

Contents

PART I BASIC OO PROGRAMMING	1		
Chapter 0. Computers and programming	3		
0.1 Computer organization	4		
0.2 Computer software	8		
Chapter 1. OO introduction to Java	15		
1.1. Types and expressions	16		
1.1.1 Type int	17		
1.1.2 Type double	18		
1.1.3 Casting between int and double	19		
1.1.4 Type boolean and arithmetic relations	20		
1.1.5 Type String	22		
1.1.6 Precedences of operators	23		
1.1.7 Function calls	23		
1.1.8 Self-review exercises	24		
1.2 Variables, declarations, assignments	26		
1.2.1 Self-review exercises	29		
1.3 Classes and objects	30		
1.3.1 The class as a file drawer of folders	30		
1.3.2 Packages and the import statement	31		
1.3.3 Objects of class JFrame	33		
1.3.4 Objects of class String	38		
1.3.5 Key concepts	39		
1.3.6 Self-review exercises	40		
1.4 Customizing a class to suit our needs	41		
1.4.1 A subclass definition	41		
1.4.2 Remembering data: adding variables	45		
1.4.3 Self-review exercises	46		
1.5 Static components	47		
1.6 Graphics in a JFrame	48		
1.7 Programming style and habits	50		
Exercises for Chapter 1	51		
Chapter 2. Methods	55		
2.1 Java methods are recipes	55		
2.2 The black-box view of a method	57		
2.2.1 The anatomy of a method header	57		
2.2.2 The procedure call	59		
2.2.3 The function call	62		
2.2.4 Self-review exercises for calls	62		
2.3 Method bodies	64		
2.3.1 The procedure body	64		
2.3.2 Executing a procedure call	66		
2.3.3 Conditional statements and blocks	68		
2.3.4 Self-review exercises for ifs	71		
2.3.5 The return statement	74		
2.3.6 The function body	75		
2.3.7 Local variables	76		
2.3.8 Processing a range of integers	78		
2.3.9 Self-review exercises for for-loops	81		
2.4 Static versus non-static methods	82		
2.5 Stepwise refinement	83		
2.5.1 Stepwise refinement: making coffee	84		
2.5.2 A summary of stepwise refinement	85		
2.5.3 Top-down development of a Java task	86		
2.6 Assertions in programs	90		
2.6.1 Relations about variables and values	90		
2.6.2 Assertions	92		
2.7 A model of execution	93		
2.7.1 Frames for method calls	93		
2.7.2 The steps in executing a method call	97		
2.8 Key concepts	97		
2.9 Self-review exercises	99		
Exercises for Chapter 2	100		
3.1 Class definitions	105		
Chapter 3. Classes	105		
3.1.1 The principal of information hiding	108		
3.1.2 The inside-out rule	109		
3.1.3 Declaration of constructors	110		
3.1.4 Function toString	112		
3.1.5 Self-review exercises	114		
3.2 Using classes	115		
3.2.1 The class as a type	115		
3.2.2 The new-expression	116		
3.2.3 Referencing components	118		
3.2.4 Equality testing and aliasing	118		
3.2.5 Making fields public	119		
3.2.6 Self-review exercises	120		
3.3 Static components	122		
3.3.1 Static variables	122		
3.3.2 Static methods	124		
3.4 Object-oriented design	124		

3.4.1	The basic idea of OO design	124	5.6	Random numbers	193
3.4.2	An example of OO design	126	5.6.1	Method Math.random	194
3.5	The model of execution	134	5.6.2	Class Random	194
3.6	Key concepts	137	5.6.3	Exercises with random numbers	195
Exercises for Chapter 3		138	5.7	Class JLiveRead for keyboard input	196
Chapter 4. Subclasses		141	5.8	GUI JLiveWindow	199
4.1	The subclass definition	142	5.9	Reading the keyboard and a file	201
4.1.1	Inheriting and overriding	143	5.9.1	Reading the keyboard	202
4.1.2	The class invariant	146	5.9.2	Reading a file	205
4.1.3	Constructors in a subclass	147	5.10	Writing and appending to a file	207
4.2	Casting about, or about casting	148	5.10.1	Exercises with files	208
4.2.1	Apparent and real classes	148	5.11	Universal resource locators	209
4.2.2	Explicit widening and narrowing	151	5.11.1	URLs	209
4.2.3	Operator instanceof	152	5.11.2	Class URL	212
4.3	The class hierarchy	153	5.11.3	Reading the file given by a URL	212
4.3.1	Class Object	154	Chapter 6. Reference on primitive types		215
4.3.2	Boolean function equals	154	6.1	Type int	216
4.4	Access modifiers	155	6.2	Types byte, short, and long	218
4.5	Object-oriented design	156	6.2.1	Types byte and short	218
4.5.1	The is-a relation	156	6.2.2	Type long	219
4.5.2	Example of object-oriented design	158	6.3	Casting among integral types	220
4.6	The final model of execution	162	6.4	Floating-point types double and float	221
4.7	Abstract classes	163	6.4.1	Type double	221
4.8	Key concepts	165	6.4.2	Type float	224
4.9	Self-review exercises	166	6.5	Type char	224
Exercises for Chapter 4		167	6.6	Casting among primitive types	225
Chapter 5. Some useful classes		171	6.7	Type boolean	226
5.1	The wrapper classes	172	PART II OTHER JAVA CONSTRUCTS		231
5.1.1	Wrapper class Integer	172	Chapter 7. Loops		233
5.1.2	Other wrapper classes	174	7.1	The while-loop	233
5.2	Classes String and StringBuffer	175	7.1.1	Syntax and semantics of while-loops	233
5.2.1	String literals	175	7.1.2	Tracing execution of a loop	234
5.2.2	The basics of String manipulation	176	7.1.3	Self-review exercises	235
5.2.3	Changing a name format	179	7.2	Understanding and developing loops	236
5.2.4	Extracting an integer from a string	180	7.2.1	Four loopy questions	237
5.2.5	Class StringBuffer	181	7.2.2	Developing a second loop	240
5.2.6	Exercises on strings	183	7.2.3	A slightly different problem	241
5.3	Class Vector	184	7.2.4	Self-review exercises	242
5.3.1	Creating and adding to a Vector	185	7.3	Examples of while-loops	244
5.3.2	Changing and retrieving elements	186	7.3.1	The roach explosion	244
5.3.3	Other methods in class Vector	188	7.3.2	Exponentiation	245
5.3.4	Exercises on class Vector	188	7.3.3	The spiral	248
5.4	Class Date	189	7.4	Loop patterns	248
5.5	Formatting numbers	190	7.4.1	Schema to process natural numbers	248
5.5.1	Class DecimalFormat	191			
5.5.2	Formatting in locales	192			

7.4.2	A loop to count the w's	249	8.6	Parallel vs. class-type arrays	295
7.4.3	Testing primality	251	8.7	Key concepts	296
7.4.4	A schema for reading	252	8.8	Self review exercises	297
7.4.5	Self-review exercises	253		Exercises for Chapter 8	297
7.5	The for-loop	253	Chapter 9. Multi-dimensional arrays	301	
7.5.1	The for-loop as an abbreviation	253	9.1	Rectangular arrays	301
7.5.2	Syntax and semantics of for-loops	254	9.2	Programs that use rectangular arrays	304
7.5.3	Developing a for-loop	255	9.2.1	Printing a two-dimensional array	304
7.5.4	A for-loop schema	256	9.2.2	A two-dimensional array schema	305
7.5.5	Self-review exercises	257	9.2.3	An interesting table	306
7.6	Making progress and stopping	257	9.2.4	Row-major search	306
7.6.1	Why use condition $i \neq n$?	257	9.2.5	Saddleback search	307
7.6.2	A case where $i < n$ is needed	258	9.3	Arrays of arrays	307
7.6.3	Off-by-one errors	260	9.3.1	Ragged arrays	308
7.6.4	The bound function of a loop	260	9.3.2	Pascal's triangle	309
7.7	Miscellaneous points about loops	261	9.4	Key concepts	311
7.7.1	There are no nested loops	261		Exercises for Chapter 9	311
7.7.2	How not to program	263	Chapter 10. Exception handling	313	
7.8	Key concepts	265	10.1	Output of thrown Exceptions	313
	Exercises for Chapter 7	265	10.2	The throw-statement	316
Chapter 8. Arrays	271		10.3	The throwable object	317
8.1	Arrays of subscripted variables	271	10.4	Catching a thrown Exception	319
8.1.1	Declarations of arrays	272	10.4.1	The try-statement	319
8.1.2	Creating an array	272	10.4.3	Propagation of a thrown exception	321
8.1.3	Referencing the length of an array	273	10.5	Checked Exceptions	323
8.1.4	Array initializers	273	10.6	Hints on using exceptions	325
8.1.5	Consequences of arrays as objects	274	10.7	Key concepts	326
8.1.6	Passing an array as an argument	275	10.8	Self-review exercises	327
8.1.7	Initializing class-type arrays	276		Exercises for Chapter 10	327
8.2	Talking about array segments	276	Chapter 11. Packages	329	
8.2.1	Range notation: h..k	276	11.1	Using packages	329
8.2.2	Horizontal descriptions of arrays.	277	11.2	Package names	331
8.2.3	Placing information in a segment	278	11.3	The packages that come with Java	333
8.3	Processing array segments	279	11.4	Key concepts	334
8.3.1	Printing an array segment	279	Chapter 12. Interface and nested class	335	
8.3.2	A schema to process arrays	280	12.1	Interfaces	335
8.3.3	A schema to process in reverse	280	12.1.1	The interface as a type	337
8.3.4	Example: using a schema	281	12.1.2	Implementing several interfaces	339
8.3.5	Checking equality of arrays	283	12.1.3	Extending an interface	340
8.3.6	Returning an array	284	12.2	Comparable and Comparator	341
8.4	Storing tables of values in arrays	285	12.3	Enumeration and Iterator	344
8.4.1	Changing the size of an array	287	12.4	Nested classes	348
8.5	Basic array algorithms	288	12.4.1	Static nested classes	348
8.5.1	Linear search	288			
8.5.2	Finding the minimum value	289			
8.5.3	Binary search	291			
8.5.4	Selection sort	293			

12.4.2 Inner classes	350	15.2.1 Tiling Elaine's kitchen	411
12.4.3 The flattened view of inner classes	356	15.2.2 Computing xy	412
12.4.4 Local inner classes	358	15.2.3 Computing Fibonacci numbers	413
12.4.5 Anonymous classes	360	15.2.4 Merge sort	414
12.5 Key concepts	361	15.3 Execution of recursive calls	415
Exercises for Chapter 12	362	15.3.1 Tail-recursive procedure calls	416
PART III ASPECTS OF PROGRAMMING	365	15.3.2 Tail-recursive function calls	417
Chapter 13. Programming style	367	15.3.3 Removing tail-recursion: procedures	418
13.1 Naming conventions	370	15.3.4 Removing tail-recursion: functions	419
13.1.1 Conventions for variable names	370	15.4 Quicksort	420
13.1.2 Conventions for naming methods	372	15.4.1 Algorithm partition	421
13.1.3 Conventions for class names	372	15.4.2 Basic quicksort	421
13.2 Conventions for indentation	373	15.4.3 Quicksort at its best	422
13.2.1 Indenting if-statements	373	15.4.4 Quicksort at its worst	424
13.2.2 Indenting assertions	374	15.4.5 Quicksort's time/space problems	425
13.2.3 Indenting loops	374	15.4.6 Removing quicksort's tail recursion	427
13.2.4 Indenting the body of a method	375	15.5 Object recursion	427
13.2.5 Indenting components of a class	375	15.6 Key concepts	428
13.3 Guidelines for writing methods	376	Exercises for Chapter 15	431
13.3.1 The specification as a contract	376	Chapter 16. Applications and applets	435
13.3.2 Keeping body and spec consistent	378	16.1 Java applications	435
13.3.3 Using statement-comments	379	16.2 Stand-alone applications	436
13.4 Describing variables	381	16.3 Java applets	438
Chapter 14. Testing and debugging	385	16.4 HTML and the web	441
14.1 An introduction to testing	386	16.5 Key concepts	444
14.1.1 Five maxims for creating test cases	387	Chapter 17. GUIs	445
14.1.2 Example of creating test cases	388	17.1 JFrames	446
14.1.3 A test driver	389	17.1.1 The basics of JFrames	446
14.1.4 Testing using JUnit	390	17.1.2 Placing components in a JFrame	448
14.1.5 Testing a class	391	17.2 Components	449
14.2 Approaches to creating test cases	392	17.2.1 JButtons	450
14.3 Approaches to testing	394	17.2.2 JLabels, JTextFields, JTextAreas	450
14.4 The Java assert statement	395	17.2.3 Other components	454
14.5 Debugging	398	17.2.4 JPanels as graphics panels	455
14.6 Key concepts	400	17.2.5 Components versus containers	456
Chapter 15. Recursion	403	17.2.6 Lightweight versus heavyweight	457
15.1 The recursive pattern	403	17.3 Containers and layout managers	458
15.1.1 A simple recursive definition	403	17.3.1 JPanels and FlowLayout managers	459
15.1.2 A recursive procedure	404	17.3.2 Boxes and BorderLayout managers	460
15.1.3 A recursive function	406	17.3.3 Using different layout managers	462
15.1.4 A function for a math definition	407	17.4 Listening to a GUI	462
15.1.5 The recursive pattern	408	17.4.1 Button events	463
15.1.6 Self-review exercises	409	17.4.2 Mouse events: class Square	466
15.2 Some interesting recursive methods	411	17.4.3 Mouse events: class MouseDemo	468
		17.4.4 Listening to other components	469
		17.4.4 Using several listeners	470

17.5	Dialog windows	470
17.5.1	Class JOptionPane	471
17.5.2	Class JDialog	474
17.6	Key concepts	474
	Exercises for Chapter 17	475
APPENDICES		479
Appendix I. Dealing with Java		481
I.1	Java SDK	482
I.2	DrJava	482
I.2.1	Using the Interactions Pane	483
I.2.2	Using the Definitions Pane	483
I.2.3	Javadoc	486
I.2.4	Using JUnit in DrJava	486
I.3	Using a command-line window	489
I.4	Making a stand-alone application	490
I.5	Java error messages	491
Appendix II. Java API and Javadoc		493
II.1	The Java API Specifications	493
II.2	Javadoc	496
Appendix III. Number systems and logs		499
III.1	Number systems	499
III.2	Base-2 logarithms	504
Appendix IV. Correctness of programs		505
IV.1	Hoare triples	506
IV.2	Two inference rules	508
IV.3	Axiomatic definition of statements	509
IV.3.1	Empty statement	510
IV.3.2	Assignment statement	510
IV.3.3	Multiple assignment statement	511
IV.3.4	Sequencing	511
IV.3.5	Conditional statements	513
IV.3.6	The while-loop	513
IV.4	Developing simple programs	514
IV.5	Developing loops	517
IV.5.1	Developing the invariant	517
IV.5.2	Developing the repetend	521
IV.6	A neat example: fusc	522
Index		525