

CONTENTS

<i>Preface</i>	<i>xvii</i>
<i>Acknowledgments</i>	<i>xvi</i>
Chapter 1 A Context-Sensitive Introduction	1
1.1 What Is Object-Oriented Programming?	2
1.1.1 Data Abstraction and Information Hiding	3
1.1.2 Inheritance	6
1.1.3 Polymorphism	9
1.1.4 Related Concepts—Overloading and Generic Programming	10
1.2 Alternative Language Paradigms	14
1.2.1 Introduction—Some Basic Terminology	14
1.2.2 The Imperative Programming Paradigm	17
1.2.3 Procedural Programming Languages	21
1.2.4 Functional Programming Languages	23
1.2.5 Modular Programming Languages	25
1.2.6 Declarative Programming Languages	27
1.3 Relevant Programming Language Concepts	30
1.3.1 Abstraction	30
1.3.2 Syntax and Semantics	32
1.3.3 Target Machines	34
1.3.4 The Semantic Mapping, Parsing, and Syntactic Discontinuities	37
1.3.5 Binding Times and Translation Strategies	39
1.3.6 Storage Classes and Storage Management	41
1.3.7 Dealing with Garbage	44
1.3.8 Scope, Visibility, Accessibility, and Lifetime	46
1.3.9 Types, Type Checking, and Type Transfers	49
1.3.10 Uninitialized Variables and Existence Checking	50
1.3.11 Pointer Semantics versus Value Semantics	53
1.3.12 Parameter Passing Mechanisms	54
1.3.13 Aliasing	57
1.3.14 I/O and the External Environment	59
1.3.15 Exception Handling	61
1.3.16 Threading	62
1.4 Simula 67 and its Historical Significance	63
1.4.1 Generalizing the Procedure Call	64
1.4.2 Inheritance in Simula	66
1.4.3 Coroutines	68
1.5 Setting the Stage: Object-Oriented Terminology	69
1.6 Summary	71

Chapter 2 Event-Driven Programming	85
2.1 Basic Definitions	86
2.2 The Hardware Model: Interrupts and Interrupt Handlers	87
2.3 Operating System Support	89
2.4 Callback Functions	90
2.5 Object-Oriented Models for EDP	91
2.6 An Example of an Event and its Lifetime	92
2.7 Programming in an Object-Oriented Windowing Environment	93
2.7.1 Bitmapped Graphics Displays	94
2.7.2 Partitioning the Display	95
2.7.3 Mice, Cursors, and the “Focus”	95
2.8 Two-Stage View Creation	96
2.9 A Warning about Code Generators	98
2.10 Generic Graphics Overview	99
2.11 Primitive Events and their Handlers	102
2.11.1 Mouse Events	102
2.11.2 Keyboard Events	103
2.11.3 Timer Events	103
2.12 Higher-Level Events	104
2.12.1 The Load Event	105
2.12.2 The Paint Event	105
2.12.3 Focus Gained and Focus Lost	106
2.12.4 The Destroy Event	106
2.12.5 Programmer-Defined Events	107
2.13 GUI Component Examples	107
2.13.1 Push Buttons and Menu Choices	107
2.13.2 Edit Boxes	108
2.13.3 List Boxes	109
2.13.4 Scroll Bars	109
2.14 GUIs and Threading	111
2.15 Summary	111
 Chapter 3 Smalltalk and the Squeak Environment	 117
3.1 A Brief History of Smalltalk	117
3.2 “Everything is an Object”—Basic Smalltalk Syntax and Semantics	118
3.2.1 Messages	119
3.2.2 Symbols	120
3.3 Names and Types	121
3.3.1 Dynamic Bindings and the Referencing Environment	121
3.3.2 Types	122
3.3.3 Type BlockContext	123

3.3.4 Basic Operators	124
3.3.5 The nil Object	125
3.3.6 Storage Management	125
3.4 Control Flow in a Smalltalk Program	126
3.5 The Squeak Dialect and Environment	128
3.5.1 Navigating Squeak	128
3.5.2 The Squeak Browser Window	129
3.6 Information Hiding in Smalltalk	134
3.7 Polymorphism in Smalltalk	134
3.8 Smalltalk Containers	135
3.9 The Model, View, Controller (MVC) and Morphic Patterns	137
3.10 Pixels, Graphics Objects, and Drawing	137
3.10.1 Important Data Types for Drawing	137
3.10.2 Drawing on a Canvas	138
3.10.3 Default Drawing Attributes	139
3.11 The Morphic Event Model	140
3.11.1 A First Exercise in the Use of the Morphic Framework	141
3.11.2 Maintaining a Persistent Display	143
3.11.3 Subclass for a Bouncing Ball	145
3.11.4 Adding a Slider Component	148
3.12 Case Study: Scoring a Bowling Game	151
3.12.1 The Interface to the Business Class	151
3.12.2 The BowlingFrameMorph Class	153
3.12.3 The BowlingScoreMorph Class	158
3.13 Summary	164
Chapter 4 C++ and Java Commonalities and Similarities	171
4.1 Introduction: the C Heritage and a Divergence of Goals	171
4.1.1 C and C++ Evolution and Backward Compatibility	172
4.1.2 Java Abandons Compatibility	174
4.2 Declarations and Static Bindings	174
4.3 Files Involved in Translation	177
4.3.1 C++ Header Files, Source Files, and Object Files	177
4.3.2 Java Source Files, Class Files, and the JVM	182
4.4 Scope and Visibility of Names	183
4.4.1 Static Scopes	183
4.4.2 Global Scope Rules and Java Packages	184
4.4.3 C++ Namespaces	185
4.4.4 File Scope in C++	186
4.4.5 Class Scope	187
4.4.6 Lifetimes of Variables	188

4.5	Low-Level Operations	188
4.5.1	Translation of Expressions and Coercions	189
4.5.2	Basic Types, Arithmetic, and Bit-Level Operations	189
4.5.3	Programmer-Defined Operators and Boolean Coercions	191
4.5.4	The C++ <code>void*</code> Data Type	192
4.5.5	Java Wrapper Classes	192
4.6	Console Output	193
4.7	Arrays and Subscripting	195
4.8	Control Flow	197
4.8.1	Expression-Level Sequence Control	197
4.8.2	Casts and Temporaries	199
4.8.3	Statement-Level Sequence Control	201
4.9	Class Concepts	206
4.9.1	Basic Syntax and Semantics	208
4.9.2	Value Semantics and Pointer Semantics	209
4.9.3	Single Inheritance	211
4.9.4	Abstract Classes vs. Concrete Classes	212
4.9.5	Overrides and Polymorphism	216
4.9.6	Run-Time Type Checking and Casting	218
4.9.7	Named Constants	219
4.10	Exception Handling	222
4.11	Library Commonalities	227
4.11.1	Heterogeneity and Genericity	227
4.11.2	The Vector Classes—Direct Access and Adjustable Size	230
4.11.3	Iterators—Traversing a Container	232
4.11.4	The List Classes—Efficient Insertions and Deletions	234
4.11.5	Sets and Maps—Efficiently Searchable Trees	238
4.12	Summary	244
Chapter 5	Additional Concepts from the C++ Language	253
5.1	L-Values, Pointers, and References	254
5.1.1	L-Values and R-Values	254
5.1.2	Pointers	255
5.1.3	References	256
5.1.4	Pointers to Members	258
5.2	Storage Management and Class Destructors	260
5.3	Arrays, Strings, and Pointer Arithmetic	265
5.4	Parameter Transmission Modes and Mutability	269
5.4.1	Reference Parameters	269
5.4.2	Mutability and <code>const</code> Functions	271
5.5	Type Coercions, Parameter Matching, and Overload Resolution	272

5.6	Stream Input and Output	275
5.6.1	Console Input	275
5.6.2	File I/O Using Streams	277
5.6.3	String Streams	278
5.7	File-Scope Variables and Procedure Definitions	280
5.8	Inline Implementation vs. Separate Compilation	283
5.9	Operator Overloading	285
5.10	More on C++ Header Files and Source Files	288
5.11	C++ Access Control and Inheritance Patterns	294
5.11.1	Protected Access and Friends	295
5.11.2	Protected and Private Inheritance	296
5.11.3	Multiple Inheritance	297
5.12	Slicing	303
5.13	Downcasting, Crosscasting, and Run-Time Type Information	304
5.14	Macros and Function Templates	306
5.15	Class Templates and Member Function Templates	309
5.16	Hash Tables in C++	311
5.17	Summary	313
Chapter 6	Visual Studio and the Microsoft Foundation Classes	327
6.1	Win32 Messages and the Message Loop	328
6.2	The MFC Library and the Document/View Architecture	329
6.3	Using the Code Generators to Develop an MFC Application	332
6.4	Device Contexts, Graphics, and Maintaining the View Window	335
6.5	Colors, Pens, and Brushes	337
6.6	Using the Visual Studio “Properties” Window	338
6.7	Timers and Animations	344
6.8	A Persistent and Traversable AVM Game Class	354
6.9	An AVM Game Application Using the Document Class	363
6.10	Designing and Using a Dialog Box	370
6.11	Snapshots, Bit Block Transfers, and Accelerator Keys	379
6.12	Summary	386
Chapter 7	Java and the Swing Library	395
7.1	Text Input Operations and the Wrapper Classes	396
7.1.1	Stream Readers, Buffered Readers, and the readLine() Method	396
7.1.2	The StringTokenizer Class	397
7.1.3	“Parsing” with the Wrapper Classes	398
7.2	Java Inheritance and Polymorphism	400
7.3	Java Interfaces	401
7.3.1	Sorting and Interface Comparable	402

7.3.2	Example: Interfaces for a Radix Sort	404
7.3.3	Interface Inheritance	407
7.3.4	Iterating through a Collection	408
7.3.5	Interface Cloneable—the Ramifications of Pointer Semantics	409
7.4	Java Enumerations	411
7.5	Inner Classes in Java	413
7.5.1	Nonlocal References in Inner Classes	414
7.5.2	Using an Inner Class to Export an Interface	415
7.5.3	Anonymous Inner Classes	418
7.5.4	Adapters	419
7.6	Java Generics	420
7.6.1	Defining a Generic Container Class	421
7.6.2	Type Relationships Revisited	424
7.7	Java Hash Tables	426
7.8	Introduction to the Swing Library	428
7.8.1	The Graphics and Component Classes	428
7.8.2	Top-Level Components	431
7.8.3	Timer Example: A “Dropped Ball” Panel	434
7.8.4	Layouts	440
7.9	Serialization	444
7.9.1	Introduction to Java Serialization	444
7.9.2	Wrapping for Storage	445
7.9.3	Some Simple Examples	446
7.10	Menus, Dialogs, and a Complete Application	448
7.10.1	The JOptionPane Class	448
7.10.2	Building a Maze	449
7.10.3	The Maze Class	451
7.10.4	The MazePanel Class	457
7.10.5	Menus on a JFrame Object—The Maze Application	464
7.10.6	Custom Dialog Boxes	471
7.11	Gaining Run-Time Access to Type Information	474
7.12	Component-Level Programming	476
7.12.1	Bean Properties	476
7.12.2	A “Drag and Drop” Example	481
7.13	Applets	483
7.13.1	The Maze Application as an Applet	485
7.13.2	Security and the “Sandbox”	486
7.13.3	Applets and Threads	487
7.14	Summary	488

Chapter 8 C# and the Common Language Infrastructure	499
8.1 Introduction to the Common Language Infrastructure	499
8.2 Some Interesting C# Design Choices	500
8.2.1 C# Control Flow	501
8.2.2 Value and Reference Types	504
8.2.3 Arrays and Multidimensional Arrays	507
8.2.4 Strings and Parsing	509
8.2.5 Parameter Transmission in C#	511
8.2.6 Variable-Sized Parameter Lists	514
8.2.7 Properties—Access Functions in Disguise	515
8.2.8 Indexers—Programmer-Defined Subscripting	516
8.2.9 Operator Overloading	519
8.2.10 C# Compilation Units	522
8.3 C# Types and Namespaces	523
8.3.1 C# Classes and Partial Classes	523
8.3.2 C# Structs	525
8.3.3 Interfaces	526
8.3.4 Enumerations	526
8.3.5 Namespaces and Namespace Aliases	527
8.3.6 Partial Methods	528
8.4 Additional Inheritance Considerations	529
8.4.1 Run-Time Polymorphism	529
8.4.2 Inline Implementation	531
8.5 C# Type Parameters	532
8.6 Some Important Library Interfaces	536
8.6.1 IDisposable and the using Statement	537
8.6.2 Anonymous IEnumerable Objects and yield Statements	537
8.6.3 Implementing IEnumerable<> and IEnumerator<>	539
8.6.4 The ICloneable Interface	541
8.6.5 The IComparable Interface	541
8.7 Function Types in C#	542
8.7.1 Delegates	542
8.7.2 Multicast Delegates	544
8.7.3 Lambda Expressions and Anonymous Methods	547
8.7.4 Captured Local Variables	549
8.7.5 Parameterized Delegates	551
8.7.6 Expression Trees	553
8.8 .NET Framework Collection Classes	554
8.9 Windows Forms and Controls	556
8.9.1 Starting a Windows Application in Visual Studio	556
8.9.2 Drawing on a Form	559

8.9.3	Events in C#	562
8.9.4	An Application Using Mouse Events and a Timer	563
8.10	Programmer-Defined Events	568
8.10.1	Event Syntax	569
8.10.2	A Bouncing Ball Class with a “Bounce” Event	569
8.10.3	Reacting to the Event—A Bouncing Ball Application	573
8.11	Unsafe Code	576
8.12	File I/O and Serialization	579
8.13	Summary	581
Chapter 9	Python	593
9.1	Python: a Highly Simulated OOPL with Unobtrusive Syntax	594
9.1.1	Indentation	594
9.1.2	Keyword Arguments	595
9.2	The IDLE Environment and Console I/O	596
9.2.1	Console Output	596
9.2.2	Console Input	597
9.3	Python Data Types	598
9.3.1	Numeric Types	599
9.3.2	NoneType and the Value None, and Uninitialized Variables	601
9.3.3	Sequence Types	602
9.3.4	String Operations	608
9.3.5	Hashing and the Built-In Keyed Collections	610
9.4	Python Scopes and Information Layering	612
9.4.1	Function Definitions and Function Objects	613
9.4.2	Class Definitions	616
9.4.3	Modules	619
9.4.4	Decorating an Object	621
9.4.5	Storage Management and Dynamic Variable Deletion	622
9.5	Sequence Control in Python	624
9.5.1	Exception Handling	624
9.5.2	The with Construct	626
9.5.3	Operator Precedence	626
9.5.4	Statement-Level Sequence Control	627
9.5.5	Generators, Iterators, and Comprehensions	631
9.5.6	Operator Overloading	634
9.6	Inheritance and Polymorphism	636
9.7	Functional Features of Python	639
9.8	The External Environment	640
9.8.1	Files and I/O	640
9.8.2	Python Features for Persistence—pickle and shelve	642

9.9	The tkinter Library and GUI Applications	644
9.9.1	Creating a Root Window in tkinter	645
9.9.2	Adding a Button and a Label	646
9.9.3	Threads and Windows	647
9.9.4	“Modal” Windows	648
9.9.5	Widgets and Events in tkinter	649
9.9.6	Canvas Objects and How to Draw with Them	650
9.10	A Postscript: Python, Smalltalk, and a Culture of Reusability	652
9.11	Summary	653
Appendix A	Event-Driven Project Ideas	665
Appendix B	Answers to Odd-Numbered Exercises	673
Appendix C	About the CD-ROM	713
Index		717

