

DETAILED CONTENTS

PREFACE TO THE SECOND EDITION

WELCOME TO THE SECOND EDITION	xxxv
IMPROVEMENTS TO THIS EDITION	xxxv
WHERE TO SEND COMMENTS AND SUGGESTIONS	xxxvi
ACKNOWLEDGMENTS	xxxvi

PREFACE TO THE FIRST EDITION

WELCOME – AND THANK YOU!	xxxvii
TARGET AUDIENCE	xxxvii
APPROACH(ES)	xxxvii
PEDAGOGY – I MEAN, HOW THIS BOOK’S ARRANGED	xxxviii
LEARNING OBJECTIVES	xxxviii
INTRODUCTION	xxxviii
CONTENT	xxxviii
QUICK REVIEWS	xxxviii
SUMMARY	xxxix
SKILL-BUILDING EXERCISES	xxxix
SUGGESTED PROJECTS	xxxix
SELF-TEST QUESTIONS	xxxix
REFERENCES	xxxix
NOTES	xxxix
TYPOGRAPHICAL FORMATS	xxxix
THIS IS AN EXAMPLE OF A FIRST LEVEL SUBHEADING	xxxix
<i>THIS IS AN EXAMPLE OF A SECOND LEVEL SUBHEADING.....</i>	<i>xxxix</i>
SOURCE CODE FORMATTING	xl
SUPPORTSITE™ WEBSITE	xl
PROBLEM REPORTING	xl
ABOUT THE AUTHOR	xl
ACKNOWLEDGMENTS	xl

PART I: THE C# STUDENT SURVIVAL GUIDE

I AN APPROACH TO THE ART OF PROGRAMING

INTRODUCTION	4
THE DIFFICULTIES YOU WILL ENCOUNTER LEARNING C#	4
<i>REQUIRED Skills</i>	<i>4</i>
<i>THE PLANETS WILL COME INTO ALIGNMENT.....</i>	<i>5</i>
HOW THIS CHAPTER WILL HELP YOU	5
PERSONALITY TRAITS FOUND IN GREAT PROGRAMMERS	5
CREATIVE	5

TENACIOUS	5
RESILIENT	6
METHODICAL	6
METICULOUS	6
HONEST	6
PROACTIVE	6
HUMBLE	6
BE A GENERALIST AND A JUST-IN-TIME SPECIALIST	6
PROJECT MANAGEMENT	7
THREE SOFTWARE DEVELOPMENT ROLES	7
<i>ANALYST</i>	7
<i>ARCHITECT</i>	7
<i>PROGRAMMER</i>	8
A PROJECT-APPROACH STRATEGY	8
<i>YOU HAVE BEEN HANDED A PROJECT – NOW WHAT?</i>	8
<i>STRATEGY AREAS OF CONCERN</i>	8
<i>THINK ABSTRACTLY</i>	10
THE STRATEGY IN A NUTSHELL	10
APPLICABILITY TO THE REAL WORLD	11
THE ART OF PROGRAMMING	11
DON'T START AT THE COMPUTER	11
INSPIRATION STRIKES AT THE WEIRDEST TIME	11
OWN YOUR OWN COMPUTER	12
<i>YOU EITHER HAVE TIME AND NO MONEY, OR MONEY AND NO TIME</i>	12
<i>THE FAMILY COMPUTER IS NOT GOING TO CUT IT!</i>	12
SET THE MOOD	12
<i>LOCATION, LOCATION, LOCATION</i>	12
CONCEPT OF THE FLOW	12
<i>THE STAGES OF FLOW</i>	13
BE EXTREME	13
<i>THE PROGRAMMING CYCLE</i>	14
<i>THE PROGRAMMING CYCLE SUMMARIZED</i>	14
A HELPFUL TRICK: STUBBING	15
FIX THE FIRST COMPILER ERROR FIRST	15
MANAGING PROJECT COMPLEXITY	15
CONCEPTUAL COMPLEXITY	15
<i>MANAGING CONCEPTUAL COMPLEXITY</i>	16
<i>THE UNIFIED MODELING LANGUAGE (UML)</i>	16
PHYSICAL COMPLEXITY	16
<i>MANAGING PHYSICAL COMPLEXITY</i>	16
THE RELATIONSHIP BETWEEN PHYSICAL AND CONCEPTUAL COMPLEXITY	17
MAXIMIZE COHESION – MINIMIZE COUPLING	17
THE ENGINEER'S NOTEBOOK	17
SUMMARY	18
SKILL-BUILDING EXERCISES	19
SUGGESTED PROJECTS	19
SELF-TEST QUESTIONS	19
REFERENCES	20
NOTES	20
2 SMALL VICTORIES: CREATING C# PROJECTS	
INTRODUCTION	22
CREATING PROJECTS WITH MICROSOFT C#.NET COMMAND-LINE TOOLS	22

Download And Installing The .NET Framework	23
Download And Installing Notepad++	23
Configuring Your Development Environment	24
<i>Environment Variables</i>	24
<i>Creating A Project Folder</i>	27
<i>Setting Folder Options</i>	28
<i>Creating A Shortcut To The Command Console And Setting Its Properties</i>	29
Testing The Configuration	31
<i>Creating The Source File</i>	31
<i>Compiling The Source File</i>	32
<i>Executing The Application</i>	33
Quick Review	34
 CREATING PROJECTS WITH MICROSOFT VISUAL C# EXPRESS	35
Download And Install Visual C# Express	36
Quick Tour Of Visual C# Express	36
<i>Select Project Type</i>	36
<i>Saving The Project</i>	38
<i>Build The Project</i>	38
<i>Locating The Project Executable File</i>	40
<i>Execute The Project</i>	41
Where To Go For More Information About Visual C# Express	41
DREAMSPARK	42
Quick Review	42
 SUMMARY	42
 Skill-Building Exercises	43
 SUGGESTED PROJECTS	43
 Self-Test Questions	43
 REFERENCES	44
 NOTES	44

3 PROJECT WALKTHROUGH

Introduction	46
The Project-Approach Strategy Summarized	46
Development Cycle	47
Project Specification	48
Analyzing The Project Specification	49
<i>Application Requirements Strategy Area</i>	50
<i>Problem-Domain Strategy Area</i>	50
<i>Language-Features Strategy Area</i>	53
<i>Design Strategy Area</i>	55
Development Cycle: First Iteration	56
Plan (First Iteration)	56
Code (First Iteration)	57
Test (First Iteration)	57
Integrate/Test (First Iteration)	57
Development Cycle: Second Iteration	58
Plan (Second Iteration)	58
Code (Second Iteration)	58
Test (Second Iteration)	59
Integrate/Test (Second Iteration)	59
Development Cycle: Third Iteration	59
Plan (Third Iteration)	60
Code (Third Iteration)	61

INTEGRATE/TEST (THIRD ITERATION)	63
A BUG IN THE PROGRAM	63
DEVELOPMENT Cycle: Fourth Iteration	65
PLAN (FOURTH ITERATION)	65
<i>IMPLEMENTING STATE TRANSITION DIAGRAMS.....</i>	<i>66</i>
<i>IMPLEMENTING THE PRINTFLOOR() METHOD.....</i>	<i>67</i>
CODE (FOURTH ITERATION)	67
PUTTING IT ALL TOGETHER	69
TEST (FOURTH ITERATION)	71
INTEGRATE/TEST (FOURTH ITERATION)	71
DEVELOPMENT Cycle: Fifth Iteration	72
PLAN (FIFTH ITERATION)	72
CODE (FIFTH ITERATION)	73
TEST (FIFTH ITERATION)	75
PUTTING IT ALL TOGETHER	75
INTEGRATE/TEST (FIFTH ITERATION)	79
FINAL CONSIDERATIONS	79
COMPLETE ROBOTRAT.CS SOURCE CODE LISTING	80
AN ALTERNATIVE MAIN() METHOD IMPLEMENTATION	86
SUMMARY	88
SKILL-BUILDING EXERCISES	88
SUGGESTED PROJECTS	88
SELF-TEST QUESTIONS	89
REFERENCES	89
NOTES	90

4 COMPUTERS, PROGRAMS, AND ALGORITHMS

INTRODUCTION	92
WHAT IS A COMPUTER?	92
COMPUTER VS. COMPUTER SYSTEM	92
<i>COMPUTER SYSTEM.....</i>	<i>92</i>
<i>PROCESSOR.....</i>	<i>94</i>
THREE ASPECTS OF PROCESSOR ARCHITECTURE	95
<i>FEATURE SET.....</i>	<i>95</i>
<i>FEATURE SET IMPLEMENTATION.....</i>	<i>96</i>
<i>FEATURE SET ACCESSIBILITY.....</i>	<i>96</i>
MEMORY ORGANIZATION	96
MEMORY BASICS	96
<i>MEMORY HIERARCHY.....</i>	<i>97</i>
<i>Bits, Bytes, Words.....</i>	<i>97</i>
ALIGNMENT AND ADDRESSABILITY	98
WHAT IS A PROGRAM?	98
TWO VIEWS OF A PROGRAM	98
<i>THE HUMAN PERSPECTIVE.....</i>	<i>99</i>
<i>THE COMPUTER PERSPECTIVE.....</i>	<i>99</i>
THE PROCESSING Cycle	100
FETCH	100
DECODE	100
EXECUTE	100
STORE	101
<i>Why A PROGRAM CRASHES.....</i>	<i>101</i>
ALGORITHMS	101
Good vs. Bad Algorithms	101

DON'T REINVENT THE WHEEL! 104

VIRTUAL MACHINES AND THE COMMON LANGUAGE INFRASTRUCTURE 104

 VIRTUAL MACHINES 105

 THE COMMON LANGUAGE INFRASTRUCTURE (CLI) 105

FOUR PARTS OF THE COMMON LANGUAGE INFRASTRUCTURE 106

THE CROSS PLATFORM PROMISE 107

SUMMARY 108

SKILL-BUILDING EXERCISES 109

SUGGESTED PROJECTS 109

SELF-TEST QUESTIONS 109

REFERENCES 110

NOTES 111

5. NAVIGATING .NET FRAMEWORK DOCUMENTATION

INTRODUCTION 114

MSDN: THE DEFINITIVE SOURCE FOR API INFORMATION 114

DISCOVERING INFORMATION ABOUT CLASSES 114

 GENERAL OVERVIEW PAGE 115

 CLASS MEMBER DOCUMENTATION 117

 GETTING INFORMATION ON OTHER CLASS MEMBERS 119

 Quick Review 119

NAMESPACES USED HEAVILY IN THIS BOOK 120

NAVIGATING AN INHERITANCE HIERARCHY 120

 Quick Review 121

BEWARE OBSOLETE APIS 122

SUMMARY 122

SKILL-BUILDING EXERCISES 122

SUGGESTED PROJECTS 123

SELF-TEST QUESTIONS 124

REFERENCES 124

NOTES 125

PART II: LANGUAGE FUNDAMENTALS

6 Simple C# PROGRAMS

INTRODUCTION 130

WHAT IS A C# PROGRAM? 130

A Simple Console Application 131

 DEFINITION OF TERMS: Application, Assembly, Module, and Entry Point 131

 STRUCTURE OF A Simple Application 131

 PURPOSE OF THE MAIN() METHOD 132

 MAIN() METHOD SIGNATURES 132

 Quick Review 133

IDENTIFIERS AND RESERVED KEYWORDS 134

 IDENTIFIER NAMING RULES 135

 Quick Review 135

Types 136

 VALUE TYPE VARIABLES VS. REFERENCE TYPE VARIABLES 136

<i>VALUE TYPE VARIABLES</i>	137
<i>REFERENCE TYPE VARIABLES</i>	137
<i>MAYBE SOME PICTURES WILL HELP</i>	138
<i>MAPPING PREDEFINED TYPES TO SYSTEM STRUCTURES</i>	138
Quick Review	140
STATEMENTS, EXPRESSIONS, AND OPERATORS	140
STATEMENT TYPES	141
OPERATORS AND THEIR USE	142
<i>OPERATOR PRECEDENCE AND ASSOCIATIVITY</i>	144
<i>FORCING OPERATOR PRECEDENCE AND ASSOCIATIVITY ORDER WITH PARENTHESES</i>	144
<i>OPERATORS AND OPERANDS</i>	144
OPERATOR USAGE EXAMPLES	144
<i>PRIMARY EXPRESSION OPERATORS</i>	144
<i>UNARY EXPRESSION OPERATORS</i>	145
<i>MULTIPLICATIVE EXPRESSION OPERATORS</i>	146
<i>ADDITIVE EXPRESSION OPERATORS</i>	146
<i>SHIFT EXPRESSION OPERATORS</i>	147
<i>RELATIONAL, TYPE-TESTING, AND EQUALITY EXPRESSION OPERATORS</i>	148
<i>LOGICAL AND, OR, AND XOR EXPRESSION OPERATORS</i>	150
<i>CONDITIONAL AND AND OR EXPRESSION OPERATORS</i>	152
<i>CONDITIONAL (TERNARY) EXPRESSION OPERATOR</i>	153
<i>ASSIGNMENT EXPRESSION OPERATORS</i>	154
Quick Review	154
SUMMARY	155
SKILL-BUILDING EXERCISES	155
SUGGESTED PROJECTS	156
SELF-TEST QUESTIONS	156
REFERENCES	157
NOTES	158

7 CONTROLLING THE FLOW OF PROGRAM EXECUTION

INTRODUCTION	160
SELECTION STATEMENTS	160
IF STATEMENT	160
<i>HANDLING PROGRAM ERROR CONDITIONS</i>	161
<i>EXECUTING CODE BLOCKS IN IF STATEMENTS</i>	163
<i>EXECUTING CONSECUTIVE IF STATEMENTS</i>	164
IF/ELSE STATEMENT	164
<i>CHAINED IF/ELSE STATEMENTS</i>	166
SWITCH STATEMENT	167
<i>IMPLICIT CASE FALL-THROUGH</i>	168
<i>NESTED SWITCH STATEMENT</i>	169
Quick Review	170
ITERATION STATEMENTS	170
WHILE STATEMENT	171
<i>PERSONALITY OF THE WHILE STATEMENT</i>	171
DO/WHILE STATEMENT	172
<i>PERSONALITY OF THE DO/WHILE STATEMENT</i>	172
FOR STATEMENT	173
<i>HOW THE FOR STATEMENT IS RELATED TO THE WHILE STATEMENT</i>	173
<i>PERSONALITY OF THE FOR STATEMENT</i>	174
NESTING ITERATION STATEMENTS	174
MIXING SELECTION AND ITERATION STATEMENTS: A POWERFUL COMBINATION	175
Quick Review	177

BREAK, CONTINUE, AND GOTO	177
BREAK STATEMENT	177
CONTINUE STATEMENT	178
GOTO STATEMENT	179
Quick Review	179
SELECTION AND ITERATION STATEMENT SELECTION TABLE	180
SUMMARY	181
SKILL-BUILDING EXERCISES	182
SUGGESTED PROJECTS	184
SELF-TEST QUESTIONS	185
REFERENCES	185
NOTES	186

8 ARRAYS

INTRODUCTION	188
WHAT IS AN ARRAY?	188
Specifying Array Types	189
Quick Review	190
FUNCTIONALITY PROVIDED BY C# ARRAY TYPES	190
ARRAY-TYPE INHERITANCE HIERARCHY	190
SPECIAL PROPERTIES OF C# ARRAYS	191
Quick Review	192
CREATING AND USING SINGLE-DIMENSIONAL ARRAYS	192
ARRAYS OF VALUE TYPES	192
<i>How Value-Type Array Objects Are Arranged In Memory</i>	193
FINDING AN ARRAY'S TYPE, RANK, AND TOTAL NUMBER OF ELEMENTS	194
CREATING SINGLE-DIMENSIONAL ARRAYS USING ARRAY LITERAL VALUES	195
DIFFERENCES BETWEEN ARRAYS OF VALUE TYPES AND ARRAYS OF REFERENCE TYPES	196
SINGLE-DIMENSIONAL ARRAYS IN ACTION	198
<i>Message Array</i>	198
<i>Calculating Averages</i>	200
<i>Histogram: Letter Frequency Counter</i>	201
Quick Review	203
CREATING AND USING MULTIDIMENSIONAL ARRAYS	203
RECTANGULAR ARRAYS	203
<i>Initializing Rectangular Arrays With Array Literals</i>	205
RAGGED ARRAYS	206
MULTIDIMENSIONAL ARRAYS IN ACTION	207
<i>Weighted Grade Tool</i>	207
Quick Review	209
THE MAIN() METHOD'S STRING ARRAY	210
PURPOSE AND USE OF THE MAIN() METHOD'S STRING ARRAY	210
MANIPULATING ARRAYS WITH THE SYSTEM.ARRAY CLASS	211
NUMERIC FORMATTING	211
SUMMARY	212
SKILL-BUILDING EXERCISES	213
SUGGESTED PROJECTS	213
SELF-TEST QUESTIONS	216
REFERENCES	216
NOTES	217

9 TOWARD PROBLEM ABSTRACTION: CREATING NEW DATA TYPES

INTRODUCTION	220
ABSTRACTION: Amplify THE ESSENTIAL, ELIMINATE THE IRRELEVANT	220
ABSTRACTION IS THE ART OF PROGRAMMING	220
WHERE PROBLEM ABSTRACTION FITS INTO THE DEVELOPMENT CYCLE	221
CREATING YOUR OWN DATA TYPES	221
CASE-STUDY PROJECT: WRITE A PEOPLE MANAGER PROGRAM	221
QUICK REVIEW	223
THE UML CLASS DIAGRAM	223
QUICK REVIEW	225
OVERVIEW OF THE CLASS CONSTRUCT	225
ELEVEN CATEGORIES OF CLASS MEMBERS	225
<i>Fields</i>	225
<i>CONSTANTS</i>	228
<i>The Difference BETWEEN const AND readonly; Compile-Time vs. Runtime Constants</i>	228
<i>PROPERTIES</i>	229
<i>METHODS</i>	231
<i>INSTANCE CONSTRUCTORS</i>	231
<i>STATIC CONSTRUCTORS</i>	231
<i>EVENTS</i>	231
<i>OPERATORS</i>	231
<i>INDEXERS</i>	231
<i>NESTED TYPE DECLARATIONS</i>	232
<i>FINALIZERS</i>	232
ACCESS MODIFIERS	232
<i>PUBLIC</i>	232
<i>PRIVATE</i>	232
<i>PROTECTED</i>	232
<i>INTERNAL</i>	232
<i>PROTECTED INTERNAL</i>	233
THE CONCEPTS OF HORIZONTAL ACCESS, INTERFACE, AND ENCAPSULATION	233
QUICK REVIEW	234
METHODS	234
METHOD NAMING: USE ACTION WORDS THAT INDICATE THE METHOD'S PURPOSE	234
MAXIMIZE METHOD COHESION	234
STRUCTURE OF A METHOD DEFINITION	235
<i>Method Modifiers (optional)</i>	235
<i>RETURN TYPE OR VOID (optional)</i>	236
<i>METHOD NAME (MANDATORY)</i>	236
<i>PARAMETER LIST (optional)</i>	236
<i>Method Body (optional FOR ABSTRACT OR EXTERNAL METHODS)</i>	236
METHOD DEFINITION EXAMPLES	237
METHOD SIGNATURES	237
OVERLOADING METHODS	237
CONSTRUCTOR METHODS	238
QUICK REVIEW	238
BUILDING AND TESTING THE PERSON CLASS	238
START BY CREATING THE SOURCE FILE AND CLASS DEFINITION SHELL	239
DEFINING PERSON INSTANCE FIELDS	239
DEFINING PERSON PROPERTIES AND CONSTRUCTOR METHOD	239
<i>Adding PROPERTIES</i>	240
<i>Adding A CONSTRUCTOR METHOD</i>	240
TESTING THE PERSON CLASS: A MINIATURE TEST PLAN	241
<i>USE THE PEOPLEMANAGERAPPLICATION CLASS AS A TEST DRIVER</i>	242
ADDING FEATURES TO THE PERSON CLASS: CALCULATING AGE	242

Adding FEATURES TO THE PERSON CLASS: CONVENIENCE PROPERTIES	244
Adding FEATURES TO THE PERSON CLASS: FINISHING TOUCHES	246
Quick Review	248
Building AND TESTING THE PEOPLEMANAGER CLASS	248
DEFINING THE PEOPLEMANAGER CLASS SHELL	249
DEFINING PEOPLEMANAGER FIELDS	249
DEFINING PEOPLEMANAGER CONSTRUCTOR METHODS	249
DEFINING ADDITIONAL PEOPLEMANAGER METHODS	250
TESTING THE PEOPLEMANAGER CLASS	251
Adding FEATURES TO THE PEOPLEMANAGER CLASS	251
Quick Review	253
MORE ABOUT METHODS	254
VALUE PARAMETERS AND REFERENCE PARAMETERS	254
<i>VALUE PARAMETERS: THE DEFAULT PARAMETER PASSING MODE</i>	254
<i>REFERENCE PARAMETERS: USING THE ref PARAMETER MODIFIER</i>	255
THE out PARAMETER MODIFIER	257
PARAMETER ARRAYS: USING THE params MODIFIER	259
LOCAL VARIABLE SCOPING	259
ANYWHERE AN OBJECT OF <type> IS REQUIRED, A METHOD THAT RETURNS <type> CAN BE USED	260
Quick Review	260
STRUCTURES VS. CLASSES	260
VALUE SEMANTICS VS. REFERENCE SEMANTICS	261
TEN AUTHORIZED MEMBERS VS. ELEVEN	261
DEFAULT VARIABLE FIELD VALUES	261
BEHAVIOR DURING ASSIGNMENT	261
this BEHAVES DIFFERENTLY	262
INHERITANCE NOT ALLOWED	262
BOXING AND UNBOXING	262
WHEN TO USE STRUCTURES	262
SUMMARY	263
SKILL-BUILDING EXERCISES	264
SUGGESTED PROJECTS	265
SELF-TEST QUESTIONS	267
REFERENCES	268
NOTES	269

10 COMPOSITIONAL DESIGN

INTRODUCTION	272
MANAGING CONCEPTUAL AND PHYSICAL COMPLEXITY	272
Compiling Multiple SOURCE FILES SIMULTANEOUSLY WITH CSC	272
Quick Review	273
DEPENDENCY VS. ASSOCIATION	273
AGGREGATION	274
Simple vs. Composite Aggregation	274
<i>THE RELATIONSHIP BETWEEN AGGREGATION AND OBJECT LIFETIME</i>	274
Quick Review	275
EXPRESSING AGGREGATION IN A UML CLASS DIAGRAM	275
Simple Aggregation Expressed In UML	275
Composite Aggregation Expressed In UML	276
AGGREGATION EXAMPLE CODE	276
Simple Aggregation Example	276
Composite Aggregation Example	278

Quick Review	279
SEQUENCE DIAGRAMS	279
MESSAGE PASSING	281
MAGIC DRAW	281
Quick Review	281
THE ENGINE SIMULATION: AN EXTENDED EXAMPLE	281
THE PURPOSE OF THE ENGINE CLASS	281
ENGINE CLASS ATTRIBUTES AND METHODS	281
ENGINE SIMULATION SEQUENCE DIAGRAMS	284
RUNNING THE ENGINE SIMULATION PROGRAM	284
Quick Review	284
COMPLETE ENGINE SIMULATION CODE LISTING	286
SUMMARY	291
SKILL-BUILDING EXERCISES	291
SUGGESTED PROJECTS	292
SELF-TEST QUESTIONS	293
REFERENCES	294
NOTES	295

II INHERITANCE AND INTERFACES

INTRODUCTION	298
THREE PURPOSES OF INHERITANCE	298
IMPLEMENTING THE “IS A” RELATIONSHIP	299
THE RELATIONSHIP BETWEEN THE TERMS TYPE, INTERFACE, AND CLASS	299
<i>MEANING OF THE TERM INTERFACE</i>	299
<i>MEANING OF THE TERM CLASS</i>	300
Quick Review	300
EXPRESSING GENERALIZATION AND SPECIALIZATION IN THE UML	300
A SIMPLE INHERITANCE EXAMPLE	300
THE UML DIAGRAM	301
BASECLASS SOURCE CODE	302
DERIVEDCLASS SOURCE CODE	302
DRIVERAPPLICATION PROGRAM	303
Quick Review	304
ANOTHER INHERITANCE EXAMPLE: PERSON - STUDENT	304
THE PERSON - STUDENT UML CLASS DIAGRAM	304
PERSON - STUDENT SOURCE CODE	304
CASTING	307
<i>USE CASTING SPARINGLY</i>	308
Quick Review	309
OVERRIDING BASE CLASS METHODS	309
Quick Review	311
ABSTRACT METHODS AND ABSTRACT BASE CLASSES	311
THE PRIMARY PURPOSE OF AN ABSTRACT BASE CLASS	311
EXPRESSING ABSTRACT BASE CLASSES IN UML	311
<i>ABSTRACT CLASS EXAMPLE</i>	312
Quick Review	314
INTERFACES	314
THE PURPOSE OF INTERFACES	314
AUTHORIZED INTERFACE MEMBERS	314
THE DIFFERENCES BETWEEN AN INTERFACE AND AN ABSTRACT CLASS	315
EXPRESSING INTERFACES IN UML	315

Expressing Realization In A UML Class Diagram	315
An Interface Example	316
Quick Review	318
Controlling Horizontal And Vertical Access	318
Quick Review	318
Sealed Classes And Methods	319
Quick Review	319
Polymorphic Behavior	319
Quick Review	319
Inheritance Example: Employee	320
Inheritance Example: Engine Simulation	323
Engine Simulation UML Diagram	323
Simulation Operational Description	325
Compiling The Engine Simulation Code	325
Complete Engine Simulation Code Listing	325
Summary	330
Skill-Building Exercises	331
Suggested Projects	332
Self-Test Questions	333
References	334
Notes	335

PART III: GUI PROGRAMMING & CUSTOM EVENTS

12 Windows Forms Programming

Introduction	340
The Form Class	340
Form Class Inheritance Hierarchy	340
A Simple Form Program	340
Quick Review	342
Application Messages, Message Pump, Events, And Event Loop	342
Message Categories	343
Messages In Action: Trapping Messages With IMessageFilter	344
Final Thoughts On Messages	345
Quick Review	345
Screen And Window (Client) Coordinate System	345
Quick Review	347
Manipulating Form Properties	348
Quick Review	349
Adding Components To Windows: Button, TextBox, And Label	350
Quick Review	351
Registering Event Handlers With GUI Components	352
Delegates And Events	352
Quick Review	354
Handling GUI Component Events In Separate Objects	355
Quick Review	357
Layout Managers	358
FlowLayoutPanel	358
TableLayoutPanel	361

Quick Review	362
MENUS	363
Quick Review	366
A LITTLE MORE ABOUT TEXTBOXES	366
Quick Review	368
THE RHYTHM OF CODING GUIs	368
SUMMARY	368
SKILL-BUILDING EXERCISES	370
SUGGESTED PROJECTS	371
SELF-TEST QUESTIONS	371
REFERENCES	372
NOTES	373

13 CUSTOM EVENTS

INTRODUCTION	376
C# EVENT PROCESSING MODEL: AN OVERVIEW	376
Quick Review	377
CUSTOM EVENTS EXAMPLE: MINUTE TICK	378
CUSTOM EVENTS EXAMPLE: AUTOMATED WATER TANK SYSTEM	380
NAMING CONVENTIONS	388
FINAL THOUGHTS ON EXTENDING THE EVENTARGS CLASS	388
SUMMARY	388
SKILL-BUILDING EXERCISES	388
SUGGESTED PROJECTS	389
SELF-TEST QUESTIONS	389
REFERENCES	390
NOTES	391

PART IV: INTERMEDIATE CONCEPTS

14 COLLECTIONS

INTRODUCTION	396
CASE STUDY: BUILDING A DYNAMIC ARRAY	396
EVALUATING DYNAMICARRAY	398
THE ARRAYLIST CLASS TO THE RESCUE	399
A QUICK PEEK AT GENERICS	399
Quick Review	400
DATA STRUCTURE PERFORMANCE CHARACTERISTICS	400
ARRAY PERFORMANCE CHARACTERISTICS	401
LINKED LIST PERFORMANCE CHARACTERISTICS	402
HASH TABLE PERFORMANCE CHARACTERISTICS	402
<i>Chained Hash Table vs. Open-Address Hash Table</i>	405
RED-BLACK TREE PERFORMANCE CHARACTERISTICS	405
STACKS AND QUEUES	406
Quick Review	406
NAVIGATING THE .NET COLLECTIONS API	407
SYSTEM.COLLECTIONS	407
SYSTEM.COLLECTIONS.GENERIC	407

SYSTEM.COLLECTIONS.OBJECTMODEL	408
SYSTEM.COLLECTIONS.SPECIALIZED	408
MAPPING NON-GENERIC TO GENERIC COLLECTIONS	408
Quick Review	409
Using Non-Generic Collection Classes - Pre .NET 2.0	409
Objects In – Objects Out: Casting 101	411
EXTENDING ARRAYLIST TO CREATE A STRONGLY-TYPED COLLECTION	412
Using Generic Collection Classes – .NET 2.0 and Beyond	414
LIST<T>: LOOK MA, NO MORE CASTING!	414
IMPLEMENTING KEYEDCOLLECTION<TKEY, TITEM>	415
Quick Review	417
SPECIAL OPERATIONS ON COLLECTIONS	417
SORTING A LIST	417
<i>IMPLEMENTING SYSTEM.ICOMPARABLE<T>.....</i>	<i>418</i>
<i>EXTENDING COMPARE<T>.....</i>	<i>421</i>
CONVERTING A COLLECTION INTO AN ARRAY	422
Quick Review	423
SUMMARY	423
SKILL-BUILDING EXERCISES	424
SUGGESTED PROJECTS	425
SELF-TEST QUESTIONS	425
REFERENCES	426
NOTES	427

15 EXCEPTIONS: WRITING FAULT-TOLERANT SOFTWARE

INTRODUCTION	430
WHAT IS AN EXCEPTION	430
.NET CLR EXCEPTION HANDLING MECHANISM	430
UNHANDLED EXCEPTIONS	431
THE EXCEPTION INFORMATION TABLE	431
Quick Review	431
EXCEPTION CLASS HIERARCHY	432
APPLICATION VS. RUNTIME EXCEPTIONS	432
RUNTIME EXCEPTION LISTING	433
DETERMINING WHAT EXCEPTIONS A .NET FRAMEWORK METHOD THROWS	433
Quick Review	433
EXCEPTION CLASS PROPERTIES	434
Quick Review	435
CREATING EXCEPTION HANDLERS: USING TRY/CATCH/FINALLY BLOCKS	435
USING A TRY/CATCH BLOCK	435
<i>FIRST LINE OF DEFENSE: USE DEFENSIVE CODING.....</i>	<i>436</i>
USING MULTIPLE CATCH BLOCKS	437
USING A FINALLY BLOCK	438
Quick Review	439
CREATING CUSTOM EXCEPTIONS	439
EXTENDING THE EXCEPTION CLASS	439
MANUALLY THROWING AN EXCEPTION WITH THE THROW KEYWORD	441
TRANSLATING LOW-LEVEL EXCEPTIONS INTO HIGH-LEVEL EXCEPTIONS	441
Quick Review	441
DOCUMENTING EXCEPTIONS	441
SUMMARY	442
SKILL-BUILDING EXERCISES	443

SUGGESTED PROJECTS	443
SELF-TEST QUESTIONS	443
REFERENCES	444
NOTES	445

16 MULTITHREADED PROGRAMMING

INTRODUCTION	448
MULTITHREADING OVERVIEW: THE TALE OF TWO VACATIONS	448
SINGLE-THREADED VACATION	448
MULTITHREADED VACATION	448
THE RELATIONSHIP BETWEEN A PROCESS AND ITS THREADS	449
VACATION GONE BAD	451
Quick Review	451
CREATING MANAGED THREADS WITH THE THREAD CLASS	452
SINGLE-THREADED VACATION EXAMPLE	452
MULTITHREADED VACATION EXAMPLE	454
THREAD STATES	455
CREATING AND STARTING MANAGED THREADS	456
<i>ThreadStart Delegate</i>	456
<i>ParameterizedThreadStart Delegate: Passing Arguments To Threads</i>	457
BLOCKING A THREAD WITH <code>Thread.Sleep()</code>	458
BLOCKING A THREAD WITH <code>Thread.Join()</code>	459
FOREGROUND VS. BACKGROUND THREADS	461
Quick Review	463
CREATING THREADS WITH THE BACKGROUNDWORKER CLASS	464
Quick Review	467
THREAD POOLS	468
Quick Review	470
ASYNCHRONOUS METHOD CALLS	470
OBTAINING RESULTS FROM AN ASYNCHRONOUS METHOD CALL	472
PROVIDING A CALLBACK METHOD TO <code>BEGININVOKE()</code>	473
Quick Review	474
WHEN TO EMPLOY THE VARIOUS THREAD TYPES	474
SUMMARY	474
SKILL-BUILDING EXERCISES	476
SUGGESTED PROJECTS	477
SELF-TEST QUESTIONS	477
REFERENCES	477
NOTES	478

17 FILE I/O

INTRODUCTION	480
MANIPULATING DIRECTORIES AND FILES	480
FILES, DIRECTORIES, AND PATHS	481
MANIPULATING DIRECTORIES AND FILES	481
VERBATIM STRING LITERALS	482
Quick Review	483
SERIALIZING OBJECTS TO DISK	483
SERIALIZABLE ATTRIBUTE	484
SERIALIZING OBJECTS WITH <code>BINARYFORMATTER</code>	485
SERIALIZING OBJECTS WITH <code>XMLSERIALIZER</code>	487

Quick Review	489
Working With Text Files	489
SOME ISSUES YOU MUST CONSIDER	490
SAVING DOG DATA TO A TEXT FILE	490
Quick Review	492
Working With Binary Data	492
Quick Review	494
RANDOM ACCESS FILE I/O	494
TOWARDS AN APPROACH TO THE ADAPTER PROJECT	496
<i>START SMALL AND TAKE BABY STEPS</i>	496
OTHER PROJECT CONSIDERATIONS	497
<i>LOCKING A RECORD FOR UPDATES AND DELETES</i>	497
<i>MONITOR.ENTER()/MONITOR.EXIT() VS. THE LOCK KEYWORD</i>	497
<i>TRANSLATING LOW-LEVEL EXCEPTIONS INTO HIGHER-LEVEL EXCEPTION ABSTRACTIONS</i>	498
WHERE TO GO FROM HERE	498
COMPLETE DATAFILE ADAPTER PROJECT SOURCE CODE LISTING	498
Quick Review	513
Working With Log Files	513
Quick Review	516
Using FileDialogs	516
Quick Review	518
SUMMARY	518
Skill-Building Exercises	520
SUGGESTED PROJECTS	521
SELF-TEST QUESTIONS	521
REFERENCES	522
NOTES	523

PART V: NETWORK & DATABASE PROGRAMMING

18 NETWORK PROGRAMMING FUNDAMENTALS

INTRODUCTION	528
WHAT IS A COMPUTER NETWORK?	528
PURPOSE OF A NETWORK	528
THE ROLE OF NETWORK PROTOCOLS	528
<i>HOMOGENEOUS VS. HETEROGENEOUS NETWORKS</i>	529
<i>THE UNIFYING NETWORK PROTOCOLS: TCP/IP</i>	529
WHAT'S SO SPECIAL ABOUT THE INTERNET?	529
Quick Review	531
SERVERS & CLIENTS	531
SERVER HARDWARE AND SOFTWARE	531
CLIENT HARDWARE AND SOFTWARE	531
Quick Review	532
Application Distribution	532
PHYSICAL DISTRIBUTION ON ONE COMPUTER	532
<i>RUNNING MULTIPLE CLIENTS ON THE SAME COMPUTER</i>	533
<i>ADDRESSING THE LOCAL MACHINE</i>	533
PHYSICAL DISTRIBUTION ACROSS MULTIPLE COMPUTERS	534
Quick Review	534
MULTILAYERED APPLICATIONS	534

Logical Application Layers	534
Physical Tier Distribution	535
Quick Review	536
INTERNET NETWORKING PROTOCOLS: NUTS & BOLTS	536
THE INTERNET PROTOCOLS: TCP, UDP, AND IP	536
<i>The Application Layer</i>	537
<i>Transport Layer</i>	537
<i>Network Layer</i>	538
<i>Data Link And Physical Layers</i>	538
<i>Putting It All Together</i>	538
WHAT YOU NEED TO KNOW	538
Quick Review	539
SUMMARY	540
Skill-Building Exercises	541
SUGGESTED PROJECTS	541
SELF-TEST QUESTIONS	541
REFERENCES	542
NOTES	543

19 NETWORKED CLIENT-SERVER APPLICATIONS

INTRODUCTION	546
Building Client-Server Applications With .NET Remoting	546
THE THREE REQUIRED COMPONENTS OF A .NET REMOTING APPLICATION	546
A SIMPLE .NET REMOTING APPLICATION	547
SINGLECALL VS. SINGLETON	550
ACCESSING A REMOTE OBJECT VIA AN INTERFACE	551
USING CONFIGURATION FILES	553
PASSING OBJECTS BETWEEN CLIENT AND SERVER	555
Quick Review	560
Client-Server Applications With TcpListener And TcpClient	560
TCP/IP CLIENT-SERVER OVERVIEW	560
A SIMPLE CLIENT-SERVER APPLICATION	561
BUILDING A MULTITHREADED SERVER	564
LISTENING ON MULTIPLE IP ADDRESSES	565
SENDING OBJECTS BETWEEN CLIENT AND SERVER	569
Quick Review	573
SUMMARY	574
Skill-Building Exercises	575
SUGGESTED PROJECTS	576
SELF-TEST QUESTIONS	576
REFERENCES	577
NOTES	577

20 DATABASE ACCESS & MULTILAYERED APPLICATIONS

INTRODUCTION	580
WHAT YOU ARE GOING TO BUILD	580
PRELIMINARIES	581
INSTALLING SQL SERVER EXPRESS EDITION	581
INSTALLING MICROSOFT SQL SERVER MANAGEMENT STUDIO EXPRESS	583
INSTALLING MICROSOFT ENTERPRISE LIBRARY	584
A SIMPLE TEST APPLICATION	586

INTRODUCTION TO RELATIONAL DATABASES AND SQL	588
TERMINOLOGY	588
STRUCTURED QUERY LANGUAGE (SQL)	589
DATA DEFINITION LANGUAGE (DDL)	589
<i>CREATING THE EMPLOYEE TRAINING DATABASE.....</i>	<i>590</i>
<i>CREATING A DATABASE WITH A SCRIPT.....</i>	<i>591</i>
<i>CREATING TABLES.....</i>	<i>592</i>
<i>SQL SERVER DATABASE TYPES.....</i>	<i>593</i>
DATA MANIPULATION LANGUAGE (DML)	595
<i>USING THE INSERT COMMAND.....</i>	<i>595</i>
<i>USING THE SELECT COMMAND.....</i>	<i>596</i>
<i>USING THE UPDATE COMMAND.....</i>	<i>598</i>
<i>USING THE DELETE COMMAND.....</i>	<i>599</i>
QUICK REVIEW	599
COMPLEX SQL QUERIES	600
CREATING A RELATED TABLE WITH A FOREIGN KEY	600
INSERTING TEST DATA INTO THE tbl_EMPLOYEE_TRAINING TABLE	602
SELECTING DATA FROM MULTIPLE TABLES	603
<i>JOIN OPERATIONS.....</i>	<i>603</i>
TESTING THE CASCADE DELETE CONSTRAINT	605
QUICK REVIEW	605
THE SERVER APPLICATION	605
PROJECT FOLDER ORGANIZATION	606
USING MICROSOFT BUILD TO MANAGE AND BUILD THE PROJECT	606
FIRST ITERATION	609
<i>CODING THE EMPLOYEEVO AND EMPLOYEEDAO.....</i>	<i>610</i>
<i>APPLICATION CONFIGURATION FILE.....</i>	<i>620</i>
<i>CREATING TEST APPLICATION.....</i>	<i>620</i>
SECOND ITERATION	625
<i>TESTING THE CODE - SECOND ITERATION.....</i>	<i>638</i>
<i>REALITY CHECK.....</i>	<i>648</i>
THIRD ITERATION	648
THE CLIENT APPLICATION	654
THIRD ITERATION (CONTINUED)	654
FOURTH ITERATION	657
FIFTH ITERATION	663
SIXTH ITERATION	669
COMPILING AND RUNNING THE MODIFIED EMPLOYEE TRAINING CLIENT PROJECT	684
WHERE TO GO FROM HERE	686
SUMMARY	687
SKILL-BUILDING EXERCISES	687
SUGGESTED PROJECTS	688
SELF-TEST QUESTIONS	689
REFERENCES	689
NOTES	690

PART VI: ADVANCED TOPICS

21 OPERATOR OVERLOADING

INTRODUCTION	694
---------------------------	------------

OPERATOR OVERLOADING	694
OVERLOADABLE OPERATORS	694
Quick Review	695
OVERLOADING UNARY OPERATORS	695
+, - OPERATORS	696
! OPERATOR	697
TRUE, FALSE OPERATORS	698
++, - OPERATORS	700
Quick Review	702
OVERLOADING BINARY OPERATORS	702
+, - OPERATORS	702
*, / OPERATORS	704
&, OPERATORS	707
Quick Review	710
OVERLOADING COMPARISON OPERATORS	710
==, !=, <, >, <=, >= OPERATORS	710
Quick Review	713
Implicit And Explicit CAST OPERATORS	713
Implicit vs. Explicit CAST	714
OVERLOADED CAST OPERATORS EXAMPLE	714
Quick Review	717
THE ASSIGNMENT OPERATORS: THINGS YOU GET FOR FREE	717
Quick Review	717
SUMMARY	718
Skill-Building EXERCISES	718
SUGGESTED PROJECTS	718
SELF-TEST QUESTIONS	719
REFERENCES	719
NOTES	720

22 Well-Behaved Objects

INTRODUCTION	722
OBJECT BEHAVIOR DEFINED	722
FUNDAMENTAL BEHAVIOR	722
COPY/ASSIGNMENT BEHAVIOR	723
EQUALITY BEHAVIOR	723
COMPARISON/ORDERING BEHAVIOR	723
SEVEN OBJECT USAGE SCENARIOS	723
FUNDAMENTAL BEHAVIOR	724
OBJECT CREATION – CONSTRUCTORS	724
<i>Default Constructor</i>	724
<i>Private Constructors</i>	724
<i>Overloaded Constructors</i>	725
MEMBER ACCESSIBILITY	725
<i>Horizontal Member Access</i>	725
<i>Vertical Member Access</i>	725
OVERRIDING OBJECT.TOSTRING()	726
STATIC VS. INSTANCE MEMBERS	726
SERIALIZATION	726
<i>Custom Serialization Example</i>	726
Quick Review	733
Copy/Assignment BEHAVIOR	733
VALUE OBJECT VS. REFERENCE OBJECT ASSIGNMENT	734

<i>Rule Of Thumb – FAVOR THE CLASS CONSTRUCT FOR COMPLEX TYPES</i>	734
SHALLOW COPY VS. DEEP COPY	734
COPY CONSTRUCTORS	735
SYSTEM.ICLONEABLE VS. OBJECT.MEMBERWISECLONE()	738
QUICK REVIEW	740
EQUALITY BEHAVIOR	741
REFERENCE EQUALITY VS. VALUE EQUALITY	741
RULES FOR OVERRIDING THE OBJECT.EQUALS() METHOD	741
OVERRIDING THE OBJECT.GETHASHCODE() METHOD	742
<i>BLOCH'S HASH CODE GENERATION ALGORITHM</i>	743
<i>ASHMORE'S HASH CODE GENERATION ALGORITHM</i>	743
OVERRIDING OBJECT.EQUALS() AND OBJECT.GETHASHCODE() METHODS IN THE PERSONVO CLASS	743
QUICK REVIEW	746
COMPARISON/ORDERING BEHAVIOR	747
IMPLEMENTING SYSTEM.ICOMPARABLE<T>	747
<i>RULES FOR IMPLEMENTING THE COMPARETO(OTHER) METHOD</i>	750
EXTENDING THE COMPARER<T> CLASS	751
QUICK REVIEW	753
SUMMARY	753
SKILL-BUILDING EXERCISES	753
SUGGESTED PROJECTS	753
SELF-TEST QUESTIONS	754
REFERENCES	754
NOTES	755

23 THREE DESIGN PRINCIPLES

INTRODUCTION	758
THE PREFERRED CHARACTERISTICS OF AN OBJECT-ORIENTED ARCHITECTURE	758
EASY TO UNDERSTAND: HOW DOES THIS THING WORK?	758
EASY TO REASON ABOUT: WHAT ARE THE EFFECTS OF CHANGE?	758
EASY TO EXTEND: WHERE DO I ADD FUNCTIONALITY?	759
THE LISKOV SUBSTITUTION PRINCIPLE & DESIGN BY CONTRACT	759
REASONING ABOUT THE BEHAVIOR OF SUPERTYPES AND SUBTYPES	759
<i>RELATIONSHIP BETWEEN THE LSP AND DbC</i>	759
<i>THE COMMON GOAL OF THE LSP AND DbC</i>	760
<i>C# SUPPORT FOR THE LSP AND DbC</i>	760
DESIGNING WITH THE LSP/DbC IN MIND	760
<i>CLASS DECLARATIONS VIEWED AS BEHAVIOR SPECIFICATIONS</i>	760
QUICK REVIEW	760
PRECONDITIONS, POSTCONDITIONS, AND CLASS INVARIANTS	761
CLASS INVARIANT	761
PRECONDITION	761
POSTCONDITION	761
AN EXAMPLE	761
<i>A NOTE ON USING THE DEBUG.ASSERT() METHOD TO ENFORCE PRE- AND POSTCONDITIONS</i>	763
USING INCREMENTER AS A BASE CLASS	763
<i>CHANGING THE PRECONDITIONS OF DERIVED CLASS METHODS</i>	765
CHANGING THE POSTCONDITIONS OF DERIVED CLASS METHODS	769
SPECIAL CASES OF PRECONDITIONS AND POSTCONDITIONS	770
<i>METHOD ARGUMENT TYPES</i>	770
<i>METHOD RETURN TYPES</i>	773
THREE RULES OF THE SUBSTITUTION PRINCIPLE	773
<i>SIGNATURE RULE</i>	773

<i>Methods Rule</i>	773
<i>Properties Rule</i>	774
Quick Review	774
The Open-Closed Principle	774
Achieving The Open-Closed Principle	774
An OCP Example	775
Quick Review	780
The Dependency Inversion Principle	780
Characteristics Of Bad Software Architecture	780
Characteristics Of Good Software Architecture	781
Selecting The Right Abstractions Takes Experience	781
Quick Review	781
Code Contracts	782
Downloading And Installing Code Contracts Extensions	782
Code Contracts Example	783
Quick Review	784
Terms And Definitions	785
Summary	786
Skill-Building Exercises	787
Suggested Projects	787
Self-Test Questions	787
References	788
Notes	789

24 INHERITANCE, COMPOSITION, INTERFACES, POLYMORPHISM

Introduction	792
Inheritance Vs. Composition: The Great Debate	792
What's The End Game?	793
<i>Flexible Application Architectures</i>	793
<i>Modularity And Reliability</i>	793
<i>Architectural Stability Via Managed Dependencies</i>	794
Knowing When To Accept A Design That's Good Enough	794
Quick Review	794
Inheritance-Based Design	794
Three Good Reasons To Use Inheritance	794
<i>As A Means To Reason About Code Behavior</i>	795
<i>To Gain A Measure Of Code Reuse</i>	795
<i>To Facilitate Incremental Development</i>	795
Forms Of Inheritance: Meyer's Inheritance Taxonomy	795
Coad's Inheritance Criteria	797
Person - Employee Example Revisited	797
Quick Review	798
The Role Of Interfaces	798
Reducing Or Limiting Intermodule Dependencies	799
Modeling Dominant, Collateral, And Dynamic Roles	799
<i>Dominant Roles</i>	799
<i>Collateral Roles</i>	800
<i>Dynamic Roles</i>	800
Quick Review	800
Applied Polymorphism	800
Quick Review	801
Composition-Based Design As A Force Multiplier	801
Two Types Of Aggregation	801

Polymorphic Containment	801
An Extended Example	802
Quick Review	810
Summary	810
Skill-Building Exercises	811
Suggested Projects	812
Self-Test Questions	813
References	813
Notes	814

25 Helpful Design Patterns

Introduction	816
Software Design Patterns And How They Came To Be	816
What Exactly Is A Software Design Pattern?	816
Origins	816
Pattern Specification	817
Applying Software Design Patterns	817
Quick Review	818
The Singleton Pattern	818
Quick Review	822
The Factory Pattern	822
The Dynamic Factory	822
Advantages Of The Dynamic Factory Pattern	824
Quick Review	824
The Model-View-Controller Pattern	825
Quick Review	827
The Command Pattern	827
Quick Review	835
A Comprehensive Pattern-Based Example	835
Complete Code Listing	835
<i>Com.PulpFreePress.Exceptions</i>	835
<i>Com.PulpFreePress.Common</i>	836
<i>Com.PulpFreePress.Utils</i>	843
<i>Com.PulpFreePress.Commands</i>	847
<i>Com.PulpFreePress.Model</i>	851
<i>Com.PulpFreePress.View</i>	853
<i>Com.PulpFreePress.Controller</i>	863
MSBuild Project File	864
Running The Application	866
Summary	867
Skill-Building Exercises	867
Suggested Projects	868
Self-Test Questions	868
References	869
Notes	869

Appendix A: Helpful Checklists And Tables

Project-Approach Strategy Check-off List	873
Development Cycle	874
Final Project Review Checklist	874

Appendix B: ASCII Table

ASCII Table	875
--------------------------	------------

Appendix C: Identifier Naming: Writing Self-Commenting Code

Identifier Naming: Writing Self-Commenting Code	879
Benefits of Self-Commenting Code	879
Coding Convention	879
<i>Type Names (Classes, Structures, Delegates, Enumerations)</i>	<i>879</i>
<i>Constant Names</i>	<i>880</i>
<i>Variable Names</i>	<i>880</i>
<i>Method Names</i>	<i>880</i>
<i>Property Names</i>	<i>881</i>