

C++

Eastern
Economy
Edition



HOW TO PROGRAM

EIGHTH EDITION



PAUL DEITEL
HARVEY DEITEL



Contents

Chapters 25–26 and Appendices F–I are PDF documents posted online at the book's Companion Website, which is accessible from www.pearsonhighered.com/deitel.

Preface xxi

I	Introduction to Computers and C++	1
1.1	Introduction	2
1.2	Computers: Hardware and Software	5
1.3	Data Hierarchy	6
1.4	Computer Organization	7
1.5	Machine Languages, Assembly Languages and High-Level Languages	9
1.6	Introduction to Object Technology	10
1.7	Operating Systems	13
1.8	Programming Languages	15
1.9	C++ and a Typical C++ Development Environment	17
1.10	Test-Driving a C++ Application	21
1.11	Web 2.0: Going Social	27
1.12	Software Technologies	29
1.13	Future of C++: TR1, the New C++ Standard and the Open Source Boost Libraries	31
1.14	Keeping Up-to-Date with Information Technologies	32
1.15	Wrap-Up	32
2	Introduction to C++ Programming	37
2.1	Introduction	38
2.2	First Program in C++: Printing a Line of Text	38
2.3	Modifying Our First C++ Program	42
2.4	Another C++ Program: Adding Integers	43
2.5	Memory Concepts	47
2.6	Arithmetic	48
2.7	Decision Making: Equality and Relational Operators	51
2.8	Wrap-Up	55

3	Introduction to Classes, Objects and Strings	64
3.1	Introduction	65
3.2	Defining a Class with a Member Function	65
3.3	Defining a Member Function with a Parameter	68
3.4	Data Members, <i>set</i> Functions and <i>get</i> Functions	71
3.5	Initializing Objects with Constructors	77
3.6	Placing a Class in a Separate File for Reusability	81
3.7	Separating Interface from Implementation	84
3.8	Validating Data with <i>set</i> Functions	90
3.9	Wrap-Up	95
4	Control Statements: Part I	101
4.1	Introduction	101
4.2	Algorithms	101
4.3	Pseudocode	101
4.4	Control Structures	101
4.5	<i>if</i> Selection Statement	101
4.6	<i>if...else</i> Double-Selection Statement	101
4.7	<i>while</i> Repetition Statement	111
4.8	Formulating Algorithms: Counter-Controlled Repetition	111
4.9	Formulating Algorithms: Sentinel-Controlled Repetition	121
4.10	Formulating Algorithms: Nested Control Statements	131
4.11	Assignment Operators	131
4.12	Increment and Decrement Operators	131
4.13	Wrap-Up	131
5	Control Statements: Part 2	151
5.1	Introduction	151
5.2	Essentials of Counter-Controlled Repetition	151
5.3	<i>for</i> Repetition Statement	151
5.4	Examples Using the <i>for</i> Statement	151
5.5	<i>do...while</i> Repetition Statement	161
5.6	<i>switch</i> Multiple-Selection Statement	161
5.7	<i>break</i> and <i>continue</i> Statements	171
5.8	Logical Operators	171
5.9	Confusing the Equality (<i>==</i>) and Assignment (<i>=</i>) Operators	171
5.10	Structured Programming Summary	181
5.11	Wrap-Up	181

6	Functions and an Introduction to Recursion	194
6.1	Introduction	195
6.2	Program Components in C++	196
6.3	Math Library Functions	197
6.4	Function Definitions with Multiple Parameters	198
6.5	Function Prototypes and Argument Coercion	203
6.6	C++ Standard Library Headers	205
6.7	Case Study: Random Number Generation	207
6.8	Case Study: Game of Chance; Introducing enum	212
6.9	Storage Classes	215
6.10	Scope Rules	218
6.11	Function Call Stack and Activation Records	221
6.12	Functions with Empty Parameter Lists	225
6.13	Inline Functions	225
6.14	References and Reference Parameters	227
6.15	Default Arguments	231
6.16	Unary Scope Resolution Operator	232
6.17	Function Overloading	234
6.18	Function Templates	236
6.19	Recursion	239
6.20	Example Using Recursion: Fibonacci Series	242
6.21	Recursion vs. Iteration	245
6.22	Wrap-Up	248
7	Arrays and Vectors	267
7.1	Introduction	268
7.2	Arrays	269
7.3	Declaring Arrays	270
7.4	Examples Using Arrays	271
7.4.1	Declaring an Array and Using a Loop to Initialize the Array's Elements	271
7.4.2	Initializing an Array in a Declaration with an Initializer List	272
7.4.3	Specifying an Array's Size with a Constant Variable and Setting Array Elements with Calculations	273
7.4.4	Summing the Elements of an Array	275
7.4.5	Using Bar Charts to Display Array Data Graphically	276
7.4.6	Using the Elements of an Array as Counters	277
7.4.7	Using Arrays to Summarize Survey Results	278
7.4.8	Static Local Arrays and Automatic Local Arrays	281

7.5	Passing Arrays to Functions	
7.6	Case Study: Class GradeBook Using an Array to Store Grades	283
7.7	Searching Arrays with Linear Search	287
7.8	Sorting Arrays with Insertion Sort	293
7.9	Multidimensional Arrays	294
7.10	Case Study: Class GradeBook Using a Two-Dimensional Array	297
7.11	Introduction to C++ Standard Library Class Template vector	300
7.12	Wrap-Up	307
		313
8	Pointers	330
8.1	Introduction	331
8.2	Pointer Variable Declarations and Initialization	331
8.3	Pointer Operators	332
8.4	Pass-by-Reference with Pointers	335
8.5	Using const with Pointers	339
8.6	Selection Sort Using Pass-by-Reference	343
8.7	sizeof Operator	347
8.8	Pointer Expressions and Pointer Arithmetic	349
8.9	Relationship Between Pointers and Arrays	352
8.10	Pointer-Based String Processing	354
8.11	Arrays of Pointers	357
8.12	Function Pointers	358
8.13	Wrap-Up	361
9	Classes: A Deeper Look, Part I	379
9.1	Introduction	380
9.2	Time Class Case Study	381
9.3	Class Scope and Accessing Class Members	388
9.4	Separating Interface from Implementation	389
9.5	Access Functions and Utility Functions	390
9.6	Time Class Case Study: Constructors with Default Arguments	393
9.7	Destructors	398
9.8	When Constructors and Destructors Are Called	399
9.9	Time Class Case Study: A Subtle Trap—Returning a Reference to a private Data Member	402
9.10	Default Memberwise Assignment	405
9.11	Wrap-Up	407
10	Classes: A Deeper Look, Part 2	414
10.1	Introduction	415

10.2	<code>const</code> (Constant) Objects and <code>const</code> Member Functions	415
10.3	Composition: Objects as Members of Classes	423
10.4	<code>friend</code> Functions and <code>friend</code> Classes	429
10.5	Using the <code>this</code> Pointer	431
10.6	<code>static</code> Class Members	436
10.7	Proxy Classes	441
10.8	Wrap-Up	445

11 Operator Overloading; Class string 451

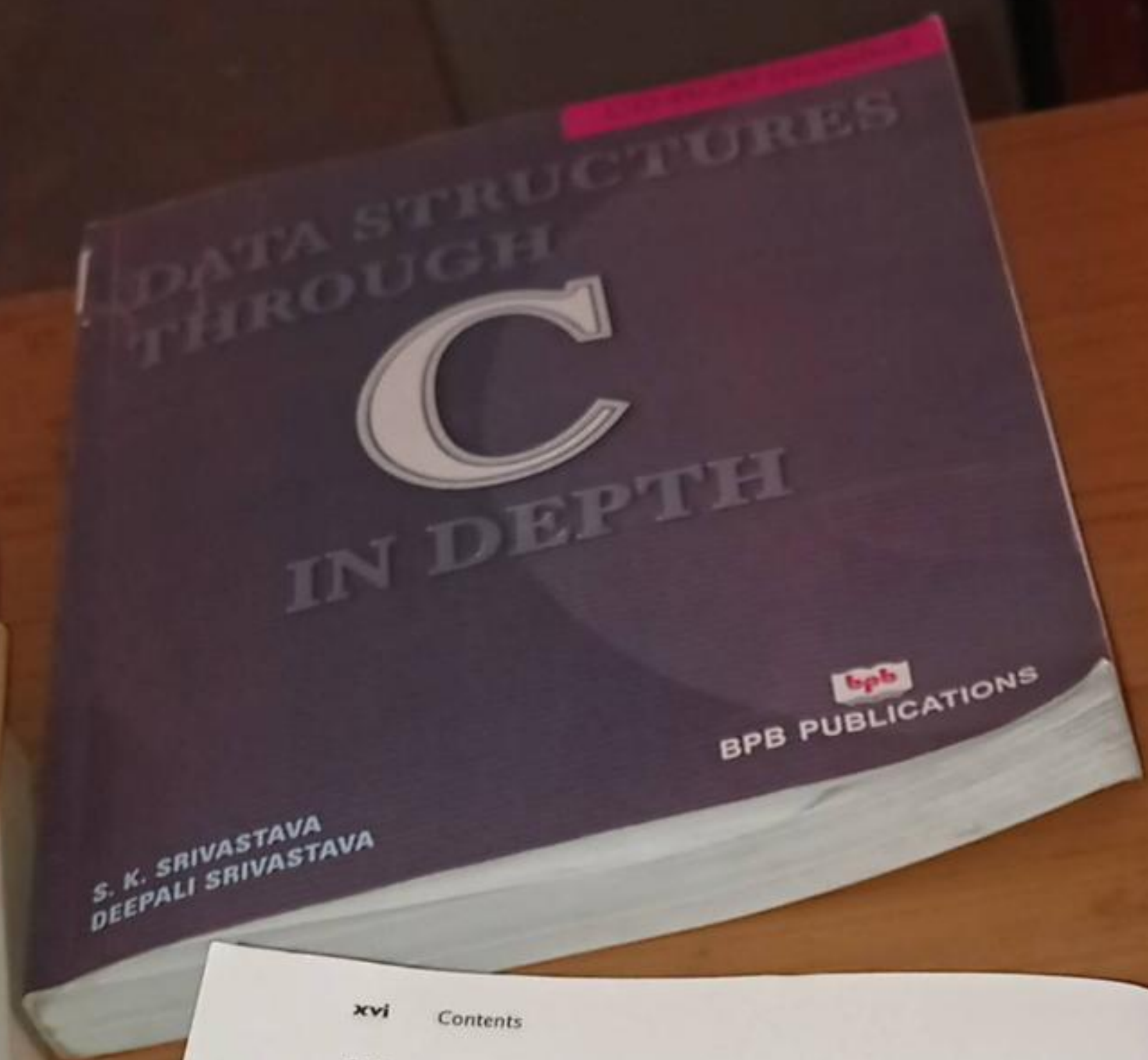
11.1	Introduction	452
11.2	Using the Overloaded Operators of Standard Library Class <code>string</code>	453
11.3	Fundamentals of Operator Overloading	456
11.4	Overloading Binary Operators	457
11.5	Overloading the Binary Stream Insertion and Stream Extraction Operators	458
11.6	Overloading Unary Operators	462
11.7	Overloading the Unary Prefix and Postfix <code>++</code> and <code>--</code> Operators	463
11.8	Case Study: A Date Class	464
11.9	Dynamic Memory Management	469
11.10	Case Study: Array Class	471
	11.10.1 Using the Array Class	472
	11.10.2 Array Class Definition	475
11.11	Operators as Member Functions vs. Non-Member Functions	483
11.12	Converting between Types	483
11.13	<code>explicit</code> Constructors	485
11.14	Building a <code>String</code> Class	487
11.15	Wrap-Up	488

12 Object-Oriented Programming: Inheritance 499

12.1	Introduction	500
12.2	Base Classes and Derived Classes	500
12.3	<code>protected</code> Members	503
12.4	Relationship between Base Classes and Derived Classes	503
	12.4.1 Creating and Using a <code>CommissionEmployee</code> Class	504
	12.4.2 Creating a <code>BasePlusCommissionEmployee</code> Class Without Using Inheritance	508
	12.4.3 Creating a <code>CommissionEmployee</code> – <code>BasePlusCommissionEmployee</code> Inheritance Hierarchy	514
	12.4.4 <code>CommissionEmployee</code> – <code>BasePlusCommissionEmployee</code> Inheritance Hierarchy Using <code>protected</code> Data	519

12.4.5	CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using private Data	52
12.5	Constructors and Destructors in Derived Classes	52
12.6	public, protected and private Inheritance	52
12.7	Software Engineering with Inheritance	52
12.8	Wrap-Up	52
13	Object-Oriented Programming: Polymorphism	534
13.1	Introduction	535
13.2	Introduction to Polymorphism: Polymorphic Video Game	536
13.3	Relationships Among Objects in an Inheritance Hierarchy	536
13.3.1	Invoking Base-Class Functions from Derived-Class Objects	537
13.3.2	Aiming Derived-Class Pointers at Base-Class Objects	540
13.3.3	Derived-Class Member-Function Calls via Base-Class Pointers	541
13.3.4	Virtual Functions	543
13.4	Type Fields and switch Statements	549
13.5	Abstract Classes and Pure virtual Functions	549
13.6	Case Study: Payroll System Using Polymorphism	551
13.6.1	Creating Abstract Base Class Employee	552
13.6.2	Creating Concrete Derived Class SalariedEmployee	556
13.6.3	Creating Concrete Derived Class CommissionEmployee	558
13.6.4	Creating Indirect Concrete Derived Class BasePlusCommissionEmployee	560
13.6.5	Demonstrating Polymorphic Processing	562
13.7	(Optional) Polymorphism, Virtual Functions and Dynamic Binding “Under the Hood”	566
13.8	Case Study: Payroll System Using Polymorphism and Runtime Type Information with Downcasting, dynamic_cast, typeid and type_info	569
13.9	Virtual Destructors	573
13.10	Wrap-Up	573
14	Templates	579
14.1	Introduction	580
14.2	Function Templates	580
14.3	Overloading Function Templates	583
14.4	Class Templates	584

14.5	Nontype Parameters and Default Types for Class Templates	590
14.6	Wrap-Up	591
15	Stream Input/Output	595
15.1	Introduction	596
15.2	Streams	597
15.2.1	Classic Streams vs. Standard Streams	597
15.2.2	iostream Library Headers	598
15.2.3	Stream Input/Output Classes and Objects	598
15.3	Stream Output	601
15.3.1	Output of char * Variables	601
15.3.2	Character Output Using Member Function put	601
15.4	Stream Input	602
15.4.1	get and getline Member Functions	602
15.4.2	istream Member Functions peek, putback and ignore	605
15.4.3	Type-Safe I/O	605
15.5	Unformatted I/O Using read, write and gcount	605
15.6	Introduction to Stream Manipulators	606
15.6.1	Integral Stream Base: dec, oct, hex and setbase	607
15.6.2	Floating-Point Precision (precision, setprecision)	607
15.6.3	Field Width (width, setw)	609
15.6.4	User-Defined Output Stream Manipulators	610
15.7	Stream Format States and Stream Manipulators	612
15.7.1	Trailing Zeros and Decimal Points (showpoint)	612
15.7.2	Justification (left, right and internal)	613
15.7.3	Padding (fill, setfill)	615
15.7.4	Integral Stream Base (dec, oct, hex, showbase)	616
15.7.5	Floating-Point Numbers; Scientific and Fixed Notation (scientific, fixed)	617
15.7.6	Uppercase/Lowercase Control (uppercase)	618
15.7.7	Specifying Boolean Format (boolalpha)	618
15.7.8	Setting and Resetting the Format State via Member Function flags	619
15.8	Stream Error States	620
15.9	Tying an Output Stream to an Input Stream	622
15.10	Wrap-Up	623
16	Exception Handling: A Deeper Look	632
16.1	Introduction	633



xvi Contents

22 Standard Template Library (STL)	850
22.1 Introduction to the Standard Template Library (STL)	851
22.2 Introduction to Containers	853
22.3 Introduction to Iterators	856
22.4 Introduction to Algorithms	861
22.5 Sequence Containers	863
22.5.1 vector Sequence Container	864
22.5.2 list Sequence Container	871
22.5.3 deque Sequence Container	875
22.6 Associative Containers	877
22.6.1 multiset Associative Container	877
22.6.2 set Associative Container	880
22.6.3 multimap Associative Container	881
22.6.4 map Associative Container	883
22.7 Container Adapters	885
22.7.1 stack Adapter	885
22.7.2 queue Adapter	887
22.7.3 priority_queue Adapter	888
22.8 Algorithms	890
22.8.1 fill, fill_n, generate and generate_n	890
22.8.2 equal, mismatch and lexicographical_compare	892
22.8.3 remove, remove_if, remove_copy and remove_copy_if	895
22.8.4 replace, replace_if, replace_copy and replace_copy_if	879
22.8.5 Mathematical Algorithms	900
22.8.6 Basic Searching and Sorting Algorithms	903
22.8.7 swap, iter_swap and swap_ranges	905
22.8.8 copy_backward, merge, unique and reverse	906
22.8.9 inplace_merge, unique_copy and reverse_copy	909
22.8.10 Set Operations	910
22.8.11 lower_bound, upper_bound and equal_range	913
22.8.12 Heapsort	915
22.8.13 min and max	918
22.8.14 STL Algorithms Not Covered in This Chapter	919
22.9 Class bitset	920
22.10 Function Objects	924
22.11 Wrap-Up	927
23 Boost Libraries, Technical Report I and C++0x	936
23.1 Introduction	937

Contents

23.2 Deitel Online C++ and Related Resource Centers	
23.3 Boost Libraries	
23.4 Boost Libraries Overview	
23.5 Regular Expressions with the regex Library	
23.5.1 Regular Expression Example	
23.5.2 Validating User Input with Regular Expressions	
23.5.3 Replacing and Splitting Strings	
23.6 Smart Pointers	
23.6.1 Reference Counted shared_ptr	
23.6.2 weak_ptr: shared_ptr Observer	
23.7 Technical Report I	
23.8 C++0x	
23.9 Core Language Changes	
23.10 Wrap-Up	
24 Other Topics	
24.1 Introduction	
24.2 const_cast Operator	
24.3 mutable Class Members	
24.4 namespaces	
24.5 Operator Keywords	
24.6 Pointers to Class Members (. * and -> *)	
24.7 Multiple Inheritance	
24.8 Multiple Inheritance and virtual Base Classes	
24.9 Wrap-Up	
Chapters on the Web	
A Operator Precedence and Assoc	
B ASCII Character Set	
C Fundamental Types	
D Number Systems	
D.1 Introduction	

23.2	Deitel Online C++ and Related Resource Centers	937
23.3	Boost Libraries	937
23.4	Boost Libraries Overview	938
23.5	Regular Expressions with the regex Library	941
23.5.1	Regular Expression Example	942
23.5.2	Validating User Input with Regular Expressions	944
23.5.3	Replacing and Splitting Strings	947
23.6	Smart Pointers	950
23.6.1	Reference Counted shared_ptr	950
23.6.2	weak_ptr: shared_ptr Observer	954
23.7	Technical Report 1	960
23.8	C++0x	961
23.9	Core Language Changes	962
23.10	Wrap-Up	967

24 Other Topics 974

24.1	Introduction	975
24.2	const_cast Operator	975
24.3	mutable Class Members	977
24.4	namespaces	979
24.5	Operator Keywords	982
24.6	Pointers to Class Members (. * and ->*)	984
24.7	Multiple Inheritance	986
24.8	Multiple Inheritance and virtual Base Classes	991
24.9	Wrap-Up	996

Chapters on the Web 1001

A Operator Precedence and Associativity 1002

B ASCII Character Set 1004

C Fundamental Types 1005

D Number Systems 1007

D.1	Introduction	1008
-----	--------------	------

D.2	Abbreviating Binary Numbers as Octal and Hexadecimal Numbers	101
D.3	Converting Octal and Hexadecimal Numbers to Binary Numbers	101
D.4	Converting from Binary, Octal or Hexadecimal to Decimal	101
D.5	Converting from Decimal to Binary, Octal or Hexadecimal	101
D.6	Negative Binary Numbers: Two's Complement Notation	101

E Preprocessor

E.1	Introduction	102
E.2	#include Preprocessor Directive	102
E.3	#define Preprocessor Directive: Symbolic Constants	102
E.4	#define Preprocessor Directive: Macros	102
E.5	Conditional Compilation	102
E.6	#error and #pragma Preprocessor Directives	102
E.7	Operators # and ##	102
E.8	Predefined Symbolic Constants	102
E.9	Assertions	102
E.10	Wrap-Up	102

Appendices on the Web

Index

Chapters 25–26 and Appendices F–I are PDF documents posted online at the book's Companion Website, which is accessible from www.pearsonhighered.com/deitel.

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

25.1	Introduction	2
25.2	Introduction to Object-Oriented Analysis and Design	2
25.3	Examining the ATM Requirements Document	2
25.4	Identifying the Classes in the ATM Requirements Document	25
25.5	Identifying Class Attributes	25
25.6	Identifying Objects' States and Activities	25
25.7	Identifying Class Operations	25
25.8	Indicating Collaboration Among Objects	25
25.9	Wrap-Up	25

26	ATM Case Study, Part 2: Implementing an Object-Oriented Design	26-1
26.1	Introduction	26-2
26.2	Starting to Program the Classes of the ATM System	26-2
26.3	Incorporating Inheritance into the ATM System	26-8
26.4	ATM Case Study Implementation	26-15
26.4.1	Class ATM	26-16
26.4.2	Class Screen	26-23
26.4.3	Class Keypad	26-25
26.4.4	Class CashDispenser	26-26
26.4.5	Class DepositSlot	26-28
26.4.6	Class Account	26-29
26.4.7	Class BankDatabase	26-31
26.4.8	Class Transaction	26-35
26.4.9	Class BalanceInquiry	26-37
26.4.10	Class Withdrawal	26-39
26.4.11	Class Deposit	26-44
26.4.12	Test Program ATMCaseStudy.cpp	26-47
26.5	Wrap-Up	26-47

F	C Legacy Code Topics	F-1
F.1	Introduction	F-2
F.2	Redirecting Input/Output on UNIX/Linux/Mac OS X and Windows Systems	F-2
F.3	Variable-Length Argument Lists	F-3
F.4	Using Command-Line Arguments	F-5
F.5	Notes on Compiling Multiple-Source-File Programs	F-7
F.6	Program Termination with <code>exit</code> and <code>atexit</code>	F-9
F.7	Type Qualifier <code>volatile</code>	F-10
F.8	Suffixes for Integer and Floating-Point Constants	F-10
F.9	Signal Handling	F-11
F.10	Dynamic Memory Allocation with <code>calloc</code> and <code>realloc</code>	F-13
F.11	Unconditional Branch: <code>goto</code>	F-14
F.12	Unions	F-15
F.13	Linkage Specifications	F-18
F.14	Wrap-Up	F-19

G	UML 2: Additional Diagram Types	G-1
G.1	Introduction	G-1
G.2	Additional Diagram Types	G-2

H Using the Visual Studio Debugger

H.1	Introduction	H-1
H.2	Breakpoints and the Continue Command	H-2
H.3	Locals and Watch Windows	H-2
H.4	Controlling Execution Using the Step Into, Step Over, Step Out and Continue Commands	H-8
H.5	Autos Window	H-11
H.6	Wrap-Up	H-13

I Using the GNU C++ Debugger

I.1	Introduction	I-1
I.2	Breakpoints and the run, stop, continue and print Commands	I-2
I.3	print and set Commands	I-2
I.4	Controlling Execution Using the step, finish and next Commands	I-8
I.5	watch Command	I-10
I.6	Wrap-Up	I-13