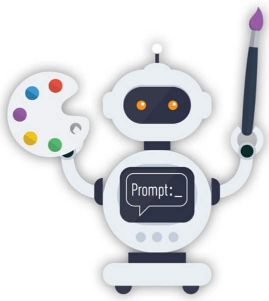

Contents

Preface xxvii

Before You Begin lxvii

1 Intro to Computers, the Internet, Java and Generative AI 1

1.1 Introduction 2
1.2 Hardware 5
  1.2.1 Computer Organization 5
  1.2.2 Moore's Law, Multi-Core Processors and Concurrent Programming 7
1.3 Data Hierarchies 10
1.4 Machine Languages, Assembly Languages and High-Level Languages 13
1.5 Operating Systems 15
1.6 Java 18
1.7 C and C++ 20
1.8 Other Popular Programming Languages 21
1.9 Introduction to Object Orientation 24
1.10 Java Application Programming Interface (API) and Open-Source Libraries 27
1.11 Typical Java Program Development Environment 28
1.12 Test-Driving a Java Program with the Java Development Kit (JDK) 31
  1.12.1 Compiling and Running the Program with the `javac` and `java` Commands 31
  1.12.2 Running the Program's Source-Code File Directly with the `java` Command 34
1.13 Internet, World Wide Web, the Cloud and IoT 34
  1.13.1 The Internet: A Network of Networks 35
  1.13.2 The World Wide Web: Making the Internet User-Friendly 35
  1.13.3 The Cloud 36
  1.13.4 The Internet of Things (IoT) 36
  1.13.5 Edge Computing 37
  1.13.6 Mashups 37
1.14 Metaverse 38
  1.14.1 Virtual Reality (VR) 38
  1.14.2 Augmented Reality (AR) 39
  1.14.3 Mixed Reality 39

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

1.14.4 Blockchain 40
1.14.5 Bitcoin and Cryptocurrency 40
1.14.6 Ethereum 41
1.14.7 Non-Fungible Tokens (NFTs) 41
1.14.8 Web3 41
1.15 Software Development Technologies 42
1.16 Data Analytics and Data Science 44
1.17 How Big Is Big Data? 45
1.17.1 Big-Data Analytics 47
1.17.2 Data Science and Big Data Are Making a Difference: Use Cases 48
1.18 AI—at the Intersection of Computer Science and Data Science 48
1.18.1 Artificial Intelligence (AI) 49
1.18.2 Artificial General Intelligence (AGI) 49
1.18.3 Artificial Intelligence Milestones 50
1.18.4 Machine Learning 51
1.18.5 Deep Learning 52
1.18.6 Reinforcement Learning 52
1.19 Generative AI 53
1.20 Wrap-Up 59

2 Intro to Java Programming 65

2.1 Introduction 66
2.2 Your First Program in Java: Printing a Line of Text 66
2.2.1 Compiling the Program 71
2.2.2 Executing the Program 72
2.3 Modifying Your First Program 72
2.4 Displaying Text with `printf` 74
2.5 Another Program: Adding Integers 76
2.6 Memory Concepts 80
2.7 Arithmetic 82
2.8 Decision Making: Equality and Relational Operators 85
2.9 Objects-Natural Case Study: Creating and Using Objects of the Java API's `String` Class 90
2.10 Wrap-Up 93

3 Algorithm Development and Control Statements: Part 1 99

3.1 Introduction 100
3.2 Algorithms 101
3.3 Pseudocode 101
3.4 Control Structures 102
3.4.1 Sequence Structure 103
3.4.2 Selection Statements 104
3.4.3 Iteration Statements 104
3.4.4 Summary of Control Statements 104

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

3.5 if Single-Selection Statement 105
3.6 if...else Double-Selection Statement 107
3.6.1 Nested if...else Statements 108
3.6.2 Blocks 110
3.6.3 Conditional Operator (?:) 112
3.7 while Iteration Statement 112
3.8 Formulating Algorithms: Counter-Controlled Iteration 115
3.8.1 Pseudocode Algorithm 115
3.8.2 Implementing the Algorithm 116
3.8.3 Integer Division and Truncation 118
3.8.4 Arithmetic Overflow 118
3.8.5 Input Validation 119
3.9 Formulating Algorithms: Sentinel-Controlled Iteration 120
3.9.1 Developing the Pseudocode Algorithm with Top-Down, Stepwise Refinement: The Top and First Refinement 120
3.9.2 Proceeding to the Second Refinement 121
3.9.3 Implementing Sentinel-Controlled Iteration 122
3.9.4 Explicitly and Implicitly Converting Between Primitive Types 124
3.9.5 Formatting Floating-Point Numbers 125
3.10 Formulating Algorithms: Nested Control Statements 127
3.10.1 Problem Statement 127
3.10.2 Top-Down, Stepwise Refinement: Pseudocode Representation of the Top 128
3.10.3 Top-Down, Stepwise Refinement: First Refinement 128
3.10.4 Top-Down, Stepwise Refinement: Second Refinement 128
3.10.5 Complete Second Refinement of Pseudocode 129
3.10.6 Implementing the Program 130
3.11 Compound Assignment Operators 132
3.12 Increment and Decrement Operators 133
3.13 Primitive Types 137
3.14 Objects-Natural Case Study: Super-Sized Integers 138
3.15 Wrap-Up 142

4 Control Statements: Part 2 149

4.1 Introduction 150
4.2 Essentials of Counter-Controlled Iteration 150
4.3 for Iteration Statement 152
4.4 Examples Using the for Statement 156
4.5 Summing the Even Integers from 2 to 20 with the for Statement 157
4.6 Calculating Compound Interest with the for Statement 158
4.7 do...while Iteration Statement 161
4.8 switch Multiple-Selection Statement 163
4.9 break and continue Statements 169
4.10 Logical Operators 171
4.10.1 Conditional AND (&&) Operator 172

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

4.10.2 Conditional OR (||) Operator 173
4.10.3 Short-Circuit Evaluation of Complex Conditions 173
4.10.4 Boolean Logical AND (&) and Inclusive OR (|) Operators 174
4.10.5 Boolean Logical Exclusive OR (^) Operator 174
4.10.6 Logical Complement (!) Operator 174
4.10.7 Example: Generating Logical-Operator Truth Tables 175
4.11 Structured-Programming Summary 178
4.12 Objects-Natural Case Study: Precise Monetary Calculations with Java API Class `BigDecimal` 182
4.13 Wrap-Up 187

5 Methods 193
5.1 Introduction 194
5.2 Declaring Methods 194
5.3 Notes on Declaring and Using Methods 199
5.4 Case Study: Die Rolling Simulation with Random-Number Generation 200
5.5 Case Study: A Game of Chance; Introducing `enums` 206
5.6 Scope of Declarations 211
5.7 Method Overloading 214
5.8 Class `Math`: `static` Methods and Variables 216
5.9 Java API Packages 219
5.10 Method-Call Stack and Activation Records 220
5.11 Argument Promotion and Casting 224
5.12 Objects-Natural Case Study: Java Date/Time API 226
5.12.1 Data/Time Manipulations 227
5.12.2 Running the Program for a Different Locale Than Your Own 232
5.13 Wrap-Up 233

6 Arrays and `ArrayLists` 243
6.1 Introduction 244
6.2 Primitive Types vs. Reference Types 244
6.3 Arrays 245
6.4 Declaring and Creating Arrays 246
6.5 Creating and Initializing an Array 248
6.6 Array Initializers 250
6.7 Calculating Array Element Values 251
6.8 Totaling Array Elements 252
6.9 Intro to Visualization: Using a Bar Chart to Display Array Data Graphically 253
6.10 Using Array Elements as Counters 255
6.11 Analyzing Survey Results 257
6.12 Exception Handling 259
6.12.1 The `try` Statement 260
6.12.2 The `catch` Block 260
6.13 Enhanced `for` Statement: Totaling Array Elements 261

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

6.14 Passing Arrays to Methods 263
6.15 Pass-By-Value vs. Pass-By-Reference 266
6.16 Multidimensional Arrays 267
6.17 Variable-Length Argument Lists 271
6.18 Command-Line Arguments 273
6.19 Class `Arrays` 276
6.20 Objects-Natural Case Study: Intro to Collections and Class `ArrayList` 279
6.21 Wrap-Up 284

7 Strings, NLP and Regex: Generative AI Foundations 299

7.1 Introduction 300
7.2 Fundamentals of Characters and Strings 301
7.3 Class `String` 302
7.3.1 Creating `String` Objects 302
7.3.2 `String` Methods `length`, `charAt` and `getChars` 304
7.3.3 Comparing `Strings` 306
7.3.4 Locating Characters and Substrings in `Strings` 311
7.3.5 Extracting Substrings from `Strings` 314
7.3.6 Concatenating `Strings` 315
7.3.7 Miscellaneous `String` Methods 316
7.3.8 `String` Method `valueOf` 318
7.4 Class `StringBuilder` 320
7.4.1 Creating `StringBuilder` Objects 320
7.4.2 Methods `length`, `capacity`, `setLength` and `ensureCapacity` 321
7.4.3 Methods `charAt`, `setCharAt`, `getChars` and `reverse` 323
7.4.4 `append` Methods 325
7.4.5 Insertion and Deletion Methods 328
7.5 Class `Character` 330
7.5.1 `Character` Testing and Case Conversion Methods 330
7.5.2 Character/Digit Conversions 332
7.5.3 Other `Character` Methods 334
7.6 Tokenizing `Strings` 335
7.7 Intro to Natural Language Processing (NLP)—at the Root of Generative AI 336
7.8 Objects-Natural Case Study: Intro to Regular Expressions in NLP 338
7.8.1 Matching Complete `Strings` to Patterns 339
7.8.2 Replacing Substrings 344
7.8.3 Searching for Matches with Classes `Pattern` and `Matcher` 345
7.8.4 Simple Data Wrangling Steps Used to Prepare Text for Training NLP and Generative AI Models 347
7.9 Objects-Natural Security Case Study: pMa5tfEKwk59dTvC04Ft1IFQz9mEXnkfYXZwxk4ujGE= 350
7.10 Wrap-Up 359

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

8 Real-World Modeling with Custom Classes 367

8.1 Introduction 368
8.2 Instance Variables, *set* Methods and *get* Methods 369
 8.2.1 `Account` Class with an Instance Variable, and *set* and *get* Methods 369
 8.2.2 `AccountTest` Class That Creates and Uses an `Account` Object 372
 8.2.3 Compiling and Executing a Program with Multiple Classes 374
 8.2.4 `Account` UML Class Diagram 375
 8.2.5 Notes on Class `AccountTest` 376
 8.2.6 Software Engineering with `private` Instance Variables and `public` *set* and *get* Methods 377
8.3 Default and Explicit Instance Variable Initialization 378
8.4 `Account` Class: Initializing Objects with Constructors 379
 8.4.1 Declaring an `Account` Constructor for Custom Object Initialization 379
 8.4.2 Class `AccountTest`: Initializing New `Account` Objects 380
8.5 `Account` Class with a Balance 382
 8.5.1 `Account` Class with a `BigDecimal` `balance` Instance Variable 382
 8.5.2 `AccountTest` Class 384
8.6 Case Study: Card Shuffling and Dealing Simulation 386
8.7 Case Study: `Time` Class 391
8.8 Controlling Access to Members 395
8.9 Referring to the Current Object's Members with the `this` Reference 396
8.10 Case Study: `Time` Class Overloaded Constructors 398
8.11 Default and No-Argument Constructors 405
8.12 Notes on *Set* and *Get* Methods 406
8.13 Composition 407
8.14 `enum` Types 410
8.15 Garbage Collection 413
8.16 `static` Class Members 413
8.17 `static` Import 417
8.18 `final` Instance Variables 418
8.19 Package Access 419
8.20 `record` Classes 421
 8.20.1 Pattern Matching for `switch` Expressions 424
 8.20.2 Decomposing `record`s in `switch` Expressions 426
8.21 Wrap-Up 427

9 Real-World Modeling with Inheritance, Polymorphism & Interfaces 433

9.1 Introduction 434
9.2 Superclasses and Subclasses 436
9.3 Relationship Between Superclasses and Subclasses 438
 9.3.1 Creating and Using a `SalariedEmployee` Class 439
 9.3.2 Creating a `SalariedEmployee`/`SalariedCommissionEmployee` Inheritance Hierarchy 443
9.4 Class `Object` 449

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

9.5 Intro to Polymorphism: Polymorphic Video Game 450
9.6 Demonstrating Polymorphic Behavior 451
9.7 abstract Classes and Methods 454
9.8 Case Study: Payroll System Using Polymorphism 456
9.8.1 abstract Superclass Employee 457
9.8.2 Concrete Subclass SalariedEmployee 458
9.8.3 Concrete Subclass CommissionEmployee 460
9.8.4 Polymorphic Processing, Operator instanceof and Downcasting 462
9.9 final Methods and Classes 466
9.10 Issues with Constructors Calling Instance Methods 467
9.11 Creating and Using Interfaces 468
9.11.1 Developing a Payable Hierarchy 470
9.11.2 Interface Payable 471
9.11.3 Class Invoice 471
9.11.4 Modifying Class Employee to Implement Interface Payable 473
9.11.5 Using Interface Payable to Process Invoices and Employees Polymorphically 475
9.11.6 Some Common Java API Interfaces 476
9.12 Other Interface Features 477
9.12.1 default Interface Methods 477
9.12.2 static Interface Methods 478
9.12.3 Functional Interfaces 478
9.12.4 private Interface Methods 479
9.13 Program to an Interface, Not an Implementation 479
9.13.1 CompensationModel Interface 481
9.13.2 An Employee Has a CompensationModel—Composition and Dependency Injection 481
9.13.3 CompensationModel Implementations 483
9.13.4 Testing the CompensationModel Hierarchy 485
9.13.5 Dependency Injection Design Benefits 486
9.13.6 interfaces vs. abstract Classes 487
9.14 sealed Classes and Interfaces 488
9.15 private Constructors 491
9.16 protected Members 492
9.17 Wrap-Up 493

10 Exception Handling: A Deeper Look 499

10.1 Introduction 500
10.2 Example: Divide by Zero without Exception Handling 502
10.3 Example: Handling ArithmeticExceptions and InputMismatchExceptions 504
10.4 Java Exception Hierarchy 509
10.5 Checked vs. Unchecked Exceptions 511
10.6 finally Block 513
10.7 Stack Unwinding and Obtaining Information from an Exception 517

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

10.8 Chained Exceptions 520
10.9 Declaring Custom Exceptions 523
10.10 Preconditions and Postconditions 524
10.11 Assertions 525
10.12 `try`-with-Resources Statement: Automatic Resource Deallocation 527
10.13 Unnamed Variables in `catch` Handlers 529
10.14 Wrap-Up 530

11 Files, I/O Streams, JSON Serialization & CSV Files 533

11.1 Introduction 534
11.2 Files and Streams 535
11.3 Using NIO Classes and Interfaces to Get File and Directory Information 537
11.4 Sequential Text Files 541
11.4.1 Creating a Sequential Text File 541
11.4.2 Reading Data from a Sequential Text File 544
11.4.3 Updating Sequential Files 545
11.5 Case Study: JavaScript Object Notation (JSON) Serialization 546
11.5.1 Serializing a `List<Account>` 548
11.5.2 Deserializing a `List<Account>` 550
11.6 Case Study: Processing a JSON Response from a Web Service 552
11.7 Case Study: Creating and Reading CSV Files 560
11.7.1 Creating a CSV File 561
11.7.2 Reading a CSV File 563
11.8 Case Study: Reading and Analyzing a CSV File Containing the *Titanic* Disaster Dataset 565
11.9 Objects-Natural Security Case Study: RSA Public-Key Cryptography 573
11.10 Wrap-Up 580

12 Generic Collections 585

12.1 Introduction 586
12.2 Collections Overview 587
12.3 Type-Wrapper Classes 588
12.4 Boxing and Unboxing 589
12.5 Lists 589
12.5.1 `ArrayList` and `Iterator` 590
12.5.2 `LinkedList` 593
12.5.3 Views into Collections and `Arrays` Method `asList` 596
12.6 `Collections` Methods 598
12.6.1 Method `sort` 599
12.6.2 Method `shuffle` 602
12.6.3 Methods `reverse`, `fill`, `copy`, `max` and `min` 605
12.6.4 Method `binarySearch` 607
12.6.5 Methods `addAll`, `frequency` and `disjoint` 608

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

12.7 Class `PriorityQueue` and Interface `Queue` 610
12.8 Computer Science Thinking: A Brief Intro to Hash Tables 611
12.9 Sets 613
12.10 Maps 616
12.11 Convenience Factory Methods for Creating Immutable Collections 619
12.12 Concurrent Collections 622
12.13 Wrap-Up 622

13 Generic Classes and Methods: A Deeper Look 645

13.1 Introduction 646
13.2 Motivation for Generic Methods 646
13.3 Generic Methods: Implementation and Compile-Time Translation 648
- 13.3.1 Generic Method `printArray` 648
- 13.3.2 Type Parameter Section of a Generic Method 649
- 13.3.3 Testing the Generic Method 650
- 13.3.4 Erasure at Compilation Time 651

13.4 Additional Compile-Time Translation Issues: Methods That Use a Type Parameter as the Return Type 651
13.5 Overloading Generic Methods 655
13.6 Generic Classes 655
- 13.6.1 Parameterized Types 655
- 13.6.2 Implementing a Generic `Stack` Class 656
- 13.6.3 Testing the Generic `Stack` Class 657
- 13.6.4 Refactoring the Test Program to Manipulate `Stack<E>` Objects with Generic Methods 661

13.7 Wildcards in Methods That Accept Type Parameters 663
- 13.7.1 Summing a `List<Number>` 663
- 13.7.2 Reimplementing Method `sum` with a Wildcard Type Argument in Its Parameter 665

13.8 Wrap-Up 667

14 Functional Programming with Lambdas & Streams 671

14.1 Introduction 672
14.2 Streams and Reduction 673
- 14.2.1 Summing the Integers from 1 through 10 with a `for` Loop 674
- 14.2.2 External Iteration with `for` Is Error Prone 674
- 14.2.3 Summing with a Stream and Reduction 674
- 14.2.4 Internal Iteration 676

14.3 Mapping and Lambdas 677
- 14.3.1 Lambda Expressions 678
- 14.3.2 Lambda Syntax 678
- 14.3.3 Intermediate and Terminal Operations 679

14.4 Filtering 680

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

14.5 How Elements Move Through Stream Pipelines 682
14.6 Method References 683
14.6.1 Creating an `IntStream` of Random Values 684
14.6.2 Performing a Task on Each Stream Element with `forEach` and a Method Reference 685
14.6.3 Mapping Integers to String Objects with `mapToObj` 685
14.6.4 Concatenating Strings with `collect` 686
14.7 `IntStream` Operations 686
14.7.1 Creating an `IntStream` and Displaying Its Values 687
14.7.2 Terminal Operations `count`, `min`, `max`, `sum` and `average` 687
14.7.3 Terminal Operation `reduce` 689
14.7.4 Sorting `IntStream` Values 691
14.8 Functional Interfaces 692
14.9 Lambdas: A Deeper Look 693
14.10 `Stream<Integer>` Manipulations 694
14.10.1 Creating a `Stream<Integer>`, Sorting It and Collecting the Results 695
14.10.2 Filtering a `Stream` and Storing the Results for Later Use 696
14.10.3 Filtering and Sorting a `Stream` and Collecting the Results 697
14.10.4 Sorting Previously Collected Results 697
14.11 `Stream<String>` Manipulations 698
14.11.1 Mapping `Strings` to Uppercase 698
14.11.2 Filtering `Strings` Then Sorting Them in Case-Insensitive Ascending Order 699
14.11.3 Filtering `Strings` Then Sorting Them in Case-Insensitive Descending Order 699
14.12 `Stream<Employee>` Manipulations 700
14.12.1 Creating and Displaying a `List<Employee>` 701
14.12.2 Filtering `Employees` with Salaries in a Specified Range 702
14.12.3 Sorting `Employees` By Multiple Fields 704
14.12.4 Mapping `Employees` to Unique-Last-Name `Strings` 706
14.12.5 Grouping `Employees` By Department 707
14.12.6 Counting the Number of `Employees` in Each Department 708
14.12.7 Summing and Averaging `Employee` Salaries 708
14.13 Creating a `Stream<String>` from a File 710
14.14 Streams of Random Values 713
14.15 Infinite Streams 714
14.16 Additional Notes on Interfaces 716
14.17 Wrap-Up 716

15 JavaFX Graphical User Interfaces: Part 1 725

15.1 Introduction 726
15.2 JavaFX Scene Builder 727
15.3 JavaFX Application Window Structure 728
15.4 `Welcome` Application: Displaying Text and an Image 729
15.4.1 Opening Scene Builder and Creating the File `Welcome.fxml` 729

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

15.4.2 Adding an Image to the Folder Containing Welcome.fxml 731
15.4.3 Creating a VBox Layout Container 731
15.4.4 Configuring the VBox Layout Container 731
15.4.5 Adding and Configuring a Label 732
15.4.6 Adding and Configuring an ImageView 732
15.4.7 Previewing the Welcome GUI 734
15.5 Tip Calculator Application: Intro to Event Handling 735
15.5.1 Test-Driving the Tip Calculator 735
15.5.2 Technologies Overview 737
15.5.3 Building the Application's GUI 740
15.5.4 TipCalculator Class 746
15.5.5 TipCalculatorController Class; Implementing an Event Handler with a Lambda 748
15.6 Features Covered in the Other JavaFX Chapters 752
15.7 Wrap-Up 752

16 JavaFX GUI: Part 2 757

16.1 Introduction 758
16.2 Laying Out Nodes in a Scene Graph 758
16.3 Painter Application: RadioButtons, Mouse Events and Shapes 760
16.3.1 Technologies Overview 761
16.3.2 Creating the Painter.fxml File 762
16.3.3 Building the GUI 763
16.3.4 Painter Subclass of Application 765
16.3.5 PainterController Class 766
16.4 Color Chooser Application: Property Bindings and Property Listeners 770
16.4.1 Technologies Overview 771
16.4.2 Building the GUI 772
16.4.3 ColorChooser Subclass of Application 773
16.4.4 ColorChooserController Class 774
16.5 Cover Viewer Application: Data-Driven GUIs with JavaFX Collections 776
16.5.1 Technologies Overview 777
16.5.2 Building the GUI 777
16.5.3 CoverViewer Subclass of Application 779
16.5.4 CoverViewerController Class 779
16.6 Cover Viewer Application: Customizing ListView Cells 781
16.6.1 Technologies Overview 782
16.6.2 Copying the CoverViewer Application 782
16.6.3 ImageTextCell Custom ListView Cell Layout 782
16.6.4 CoverViewerController Class 784
16.7 FileChooser and DirectoryChooser Dialogs 785
16.8 Other JavaFX Capabilities and JavaFX Accessibility 791
16.9 JavaFX Updates 793
16.10 JavaFX Resources and Libraries 794
16.11 Wrap-Up 796

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

17 JavaFX Graphics and Multimedia 801

17.1 Introduction 802
17.2 Controlling Fonts with Cascading Style Sheets (CSS) 803
17.2.1 CSS That Styles the GUI 804
17.2.2 FXML That Defines the GUI—Introduction to XML Markup 807
17.2.3 Referencing the CSS File from FXML 810
17.2.4 Specifying the `VBox`'s Style Class 810
17.2.5 Programmatically Loading CSS 810
17.3 Displaying Two-Dimensional Shapes 811
17.3.1 Defining Two-Dimensional Shapes with FXML 811
17.3.2 CSS That Styles the Two-Dimensional Shapes 813
17.4 `Polyline`s, `Polygon`s and `Path`s 817
17.4.1 GUI and CSS 817
17.4.2 `PolyShapesController` Class 818
17.5 Transforms 822
17.6 Playing Video with `Media`, `MediaPlayer` and `MediaView` 824
17.6.1 VideoPlayer GUI 825
17.6.2 `VideoPlayerController` Class 827
17.7 `Transition` Animations 831
17.7.1 `TransitionAnimations.fxml` 831
17.7.2 `TransitionAnimationsController` Class 833
17.8 `Timeline` Animations 837
17.9 Frame-by-Frame Animation with `AnimationTimer` 839
17.10 CSS Transitions 842
17.11 Drawing on a `Canvas` 847
17.12 Three-Dimensional Shapes 852
17.13 FXGL: A Brief Intro to Game Programming with JavaFX 856
17.14 Wrap-Up 857

18 Concurrency: Platform Threads to Virtual Threads 869

18.1 Introduction 870
18.2 `sort`/`parallelSort` Timings with the Date/Time API 872
18.3 Sequential vs. Parallel Streams 875
18.4 Creating and Executing Platform Threads with the `Executor` Framework 879
18.5 Project Loom Overview 883
18.6 Creating and Executing Virtual Threads with the `Executor` Framework 884
18.6.1 Launching Virtual Threads with a Thread-Per-Task Executor 885
18.6.2 Other Ways to Launch Virtual Threads 887
18.7 Profiling Platform vs. Virtual Threads 889
18.7.1 `Future` and `Callable` Interfaces 889
18.7.2 Threading Approaches We'll Use 890
18.7.3 Launching and Timing Large Numbers of Concurrent Tasks 890
18.7.4 Testing the Threading Approaches 892
18.7.5 Performance Analysis 893

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

18.8 Structured Concurrency and Scoped Values 895
18.9 Thread Synchronization Overview 903
18.10 Producer/Consumer Relationship with `ArrayBlockingQueue` 905
18.10.1 Synchronization and State Dependence 905
18.10.2 Class `ArrayBlockingQueue` 906
18.10.3 Overview of the Producer/Consumer Example 906
18.10.4 Interface `Buffer` 907
18.10.5 Class `BlockingBuffer` 907
18.10.6 Class `BlockingBufferTest` 908
18.10.7 Bounded Buffers 911
18.10.8 Other Concurrent Collections 912
18.11 Multithreading in JavaFX 913
18.11.1 Performing Computations in a Worker Thread: Fibonacci Numbers 914
18.11.2 Processing Intermediate Results: Sieve of Eratosthenes 919
18.12 Wrap-Up 925

19 Building API-Based Java Generative AI Applications 933

19.1 Introduction 934
19.2 OpenAI APIs 936
19.2.1 Some of OpenAI's Models 936
19.2.2 OpenAI API Capabilities 936
19.2.3 Get an OpenAI Developer Account 937
19.2.4 OpenAI Developer API Key 938
19.3 Setting Up a Java Environment 939
19.3.1 Simple-OpenAI Open Source Library: Connecting to OpenAI from Java 939
19.3.2 IntelliJ Maven-Based Java Project 939
19.4 Text Generation Via Chat Completions 940
19.4.1 Text Summarization 940
19.4.2 Sentiment Analysis 944
19.4.3 Accessible Image Descriptions 947
19.4.4 Language Detection and Translation 950
19.4.5 Code Generation 952
19.4.6 Named Entity Recognition (NER) and Structured Outputs 956
19.5 Speech Synthesis and Speech Recognition 959
19.5.1 English Speech-to-Text for Audio Transcription 959
19.5.2 Text-to-Speech 960
19.6 Image Generation 962
19.7 Video 965
19.7.1 Generating Video Closed Captions 965
19.7.2 Sora 968
19.7.3 Other Video GenAIs 970
19.8 Moderation 970

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

19.9 Class OpenAIUtilities 973
19.9.1 Builder Design Pattern 973
19.9.2 package and import Statements 974
19.9.3 Nested record Class Message 975
19.9.4 SimpleOpenAI Client Object 975
19.9.5 chat Method 975
19.9.6 Helper Method makeChatMessages 977
19.9.7 describeImage Method 978
19.9.8 translate Method 979
19.9.9 chatWithStructuredOutput Method 980
19.9.10 speechToText Method 982
19.9.11 textToSpeech Method 983
19.9.12 image Method 984
19.9.13 speechToVTT Method 985
19.9.14 checkPrompt Method 987
19.10 Wrap-Up 988

20 Accessing Databases with JDBC and SQLite 1001
20.1 Introduction 1002
20.2 Relational Databases 1003
20.3 Setting Up the SQLite RDBMS 1004
20.4 A books Database 1004
20.4.1 SELECT Queries 1008
20.4.2 WHERE Clause 1009
20.4.3 ORDER BY Clause 1010
20.4.4 Merging Data from Multiple Tables with INNER JOIN 1011
20.4.5 INSERT INTO Statement 1011
20.4.6 UPDATE Statement 1012
20.4.7 DELETE FROM Statement 1013
20.5 Connecting to and Querying a Database with JDBC 1014
20.5.1 Automatic Driver Discovery 1015
20.5.2 Connecting to the Database 1015
20.5.3 Creating a Statement for Executing Queries 1016
20.5.4 Executing a Query 1016
20.5.5 Processing a Query's ResultSet 1016
20.6 Querying the books Database 1018
20.6.1 DisplayQueryResults App's GUI 1018
20.6.2 DisplayQueryResultsController Class 1020
20.7 PreparedStatements 1024
20.7.1 AddressBook App That Uses PreparedStatements 1025
20.7.2 record Class Person 1026
20.7.3 Class PersonQueries 1026
20.7.4 AddressBook GUI 1030
20.7.5 Class AddressBookController 1032
20.8 Stored Procedures 1036

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

20.9 Transaction Processing 1036
20.10 Wrap-Up 1037

21 Java Platform Module System 1041

21.1 Introduction 1042
21.2 Module Declarations 1047
21.2.1 `requires` 1047
21.2.2 `requires transitive`—Implied Readability 1047
21.2.3 `exports` and `exports…to` 1048
21.2.4 `uses` 1048
21.2.5 `provides…with` 1048
21.2.6 `open`, `opens` and `opens…to` 1048
21.2.7 Restricted Keywords 1050
21.3 Modularized `Welcome` App 1050
21.3.1 `Welcome` App's Structure 1050
21.3.2 Class `Welcome` 1053
21.3.3 `module-info.java` 1054
21.3.4 Module-Dependency Graph 1054
21.3.5 Compiling a Module 1054
21.3.6 Running an App from a Module's Exploded Folders 1056
21.3.7 Packaging a Module into a Modular JAR File 1057
21.3.8 Running the `Welcome` App from a Modular JAR File 1058
21.3.9 Aside: Classpath vs. Module Path 1058
21.4 Creating and Using a Custom Module 1059
21.4.1 Exporting a Package for Use in Other Modules 1059
21.4.2 Using a Class from a Package in Another Module 1060
21.4.3 Compiling and Running the Example 1062
21.4.4 Packaging the App into Modular JAR Files 1063
21.4.5 Strong Encapsulation and Accessibility 1064
21.5 Module-Dependency Graphs: A Deeper Look 1064
21.5.1 `java.sql` 1065
21.5.2 `java.se` 1065
21.5.3 Error: Module Graph with a Cycle 1067
21.6 Migrating Code to Modules 1068
21.6.1 Unnamed Module 1069
21.6.2 Automatic Modules 1069
21.6.3 `jdeps` Tool for Java Dependency Analysis 1070
21.7 Resources in Modules 1072
21.7.1 Requiring Multiple Modules 1074
21.7.2 Opening a Module for Reflection 1074
21.7.3 Compiling the Module 1074
21.7.4 Running a Modularized App 1074
21.7.5 Module-Dependency Graph 1075
21.8 Creating Custom Runtimes with `jlink` 1076
21.8.1 Custom Runtime Containing Only `java.base` 1076
21.8.2 Creating a Custom Runtime for the `Welcome` App 1077

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

21.8.3 Executing the `Welcome` App Using a Custom Runtime 1078
21.8.4 Using the Module Resolver on a Custom Runtime 1078
21.9 Services and `ServiceLoader` 1079
21.9.1 Service-Provider Interface 1081
21.9.2 Loading and Consuming Service Providers 1082
21.9.3 `uses` Module Directive and Service Consumers 1085
21.9.4 Running the App with No Service Providers 1085
21.9.5 Implementing a Service Provider 1086
21.9.6 `provides…with` Module Directive and Declaring a Service Provider 1086
21.9.7 Running the App with One Service Provider 1087
21.9.8 Implementing a Second Service Provider 1088
21.9.9 Running the App with Two Service Providers 1089
21.10 Wrap-Up 1089

22 Computer Science Thinking: Recursion, Searching, Sorting, Big O 1093

22.1 Introduction 1094
22.2 Recursion Concepts 1095
22.3 Recursion Example: Factorials 1095
22.4 Recursion Example: Fibonacci Series 1098
22.5 Recursion vs. Iteration 1101
22.6 Towers of Hanoi 1103
22.7 Fractals 1105
22.7.1 Koch Curve Fractal 1106
22.7.2 (Optional) Case Study: Lo Feather Fractal 1107
22.7.3 (Optional) `Fractal` App GUI 1109
22.7.4 (Optional) `FractalController` Class 1111
22.8 Recursive Backtracking 1116
22.9 Linear Search 1117
22.10 Big O Notation 1119
22.10.1 $O(1)$ Algorithms 1119
22.10.2 $O(n)$ Algorithms 1119
22.10.3 $O(n^2)$ Algorithms 1119
22.10.4 Big O of the Linear Search 1120
22.11 Binary Search 1121
22.11.1 Binary Search Implementation 1121
22.11.2 Efficiency of the Binary Search 1124
22.12 Sorting Algorithms 1125
22.13 Selection Sort 1126
22.13.1 Selection Sort Implementation 1126
22.13.2 Efficiency of the Selection Sort 1129
22.14 Insertion Sort 1129
22.14.1 Insertion Sort Implementation 1130
22.14.2 Efficiency of the Insertion Sort 1132

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

22.15 Merge Sort 1132
22.15.1 Merge Sort Implementation 1133
22.15.2 Efficiency of the Merge Sort 1137
22.16 Big O Summary for This Chapter's Searching and Sorting Algorithms 1138
22.17 Wrap-Up 1139

A Introduction to JShell for Interactive Java 1149

A.1 Introduction 1150
A.2 Introduction to JShell 1152
A.2.1 Starting a JShell Session 1152
A.2.2 Executing Statements 1153
A.2.3 Declaring Variables Explicitly 1154
A.2.4 Listing and Executing Prior Snippets 1156
A.2.5 Evaluating Expressions and Declaring Variables Implicitly 1157
A.2.6 Using Implicitly Declared Variables 1158
A.2.7 Viewing a Variable's Value 1158
A.2.8 Resetting a JShell Session 1159
A.2.9 Writing Multiline Statements 1159
A.2.10 Editing Code Snippets 1160
A.2.11 Exiting JShell 1163
A.3 Command-Line Input in JShell 1163
A.4 Declaring and Using Classes 1164
A.4.1 Creating a Class in JShell 1164
A.4.2 Explicitly Declaring Reference-Type Variables 1165
A.4.3 Creating Objects 1165
A.4.4 Manipulating Objects 1166
A.4.5 Creating a Meaningful Variable Name for an Expression 1166
A.4.6 Saving and Opening Code-Snippet Files 1167
A.5 Discovery with JShell Auto-Completion 1168
A.5.1 Auto-Completing Identifiers 1168
A.5.2 Auto-Completing JShell Commands 1169
A.6 Exploring a Class's Members and Viewing Documentation 1170
A.6.1 Listing Class `Math`'s `static` Members 1170
A.6.2 Viewing a Method's Parameters 1171
A.6.3 Viewing a Method's Documentation 1171
A.6.4 Viewing a `public` Field's Documentation 1172
A.6.5 Viewing a Class's Documentation 1172
A.6.6 Viewing Method Overloads 1173
A.6.7 Exploring Members of a Specific Object 1173
A.7 Declaring Methods 1175
A.7.1 Forward Referencing an Undeclared Method—Declaring Method `displayCubes` 1175
A.7.2 Declaring a Previously Undeclared Method 1176
A.7.3 Testing `cube` and Replacing Its Declaration 1176
A.7.4 Testing Updated Method `cube` and Method `displayCubes` 1177

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.

A.8 Exceptions 1177
A.9 Importing Classes and Adding Packages to the `CLASSPATH` 1178
A.10 Using an External Editor 1181
A.11 Summary of JShell Commands 1181
A.11.1 Getting Help in JShell 1182
A.11.2 `/edit` Command: Additional Features 1183
A.11.3 `/reload` Command 1183
A.11.4 `/drop` Command 1184
A.11.5 Feedback Modes 1184
A.11.6 Other JShell Features Configurable with `/set` 1186
A.12 Keyboard Shortcuts for Snippet Editing 1187
A.13 How JShell Reinterprets Java for Interactive Use 1187
A.14 IDE JShell Support 1188
A.15 Wrap-Up 1188

B Formatted Output 1205

B.1 Introduction 1206
B.2 Output with `printf` 1206
B.3 Integer Formatting 1207
B.4 Floating-Point Number Formatting 1208
B.5 String and Character Formatting 1209
B.6 Other Conversion Characters 1210
B.7 Field Widths and Precisions 1211
B.8 Formatting Flags 1213
B.9 Argument Indices for Explicit Positioning in Format Strings 1216
B.10 Escape Sequences 1217
B.11 Formatting `String`s in Memory 1218
B.12 Wrap-Up 1219

C Number Systems 1221

C.1 Introduction 1222
C.2 Abbreviating Binary Numbers as Octal and Hexadecimal Numbers 1225
C.3 Converting Octal and Hexadecimal Numbers to Binary Numbers 1226
C.4 Converting from Binary, Octal or Hexadecimal to Decimal 1226
C.5 Converting from Decimal to Binary, Octal or Hexadecimal 1228
C.6 Negative Binary Numbers: Two's Complement Notation 1229

Index 1231

© Copyright 1992-2025 by Pearson Education, Inc. All Rights Reserved.