# Detailed table of contents

## Chapter 1: Background, 1

1.1 Introduction, 2
1.2 Computer organization, 3
   1.2.1 Computing units of measure, 5
   1.2.2 Buying a personal computer, 6
1.3 Internet computing, 10
   1.3.1 Networks, 11
1.4 Software and Java, 14
   1.4.1 Programming languages, 14
   1.4.2 Running a Java program, 15
1.5 Engineering software, 18
   1.5.1 Software engineering principles, 20
1.6 Object-oriented design, 23
1.7 Problem solving, 26
   1.7.1 Ask questions as needed, 29
   1.7.2 Find out as much as you can, 30
   1.7.3 Break complex problems into subproblems, 31
   1.7.4 Reuse and expect future reuse, 32
   1.7.5 Further reading, 33
1.8 Review, 34
1.9 Self-test, 35
1.10 Exercises, 35
1.11 Self-test answers, 38

## Chapter 2: Java Basics, 39

2.1 A first program, 40
   2.1.1 Commenting and whitespace, 41
   2.1.2 Classes, keywords, and identifiers, 43
   2.1.3 Methods, 44
   2.1.4 Selecting methods print() and println(), 46
   2.1.5 Escape sequences, 49
2.2 Simple computations, 52
   2.2.1 Constants, 54
   2.2.2 Variables, 56
   2.2.3 Operations, 57
2.3 Primitive types, 61
   2.3.1 int type, 61
   2.3.2 char type, 63
<table>
<thead>
<tr>
<th>Chapter 5: Decisions, 177</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1 Boolean algebra and truth tables, 178</td>
</tr>
<tr>
<td>5.1.1 Logical expressions, 179</td>
</tr>
<tr>
<td>5.2 Boolean type, 180</td>
</tr>
<tr>
<td>5.2.1 Boolean equality and ordering operators, 181</td>
</tr>
<tr>
<td>5.2.2 Operator precedence revisited, 184</td>
</tr>
<tr>
<td>5.3 If statement, 185</td>
</tr>
<tr>
<td>5.3.1 Avoiding gotchas, 189</td>
</tr>
<tr>
<td>5.4 If-else statement, 190</td>
</tr>
<tr>
<td>5.5 Nested constructs, 193</td>
</tr>
<tr>
<td>5.6 If-else-if construct, 195</td>
</tr>
<tr>
<td>5.7 Testing objects for equality, 200</td>
</tr>
<tr>
<td>5.8 Switch statement, 204</td>
</tr>
<tr>
<td>5.9 Case study — checksum validation, 213</td>
</tr>
<tr>
<td>5.10 Case study — triangles, 217</td>
</tr>
<tr>
<td>5.11 Review, 226</td>
</tr>
<tr>
<td>5.12 Self-test, 228</td>
</tr>
<tr>
<td>5.13 Exercises, 230</td>
</tr>
<tr>
<td>5.14 Programming project — medical assistant, 235</td>
</tr>
<tr>
<td>5.15 Self-test answers, 239</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 6: Iteration, 241</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 While statement, 242</td>
</tr>
<tr>
<td>6.2 For statement, 252</td>
</tr>
<tr>
<td>6.2.1 Index variable scope, 256</td>
</tr>
<tr>
<td>6.2.2 Computing the number of combinations, 256</td>
</tr>
<tr>
<td>6.3 Do-while statement, 258</td>
</tr>
<tr>
<td>6.4 Nested loops, 262</td>
</tr>
<tr>
<td>6.5 Simple file processing, 266</td>
</tr>
<tr>
<td>6.6 Case study — data set analysis, 273</td>
</tr>
<tr>
<td>6.7 Review, 281</td>
</tr>
<tr>
<td>6.8 Self-test, 282</td>
</tr>
<tr>
<td>6.9 Exercises, 283</td>
</tr>
<tr>
<td>6.10 Programming project — four hobo problem, 287</td>
</tr>
<tr>
<td>6.11 Self-test answers, 288</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Graphics Interlude I: GUI-based Programming, 291</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.12 GUI and event-driven programming, 292</td>
</tr>
<tr>
<td>6.13 Windchill calculator, 294</td>
</tr>
<tr>
<td>6.13.1 Class constants and instance variables, 299</td>
</tr>
<tr>
<td>6.13.2 GUI construction, 303</td>
</tr>
<tr>
<td>6.13.3 Event handling and actionPerformed(), 306</td>
</tr>
<tr>
<td>6.13.4 Method main(), 308</td>
</tr>
<tr>
<td>6.14 Review, 310</td>
</tr>
<tr>
<td>6.15 Self-test, 311</td>
</tr>
</tbody>
</table>
CHAPTER 7: PROGRAMMING WITH METHODS AND CLASSES, 317

7.1 Modifier static, 318
7.2 Parameter passing, 325
7.3 This, 334
7.4 Inherited methods and overriding, 338
7.5 Scope and reuse, 343
  7.5.1 Local scope rules, 343
  7.5.2 Name reuse, 344
7.6 Overloading, 347
7.7 Illustrations, 354
7.8 Review, 362
7.9 Self-test, 364
7.10 Exercises, 366
7.11 Programming project — automobile financing, 374
7.12 Self-test answers, 378

CHAPTER 8: ARRAYS AND COLLECTIONS, 381

8.1 Basic list requirements, 382
8.2 One-dimensional arrays, 382
  8.2.1 Definitions, 383
  8.2.2 Element access, 385
  8.2.3 Explicit initialization, 388
  8.2.4 Constant arrays, 389
  8.2.5 Members, 390
8.3 Simple array processing, 393
  8.3.1 Extraction, 393
  8.3.2 Searching for a key value, 396
  8.3.3 Searching for the minimum value, 398
8.4 Arrays and methods, 399
  8.4.1 Sequential and binary search, 400
  8.4.2 Zeroing, 403
  8.4.3 Display, 406
  8.4.4 Extraction and reversal, 406
8.5 Increasing representation capacity, 409
8.6 Sorting, 410
  8.6.1 Method selectionSort(), 411
  8.6.2 Quality of selectionSort(), 413
8.7 Command-line parameters, 414
8.8 Multidimensional arrays, 415
  8.8.1 Matrices, 419
8.9 Collections framework, 420
8.10 ArrayList, 421
8.11 Collections algorithms, 427
**CHAPTER 9: INHERITANCE AND POLYMORPHISM, 455**

9.1 Object-oriented design, 456  
  9.1.1 ThreeDimensionalPoint, 458  
  9.1.2 ColoredPoint, 465  
9.2 Polymorphism, 469  
9.3 Inheritance nuances, 471  
  9.3.1 Controlling access, 472  
  9.3.2 Data fields, 475  
  9.3.3 Typing, 478  
  9.3.4 Late binding, 480  
  9.3.5 Finality, 481  
9.4 Case study — extending BufferedReader, 483  
9.5 Abstract base classes, 487  
9.6 Interfaces, 491  
9.7 Preparing the aquarium, 495  
9.8 Review, 507  
9.9 Self-test, 509  
9.10 Exercises, 511  
9.11 Programming project — change maker, 515  
9.12 Self-test answers, 518

**GRAPHICS INTERLUDE II: GUI-BASED PROGRAMMING, 519**

9.13 Case study — personality typing, 520  
  9.13.1 Background, 520  
9.14 Programming project — Smiley guessing game, 535

**CHAPTER 10: EXCEPTIONS, 551**

10.1 Exception handling, 552  
10.2 Finally and the command type, 561  
10.3 Creating and throwing exceptions, 563  
10.4 Case study — value-by-value extraction, 569  
10.5 Review, 580  
10.6 Self-test, 581  
10.7 Exercises, 581  
10.8 Programming project — a second look, 582  
10.9 Self-test answers, 582

**CHAPTER 11: RECURSIVE PROBLEM SOLVING, 587**

11.1 Recursive methods, 588  
11.2 Fibonacci numbers and squares, 591  
11.3 Case study — recursive binary search, 595
CHAPTER 11: SORTING, 600
11.4 Method mergeSort(), 601
11.5 How fast can we sort?, 606
11.6 Recursion versus iteration, 607
11.7 Case study — string permutation, 609
11.8 Review, 615
11.9 Self-test, 615
11.10 Exercises, 616
11.11 Programming project — Sierpinski fractal, 619
11.12 Self-test answers, 622

CHAPTER 12: THREADS, 623
12.1 Scheduling, 624
  12.1.1 Running after a delay, 626
  12.1.2 Running repeatedly, 629
  12.1.3 Running at a chosen time, 633
12.2 Sleeping, 638
12.3 Animation, 640
12.4 Case study — swimming fish, 647
12.5 Review, 653
12.6 Self-test, 654
12.7 Exercises, 654
12.8 Programming project — better fish, 656
12.9 Self-test answers, 658

CHAPTER 13: TESTING AND DEBUGGING, 661
13.1 Testing, 662
  13.1.1 Testing — an example, 663
  13.1.2 Testing fundamentals, 672
  13.1.3 Reviews and inspections, 675
  13.1.4 Black-box and white-box testing, 677
  13.1.5 Integration and system testing, 682
13.2 Debugging, 683
  13.2.1 Scientific method, 683
  13.2.2 Debugging tips and techniques, 687
13.3 References, 691
13.4 Review, 691
13.5 Self-test, 692
13.6 Exercises, 692
13.7 Programming project — getListO, 693
13.8 Self-test answers, 695

APPENDIX A: TABLES AND OPERATORS, 697
A.1 Unicode character set, 698
A.2 Operators and precedence, 699
A.3 Reserved words, 699

APPENDIX B: NUMBER REPRESENTATION, 703
B.1 Binary number representation, 704
APPENDIX C: APPELTS, 707

C.1 A simple applet, 708
C.2 Applet methods, 710
  C.2.1 Method init(), 710
  C.2.2 Method start(), 711
  C.2.3 Method stop(), 711
  C.2.4 Method destroy(), 711
  C.2.5 Method paint(), 711
C.3 Applets and threads, 715
C.4 Applet security, 717
C.5 Summary, 718

APPENDIX D: STANDARD JAVA PACKAGES, 719

D.1 java.io, 720
D.2 java.lang, 729
D.3 java.math, 773
D.4 java.net, 779
D.5 java.text, 781
D.6 java.util, 789

APPENDIX E: STANDARD JAVA GRAPHICAL PACKAGES, 821

E.1 java.applet, 822
E.2 java.awt, 823
E.3 javax.swing, 858

INDEX, 895