



Contents

Preface

xxiii

Before You Begin

xxxvii

I	Introduction to Computers, the Internet and Visual C#	I
1.1	Introduction	2
1.2	Computers and the Internet in Industry and Research	2
1.3	Hardware and Software	5
1.3.1	Moore's Law	5
1.3.2	Computer Organization	6
1.4	Data Hierarchy	7
1.5	Machine Languages, Assembly Languages and High-Level Languages	10
1.6	Object Technology	11
1.7	Internet and World Wide Web	13
1.8	C#	15
1.8.1	Object-Oriented Programming	16
1.8.2	Event-Driven Programming	16
1.8.3	Visual Programming	16
1.8.4	An International Standard	16
1.8.5	C# on Other Platforms	17
1.8.6	Internet and Web Programming	17
1.8.7	Asynchronous Programming with async and await	17
1.8.8	Other Key Programming Languages	18
1.9	Microsoft's .NET	20
1.9.1	.NET Framework	20
1.9.2	Common Language Runtime	20
1.9.3	Platform Independence	21
1.9.4	Language Interoperability	21
1.10	Microsoft's Windows® Operating System	21
1.11	Visual Studio Integrated Development Environment	23
1.12	Painter Test-Drive in Visual Studio Community	23

2	Introduction to Visual Studio and Visual Programming	33
2.1	Introduction	34
2.2	Overview of the Visual Studio Community 2015 IDE	34
2.2.1	Introduction to Visual Studio Community 2015	34
2.2.2	Visual Studio Themes	35
2.2.3	Links on the Start Page	35
2.2.4	Creating a New Project	36
2.2.5	New Project Dialog and Project Templates	37
2.2.6	Forms and Controls	38
2.3	Menu Bar and Toolbar	39
2.4	Navigating the Visual Studio IDE	42
2.4.1	Solution Explorer	43
2.4.2	Toolbox	44
2.4.3	Properties Window	44
2.5	Help Menu and Context-Sensitive Help	46
2.6	Visual Programming: Creating a Simple App that Displays Text and an Image	47
2.7	Wrap-Up	56
2.8	Web Resources	57
3	Introduction to C# App Programming	65
3.1	Introduction	66
3.2	Simple App: Displaying a Line of Text	66
3.2.1	Comments	67
3.2.2	<code>using</code> Directive	68
3.2.3	Blank Lines and Whitespace	69
3.2.4	Class Declaration	69
3.2.5	Main Method	71
3.2.6	Displaying a Line of Text	71
3.2.7	Matching Left ({) and Right (}) Braces	72
3.3	Creating a Simple App in Visual Studio	72
3.3.1	Creating the Console App	72
3.3.2	Changing the Name of the App File	74
3.3.3	Writing Code and Using <i>IntelliSense</i>	74
3.3.4	Compiling and Running the App	76
3.3.5	Errors, Error Messages and the Error List Window	77
3.4	Modifying Your Simple C# App	77
3.4.1	Displaying a Single Line of Text with Multiple Statements	78
3.4.2	Displaying Multiple Lines of Text with a Single Statement	78
3.5	String Interpolation	80
3.6	Another C# App: Adding Integers	81
3.6.1	Declaring the <code>int</code> Variable <code>number1</code>	82
3.6.2	Declaring Variables <code>number2</code> and <code>sum</code>	83

3.6.3	Prompting the User for Input	83
3.6.4	Reading a Value into Variable <code>number1</code>	83
3.6.5	Prompting the User for Input and Reading a Value into <code>number2</code>	84
3.6.6	Summing <code>number1</code> and <code>number2</code>	84
3.6.7	Displaying the sum with <code>string</code> Interpolation	85
3.6.8	Performing Calculations in Output Statements	85
3.7	Memory Concepts	85
3.8	Arithmetic	86
3.8.1	Arithmetic Expressions in Straight-Line Form	87
3.8.2	Parentheses for Grouping Subexpressions	87
3.8.3	Rules of Operator Precedence	87
3.8.4	Sample Algebraic and C# Expressions	88
3.8.5	Redundant Parentheses	89
3.9	Decision Making: Equality and Relational Operators	89
3.10	Wrap-Up	94

4 Introduction to Classes, Objects, Methods and strings 106

4.1	Introduction	107
4.2	Test-Driving an Account Class	108
4.2.1	Instantiating an Object—Keyword <code>new</code>	108
4.2.2	Calling Class <code>Account</code> 's <code>GetName</code> Method	109
4.2.3	Inputting a Name from the User	109
4.2.4	Calling Class <code>Account</code> 's <code>SetName</code> Method	110
4.3	<code>Account</code> Class with an Instance Variable and <i>Set</i> and <i>Get</i> Methods	110
4.3.1	<code>Account</code> Class Declaration	110
4.3.2	Keyword <code>class</code> and the Class Body	111
4.3.3	Instance Variable <code>name</code> of Type <code>string</code>	111
4.3.4	<code>SetName</code> Method	112
4.3.5	<code>GetName</code> Method	114
4.3.6	Access Modifiers <code>private</code> and <code>public</code>	114
4.3.7	<code>Account</code> UML Class Diagram	115
4.4	Creating, Compiling and Running a Visual C# Project with Two Classes	116
4.5	Software Engineering with <i>Set</i> and <i>Get</i> Methods	117
4.6	<code>Account</code> Class with a Property Rather Than <i>Set</i> and <i>Get</i> Methods	118
4.6.1	Class <code>AccountTest</code> Using <code>Account</code> 's <code>Name</code> Property	118
4.6.2	<code>Account</code> Class with an Instance Variable and a Property	120
4.6.3	<code>Account</code> UML Class Diagram with a Property	122
4.7	Auto-Implemented Properties	122
4.8	<code>Account</code> Class: Initializing Objects with Constructors	123
4.8.1	Declaring an <code>Account</code> Constructor for Custom Object Initialization	123
4.8.2	Class <code>AccountTest</code> : Initializing <code>Account</code> Objects When They're Created	124

4.9	Account Class with a Balance; Processing Monetary Amounts	126
4.9.1	Account Class with a decimal balance Instance Variable	126
4.9.2	AccountTest Class That Uses Account Objects with Balances	129
4.10	Wrap-Up	133

5 Algorithm Development and Control Statements: Part I 142

5.1	Introduction	143
5.2	Algorithms	144
5.3	Pseudocode	144
5.4	Control Structures	145
5.4.1	Sequence Structure	145
5.4.2	Selection Statements	146
5.4.3	Iteration Statements	147
5.4.4	Summary of Control Statements	147
5.5	if Single-Selection Statement	147
5.6	if...else Double-Selection Statement	148
5.6.1	Nested if...else Statements	149
5.6.2	Dangling-else Problem	151
5.6.3	Blocks	151
5.6.4	Conditional Operator (?:)	152
5.7	Student Class: Nested if...else Statements	153
5.8	while Iteration Statement	156
5.9	Formulating Algorithms: Counter-Controlled Iteration	157
5.9.1	Pseudocode Algorithm with Counter-Controlled Iteration	157
5.9.2	Implementing Counter-Controlled Iteration	158
5.9.3	Integer Division and Truncation	160
5.10	Formulating Algorithms: Sentinel-Controlled Iteration	161
5.10.1	Top-Down, Stepwise Refinement: The Top and First Refinement	161
5.10.2	Second Refinement	162
5.10.3	Implementing Sentinel-Controlled Iteration	164
5.10.4	Program Logic for Sentinel-Controlled Iteration	165
5.10.5	Braces in a while Statement	166
5.10.6	Converting Between Simple Types Explicitly and Implicitly	166
5.10.7	Formatting Floating-Point Numbers	167
5.11	Formulating Algorithms: Nested Control Statements	168
5.11.1	Problem Statement	168
5.11.2	Top-Down, Stepwise Refinement: Pseudocode Representation of the Top	169
5.11.3	Top-Down, Stepwise Refinement: First Refinement	169
5.11.4	Top-Down, Stepwise Refinement: Second Refinement	169
5.11.5	Complete Second Refinement of the Pseudocode	170
5.11.6	App That Implements the Pseudocode Algorithm	171
5.12	Compound Assignment Operators	173

5.13	Increment and Decrement Operators	173
5.13.1	Prefix Increment vs. Postfix Increment	174
5.13.2	Simplifying Increment Statements	175
5.13.3	Operator Precedence and Associativity	176
5.14	Simple Types	176
5.15	Wrap-Up	177

6 Control Statements: Part 2 **192**

6.1	Introduction	193
6.2	Essentials of Counter-Controlled Iteration	193
6.3	for Iteration Statement	195
6.3.1	A Closer Look at the for Statement's Header	196
6.3.2	General Format of a for Statement	196
6.3.3	Scope of a for Statement's Control Variable	196
6.3.4	Expressions in a for Statement's Header Are Optional	197
6.3.5	Placing Arithmetic Expressions in a for Statement's Header	197
6.3.6	Using a for Statement's Control Variable in the Statement's Body	198
6.3.7	UML Activity Diagram for the for Statement	198
6.4	Examples Using the for Statement	198
6.5	App: Summing Even Integers	199
6.6	App: Compound-Interest Calculations	200
6.6.1	Performing the Interest Calculations with Math Method pow	201
6.6.2	Formatting with Field Widths and Alignment	202
6.6.3	Caution: Do Not Use float or double for Monetary Amounts	203
6.7	do...while Iteration Statement	204
6.8	switch Multiple-Selection Statement	205
6.8.1	Using a switch Statement to Count A, B, C, D and F Grades	205
6.8.2	switch Statement UML Activity Diagram	209
6.8.3	Notes on the Expression in Each case of a switch	210
6.9	Class AutoPolicy Case Study: strings in switch Statements	211
6.10	break and continue Statements	213
6.10.1	break Statement	213
6.10.2	continue Statement	214
6.11	Logical Operators	215
6.11.1	Conditional AND (&&) Operator	216
6.11.2	Conditional OR () Operator	216
6.11.3	Short-Circuit Evaluation of Complex Conditions	217
6.11.4	Boolean Logical AND (&) and Boolean Logical OR () Operators	217
6.11.5	Boolean Logical Exclusive OR (^)	218
6.11.6	Logical Negation (!) Operator	218
6.11.7	Logical Operators Example	219
6.12	Structured-Programming Summary	221
6.13	Wrap-Up	226

7	Methods: A Deeper Look	237
7.1	Introduction	238
7.2	Packaging Code in C#	239
7.2.1	Modularizing Programs	239
7.2.2	Calling Methods	240
7.3	static Methods, static Variables and Class Math	240
7.3.1	Math Class Methods	241
7.3.2	Math Class Constants PI and E	242
7.3.3	Why Is Main Declared static?	242
7.3.4	Additional Comments About Main	243
7.4	Methods with Multiple Parameters	243
7.4.1	Keyword static	245
7.4.2	Method Maximum	245
7.4.3	Assembling strings with Concatenation	245
7.4.4	Breaking Apart Large string Literals	246
7.4.5	When to Declare Variables as Fields	247
7.4.6	Implementing Method Maximum by Reusing Method Math.Max	247
7.5	Notes on Using Methods	247
7.6	Argument Promotion and Casting	248
7.6.1	Promotion Rules	249
7.6.2	Sometimes Explicit Casts Are Required	249
7.7	The .NET Framework Class Library	250
7.8	Case Study: Random-Number Generation	252
7.8.1	Creating an Object of Type Random	252
7.8.2	Generating a Random Integer	252
7.8.3	Scaling the Random-Number Range	253
7.8.4	Shifting Random-Number Range	253
7.8.5	Combining Shifting and Scaling	253
7.8.6	Rolling a Six-Sided Die	253
7.8.7	Scaling and Shifting Random Numbers	256
7.8.8	Repeatability for Testing and Debugging	257
7.9	Case Study: A Game of Chance; Introducing Enumerations	257
7.9.1	Method RollDice	260
7.9.2	Method Main's Local Variables	260
7.9.3	enum Type Status	261
7.9.4	The First Roll	261
7.9.5	enum Type DiceNames	261
7.9.6	Underlying Type of an enum	261
7.9.7	Comparing Integers and enum Constants	262
7.10	Scope of Declarations	262
7.11	Method-Call Stack and Activation Records	265
7.11.1	Method-Call Stack	265
7.11.2	Stack Frames	265
7.11.3	Local Variables and Stack Frames	266
7.11.4	Stack Overflow	266
7.11.5	Method-Call Stack in Action	266

7.12	Method Overloading	269
7.12.1	Declaring Overloaded Methods	269
7.12.2	Distinguishing Between Overloaded Methods	270
7.12.3	Return Types of Overloaded Methods	271
7.13	Optional Parameters	271
7.14	Named Parameters	272
7.15	C# 6 Expression-Bodied Methods and Properties	273
7.16	Recursion	274
7.16.1	Base Cases and Recursive Calls	274
7.16.2	Recursive Factorial Calculations	274
7.16.3	Implementing Factorial Recursively	275
7.17	Value Types vs. Reference Types	277
7.18	Passing Arguments By Value and By Reference	278
7.18.1	ref and out Parameters	279
7.18.2	Demonstrating ref, out and Value Parameters	280
7.19	Wrap-Up	282

8 Arrays; Introduction to Exception Handling 299

8.1	Introduction	300
8.2	Arrays	301
8.3	Declaring and Creating Arrays	302
8.4	Examples Using Arrays	303
8.4.1	Creating and Initializing an Array	303
8.4.2	Using an Array_INITIALIZER	304
8.4.3	Calculating a Value to Store in Each Array Element	305
8.4.4	Summing the Elements of an Array	307
8.4.5	Iterating Through Arrays with foreach	307
8.4.6	Using Bar Charts to Display Array Data Graphically; Introducing Type Inference with var	309
8.4.7	Using the Elements of an Array as Counters	312
8.5	Using Arrays to Analyze Survey Results; Introduction to Exception Handling	313
8.5.1	Summarizing the Results	314
8.5.2	Exception Handling: Processing the Incorrect Response	315
8.5.3	The try Statement	315
8.5.4	Executing the catch Block	315
8.5.5	Message Property of the Exception Parameter	316
8.6	Case Study: Card Shuffling and Dealing Simulation	316
8.6.1	Class Card and Getter-Only Auto-Implemented Properties	316
8.6.2	Class DeckOfCards	317
8.6.3	Shuffling and Dealing Cards	320
8.7	Passing Arrays and Array Elements to Methods	321
8.8	Case Study: GradeBook Using an Array to Store Grades	323
8.9	Multidimensional Arrays	329
8.9.1	Rectangular Arrays	329

8.9.2	Jagged Arrays	330
8.9.3	Two-Dimensional Array Example: Displaying Element Values	331
8.10	Case Study: GradeBook Using a Rectangular Array	334
8.11	Variable-Length Argument Lists	340
8.12	Using Command-Line Arguments	342
8.13	(Optional) Passing Arrays by Value and by Reference	344
8.14	Wrap-Up	348

9 Introduction to LINQ and the List Collection 370

9.1	Introduction	371
9.2	Querying an Array of <code>int</code> Values Using LINQ	372
9.2.1	The <code>from</code> Clause	374
9.2.2	The <code>where</code> Clause	375
9.2.3	The <code>select</code> Clause	375
9.2.4	Iterating Through the Results of the LINQ Query	375
9.2.5	The <code>orderby</code> Clause	375
9.2.6	Interface <code>IEnumerable<T></code>	376
9.3	Querying an Array of <code>Employee</code> Objects Using LINQ	376
9.3.1	Accessing the Properties of a LINQ Query's Range Variable	380
9.3.2	Sorting a LINQ Query's Results by Multiple Properties	380
9.3.3	<code>Any</code> , <code>First</code> and <code>Count</code> Extension Methods	380
9.3.4	Selecting a Property of an Object	380
9.3.5	Creating New Types in the <code>select</code> Clause of a LINQ Query	380
9.4	Introduction to Collections	381
9.4.1	<code>List<T></code> Collection	381
9.4.2	Dynamically Resizing a <code>List<T></code> Collection	382
9.5	Querying the Generic <code>List</code> Collection Using LINQ	386
9.5.1	The <code>let</code> Clause	388
9.5.2	Deferred Execution	388
9.5.3	Extension Methods <code>ToArray</code> and <code>ToList</code>	388
9.5.4	Collection Initializers	388
9.6	Wrap-Up	389
9.7	Deitel LINQ Resource Center	389

10 Classes and Objects: A Deeper Look 394

10.1	Introduction	395
10.2	Time Class Case Study; Throwing Exceptions	395
10.2.1	Time1 Class Declaration	396
10.2.2	Using Class <code>Time1</code>	397
10.3	Controlling Access to Members	399
10.4	Referring to the Current Object's Members with the <code>this</code> Reference	400
10.5	Time Class Case Study: Overloaded Constructors	402
10.5.1	Class <code>Time2</code> with Overloaded Constructors	402
10.5.2	Using Class <code>Time2</code> 's Overloaded Constructors	406

10.6	Default and Parameterless Constructors	408
10.7	Composition	409
10.7.1	Class Date	409
10.7.2	Class Employee	411
10.7.3	Class EmployeeTest	412
10.8	Garbage Collection and Destructors	413
10.9	static Class Members	413
10.10	readonly Instance Variables	417
10.11	Class View and Object Browser	418
10.11.1	Using the Class View Window	418
10.11.2	Using the Object Browser	419
10.12	Object Initializers	420
10.13	Operator Overloading; Introducing struct	420
10.13.1	Creating Value Types with struct	421
10.13.2	Value Type ComplexNumber	421
10.13.3	Class ComplexTest	423
10.14	Time Class Case Study: Extension Methods	424
10.15	Wrap-Up	427

I I Object-Oriented Programming: Inheritance 435

11.1	Introduction	436
11.2	Base Classes and Derived Classes	437
11.3	protected Members	439
11.4	Relationship between Base Classes and Derived Classes	440
11.4.1	Creating and Using a CommissionEmployee Class	441
11.4.2	Creating a BasePlusCommissionEmployee Class without Using Inheritance	445
11.4.3	Creating a CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy	450
11.4.4	CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using protected Instance Variables	453
11.4.5	CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using private Instance Variables	456
11.5	Constructors in Derived Classes	460
11.6	Software Engineering with Inheritance	460
11.7	Class object	461
11.8	Wrap-Up	462

I 2 OOP: Polymorphism and Interfaces 468

12.1	Introduction	469
12.2	Polymorphism Examples	471
12.3	Demonstrating Polymorphic Behavior	472
12.4	Abstract Classes and Methods	475
12.5	Case Study: Payroll System Using Polymorphism	477
12.5.1	Creating Abstract Base Class Employee	478

12.5.2	Creating Concrete Derived Class <code>SalariedEmployee</code>	480
12.5.3	Creating Concrete Derived Class <code>HourlyEmployee</code>	482
12.5.4	Creating Concrete Derived Class <code>CommissionEmployee</code>	483
12.5.5	Creating Indirect Concrete Derived Class <code>BasePlusCommissionEmployee</code>	485
12.5.6	Polymorphic Processing, Operator <code>is</code> and Downcasting	486
12.5.7	Summary of the Allowed Assignments Between Base-Class and Derived-Class Variables	491
12.6	<code>sealed</code> Methods and Classes	492
12.7	Case Study: Creating and Using Interfaces	493
12.7.1	Developing an <code>IPayable</code> Hierarchy	494
12.7.2	Declaring Interface <code>IPayable</code>	496
12.7.3	Creating Class <code>Invoice</code>	496
12.7.4	Modifying Class <code>Employee</code> to Implement Interface <code>IPayable</code>	498
12.7.5	Using Interface <code>IPayable</code> to Process Invoices and <code>Employees</code> Polymorphically	499
12.7.6	Common Interfaces of the .NET Framework Class Library	501
12.8	Wrap-Up	502

13 Exception Handling: A Deeper Look **507**

13.1	Introduction	508
13.2	Example: Divide by Zero without Exception Handling	509
13.2.1	Dividing By Zero	510
13.2.2	Enter a Non-Numeric Denominator	511
13.2.3	Unhandled Exceptions Terminate the App	512
13.3	Example: Handling <code>DivideByZeroExceptions</code> and <code>FormatExceptions</code>	512
13.3.1	Enclosing Code in a <code>try</code> Block	514
13.3.2	Catching Exceptions	514
13.3.3	Uncaught Exceptions	515
13.3.4	Termination Model of Exception Handling	516
13.3.5	Flow of Control When Exceptions Occur	516
13.4	.NET Exception Hierarchy	517
13.4.1	Class <code>SystemException</code>	517
13.4.2	Which Exceptions Might a Method Throw?	518
13.5	<code>finally</code> Block	519
13.5.1	Moving Resource-Release Code to a <code>finally</code> Block	519
13.5.2	Demonstrating the <code>finally</code> Block	520
13.5.3	Throwing Exceptions Using the <code>throw</code> Statement	524
13.5.4	Rethrowing Exceptions	524
13.5.5	Returning After a <code>finally</code> Block	525
13.6	The <code>using</code> Statement	526
13.7	Exception Properties	527
13.7.1	Property <code>InnerException</code>	527
13.7.2	Other Exception Properties	528
13.7.3	Demonstrating Exception Properties and Stack Unwinding	528

13.7.4	Throwing an Exception with an InnerException	530
13.7.5	Displaying Information About the Exception	531
13.8	User-Defined Exception Classes	531
13.9	Checking for null References; Introducing C# 6's ?. Operator	535
13.9.1	Null-Conditional Operator (?.)	535
13.9.2	Revisiting Operators is and as	536
13.9.3	Nullable Types	536
13.9.4	Null Coalescing Operator (??)	537
13.10	Exception Filters and the C# 6 when Clause	537
13.11	Wrap-Up	538

14 Graphical User Interfaces with Windows Forms: Part 1

544

14.1	Introduction	545
14.2	Windows Forms	546
14.3	Event Handling	548
14.3.1	A Simple Event-Driven GUI	549
14.3.2	Auto-Generated GUI Code	551
14.3.3	Delegates and the Event-Handling Mechanism	553
14.3.4	Another Way to Create Event Handlers	554
14.3.5	Locating Event Information	555
14.4	Control Properties and Layout	556
14.4.1	Anchoring and Docking	557
14.4.2	Using Visual Studio To Edit a GUI's Layout	559
14.5	Labels, TextBoxes and Buttons	560
14.6	GroupBoxes and Panels	563
14.7	CheckBoxes and RadioButtons	566
14.7.1	CheckBoxes	566
14.7.2	Combining Font Styles with Bitwise Operators	568
14.7.3	RadioButtons	569
14.8	PictureBoxes	574
14.9	ToolTips	577
14.10	NumericUpDown Control	578
14.11	Mouse-Event Handling	581
14.12	Keyboard-Event Handling	583
14.13	Wrap-Up	587

15 Graphical User Interfaces with Windows Forms: Part 2

597

15.1	Introduction	598
15.2	Menus	598
15.3	MonthCalendar Control	608
15.4	DateTimePicker Control	609

15.5	LinkLabel Control	612
15.6	ListBox Control	615
15.7	CheckedListBox Control	620
15.8	ComboBox Control	623
15.9	TreeView Control	627
15.10	ListView Control	633
15.11	TabControl Control	639
15.12	Multiple Document Interface (MDI) Windows	643
15.13	Visual Inheritance	651
15.14	User-Defined Controls	656
15.15	Wrap-Up	659

16 Strings and Characters: A Deeper Look **667**

16.1	Introduction	668
16.2	Fundamentals of Characters and Strings	669
16.3	string Constructors	670
16.4	string Indexer, Length Property and CopyTo Method	671
16.5	Comparing strings	672
16.6	Locating Characters and Substrings in strings	676
16.7	Extracting Substrings from strings	679
16.8	Concatenating strings	680
16.9	Miscellaneous string Methods	680
16.10	Class StringBuilder	682
16.11	Length and Capacity Properties, EnsureCapacity Method and Indexer of Class StringBuilder	683
16.12	Append and AppendFormat Methods of Class StringBuilder	685
16.13	Insert, Remove and Replace Methods of Class StringBuilder	687
16.14	Char Methods	690
16.15	Introduction to Regular Expressions (Online)	692
16.16	Wrap-Up	692

17 Files and Streams **699**

17.1	Introduction	700
17.2	Files and Streams	700
17.3	Creating a Sequential-Access Text File	701
17.4	Reading Data from a Sequential-Access Text File	710
17.5	Case Study: Credit-Inquiry Program	714
17.6	Serialization	719
17.7	Creating a Sequential-Access File Using Object Serialization	720
17.8	Reading and Deserializing Data from a Binary File	724
17.9	Classes File and Directory	727
	17.9.1 Demonstrating Classes File and Directory	728
	17.9.2 Searching Directories with LINQ	731
17.10	Wrap-Up	735

18 Searching and Sorting 742

18.1	Introduction	743
18.2	Searching Algorithms	744
18.2.1	Linear Search	744
18.2.2	Binary Search	748
18.3	Sorting Algorithms	752
18.3.1	Selection Sort	753
18.3.2	Insertion Sort	756
18.3.3	Merge Sort	760
18.4	Summary of the Efficiency of Searching and Sorting Algorithms	766
18.5	Wrap-Up	767

19 Custom Linked Data Structures 772

19.1	Introduction	773
19.2	Simple-Type structs, Boxing and Unboxing	773
19.3	Self-Referential Classes	774
19.4	Linked Lists	775
19.5	Stacks	788
19.6	Queues	792
19.7	Trees	795
19.7.1	Binary Search Tree of Integer Values	796
19.7.2	Binary Search Tree of <code>IComparable</code> Objects	803
19.8	Wrap-Up	809

20 Generics 815

20.1	Introduction	816
20.2	Motivation for Generic Methods	817
20.3	Generic-Method Implementation	819
20.4	Type Constraints	822
20.4.1	<code>IComparable<T></code> Interface	822
20.4.2	Specifying Type Constraints	822
20.5	Overloading Generic Methods	825
20.6	Generic Classes	825
20.7	Wrap-Up	835

21 Generic Collections; Functional Programming with LINQ/PLINQ 841

21.1	Introduction	842
21.2	Collections Overview	844
21.3	Class <code>Array</code> and Enumerators	846
21.3.1	C# 6 using <code>static</code> Directive	848
21.3.2	Class <code>UsingArray</code> 's <code>static</code> Fields	849
21.3.3	<code>Array</code> Method <code>Sort</code>	849

21.3.4	Array Method Copy	849
21.3.5	Array Method BinarySearch	849
21.3.6	Array Method GetEnumerator and Interface IEnumerator	849
21.3.7	Iterating Over a Collection with foreach	850
21.3.8	Array Methods Clear, IndexOf, LastIndexOf and Reverse	850
21.4	Dictionary Collections	850
21.4.1	Dictionary Fundamentals	851
21.4.2	Using the SortedDictionary Collection	852
21.5	Generic LinkedList Collection	856
21.6	C# 6 Null Conditional Operator ?[]	860
21.7	C# 6 Dictionary Initializers and Collection Initializers	861
21.8	Delegates	861
21.8.1	Declaring a Delegate Type	863
21.8.2	Declaring a Delegate Variable	863
21.8.3	Delegate Parameters	864
21.8.4	Passing a Method Name Directly to a Delegate Parameter	864
21.9	Lambda Expressions	864
21.9.1	Expression Lambdas	866
21.9.2	Assigning Lambdas to Delegate Variables	867
21.9.3	Explicitly Typed Lambda Parameters	867
21.9.4	Statement Lambdas	867
21.10	Introduction to Functional Programming	867
21.11	Functional Programming with LINQ Method-Call Syntax and Lambdas	869
21.11.1	LINQ Extension Methods Min, Max, Sum and Average	872
21.11.2	Aggregate Extension Method for Reduction Operations	872
21.11.3	The Where Extension Method for Filtering Operations	874
21.11.4	Select Extension Method for Mapping Operations	875
21.12	PLINQ: Improving LINQ to Objects Performance with Multicore	875
21.13	(Optional) Covariance and Contravariance for Generic Types	879
21.14	Wrap-Up	881

22 Databases and LINQ

893

22.1	Introduction	894
22.2	Relational Databases	895
22.3	A Books Database	896
22.4	LINQ to Entities and the ADO.NET Entity Framework	900
22.5	Querying a Database with LINQ	901
22.5.1	Creating the ADO.NET Entity Data Model Class Library	903
22.5.2	Creating a Windows Forms Project and Configuring It to Use the Entity Data Model	907
22.5.3	Data Bindings Between Controls and the Entity Data Model	909
22.6	Dynamically Binding Query Results	915
22.6.1	Creating the Display Query Results GUI	916
22.6.2	Coding the Display Query Results App	917
22.7	Retrieving Data from Multiple Tables with LINQ	919

22.8	Creating a Master/Detail View App	925
22.8.1	Creating the Master/Detail GUI	925
22.8.2	Coding the Master/Detail App	927
22.9	Address Book Case Study	928
22.9.1	Creating the Address Book App's GUI	930
22.9.2	Coding the Address Book App	931
22.10	Tools and Web Resources	935
22.11	Wrap-Up	935

23 Asynchronous Programming with async and await **942**

23.1	Introduction	943
23.2	Basics of async and await	945
23.2.1	async Modifier	945
23.2.2	await Expression	945
23.2.3	async, await and Threads	945
23.3	Executing an Asynchronous Task from a GUI App	946
23.3.1	Performing a Task Asynchronously	946
23.3.2	Method calculateButton_Click	948
23.3.3	Task Method Run: Executing Asynchronously in a Separate Thread	949
23.3.4	awaiting the Result	949
23.3.5	Calculating the Next Fibonacci Value Synchronously	949
23.4	Sequential Execution of Two Compute-Intensive Tasks	950
23.5	Asynchronous Execution of Two Compute-Intensive Tasks	952
23.5.1	awaiting Multiple Tasks with Task Method WhenAll	955
23.5.2	Method StartFibonacci	956
23.5.3	Modifying a GUI from a Separate Thread	956
23.5.4	awaiting One of Several Tasks with Task Method WhenAny	956
23.6	Invoking a Flickr Web Service Asynchronously with HttpClient	957
23.6.1	Using Class HttpClient to Invoke a Web Service	961
23.6.2	Invoking the Flickr Web Service's flickr.photos.search Method	961
23.6.3	Processing the XML Response	962
23.6.4	Binding the Photo Titles to the ListBox	963
23.6.5	Asynchronously Downloading an Image's Bytes	964
23.7	Displaying an Asynchronous Task's Progress	964
23.8	Wrap-Up	968

Chapters on the Web **975**

A Operator Precedence Chart **976**

B Simple Types **978**

C ASCII Character Set	980
Appendices on the Web	981
Index	983

Online Topics

PDFs presenting additional topics for advanced college courses and professionals are available through the book's Companion Website:

<http://www.pearsonhighered.com/deitel>

New copies of this book come with a Companion Website access code that's located on the book's inside front cover. If the access code is already visible or there is no card, you purchased a used book or an edition that does not come with an access code—you can purchase access directly from the Companion Website.

Web App Development with ASP.NET

XML and LINQ to XML

Universal Windows Platform (UWP) GUI, Graphics, Multimedia and XAML

REST Web Services

Cloud Computing with Microsoft Azure™

Windows Presentation Foundation (WPF) GUI, Graphics, Multimedia and XAML

ATM Case Study, Part 1: Object-Oriented Design with the UML

ATM Case Study, Part 2: Implementing an Object-Oriented Design in C#

Using the Visual Studio Debugger