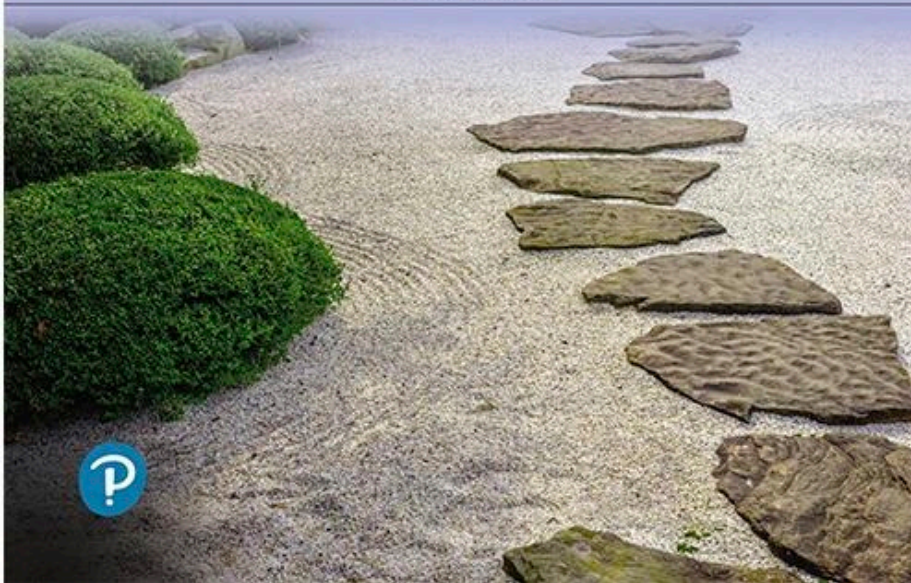
LEWIS • LOFTUS

JAVATM

Software Solutions

FOUNDATIONS OF PROGRAM DESIGN

Tenth Edition



Tingfen/123RF

Java™ Software Solutions

Foundations of Program Design

Tenth Edition

John Lewis

Virginia Tech

William Loftus

Accenture



Copyright © 2024 by Pearson Education Inc. or its affiliates. All Rights Reserved. This digital publication is protected by copyright, and permission should be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise except as authorized for use under the product subscription through which this digital application is accessed. For information regarding permissions, request forms and the appropriate contacts within the Pearson Education Global Rights & permissions department, please visit www.pearsoned.com/permissions/.

PEARSON, ALWAYS LEARNING, and Revel™ are exclusive trademarks in the U.S. and/or other countries and is owned by Pearson Education, Inc. or its affiliates.

Unless otherwise indicated herein, any third-party trademarks that may appear in this work are the property of their respective owners and any references to third-party trademarks, logos or other trade dress are for demonstrative or descriptive purposes only. Such references are not intended to imply sponsorship, endorsement, authorization, or promotion of Pearson's products by the owners of such marks, or any relationship between the owner and Pearson Education, Inc. or its affiliates, authors, licensees or distributors.

Credits and acknowledgments borrowed from other sources and reproduced, with permission, appear on the appropriate page within text. Many of the designations by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear, the publisher was aware of a trademark claim, the designations have been printed in initial caps or all caps.

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's Commitment to Diversity, Equity, and Inclusion

Pearson's Commitment to Diversity, Equity, and Inclusion

We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, sex, sexual orientation, socioeconomic status, ability, age, and religious or political beliefs.

Our ambition is to purposefully contribute to a world where:

- Everyone has an equitable and lifelong opportunity to succeed through learning.
- Our educational content accurately reflects the histories and lived experiences of the learners we serve.
- Our educational products and services are inclusive and represent the rich diversity of learners.
- Our educational content prompts deeper discussions with students and motivates them to expand their own learning (and worldview).

Accessibility

We are also committed to providing products that are fully accessible to all learners. As per Pearson's guidelines for accessible educational Web media, we test and retest the capabilities of our products against the highest standards for every release, following the WCAG guidelines in developing new products for copyright year 2022 and beyond.

You can learn more about Pearson's commitment to accessibility at <https://www.pearson.com/us/accessibility.html>

Contact Us

While we work hard to present unbiased, fully accessible content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

Please contact us with concerns about any potential bias at <https://www.pearson.com/report-bias.html>

For accessibility-related issues, such as using assistive technology with Pearson products, alternative text requests, or accessibility documentation, email the Pearson Disability Support team at disability.support@pearson.com.



This book is dedicated to our families.
Sharon, Justin, Kayla, Nathan, and Samantha Lewis
and
Veena, Isaac, and Dévi Loftus

Contents in a Glance

Preface

Chapter 1: Introduction

Chapter 2: Data and Expressions

Chapter 3: Using Classes and Objects

Chapter 4: Writing Classes

Chapter 5: Conditionals and Loops

Chapter 6: More Conditionals and Loops

Chapter 7: Object-Oriented Design

Chapter 8: Arrays

Chapter 9: Inheritance

Chapter 10: Polymorphism

Chapter 11: Exceptions

Chapter 12: Recursion

Chapter 13: Collections

Appendix A: Number Systems

Appendix B: The Unicode Character Set

Appendix C: Java Operators

Appendix D: Java Modifiers

Appendix E: Java Coding Guidelines

Appendix F: JavaFX Layout Panes

Appendix G: JavaFX Scene Builder

Appendix H: Regular Expressions

Appendix I: Javadoc Documentation Generator

Appendix J: Java Syntax

Appendix K: Answers to Self-Review Questions

Student Supplemental Materials

Glossary

Location of VideoNotes

VideoNotes are narrated step-by-step video tutorials that show how to solve problems completely, from design through coding.

Chapter 1: Introduction

VideoNote 1-1: Overview of Program Elements

VideoNote 1-2: Examples of Error Types

VideoNote 1-3: Developing a Solution to PP 1.2 - Part 1

VideoNote 1-4: Developing a Solution to PP 1.2 - Part 2

Chapter 2: Data and Expressions

VideoNote 2-1: Example using Strings and Escape Sequences - Part 1

VideoNote 2-2: Example using Strings and Escape Sequences - Part 2

VideoNote 2-3: Review of Primitive Data and Expressions - Part 1

VideoNote 2-4: Review of Primitive Data and Expressions - Part 2

VideoNote 2-5: Example using the `Scanner` class

Chapter 3: Using Classes and Objects

VideoNote 3-1: Creating Objects

VideoNote 3-2: Using the Strings

VideoNote 3-3: Example using the `Random` and `Math` Classes

Chapter 4: Writing Classes

VideoNote 4-1: Dissecting the `Die` Class - Part 1

VideoNote 4-2: Dissecting the `Die` Class - Part 2

VideoNote 4-3: Discussion of the `Account` Class

VideoNote 4-4: Developing a Solution to PP 4.2

Chapter 5: Conditionals and Loops

VideoNote 5-1: Examples using Conditionals

VideoNote 5-2: Examples using `while` Loops

VideoNote 5-3: Developing a Solution for PP 5.4 - Part 1

VideoNote 5-4: Developing a Solution for PP 5.4 - Part 2

Chapter 6: More Conditionals and Loops

VideoNote 6-1: Examples using `for` Loops

Chapter 7: Object-Oriented Design

VideoNote 7-1: Exploring the `static` Modifier

VideoNote 7-2: Example of Method Overloading

VideoNote 7-3: Developing a Solution for PP 7.1

Chapter 8: Arrays

VideoNote 8-1: Overview of Arrays

VideoNote 8-2: Discussion of the LetterCount Example

VideoNote 8-3: Developing a Solution for PP 8.5

Chapter 9: Inheritance

VideoNote 9-1: Overview of Inheritance - Part 1

VideoNote 9-2: Overview of Inheritance - Part 2

VideoNote 9-3: Example using a Class Hierarchy

Chapter 10: Polymorphism

VideoNote 10-1: Exploring the Firm Program

VideoNote 10-2: Sorting Comparable Objects

VideoNote 10-3: Developing a Solution for PP 10.1

Chapter 11: Exceptions

VideoNote 11-1: Proper Exception Handling

VideoNote 11-2: Developing a Solution for PP 11.1

Chapter 12: Recursion

VideoNote 12-1: Tracing the MazeSearch Program - Part 1

VideoNote 12-2: Tracing the MazeSearch Program - Part 2

VideoNote 12-3: Exploring the Towers of Hanoi

VideoNote 12-4: Developing a Solution for PP 12.1

Chapter 13: Collections

VideoNote 13-1: Example using a Linked List

VideoNote 13-2: Implementing a Queue

VideoNote 13-3: Developing a Solution for PP 13.3

Contents

Welcome

Cover.....	1
Title Page.....	2
Copyright Page.....	3
Pearson's Commitment to Diversity, Equity, and Inclusion.....	4
Dedication.....	5

Contents in a Glance

Contents in a Glance.....	6
Location of VideoNotes.....	7

Preface

Preface.....	18
New to This Edition.....	18
Cornerstones of the Text.....	18
Chapter Breakdown.....	19
Supplements.....	20
Instructor Resources.....	20
Features of the Text.....	21
Acknowledgments.....	22
Ordering Options.....	25

1: Introduction

1: Learning Objectives.....	27
1.1: Computer Processing	
1.1: Computer Processing.....	27
Software Categories.....	29
Digital Computers.....	31
Binary Numbers.....	33
1.1: Self-Review Questions.....	36
1.2: Hardware Components	
1.2: Hardware Components.....	37
Computer Architecture.....	37
Input/Output Devices.....	38
Main Memory and Secondary Memory.....	39
The Central Processing Unit.....	42
1.2: Self-Review Questions.....	43

1.3: Networks

1.3: Networks.....	44
Network Connections.....	45
Local-Area Networks and Wide-Area Networks..	46
The Internet.....	47
The World Wide Web.....	48
Uniform Resource Locators.....	49
1.3: Self-Review Questions.....	50

1.4: The Java Programming Language

1.4: The Java Programming Language.....	50
A Java Program.....	52
Comments.....	54
Identifiers and Reserved Words.....	55
White Space.....	58
1.4: Self-Review Questions.....	61

1.5: Program Development

1.5: Program Development.....	63
Programming Language Levels.....	63
Editors, Compilers, and Interpreters.....	65
Development Environments.....	67
Syntax and Semantics.....	68
Errors.....	69
1.5: Self-Review Questions.....	70

1.6: Object-Oriented Programming

1.6: Object-Oriented Programming.....	71
Problem Solving.....	71
Object-Oriented Software Principles.....	72
1.6: Self-Review Questions.....	75

Chapter 1: Summary of Key Concepts

Chapter 1: Summary of Key Concepts.....	75
---	----

Chapter 1: Exercises

Chapter 1: Exercises.....	77
---------------------------	----

Chapter 1: Programming Projects from the Book

Chapter 1: Programming Projects from the Book.....	79
--	----

2: Data and Expressions

2: Learning Objectives.....	82
2.1: Character Strings	
2.1: Character Strings.....	82

The print and println Methods	83
String Concatenation	84
Escape Sequences.....	89
2.1: Self-Review Questions.....	91
2.2: Variables and Assignment	
2.2: Variables and Assignment	92
Variables.....	92
The Assignment Statement.....	94
Constants	97
2.2: Self-Review Questions.....	97
2.3: Primitive Data Types	
2.3: Primitive Data Types.....	98
Integers and Floating Points.....	98
Characters	100
Booleans	102
2.3: Self-Review Questions.....	102
2.4: Expressions	
2.4: Expressions.....	103
Arithmetic Operators	103
Operator Precedence	104
Increment and Decrement Operators.....	109
Assignment Operators	110
2.4: Self-Review Questions.....	111
2.5: Data Conversion	
2.5: Data Conversion.....	114
Conversion Techniques	115
2.5: Self-Review Questions.....	117
2.6: Interactive Programs	
2.6: Interactive Programs	118
The Scanner Class	119
2.6: Self-Review Questions.....	124
Chapter 2: Summary of Key Concepts	
Chapter 2: Summary of Key Concepts	124
Chapter 2: Exercises	
Chapter 2: Exercises.....	125
Chapter 2: Programming Projects from the Book	
Chapter 2: Programming Projects from the Book.....	128
Chapter 2: Software Failure: NASA Mars Climate Orbiter and Polar Lander	
Chapter 2: Software Failure: NASA Mars Climate Orbiter and Polar Lander	130

3: Using Classes and Objects

3: Learning Objectives.....	132
3.1: Creating Objects	
3.1: Creating Objects	132
Aliases	135
3.1: Self-Review Questions.....	136
3.2: The String Class	
3.2: The String Class	137
3.2: Self-Review Questions.....	142
3.3: Packages	
3.3: Packages	143
The import Declaration	145
3.3: Self-Review Questions.....	146
3.4: The Random Class	
3.4: The Random Class	147
3.4: Self-Review Questions.....	150
3.5: The Math Class	
3.5: The Math Class.....	151
3.5: Self-Review Questions.....	154
3.6: Formatting Output	
3.6: Formatting Output.....	155
The NumberFormat Class.....	155
The DecimalFormat Class	158
The printf Method.....	161
3.6: Self-Review Questions.....	162
3.7: Enumerated Types	
3.7: Enumerated Types.....	162
3.7: Self-Review Questions.....	165
3.8: Wrapper Classes	
3.8: Wrapper Classes	166
3.8: Self-Review Questions.....	169
3.9: Introduction to JavaFX	
3.9: Introduction to JavaFX	170
3.9: Self-Review Questions.....	174
3.10: Basic Shapes	
3.10: Basic Shapes.....	174
3.10: Self-Review Questions.....	181
3.11: Representing Colors	
3.11: Representing Colors	182
3.11: Self-Review Questions.....	183
Chapter 3: Summary of Key Concepts	
Chapter 3: Summary of Key Concepts	183

Chapter 3: Exercises	4.9: Text Fields	227
Chapter 3: Exercises.....	4.9: Self-Review Questions.....	232
Chapter 3: Programming Projects from the Book	Chapter 4: Summary of Key Concepts	
Chapter 3: Programming Projects from the Book.....	Chapter 4: Summary of Key Concepts	232
4: Writing Classes	Chapter 4: Exercises	
4: Learning Objectives.....	Chapter 4: Exercises.....	233
4.1: Classes and Objects Revisited	Chapter 4: Programming Projects from the Book	
4.1: Classes and Objects Revisited	Chapter 4: Programming Projects from the Book.....	234
4.1: Self-Review Questions.....	Chapter 4: Software Failure: Denver Airport Baggage Handling System	
4.2: Anatomy of a Class	Chapter 4: Software Failure: Denver Airport Baggage Handling System.....	237
4.2: Anatomy of a Class.....	5: Conditionals and Loops	
Instance Data	5: Learning Objectives.....	239
UML Class Diagrams	5.1: Boolean Expressions	
4.2: Self-Review Questions.....	5.1: Boolean Expressions	239
4.3: Encapsulation	Equality and Relational Operators	241
4.3: Encapsulation	Logical Operators.....	242
Visibility Modifiers	5.1: Self-Review Questions.....	245
Accessors and Mutators	5.2: The If Statement	
4.3: Self-Review Questions.....	5.2: The If Statement	246
4.4: Anatomy of a Method	The if-else Statement	249
4.4: Anatomy of a Method	Using Block Statements.....	255
The return Statement.....	Nested if Statements.....	258
Parameters	5.2: Self-Review Questions.....	261
Local Data	5.3: Comparing Data	
Bank Account Example	5.3: Comparing Data.....	262
4.4: Self-Review Questions.....	Comparing Floats	262
4.5: Constructors Revisited	Comparing Character Data.....	263
4.5: Constructors Revisited	Comparing Objects	264
4.5: Self-Review Questions.....	5.3: Self-Review Questions.....	265
4.6: Arcs	5.4: The While Statement	
4.6: Arcs.....	5.4: The While Statement.....	266
4.6: Self-Review Questions.....	Infinite Loops	272
4.7: Images	Nested Loops.....	273
4.7: Images	The break and continue Statements.....	277
Viewports.....	5.4: Self-Review Questions.....	278
4.7: Self-Review Questions.....	5.5: Iterators	
4.8: Graphical User Interfaces	5.5: Iterators	279
4.8: Graphical User Interfaces.....	Reading Text Files.....	280
Alternate Ways to Specify Event Handlers.....	5.5: Self-Review Questions.....	283
4.8: Self-Review Questions.....		
4.9: Text Fields		

5.6: The ArrayList Class		6.5: Using Loops and Conditionals with Graphics	333
5.6: The ArrayList Class	283	6.5: Self-Review Questions.....	338
5.6: Self-Review Questions.....	288		
5.7: Determining Event Sources		6.6: Graphic Transformations	
5.7: Determining Event Sources	288	6.6 Graphic Transformations	339
5.7: Self-Review Questions.....	292	Translation.....	339
5.8: Managing Fonts		Scaling	340
5.8: Managing Fonts.....	292	Rotation.....	341
5.8: Self-Review Questions.....	295	Shearing	341
5.9: Check Boxes		Applying Transformations on Groups.....	342
5.9: Check Boxes.....	295	6.6: Self-Review Questions.....	346
5.9: Self-Review Questions.....	300		
5.10: Radio Buttons		Chapter 6: Summary of Key Concepts	
5.10: Radio Buttons	300	Chapter 6: Summary of Key Concepts	347
5.10: Self-Review Questions.....	305		
Chapter 5: Summary of Key Concepts		Chapter 6: Exercises	
Chapter 5: Summary of Key Concepts	305	Chapter 6: Exercises.....	347
Chapter 5: Exercises		Chapter 6: Programming Projects from the Book	
Chapter 5: Exercises.....	306	Chapter 6: Programming Projects from the Book.....	350
Chapter 5: Programming Projects from the Book			
Chapter 5: Programming Projects from the Book.....	309		
Chapter 5: Software Failure: Therac-25			
Chapter 5: Software Failure: Therac-25.....	312		
6: More Conditionals and Loops		7: Object-Oriented Design	
6: Learning Objectives.....	314	7: Learning Objectives.....	354
6.1: The Switch Statement		7.1: Software Development Activities	
6.1: The Switch Statement	314	7.1: Software Development Activities	354
6.1: Self-Review Questions.....	319	7.1: Self-Review Questions.....	356
6.2: The Conditional Operator		7.2: Identifying Classes and Objects	
6.2: The Conditional Operator.....	319	7.2: Identifying Classes and Objects	356
6.2: Self-Review Questions.....	321	Assigning Responsibilities.....	357
6.3: The Do Statement		7.2: Self-Review Questions.....	357
6.3: The Do Statement.....	321		
6.3: Self-Review Questions.....	324	7.3: Static Class Members	
6.4: The For Statement		7.3: Static Class Members.....	358
6.4 The for Statement	325	Static Variables	358
The for-each Loop	331	Static Methods.....	359
Comparing Loops	332	7.3: Self-Review Questions.....	363
6.4: Self-Review Questions.....	332		
6.5: Using Loops and Conditionals with Graphics		7.4: Class Relationships	
		7.4: Class Relationships	363
		Dependency	363
		Dependencies Among Objects of the Same Class	364
		Aggregation.....	371
		The this Reference.....	376
		7.4: Self-Review Questions.....	378
		7.5: Interfaces	
		7.5: Interfaces.....	378

The Comparable Interface	384	8.1: Array Elements	425
The Iterator Interface	385	8.1: Self-Review Questions	427
7.5: Self-Review Questions.....	385	8.2: Declaring and Using Arrays	
7.6: Enumerated Types Revisited		8.2: Declaring and Using Arrays	428
7.6: Enumerated Types Revisited.....	386	Bounds Checking	431
7.6: Self-Review Question	388	Alternate Array Syntax	437
7.7: Method Design		Initializer Lists.....	437
7.7: Method Design	389	Arrays as Parameters.....	439
Method Decomposition	389	8.2: Self-Review Questions.....	440
Method Parameters Revisited	395	8.3: Arrays of Objects	
7.7: Self-Review Questions.....	400	8.3: Arrays of Objects.....	441
7.8: Method Overloading		Array of Comic Books	446
7.8: Method Overloading	401	8.3: Self-Review Questions.....	453
7.8: Self-Review Questions.....	403	8.4: Command-Line Arguments	
7.9: Testing		8.4: Command-Line Arguments.....	453
7.9: Testing	404	8.4: Self-Review Questions.....	455
Reviews	404	8.5: Variable Length Parameter Lists	
Defect Testing.....	405	8.5: Variable Length Parameter Lists	455
7.9: Self-Review Questions.....	406	8.5: Self-Review Questions.....	461
7.10: GUI Design		8.6: Two-Dimensional Arrays	
7.10: GUI Design	407	8.6: Two-Dimensional Arrays.....	461
7.10: Self-Review Questions.....	408	Multidimensional Arrays.....	466
7.11: Mouse Events		8.6: Self-Review Questions.....	467
7.11: Mouse Events	408	8.7: Polygons and Polylines	
7.11: Self-Review Questions.....	414	8.7: Polygons and Polylines	467
7.12: Key Events		8.7: Self-Review Questions.....	470
7.12: Key Events	414	8.8: An Array of Color Objects	
7.12: Self-Review Questions.....	418	8.8: An Array of Color Objects	470
Chapter 7: Summary of Key Concepts		8.8: Self-Review Questions.....	474
Chapter 7: Summary of Key Concepts	418	8.9: Choice Boxes	
Chapter 7: Exercises		8.9: Choice Boxes.....	474
Chapter 7: Exercises.....	419	8.9: Self-Review Questions.....	479
Chapter 7: Programming Projects from the Book		Chapter 8: Summary of Key Concepts	
Chapter 7: Programming Projects from the		Chapter 8: Summary of Key Concepts	479
Book.....	420	Chapter 8: Exercises	
Chapter 7: Software Failure: 2003 Northeast Blackout		Chapter 8: Exercises.....	479
Chapter 7: Software Failure: 2003 Northeast		Chapter 8: Programming Projects from the Book	
Blackout	423	Chapter 8: Programming Projects from the	
		Book.....	481
8: Arrays		Chapter 8: Software Failure: LA Air Traffic Control	
8: Learning Objectives	425	Chapter 8: Software Failure: LA Air Traffic	
8.1: Array Elements		Control	484

9: Inheritance

9: Learning Objectives.....	486
9.1: Creating Subclasses	
9.1: Creating Subclasses	486
The protected Modifier	492
The super Reference	493
Multiple Inheritance	497
9.1: Self-Review Questions.....	498
9.2: Overriding Methods	
9.2: Overriding Methods	499
Shadowing Variables	502
9.2: Self-Review Questions.....	502
9.3: Class Hierarchies	
9.3: Class Hierarchies	502
The Object Class	504
Abstract Classes	506
Inheritance Hierarchies.....	507
9.3: Self-Review Questions.....	508
9.4: Visibility	
9.4: Visibility	508
9.4: Self-Review Questions.....	512
9.5: Designing for Inheritance	
9.5: Designing for Inheritance	512
Restricting Inheritance	513
9.5: Self-Review Questions.....	513
9.6: Inheritance in JavaFX	
9.6: Inheritance in JavaFX.....	514
9.6: Self-Review Questions.....	515
9.7: Color and Date Pickers	
9.7: Color and Date Pickers.....	515
9.7: Self-Review Questions.....	519
9.8: Dialog Boxes	
9.8: Dialog Boxes	520
File Choosers	523
9.8: Self-Review Questions.....	527
Chapter 9: Summary of Key Concepts	
Chapter 9: Summary of Key Concepts	527
Chapter 9: Exercises	
Chapter 9: Exercises.....	528

Chapter 9: Programming Projects from the Book

Chapter 9: Programming Projects from the Book.....	529
--	-----

Chapter 9: Software Failure: Ariane 5 Flight 501

Chapter 9: Software Failure: Ariane 5 Flight 501.....	531
---	-----

10: Polymorphism

10: Learning Objectives.....	533
10.1: Late Binding	
10.1: Late Binding	533
10.1: Self-Review Questions.....	534
10.2: Polymorphism via Inheritance	
10.2: Polymorphism via Inheritance	535
10.2: Self-Review Questions.....	550
10.3: Polymorphism via Interfaces	
10.3: Polymorphism via Interfaces.....	551
10.3: Self-Review Questions.....	553
10.4: Sorting	
10.4: Sorting	554
Selection Sort	555
Insertion Sort	562
Comparing Sorts	563
10.4: Self-Review Questions.....	564
10.5: Searching	
10.5: Searching	564
Linear Search.....	564
Binary Search.....	570
Comparing Searches	571
10.5: Self-Review Questions.....	571
10.6: Designing for Polymorphism	
10.6: Designing for Polymorphism	572
10.6: Self-Review Questions.....	573
10.7: Properties	
10.7: Properties	573
Change Listeners.....	576
10.7: Self-Review Questions.....	580
10.8: Sliders	
10.8: Sliders	580
10.8: Self-Review Questions	584
10.9: Spinners	
10.9: Spinners.....	584
10.9: Self-Review Questions.....	587

Chapter 10: Summary of Key Concepts		Chapter 11: Exercises	
Chapter 10: Summary of Key Concepts	588	Chapter 11: Exercises.....	627
Chapter 10: Exercises		Chapter 11: Programming Projects from the Book	
Chapter 10: Exercises.....	588	Chapter 11: Programming Projects from the Book.....	628
Chapter 10: Programming Projects from the Book			
Chapter 10: Programming Projects from the Book	589		
11: Exceptions		12: Recursion	
11: Learning Objectives.....	590	12: Learning Objectives.....	630
11.1: Exception Handling		12.1: Recursive Thinking	
11.1: Exception Handling	590	12.1: Recursive Thinking	630
11.1: Self-Review Questions.....	591	Infinite Recursion.....	631
11.2: Uncaught Exceptions		Recursion in Math.....	632
11.2: Uncaught Exceptions.....	591	12.1: Self-Review Questions.....	633
11.2: Self-Review Questions.....	593	12.2: Recursive Programming	
11.3: The try-catch Statement		12.2: Recursive Programming.....	633
11.3: The try-catch Statement.....	593	Recursion vs. Iteration.....	635
The finally Clause	597	Direct vs. Indirect Recursion	636
11.3: Self-Review Questions.....	598	12.2: Self-Review Questions.....	637
11.4: Exception Propagation		12.3: Using Recursion	
11.4: Exception Propagation	599	12.3: Using Recursion	637
11.4: Self-Review Questions.....	604	Traversing a Maze	638
11.5: The Exception Class Hierarchy		The Towers of Hanoi	644
11.5: The Exception Class Hierarchy	604	12.3: Self-Review Questions.....	650
Checked and Unchecked Exceptions.....	608	12.4: Tiled Images	
11.5: Self-Review Questions.....	608	12.4: Tiled Images	650
11.6: I/O Exceptions		12.4: Self-Review Questions.....	654
11.6: I/O Exceptions	608	12.5: Fractals	
11.6: Self-Review Questions.....	613	12.5: Fractals	654
11.7: Tool Tips and Disabling Controls		12.5: Self-Review Questions.....	663
11.7: Tool Tips and Disabling Controls	613	Chapter 12: Summary of Key Concepts	
11.7: Self-Review Questions.....	619	Chapter 12: Summary of Key Concepts	663
11.8: Scroll Panes		Chapter 12: Exercises	
11.8: Scroll Panes	619	Chapter 12: Exercises.....	663
11.8: Self-Review Questions.....	621	Chapter 12: Programming Projects from the Book	
11.9: Split Panes and List Views		Chapter 12: Programming Projects from the Book.....	664
11.9: Split Panes and List Views	622		
11.9: Self-Review Questions.....	626	13: Collections	
Chapter 11: Summary of Key Concepts		13: Learning Objectives.....	667
Chapter 11: Summary of Key Concepts	627	13.1: Collections and Data Structures	
		13.1: Collections and Data Structures	667
		Separating Interface from Implementation	668
		13.1: Self-Review Questions.....	668

13.2: Dynamic Representations	
13.2: Dynamic Representations	668
Dynamic Structures	669
A Dynamically Linked List	670
Other Dynamic List Representations	676
13.2: Self-Review Questions.....	678
13.3: Linear Collections	
13.3: Linear Collections	678
Queues	679
Stacks	680
13.3: Self-Review Questions.....	682
13.4: Non-Linear Data Structures	
13.4: Non-Linear Data Structures.....	683
Trees	683
Graphs	685
13.4: Self-Review Questions	686
13.5: The Java Collections API	
13.5: The Java Collections API.....	687
Generics	687
13.5: Self-Review Questions	688
13.6: Maps	
13.6: Maps	688
13.6: Self-Review Questions.....	690
13.7: Simplifying Declarations Using var	
13.7: Simplifying Declarations Using var.....	690
13.7: Self-Review Questions.....	692
13.8: Lambdas and Collections	
13.8: Lambdas and Collections.....	692
13.8: Self-Review Questions.....	694
Chapter 13: Summary of Key Concepts	
Chapter 13: Summary of Key Concepts	694
Chapter 13: Exercises	
Chapter 13: Exercises.....	695
Chapter 13: Programming Projects from the Book	
Chapter 13: Programming Projects from the Book	697
Appendix A: Number Systems	
Appendix A: Number Systems.....	699
Place Value	699
Bases Higher Than 10.....	700
Conversions.....	703
Shortcut Conversions.....	704
Appendix B: The Unicode Character Set	
Appendix B: The Unicode Character Set	706
Appendix C: Java Operators	
Appendix C: Java Operators	710
Java Bitwise Operators.....	713
Appendix D: Java Modifiers	
Appendix D: Java Modifiers.....	717
Java Visibility Modifiers	717
A Visibility Example	718
Other Java Modifiers	718
Appendix E: Java Coding Guidelines	
Appendix E: Java Coding Guidelines	721
Design Guidelines	721
Style Guidelines.....	722
Documentation Guidelines	723
Appendix F: JavaFX Layout Panes	
Appendix F: JavaFX Layout Panes	725
Flow Pane	725
Tile Pane	727
Stack Pane.....	727
HBox and VBox	729
Anchor Pane.....	730
Border Pane	731
Grid Pane.....	733
Appendix G: JavaFX Scene Builder	
Appendix G: JavaFX Scene Builder	736
Hello Moon	736
Handling Events in JavaFX Scene Builder.....	741
Appendix H: Regular Expressions	
Appendix H: Regular Expressions	746
Appendix I: Javadoc Documentation Generator	
Appendix I: Javadoc Documentation Generator	748
Doc Comments	748

Tags	748
Files Generated	751

Appendix J: Java Syntax

Appendix J: Java Syntax	753
-------------------------------	-----

Appendix K: Answers to Self-Review Questions

Chapter 1: Introduction	764
Chapter 2: Data and Expressions.....	768
Chapter 3: Using Classes and Objects.....	773
Chapter 4: Writing Classes.....	780
Chapter 5: Conditionals and Loops	784
Chapter 6: More Conditionals and Loops	792
Chapter 7: Object-Oriented Design.....	795
Chapter 8: Arrays	800
Chapter 9: Inheritance.....	806
Chapter 10: Polymorphism	810
Chapter 11: Exceptions	814
Chapter 12: Recursion.....	817
Chapter 13: Collections.....	820

Student Supplemental Materials

Student Supplemental Materials	823
--------------------------------------	-----

Preface

Welcome to the Tenth 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 has been tailored further to improve the coverage of topics key to introductory computing.

New to This Edition

The biggest change in the 10th edition of Java Software Solutions is the migration to digital format and inclusion of interactive coding activities and scored programming assessments (available in Revel Courseware and Pearson+ ebook) that provide immediate and personalized feedback to students on what they got wrong. The immediate feedback coaches students to the correct answer which increases persistence. This change reflects the need to make textbooks, especially computer science books, interactive and engaging. The tested and refined presentation of core material is now integrated with dynamic features and assessments.

- 21 Coding Animations to help students understand what is going on in the program and in the computer.
- 42 NEW Videonotes. Engaging video tutorials that take students from design to code completion.
- LiveCodeExamples interspersed throughout the chapters that provide coding practice.
- Over 388 Checkpoint self-check questions for students to test understanding of what they just read.
- Over 238 scored quiz assessment questions.
- Over 27 Summative end-of-chapter Programming Projects.

In addition to the digital/interactive transformation, several content changes were made for the tenth edition to stay on top of the ever-evolving Java language as it impacts introductory programmer, as well as updates to keep the examples fresh and improve the discussion of individual topics. These changes include:

- A new example demonstrating an Array of Objects in **Chapter 8**.
- A new section covering Maps and their implementation in the Java API.
- Coverage of the use of the `var` keyword to simplify complex variable declarations.
- A new section on lambda expressions, especially as they relate to the management of collections, in **Chapter 13**.

We're excited about the opportunities this new edition of Java Software Solutions provides for both students and instructors. Questions and comments are always welcome.

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. The **Software Failure** vignettes reiterate these lessons by demonstrating the perils of not following these sound practices.
- *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. The **VideoNotes** provide additional examples in a live presentation format.
- *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. The book fully embraces the JavaFX API, the preferred and fully-supported approach to Java graphics 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.

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. Then the **if** statement and **while** loop are explored in detail. Once loops are established, the concept of an iterator is introduced and the **Scanner** class is revisited for additional input parsing and the reading of text files. Finally, the **ArrayList** class is introduced, which provides the option for managing a large number of objects.

Chapter 6 (More Conditionals and Loops) examines the rest of Java's conditional (**switch**) and loop (**do**, **for**) statements. All related statements for conditionals and loops are discussed, including the enhanced version of the **for** loop. The for-each loop is also used to process iterators and **ArrayList** objects.

Chapter 7 (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 8 (Arrays) contains extensive coverage of arrays and array processing. The nature of an array as a low-level programming structure is contrasted to the higher-level object management approach. Additional topics include command-line arguments, variable length parameter lists, and multidimensional arrays.

Chapter 9 (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 10 (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 11 (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 12 (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 13 (Collections) 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, map collections, the use of the **var** keyword, and lambda expressions are also introduced. This chapter serves as an introduction to a CS2 course.

Supplements

Student Online Resources

These student resources can be accessed at the book's Companion Website, "https://media.pearsoncmg.com/ph/esm/ecs_lewis_java10e/cw/"

- Source Code for all the programs in the text
- Links to Java development environments
- VideoNotes: short step-by-step videos demonstrating how to solve problems from design through coding. VideoNotes allow for self-paced instruction with easy navigation including the ability to select, play, rewind, fast-forward, and stop within each VideoNote exercise. Margin icons in your textbook let you know when a VideoNote video is available for a particular concept or homework problem.

Instructor Resources

The following supplements are available to qualified instructors only. Visit the Pearson Education Instructor Resource Center (www.pearsonhighered.com/irc) for information on how to access them:

- Presentation Slides—in PowerPoint.
- Solutions to end-of-chapter Exercises.
- Solutions to end-of-chapter Programming Projects.

- Lab Manual Solutions
- Solutions for the Revel

Features of the Text

Key Concepts

Throughout the text, the Key Concept boxes highlight fundamental ideas and important guidelines. These concepts are summarized at the end of each chapter.

Listings

All programming examples are presented in clearly labeled listings, followed by the program output, a sample run, or screen shot display as appropriate. The code is colored to visually distinguish comments and reserved words.

Syntax Diagrams

At appropriate points in the text, syntactic elements of the Java language are discussed in special highlighted sections with diagrams that clearly identify the valid forms for a statement or construct. Syntax diagrams for the entire Java language are presented in **Appendix J**.

Graphics Track

All processing that involves graphics and graphical user interfaces is discussed in one or two sections at the end of each chapter that we collectively refer to as the Graphics Track. This material can be skipped without loss of continuity, or focused on specifically as desired. The material in any Graphics Track section relates to the main topics of the chapter in which it is found. Graphics Track sections are indicated by a brown border on the edge of the page.

Summary of Key Concepts

The Key Concepts presented throughout a chapter are summarized at the end of the chapter.

Self-Review Questions and Answers

These short-answer questions review the fundamental ideas and terms established in the preceding section. They are designed to allow students to assess their own basic grasp of the material. The answers to these questions can be found at the end of the book in **Appendix K**.

Exercises

These intermediate problems require computations, the analysis or writing of code fragments, and a thorough grasp of the chapter content. While the exercises may deal with code, they generally do not require any online activity.

Programming Projects

These problems require the design and implementation of Java programs. They vary widely in level of difficulty.

VideoNotes

VideoNotes are narrated step-by-step video tutorials that show how to solve problems completely, from design through coding.

Software Failures

These between-chapter vignettes discuss real-world flaws in software design, encouraging students to adopt sound design practices from the beginning.

Acknowledgments

I am most grateful to the faculty and students from around the world who have provided their feedback on previous editions of this book. I am 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.

I am particularly thankful for the assistance, insight, and attention to detail of Robert Burton from Brigham Young University. For years, Robert has consistently provided valuable feedback that helps shape and evolve this textbook.

Bradley Richards, at the University of Applied Sciences in Northwestern Switzerland, provided helpful advice and resources during the transition to JavaFX. Brian Fraser of Simon Fraser University also has provided some excellent feedback that helped clarify some issues. Such interaction with computing educators is incredibly valuable.

I also want to thank Dan Joyce from Villanova University, who developed the original Self-Review questions, ensuring that each relevant topic had enough review material, as well as developing the answers to each.

I continue to be amazed at the talent and effort demonstrated by the team at Pearson. Tracy Johnson, our editor, has amazing insight and commitment. Her assistant, Erin Sullivan, is a source of consistent and helpful support. Marketing Manager Krista Clark makes sure that instructors understand the pedagogical advantages of the text. The cover was designed by the skilled talents of Marta Samsel. Scott Disanno, Rajinder Singh, Bob Engelhardt and Abhijeet Gope led the production effort. We thank all of these people for ensuring that this book meets the highest quality standards.

Special thanks go to the following people who provided valuable advice to us about this book via their participation in focus groups, interviews, and reviews. They, as well as many other instructors and friends, have provided valuable feedback. They include:

Elizabeth Adams	James Madison University
Hossein Assadipour	Rutgers University
David Atkins	University of Oregon
Lewis Barnett	University of Richmond
Thomas W. Bennet	Mississippi College
Gian Mario Besana	DePaul University
Hans-Peter Bischof	Rochester Institute of Technology
Don Braffitt	Radford University
Robert Burton	Brigham Young University
John Chandler	Oklahoma State University
Robert Cohen	University of Massachusetts, Boston
Dodi Coreson	Linn Benton Community College
James H. Cross II	Auburn University
Eman El-Sheikh	University of West Florida
Sherif Elfayoumy	University of North Florida
Christopher Eliot	University of Massachusetts, Amherst
Wanda M. Eanes	Macon State College
Stephanie Elzer	Millersville University
Matt Evett	Eastern Michigan University
Marj Feroe	Delaware County Community College, Pennsylvania
John Gauch	University of Kansas
Chris Haynes	Indiana University
James Heliotis	Rochester Institute of Technology

Laurie Hendren	McGill University
Mike Higgs	Austin College
Stephen Hughes	Roanoke College
Daniel Joyce	Villanova University
Saroja Kanchi	Kettering University
Gregory Kapfhammer	Allegheny College
Karen Kluge	Dartmouth College
Jason Levy	University of Hawaii
Peter MacKenzie	McGill University
Jerry Marsh	Oakland University
Blayne Mayfield	Oklahoma State University
Gheorghe Muresan	Rutgers University
Laurie Murphy Pacific	Lutheran University
Dave Musicant	Carleton College
Faye Navabi-Tadayon	Arizona State University
Lawrence Osborne	Lamar University
Barry Pollack	City College of San Francisco
B. Ravikumar	University of Rhode Island
David Riley	University of Wisconsin (La Crosse)
Bob Roos	Allegheny College
Carolyn Rosiene	University of Hartford
Jerry Ross Lane	Community College
Patricia Roth	Southeastern Polytechnic State University

Carolyn Schauble	Colorado State University
Arjit Sengupta	Georgia State University
Bennet Setzer	Kennesaw State University
Vijay Srinivasan	JavaSoft, Sun Microsystems, Inc.
Stuart Steiner	Eastern Washington University
Katherine St. John	Lehman College, CUNY
Alexander Stoytchev	Iowa State University
Ed Timmerman	University of Maryland, University College
Shengru Tu	University of New Orleans
Paul Tymann	Rochester Institute of Technology
John J. Wegis	JavaSoft, Sun Microsystems, Inc.
Ken Williams	North Carolina Agricultural and Technical University
Linda Wilson	Dartmouth College
David Wittenberg	Brandeis University
Wang-Chan Wong	California State University (Dominguez Hills)

Thanks also go to my friends and former colleagues at Villanova University who have provided so much wonderful feedback. They include Bob Beck, Cathy Helwig, Anany Levitin, Najib Nadi, Beth Taddei, and Barbara Zimmerman. Thanks also to Pete DePasquale, formerly of The College of New Jersey and now with SailThru, Inc.

Many other people have helped in various ways. They include Ken Arnold, Mike Czepiel, John Loftus, Sebastian Niezgoda, and Saverio Perugini. Our apologies to anyone we may have omitted.

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.

Ordering Options

Revel

For more information about the Revel for this title, please visit: <https://www.pearsonhighered.com/revel/educators/features/>

Enhanced Revel Features: <https://www.pearson.com/au/revel/educators/enhanced-features/index.html>

Revel Instructor Help: <https://help.pearsoncmg.com/glp/en-us/pearson/instructor/Content/index.htm>

Pearson+

For more information about Pearson+ for this title, please visit <https://www.pearson.com/en-us/pearsonplus.html> and <https://plc.pearson.com/en-US/our-products-and-services/pearson-plus>

Pearson+ Help Center: <https://www.pearson.com/en-us/pearsonplus/support.html>

Chapter 1

Introduction

Learning Objectives

1. Describe the relationship between hardware and software.
2. Define various types of software and how they are used.
3. Identify the core hardware components of a computer and explain their roles.
4. Explain how the hardware components interact to execute programs and manage data.
5. Describe how computers are connected into networks to share information.
6. Introduce the Java programming language.
7. Describe the steps involved in program compilation and execution.
8. Present an overview of object-oriented principles.

This book is about writing well-designed software. To understand software, we must first have a fundamental understanding of its role in a computer system. Hardware and software cooperate in a computer system to accomplish complex tasks. The purpose of various hardware components and the way those components are connected into networks are important prerequisites to the study of software development. This chapter first discusses basic computer processing and then begins our exploration of software development by introducing the Java programming language and the principles of object-oriented programming.

1.1: Computer Processing

All computer systems, whether it's a desktop, laptop, tablet, smartphone, gaming console, or a special-purpose device such as a car's navigation system, share certain characteristics. The details vary, but they all process data in similar ways. While the majority of this book deals with the development of software, we'll begin with an overview of computer processing to set the context. It's important to establish some fundamental terminology and see how key pieces of a computer system interact.

A computer system is made up of hardware and software. The **hardware** components of a computer system are the physical, tangible pieces that support the computing effort. They include chips, boxes, wires, keyboards, speakers, disks, memory cards, universal serial bus (USB) flash drives (also called jump drives), cables, plugs, printers, mice, monitors, routers, and so on. If you can physically touch it and it can be considered part of a computer system, then it is computer hardware.

KEY CONCEPT

A computer system consists of hardware and software that work in concert to help us solve problems.

The hardware components of a computer are essentially useless without instructions to tell them what to do. A **program** is a series of instructions that the hardware executes one after another. **Software** consists of programs and the data that programs use. Software is the intangible counterpart to the physical hardware components. Together they form a tool that we can use to help solve problems.

The key hardware components in a computer system are

- central processing unit (CPU)
- input/output (I/O) devices
- main memory
- secondary memory devices

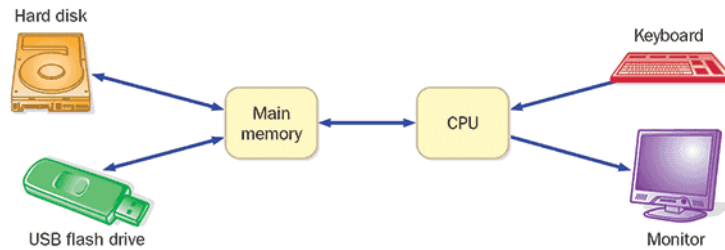
Each of these hardware components is described in detail in the next section. For now, let's simply examine their basic roles. The **central processing unit (CPU)** is a device that executes the individual commands of a program. **Input/output (I/O) devices**, such as the keyboard, mouse, trackpad, and **monitor**, allow a human being to interact with the computer.

Programs and data are held in storage devices called memory, which fall into two categories: main memory and secondary memory. **Main memory** is the storage device that holds the software while it is being processed by the **CPU**. **Secondary memory** devices store software in a relatively permanent manner. The most important secondary memory device of a typical computer system is the hard disk that resides inside the main computer box. A USB flash drive is also an important secondary memory device. A typical USB flash drive cannot store nearly as much information as a hard disk. USB flash drives have the advantage of **portability**; they can be removed temporarily or moved from computer to computer as needed.

Figure 1.1 shows how information moves among the basic hardware components of a computer. Suppose you have an executable program you wish to run. The program is stored on some secondary memory device, such as a hard disk. When you instruct the computer to execute your program, a copy of the program is brought in from secondary memory and stored in main memory. The CPU reads the individual program instructions from main memory. The CPU then executes the instructions one at a time until the program ends. The data that the instructions use, such as two numbers that will be added together, also are stored in main memory. They are either brought in from secondary memory or read from an input device such as the keyboard. During execution, the program may display information to an output device such as a monitor.

FIGURE 1.1

A simplified view of a computer system



KEY CONCEPT

The CPU reads the program instructions from main memory, executing them one at a time until the program ends.

The process of executing a program is fundamental to the operation of a computer. All computer systems basically work in the same way.

Software Categories

Software can be classified into many categories using various criteria. At this point, we will simply differentiate between system programs and application programs.

The **operating system** is the core software of a computer. It performs two important functions. First, it provides a **user interface** that allows the user to interact with the machine. Second, the operating system manages computer resources such as the CPU and main memory. It determines when programs are allowed to run, where they are loaded into memory, and how hardware devices communicate. It is the operating system's job to make the computer easy to use and to ensure that it runs efficiently.

KEY CONCEPT

The operating system provides a user interface and manages computer resources.

Several popular operating systems are in use today. The Windows operating system was developed for personal computers by Microsoft, which has captured the lion's share of the operating systems market. Various versions of the Unix operating system are also quite popular, especially in larger computer systems. A version of Unix called Linux was developed as an open-source project, which means that many people contributed to its development and its code is freely available. Because of that Linux has become a particular favorite among some users. Mac OS is an operating system used for computing systems developed by Apple Computers.

Operating systems are often specialized for mobile devices such as smartphones and tablets. The iOS operating system from Apple is used on the iPhone, iPad, and iPod. It is similar in functionality and appearance to the desktop Mac OS, but tailored for the smaller devices. The Android operating system developed by Google currently dominates the smartphone market.

An **application** (often shortened in conversation to “app”) is a generic term for just about any software other than the operating system. Word processors, missile control systems, database managers, Web browsers, and games all can be considered application programs. Each application program has its own user interface that allows the user to interact with that particular program.

The user interface for most modern operating systems and applications is a **graphical user interface** (GUI, pronounced “gooey”), which, as the name implies, makes use of graphical screen elements. Among many others, these elements include

- **windows**, which are used to separate the screen into distinct work areas
- **icons**, which are small images that represent computer resources, such as a **file**
- **menus, checkboxes, and radio buttons**, which provide the user with selectable options
- **sliders**, which allow the user to select from a range of values
- **buttons**, which can be “pushed” with a mouse click to indicate a user selection

A mouse or trackpad is the primary input device used with GUIs; thus, GUIs are sometimes called *point-and-click interfaces*. The screenshot in **Figure 1.2** shows an example of a GUI.

FIGURE 1.2

An example of a graphical user interface (GUI)

