Preface xxiii

Before You Begin lix

1 Introduction to Computers, the Internet and World Wide Web 1

1.1 Introduction 2
1.2 What Is a Computer? 3
1.3 Computer Organization 4
1.4 Early Operating Systems 5
1.5 Personal, Distributed and Client/Server Computing 5
1.6 The Internet and the World Wide Web 6
1.7 Machine Languages, Assembly Languages and High-Level Languages 6
1.8 History of C and C++ 8
1.9 C++ Standard Library 8
1.10 History of Java 9
1.11 FORTRAN, COBOL, Pascal and Ada 10
1.12 Basic, Visual Basic, Visual C++, C# and .NET 11
1.13 Key Software Trend: Object Technology 11
1.14 Typical C++ Development Environment 12
1.15 Notes About C++ and *C++ How to Program, 5/e* 15
1.16 Test-Driving a C++ Application 16
1.17 Software Engineering Case Study: Introduction to Object Technology and the UML (Required) 22
1.18 Wrap-Up 27
1.19 Web Resources 27

2 Introduction to C++ Programming 36

2.1 Introduction 37
2.2 First Program in C++: Printing a Line of Text 37
2.3 Modifying Our First C++ Program 41
2.4 Another C++ Program: Adding Integers 42
2.5 Memory Concepts 46
2.6 Arithmetic 48
2.7 Decision Making: Equality and Relational Operators 51

2.8 (Optional) Software Engineering Case Study: Examining the ATM Requirements Document 56
2.9 Wrap-Up 65

3 Introduction to Classes and Objects 74

3.1 Introduction 75
3.2 Classes, Objects, Member Functions and Data Members 75
3.3 Overview of the Chapter Examples 77
3.4 Defining a Class with a Member Function 77
3.5 Defining a Member Function with a Parameter 81
3.6 Data Members, *set* Functions and *get* Functions 84
3.7 Initializing Objects with Constructors 91
3.8 Placing a Class in a Separate File for Reusability 95
3.9 Separating Interface from Implementation 99
3.10 Validating Data with *set* Functions 105
3.11 (Optional) Software Engineering Case Study: Identifying the Classes in the ATM Requirements Document 110
3.12 Wrap-Up 118

4 Control Statements: Part 1 124

4.1 Introduction 125
4.2 Algorithms 125
4.3 Pseudocode 126
4.4 Control Structures 127
4.5 `if` Selection Statement 131
4.6 `if...else` Double-Selection Statement 132
4.7 `while` Repetition Statement 137
4.8 Formulating Algorithms: Counter-Controlled Repetition 139
4.9 Formulating Algorithms: Sentinel-Controlled Repetition 145
4.10 Formulating Algorithms: Nested Control Statements 156
4.11 Assignment Operators 161
4.12 Increment and Decrement Operators 161
4.13 (Optional) Software Engineering Case Study: Identifying Class Attributes in the ATM System 165
4.14 Wrap-Up 169

5 Control Statements: Part 2 185

5.1 Introduction 186
5.2 Essentials of Counter-Controlled Repetition 186
5.3 `for` Repetition Statement 188
5.4 Examples Using the `for` Statement 193
5.5 `do...while` Repetition Statement 197
5.6 `switch` Multiple-Selection Statement 199
5.7 `break` and `continue` Statements

5.8 Logical Operators 211
5.9 Confusing Equality (==) and Assignment (=) Operators 216
5.10 Structured Programming Summary 217
5.11 (Optional) Software Engineering Case Study: Identifying Objects' States and Activities in the ATM System 222
5.12 Wrap-Up 226

6 Functions and an Introduction to Recursion 238

6.1 Introduction 239
6.2 Program Components in C++ 240
6.3 Math Library Functions 241
6.4 Function Definitions with Multiple Parameters 243
6.5 Function Prototypes and Argument Coercion 248
6.6 C++ Standard Library Header Files 250
6.7 Case Study: Random Number Generation 252
6.8 Case Study: Game of Chance and Introducing `enum` 258
6.9 Storage Classes 262
6.10 Scope Rules 265
6.11 Function Call Stack and Activation Records 268
6.12 Functions with Empty Parameter Lists 272
6.13 Inline Functions 273
6.14 References and Reference Parameters 275
6.15 Default Arguments 280
6.16 Unary Scope Resolution Operator 282
6.17 Function Overloading 283
6.18 Function Templates 286
6.19 Recursion 288
6.20 Example Using Recursion: Fibonacci Series 292
6.21 Recursion vs. Iteration 295
6.22 (Optional) Software Engineering Case Study: Identifying Class Operations in the ATM System 298
6.23 Wrap-Up 305

7 Arrays and Vectors 326

7.1 Introduction 327
7.2 Arrays 328
7.3 Declaring Arrays 329
7.4 Examples Using Arrays 330
7.5 Passing Arrays to Functions 346
7.6 Case Study: Class `GradeBook` Using an Array to Store Grades 351
7.7 Searching Arrays with Linear Search 358
7.8 Sorting Arrays with Insertion Sort 359
7.9 Multidimensional Arrays 362
7.10 Case Study: Class `GradeBook` Using a Two-Dimensional Array 365
7.11 Introduction to C++ Standard Library Class Template `vector` 372

7.12 (Optional) Software Engineering Case Study: Collaboration Among Objects in the ATM System 377
7.13 Wrap-Up 385

8 Pointers and Pointer-Based Strings 401

8.1 Introduction 402
8.2 Pointer Variable Declarations and Initialization 403
8.3 Pointer Operators 404
8.4 Passing Arguments to Functions by Reference with Pointers 407
8.5 Using `const` with Pointers 411
8.6 Selection Sort Using Pass-by-Reference 418
8.7 `sizeof` Operators 421
8.8 Pointer Expressions and Pointer Arithmetic 424
8.9 Relationship Between Pointers and Arrays 427
8.10 Arrays of Pointers 431
8.11 Case Study: Card Shuffling and Dealing Simulation 432
8.12 Function Pointers 438
8.13 Introduction to Pointer-Based String Processing 443
8.13.1 Fundamentals of Characters and Pointer-Based Strings 444
8.13.2 String Manipulation Functions of the String-Handling Library 446
8.14 Wrap-Up 454

9 Classes: A Deeper Look, Part 1 480

9.1 Introduction 481
9.2 `Time` Class Case Study 482
9.3 Class Scope and Accessing Class Members 487
9.4 Separating Interface from Implementation 489
9.5 Access Functions and Utility Functions 491
9.6 `Time` Class Case Study: Constructors with Default Arguments 493
9.7 Destructors 499
9.8 When Constructors and Destructors Are Called 500
9.9 `Time` Class Case Study: A Subtle Trap—Returning a Reference to a `private` Data Member 503
9.10 Default Memberwise Assignment 506
9.11 Software Reusability 508
9.12 (Optional) Software Engineering Case Study: Starting to Program the Classes of the ATM System 509
9.13 Wrap-Up 516

10 Classes: A Deeper Look, Part 2 523

10.1 Introduction 524
10.2 `const` (Constant) Objects and `const` Member Functions 524
10.3 Composition: Objects as Members of Classes 534
10.4 `friend` Functions and `friend` Classes 541

10.5 Using the this Pointer 545
10.6 Dynamic Memory Management with Operators new and delete 550
10.7 static Class Members 552
10.8 Data Abstraction and Information Hiding 558
10.8.1 Example: Array Abstract Data Type 559
10.8.2 Example: String Abstract Data Type 560
10.8.3 Example: Queue Abstract Data Type 560
10.9 Container Classes and Iterators 561
10.10 Proxy Classes 562
10.11 Wrap-Up 565

11 Operator Overloading; String and Array Objects 571

11.1 Introduction 572
11.2 Fundamentals of Operator Overloading 573
11.3 Restrictions on Operator Overloading 574
11.4 Operator Functions as Class Members vs. Global Functions 576
11.5 Overloading Stream Insertion and Stream Extraction Operators 577
11.6 Overloading Unary Operators 581
11.7 Overloading Binary Operators 581
11.8 Case Study: Array Class 582
11.9 Converting between Types 594
11.10 Case Study: String Class 595
11.11 Overloading ++ and -- 607
11.12 Case Study: A Date Class 609
11.13 Standard Library Class string 613
11.14 explicit Constructors 617
11.15 Wrap-Up 621

12 Object-Oriented Programming: Inheritance 633

12.1 Introduction 634
12.2 Base Classes and Derived Classes 635
12.3 protected Members 638
12.4 Relationship between Base Classes and Derived Classes 638
12.4.1 Creating and Using a CommissionEmployee Class 639
12.4.2 Creating a BasePlusCommissionEmployee Class Without Using Inheritance 644
12.4.3 Creating a CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy 650
12.4.4 CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using protected Data 655
12.4.5 CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using private Data 662
12.5 Constructors and Destructors in Derived Classes 670
12.6 public, protected and private Inheritance 678

12.7 Software Engineering with Inheritance 678
12.8 Wrap-Up 680

13 Object-Oriented Programming: Polymorphism 686

13.1 Introduction 687
13.2 Polymorphism Examples 689
13.3 Relationships Among Objects in an Inheritance Hierarchy 690
13.3.1 Invoking Base-Class Functions from Derived-Class Objects 690
13.3.2 Aiming Derived-Class Pointers at Base-Class Objects 698
13.3.3 Derived-Class Member-Function Calls via Base-Class Pointers 699
13.3.4 Virtual Functions 701
13.3.5 Summary of the Allowed Assignments Between Base-Class and Derived-Class Objects and Pointers 707
13.4 Type Fields and `switch` Statements 707
13.5 Abstract Classes and Pure `virtual` Functions 708
13.6 Case Study: Payroll System Using Polymorphism 710
13.6.1 Creating Abstract Base Class `Employee` 712
13.6.2 Creating Concrete Derived Class `SalariedEmployee` 715
13.6.3 Creating Concrete Derived Class `HourlyEmployee` 717
13.6.4 Creating Concrete Derived Class `CommissionEmployee` 720
13.6.5 Creating Indirect Concrete Derived Class `BasePlusCommissionEmployee` 722
13.6.6 Demonstrating Polymorphic Processing 724
13.7 (Optional) Polymorphism, Virtual Functions and Dynamic Binding "Under the Hood" 728
13.8 Case Study: Payroll System Using Polymorphism and Run-Time Type Information with Downcasting, `dynamic_cast`, `typeid` and `type_info` 732
13.9 Virtual Destructors 735
13.10 (Optional) Software Engineering Case Study: Incorporating Inheritance into the ATM System 736
13.11 Wrap-Up 744

14 Templates 749

14.1 Introduction 750
14.2 Function Templates 751
14.3 Overloading Function Templates 754
14.4 Class Templates 754
14.5 Nontype Parameters and Default Types for Class Templates 761
14.6 Notes on Templates and Inheritance 762
14.7 Notes on Templates and Friends 762
14.8 Notes on Templates and `static` Members 763
14.9 Wrap-Up 764

15 Stream Input/Output 769

15.1 Introduction 770

15.2 Streams 771
15.2.1 Classic Streams vs. Standard Streams 772
15.2.2 iostream Library Header Files 772
15.2.3 Stream Input/Output Classes and Objects 772
15.3 Stream Output 775
15.3.1 Output of char * Variables 775
15.3.2 Character Output using Member Function put 775
15.4 Stream Input 776
15.4.1 get and getline Member Functions 777
15.4.2 istream Member Functions peek, putback and ignore 780
15.4.3 Type-Safe I/O 780
15.5 Unformatted I/O using read, write and gcount 780
15.6 Introduction to Stream Manipulators 781
15.6.1 Integral Stream Base: dec, oct, hex and setbase 782
15.6.2 Floating-Point Precision (precision, setprecision) 783
15.6.3 Field Width (width, setw) 784
15.6.4 User-Defined Output Stream Manipulators 786
15.7 Stream Format States and Stream Manipulators 787
15.7.1 Trailing Zeros and Decimal Points (showpoint) 788
15.7.2 Justification (left, right and internal) 789
15.7.3 Padding (fill, setfill) 791
15.7.4 Integral Stream Base (dec, oct, hex, showbase) 792
15.7.5 Floating-Point Numbers; Scientific and Fixed Notation (scientific, fixed) 793
15.7.6 Uppercase/Lowercase Control (uppercase) 793
15.7.7 Specifying Boolean Format (boolalpha) 795
15.7.8 Setting and Resetting the Format State via Member Function flags 796
15.8 Stream Error States 797
15.9 Tying an Output Stream to an Input Stream 800
15.10 Wrap-Up 800

16 Exception Handling 810

16.1 Introduction 811
16.2 Exception-Handling Overview 812
16.3 Example: Handling an Attempt to Divide by Zero 812
16.4 When to Use Exception Handling 819
16.5 Rethrowing an Exception 820
16.6 Exception Specifications 821
16.7 Processing Unexpected Exceptions 822
16.8 Stack Unwinding 823
16.9 Constructors, Destructors and Exception Handling 824
16.10 Exceptions and Inheritance 825
16.11 Processing new Failures 825
16.12 Class auto_ptr and Dynamic Memory Allocation 829
16.13 Standard Library Exception Hierarchy 832

16.14 Other Error-Handling Techniques 834
16.15 Wrap-Up 834

17 File Processing 841

17.1 Introduction 842
17.2 The Data Hierarchy 842
17.3 Files and Streams 844
17.4 Creating a Sequential File 845
17.5 Reading Data from a Sequential File 849
17.6 Updating Sequential Files 856
17.7 Random-Access Files 856
17.8 Creating a Random-Access File 857
17.9 Writing Data Randomly to a Random-Access File 862
17.10 Reading from a Random-Access File Sequentially 864
17.11 Case Study: A Transaction-Processing Program 867
17.12 Input/Output of Objects 874
17.13 Wrap-Up 874

18 Class `string` and String Stream Processing 883

18.1 Introduction 884
18.2 `string` Assignment and Concatenation 885
18.3 Comparing `string`s 887
18.4 Substrings 890
18.5 Swapping `string`s 891
18.6 `string` Characteristics 892
18.7 Finding Strings and Characters in a `string` 894
18.8 Replacing Characters in a `string` 896
18.9 Inserting Characters into a `string` 898
18.10 Conversion to C-Style Pointer-Based `char *` Strings 899
18.11 Iterators 901
18.12 String Stream Processing 902
18.13 Wrap-Up 905

19 Web Programming 911

19.1 Introduction 912
19.2 HTTP Request Types 913
19.3 Multitier Architecture 914
19.4 Accessing Web Servers 915
19.5 Apache HTTP Server 916
19.6 Requesting XHTML Documents 917
19.7 Introduction to CGI 917
19.8 Simple HTTP Transactions 918
19.9 Simple CGI Scripts 920
19.10 Sending Input to a CGI Script 928

19.11 Using XHTML Forms to Send Input 928
19.12 Other Headers 938
19.13 Case Study: An Interactive Web Page 939
19.14 Cookies 943
19.15 Server-Side Files 949
19.16 Case Study: Shopping Cart 954
19.17 Wrap-Up 969
19.18 Internet and Web Resources 969

20 Searching and Sorting 975

20.1 Introduction 976
20.2 Searching Algorithms 976
20.2.1 Efficiency of Linear Search 976
20.2.2 Binary Search 978
20.3 Sorting Algorithms 982
20.3.1 Efficiency of Selection Sort 984
20.3.2 Efficiency of Insertion Sort 985
20.3.3 Merge Sort (A Recursive Implementation) 985
20.4 Wrap-Up 992

21 Data Structures 998

21.1 Introduction 999
21.2 Self-Referential Classes 1000
21.3 Dynamic Memory Allocation and Data Structures 1001
21.4 Linked Lists 1001
21.5 Stacks 1016
21.6 Queues 1021
21.7 Trees 1025
21.8 Wrap-Up 1033

22 Bits, Characters, C-Strings and `structs` 1057

22.1 Introduction 1058
22.2 Structure Definitions 1058
22.3 Initializing Structures 1061
22.4 Using Structures with Functions 1061
22.5 `typedef` 1061
22.6 Example: High-Performance Card Shuffling and Dealing Simulation 1062
22.7 Bitwise Operators 1065
22.8 Bit Fields 1074
22.9 Character-Handling Library 1078
22.10 Pointer-Based String-Conversion Functions 1084
22.11 Search Functions of the Pointer-Based String-Handling Library 1089
22.12 Memory Functions of the Pointer-Based String-Handling Library 1094
22.13 Wrap-Up 1099

23 Standard Template Library (STL) 1110

23.1 Introduction to the Standard Template Library (STL) 1112
23.1.1 Introduction to Containers 1113
23.1.2 Introduction to Iterators 1117
23.1.3 Introduction to Algorithms 1122
23.2 Sequence Containers 1124
23.2.1 `vector` Sequence Container 1125
23.2.2 `list` Sequence Container 1133
23.2.3 `deque` Sequence Container 1136
23.3 Associative Containers 1138
23.3.1 `multiset` Associative Container 1139
23.3.2 `set` Associative Container 1142
23.3.3 `multimap` Associative Container 1143
23.3.4 `map` Associative Container 1145
23.4 Container Adapters 1147
23.4.1 `stack` Adapter 1147
23.4.2 `queue` Adapter 1149
23.4.3 `priority_queue` Adapter 1151
23.5 Algorithms 1152
23.5.1 `fill`, `fill_n`, `generate` and `generate_n` 1153
23.5.2 `equal`, `mismatch` and `lexicographical_compare` 1154
23.5.3 `remove`, `remove_if`, `remove_copy` and `remove_copy_if` 1157
23.5.4 `replace`, `replace_if`, `replace_copy` and `replace_copy_if` 1159
23.5.5 Mathematical Algorithms 1162
23.5.6 Basic Searching and Sorting Algorithms 1165
23.5.7 `swap`, `iter_swap` and `swap_ranges` 1167
23.5.8 `copy_backward`, `merge`, `unique` and `reverse` 1169
23.5.9 `inplace_merge`, `unique_copy` and `reverse_copy` 1171
23.5.10 Set Operations 1173
23.5.11 `lower_bound`, `upper_bound` and `equal_range` 1176
23.5.12 Heapsort 1178
23.5.13 `min` and `max` 1181
23.5.14 STL Algorithms Not Covered in This Chapter 1181
23.6 Class `bitset` 1183
23.7 Function Objects 1187
23.8 Wrap-Up 1190
23.9 STL Internet and Web Resources 1191

24 Other Topics 1200

24.1 Introduction 1201
24.2 `const_cast` Operator 1201
24.3 `namespaces` 1203
24.4 Operator Keywords 1207
24.5 `mutable` Class Members 1209
24.6 Pointers to Class Members (`.*` and `->*`) 1211

24.7 Multiple Inheritance 1213
24.8 Multiple Inheritance and `virtual` Base Classes 1218
24.9 Wrap-Up 1222
24.10 Closing Remarks 1223

A Operator Precedence and Associativity Chart 1228
A.1 Operator Precedence 1228

B ASCII Character Set 1231

C Fundamental Types 1232

D Number Systems 1234
D.1 Introduction 1235
D.2 Abbreviating Binary Numbers as Octal and Hexadecimal Numbers 1238
D.3 Converting Octal and Hexadecimal Numbers to Binary Numbers 1239
D.4 Converting from Binary, Octal or Hexadecimal to Decimal 1239
D.5 Converting from Decimal to Binary, Octal or Hexadecimal 1240
D.6 Negative Binary Numbers: Two's Complement Notation 1242

E C Legacy Code Topics 1247
E.1 Introduction 1248
E.2 Redirecting Input/Output on UNIX/Linux/Mac OS X and Windows Systems 1248
E.3 Variable-Length Argument Lists 1249
E.4 Using Command-Line Arguments 1252
E.5 Notes on Compiling Multiple-Source-File Programs 1253
E.6 Program Termination with `exit` and `atexit` 1255
E.7 The `volatile` Type Qualifier 1257
E.8 Suffixes for Integer and Floating-Point Constants 1257
E.9 Signal Handling 1257
E.10 Dynamic Memory Allocation with `calloc` and `realloc` 1260
E.11 The Unconditional Branch: `goto` 1261
E.12 Unions 1262
E.13 Linkage Specifications 1265
E.14 Wrap-Up 1266

F Preprocessor 1272
F.1 Introduction 1273
F.2 The `#include` Preprocessor Directive 1273
F.3 The `#define` Preprocessor Directive: Symbolic Constants 1274
F.4 The `#define` Preprocessor Directive: Macros 1275

F.5 Conditional Compilation 1277
F.6 The #error and #pragma Preprocessor Directives 1278
F.7 The # and ## Operators 1278
F.8 Predefined Symbolic Constants 1279
F.9 Assertions 1279
F.10 Wrap-Up 1280

G **ATM Case Study Code** **1285**

G.1 ATM Case Study Implementation 1285
G.2 Class ATM 1286
G.3 Class Screen 1293
G.4 Class Keypad 1294
G.5 Class CashDispenser 1295
G.6 Class DepositSlot 1297
G.7 Class Account 1298
G.8 Class BankDatabase 1300
G.9 Class Transaction 1304
G.10 Class BalanceInquiry 1306
G.11 Class Withdrawal 1308
G.12 Class Deposit 1313
G.13 Test Program ATMCaseStudy.cpp 1316
G.14 Wrap-Up 1317

H **UML 2: Additional Diagram Types** **1318**

H.1 Introduction 1318
H.2 Additional Diagram Types 1318

I **C++ Internet and Web Resources** **1320**

I.1 Resources 1320
I.2 Tutorials 1322
I.3 FAQs 1322
I.4 Visual C++ 1322
I.5 Newsgroups 1323
I.6 Compilers and Development Tools 1323
I.7 Standard Template Library 1324

J **Introduction to XHTML** **1325**

J.1 Introduction 1326
J.2 Editing XHTML 1326
J.3 First XHTML Example 1327
J.4 Headers 1330
J.5 Linking 1331
J.6 Images 1333

J.7 Special Characters and More Line Breaks 1338
J.8 Unordered Lists 1340
J.9 Nested and Ordered Lists 1340
J.10 Basic XHTML Tables 1341
J.11 Intermediate XHTML Tables and Formatting 1346
J.12 Basic XHTML Forms 1349
J.13 More Complex XHTML Forms 1352
J.14 Internet and World Wide Web Resources 1359

K XHTML Special Characters 1363

L Using the Visual Studio® .NET Debugger 1364

L.1 Introduction 1365
L.2 Breakpoints and the **Continue** Command 1365
L.3 The **Locals** and **Watch** Windows 1371
L.4 Controlling Execution Using the **Step Into**, **Step Over**, **Step Out** and **Continue** Commands 1374
L.5 The **Autos** Window 1377
L.6 Wrap-Up 1378

M Using the GNU™ C++ Debugger 1381

M.1 Introduction 1382
M.2 Breakpoints and the `run`, `stop`, `continue` and `print` Commands 1382
M.3 The `print` and `set` Commands 1389
M.4 Controlling Execution Using the `step`, `finish` and `next` Commands 1391
M.5 The `watch` Command 1393
M.6 Wrap-Up 1396
M.7 Summary 1397

Bibliography 1399

Index 1405