

## Table of Contents

- Preface p. xxxiii
- 1 Introduction to Computers and C++ Programming p. 1
- 1.1 Introduction p. 2
- 1.2 What is a Computer? p. 5
- 1.3 Computer Organization p. 5
- 1.4 Evolution of Operating Systems p. 6
- 1.5 Personal Computing, Distributed Computing and Client/Server Computing p. 7
- 1.6 Machine Languages, Assembly Languages, and High-level Languages p. 8
- 1.7 History of C and C++ p. 9
- 1.8 C++ Standard Library p. 10
- 1.9 Java and Java How to Program p. 11
- 1.10 Other High-level Languages p. 11
- 1.11 Structured Programming p. 12
- 1.12 The Key Software Trend: Object Technology p. 12
- 1.13 Basics of a Typical C++ Environment p. 15
- 1.14 Hardware Trends p. 17
- 1.15 History of the Internet p. 18
- 1.16 History of the World Wide Web p. 19
- 1.17 General Notes About C++ and This Book p. 19
- 1.18 Introduction to C++ Programming p. 20
- 1.19 A Simple Program: Printing a Line of Text p. 21
- 1.20 Another Simple Program: Adding Two Integers p. 25
- 1.21 Memory Concepts p. 29
- 1.22 Arithmetic p. 30
- 1.23 Decision Making: Equality and Relational Operators p. 34
- 1.24 Thinking About Objects: Introduction to Object Technology and the Unified Modeling Language p. 38
- 2 Control Structures p. 58
- 2.1 Introduction p. 59
- 2.2 Algorithms p. 60
- 2.3 Pseudocode p. 60
- 2.4 Control Structures p. 61
- 2.5 The if Selection Structure p. 63
- 2.6 The if/else Selection Structure p. 65
- 2.7 The while Repetition Structure p. 69
- 2.8 Formulating Algorithms: Case Study 1 (Counter-Controlled Repetition) p. 70
- 2.9 Formulating Algorithms with Top-Down, Stepwise Refinement: Case Study 2 (Sentinel-Controlled Repetition) p. 73
- 2.10 Formulating Algorithms with Top-Down, Stepwise Refinement: Case Study 3 (Nested Control Structures) p. 81
- 2.11 Assignment Operators p. 85
- 2.12 Increment and Decrement Operators p. 86
- 2.13 Essentials of Counter-Controlled Repetition p. 88
- 2.14 The for Repetition Structure p. 91
- 2.15 Examples Using the for Structure p. 95
- 2.16 The switch Multiple-Selection Structure p. 99
- 2.17 The do/while Repetition Structure p. 105
- 2.18 The break and continue Statements p. 107
- 2.19 Logical Operators p. 109
- 2.20 Confusing Equality (==) and Assignment (=) Operators p. 113

- 2.21 Structured-Programming Summary p. 114
- 2.22 [Optional Case Study] Thinking About Objects: Identifying the Classes in a Problem p. 119
- 3 Functions p. 157
  - 3.1 Introduction p. 158
  - 3.2 Program Components in C++ p. 158
  - 3.3 Math Library Functions p. 159
  - 3.4 Functions p. 160
  - 3.5 Function Definitions p. 162
  - 3.6 Function Prototypes p. 166
  - 3.7 Header Files p. 168
  - 3.8 Random Number Generation p. 170
  - 3.9 Example: A Game of Chance and Introducing enum p. 176
  - 3.10 Storage Classes p. 179
  - 3.11 Scope Rules p. 182
  - 3.12 Recursion p. 185
  - 3.13 Example Using Recursion: The Fibonacci Series p. 188
  - 3.14 Recursion vs. Iteration p. 192
  - 3.15 Functions with Empty Parameter Lists p. 194
  - 3.16 Inline Functions p. 195
  - 3.17 References and Reference Parameters p. 196
  - 3.18 Default Arguments p. 201
  - 3.19 Unary Scope Resolution Operator p. 202
  - 3.20 Function Overloading p. 203
  - 3.21 Function Templates p. 206
  - 3.22 [Optional Case Study] Thinking About Objects: Identifying a Class's Attributes p. 208
- 4 Arrays p. 239
  - 4.1 Introduction p. 240
  - 4.2 Arrays p. 240
  - 4.3 Declaring Arrays p. 242
  - 4.4 Examples Using Arrays p. 242
  - 4.5 Passing Arrays to Functions p. 258
  - 4.6 Sorting Arrays p. 262
  - 4.7 Case Study: Computing Mean, Median and Mode Using Arrays p. 264
  - 4.8 Searching Arrays: Linear Search and Binary Search p. 269
  - 4.9 Multiple-Subscripted Arrays p. 273
  - 4.10 [Optional Case Study] Thinking About Objects: Identifying the Operations of a Class p. 280
- 5 Pointers and Strings p. 304
  - 5.1 Introduction p. 305
  - 5.2 Pointer Variable Declarations and Initialization p. 306
  - 5.3 Pointer Operators p. 307
  - 5.4 Calling Functions by Reference p. 310
  - 5.5 Using the const Qualifier with Pointers p. 314
  - 5.6 Bubble Sort Using Call-by-reference p. 321
  - 5.7 Pointer Expressions and Pointer Arithmetic p. 326
  - 5.8 The Relationship Between Pointers and Arrays p. 328
  - 5.9 Arrays of Pointers p. 333
  - 5.10 Case Study: A Card Shuffling and Dealing Simulation p. 333
  - 5.11 Function Pointers p. 338
  - 5.12 Introduction to Character and String Processing p. 343
    - 5.12.1 Fundamentals of Characters and Strings p. 343

- 5.12.2 String Manipulation Functions of the String-handling Library p. 345
- 5.13 [Optional Case Study] Thinking About Objects: Collaborations Among Objects p. 353
- 6 Classes and Data Abstraction p. 389
- 6.1 Introduction p. 390
- 6.2 Structure Definitions p. 391
- 6.3 Accessing Members of Structures p. 392
- 6.4 Implementing a User-Defined Type Time with a struct p. 393
- 6.5 Implementing a Time Abstract Data Type with a class p. 395
- 6.6 Class Scope and Accessing Class Members p. 402
- 6.7 Separating Interface from Implementation p. 404
- 6.8 Controlling Access to Members p. 407
- 6.9 Access Functions and Utility Functions p. 410
- 6.10 Initializing Class Objects: Constructors p. 413
- 6.11 Using Default Arguments with Constructors p. 414
- 6.12 Using Destructors p. 418
- 6.13 When Constructors and Destructors Are Called p. 418
- 6.14 Using Data Members and Member Functions p. 421
- 6.15 A Subtle Trap: Returning a Reference to a Private Data Member p. 426
- 6.16 Assignment by Default Memberwise Copy p. 429
- 6.17 Software Reusability p. 430
- 6.18 [Optional Case Study] Thinking About Objects: Starting to Program the Classes for the Elevator Simulator p. 431
- 7 Classes: Part II p. 452
- 7.1 Introduction p. 453
- 7.2 const (Constant) Objects and const Member Functions p. 453
- 7.3 Composition: Objects as Members of Classes p. 462
- 7.4 friend Functions and friend Classes p. 467
- 7.5 Using the this Pointer p. 471
- 7.6 Dynamic Memory Allocation with Operators new and delete p. 476
- 7.7 static Class Members p. 477
- 7.8 Data Abstraction and Information Hiding p. 483
- 7.8.1 Example: Array Abstract Data Type p. 484
- 7.8.2 Example: String Abstract Data Type p. 485
- 7.8.3 Example: Queue Abstract Data Type p. 485
- 7.9 Container Classes and Iterators p. 486
- 7.10 Proxy Classes p. 486
- 7.10 [Optional Case Study] Thinking About Objects: Programming the Classes for the Elevator Simulator p. 488
- 8 Operator Overloading p. 523
- 8.1 Introduction p. 524
- 8.2 Fundamentals of Operator Overloading p. 525
- 8.3 Restrictions on Operator Overloading p. 526
- 8.4 Operator Functions as Class Members vs. as friend Functions p. 528
- 8.5 Overloading Stream-Insertion and Stream-Extraction Operators p. 529
- 8.6 Overloading Unary Operators p. 532
- 8.7 Overloading Binary Operators p. 532
- 8.8 Case Study: An Array Class p. 533
- 8.9 Converting between Types p. 545
- 8.10 Case Study: A String Class p. 546
- 8.11 Overloading ++ and -- p. 558
- 8.12 Case Study: A Date Class p. 559
- 9 Inheritance p. 576

- 9.1 Introduction p. 577
- 9.2 Inheritance: Base Classes and Derived Classes p. 579
- 9.3 Protected Members p. 581
- 9.4 Casting Base-Class Pointers to Derived-Class Pointers p. 581
- 9.5 Using Member Functions p. 587
- 9.6 Overriding Base-Class Members in a Derived Class p. 587
- 9.7 public, protected and private Inheritance p. 592
- 9.8 Direct Base Classes and Indirect Base Classes p. 593
- 9.9 Using Constructors and Destructors in Derived Classes p. 593
- 9.10 Implicit Derived-Class Object to Base-Class Object Conversion p. 597
- 9.11 Software Engineering with Inheritance p. 598
- 9.12 Composition vs. Inheritance p. 599
- 9.13 "Uses A" and "Knows A" Relationships p. 600
- 9.14 Case Study: Point, Circle, Cylinder p. 600
- 9.15 Multiple Inheritance p. 607
- 9.16 [Optional Case Study] Thinking About Objects: Incorporating Inheritance into the Elevator Simulation p. 612
- 10 Virtual Functions and Polymorphism p. 625
- 10.1 Introduction p. 626
- 10.2 Type Fields and switch Statements p. 626
- 10.3 virtual Functions p. 627
- 10.4 Abstract Base Classes and Concrete Classes p. 628
- 10.5 Polymorphism p. 628
- 10.6 Case Study: A Payroll System Using Polymorphism p. 630
- 10.7 New Classes and Dynamic Binding p. 642
- 10.8 virtual Destructors p. 642
- 10.9 Case Study: Inheriting Interface and Implementation p. 643
- 10.10 Polymorphism, virtual Functions and Dynamic Binding "Under the Hood" p. 651
- 11 C++ Stream Input/Output p. 659
- 11.1 Introduction p. 661
- 11.2 Streams p. 661
- 11.2.1 ostream Library Header Files p. 662
- 11.2.2 Stream Input/Output Classes and Objects p. 662
- 11.3 Stream Output p. 664
- 11.3.1 Stream-Insertion Operator p. 664
- 11.3.2 Cascading Stream-Insertion/Extraction Operators p. 666
- 11.3.3 Output of char \* Variables p. 667
- 11.3.4 Character Output with Member Function put; Cascading puts p. 668
- 11.4 Stream Input p. 668
- 11.4.1 Stream-Extraction Operator p. 668
- 11.4.2 get and getline Member Functions p. 671
- 11.4.3 istream Member Functions peek, putback and ignore p. 674
- 11.4.4 Type-Safe I/O p. 674
- 11.5 Unformatted I/O with read, gcount and write p. 674
- 11.6 Stream Manipulators p. 675
- 11.6.1 Integral Stream Base: dec, oct, hex and setbase p. 675
- 11.6.2 Floating-Point Precision (precision, setprecision) p. 676
- 11.6.3 Field Width (setw, width) p. 678
- 11.6.4 User-Defined Manipulators p. 679
- 11.7 Stream Format States p. 680
- 11.7.1 Format State Flags p. 680
- 11.7.2 Trailing Zeros and Decimal Points (ios::showpoint) p. 681

- 11.7.3 Justification (ios::left, ios::right, ios::internal) p. 682
- 11.7.4 Padding (fill, setfill) p. 684
- 11.7.5 Integral Stream Base (ios::dec, ios::oct, ios::hex, ios::showbase) p. 686
- 11.7.6 Floating-Point Numbers; Scientific Notation (ios::scientific, ios::fixed) p. 687
- 11.7.7 Uppercase/Lowercase Control (ios::uppercase) p. 688
- 11.7.8 Setting and Resetting the Format Flags (flags, setiosflags, resetiosflags) p. 688
- 11.8 Stream Error States p. 690
- 11.9 Tying an Output Stream to an Input Stream p. 692
- 12 Templates p. 704
- 12.1 Introduction p. 705
- 12.2 Function Templates p. 706
- 12.3 Overloading Template Functions p. 709
- 12.4 Class Templates p. 709
- 12.5 Class Templates and Nontype Parameters p. 715
- 12.6 Templates and Inheritance p. 716
- 12.7 Templates and friends p. 716
- 12.8 Templates and static Members p. 717
- 13 Exception Handling p. 723
- 13.1 Introduction p. 724
- 13.2 When Exception Handling Should Be Used p. 726
- 13.3 Other Error-Handling Techniques p. 727
- 13.4 Basics of C++ Exception Handling: try, throw, catch p. 728
- 13.5 A Simple Exception-Handling Example: Divide by Zero p. 728
- 13.6 Throwing an Exception p. 731
- 13.7 Catching an Exception p. 732
- 13.8 Rethrowing an Exception p. 735
- 13.9 Exception Specifications p. 737
- 13.10 Processing Unexpected Exceptions p. 737
- 13.11 Stack Unwinding p. 738
- 13.12 Constructors, Destructors and Exception Handling p. 739
- 13.13 Exceptions and Inheritance p. 740
- 13.14 Processing new Failures p. 740
- 13.15 Class auto\_ptr and Dynamic Memory Allocation p. 744
- 13.16 Standard Library Exception Hierarchy p. 746
- 14 File Processing p. 757
- 14.1 Introduction p. 758
- 14.2 The Data Hierarchy p. 758
- 14.3 Files and Streams p. 760
- 14.4 Creating a Sequential Access File p. 761
- 14.5 Reading Data from a Sequential Access File p. 765
- 14.6 Updating Sequential Access Files p. 771
- 14.7 Random-Access Files p. 772
- 14.8 Creating a Random-Access File p. 773
- 14.9 Writing Data Randomly to a Random-Access File p. 775
- 14.10 Reading Data Sequentially from a Random-Access File p. 777
- 14.11 Example: A Transaction Processing Program p. 779
- 14.12 Input/Output of Objects p. 785
- 15 Data Structures p. 794
- 15.1 Introduction p. 795
- 15.2 Self-Referential Classes p. 796
- 15.3 Dynamic Memory Allocation p. 797
- 15.4 Linked Lists p. 798

- 15.5 Stacks p. 811
- 15.6 Queues p. 815
- 15.7 Trees p. 818
- 16 Bits, Characters, Strings and Structures p. 849
- 16.1 Introduction p. 850
- 16.2 Structure Definitions p. 850
- 16.3 Initializing Structures p. 852
- 16.4 Using Structures with Functions p. 853
- 16.5 typedef p. 853
- 16.6 Example: High-Performance Card-shuffling and Dealing Simulation p. 854
- 16.7 Bitwise Operators p. 856
- 16.8 Bit Fields p. 865
- 16.9 Character-handling Library p. 868
- 16.10 String Conversion Functions p. 874
- 16.11 Search Functions of the String-handling Library p. 879
- 16.12 Memory Functions of the String-handling Library p. 884
- 16.13 Another Function of the String-handling Library p. 888
- 17 The Preprocessor p. 902
- 17.1 Introduction p. 903
- 17.2 The #include Preprocessor Directive p. 903
- 17.3 The #define Preprocessor Directive: Symbolic Constants p. 904
- 17.4 The #define Preprocessor Directive: Macros p. 905
- 17.5 Conditional Compilation p. 906
- 17.6 The #error and #pragma Preprocessor Directives p. 908
- 17.7 The # and ## Operators p. 908
- 17.8 Line Numbers p. 908
- 17.9 Predefined Symbolic Constants p. 909
- 17.10 Assertions p. 909
- 18 C Legacy Code Topics p. 915
- 18.1 Introduction p. 916
- 18.2 Redirecting Input/Output on UNIX and DOS Systems p. 916
- 18.3 Variable-Length Argument Lists p. 917
- 18.4 Using Command-Line Arguments p. 919
- 18.5 Notes on Compiling Multiple-Source-File Programs p. 921
- 18.6 Program Termination with exit and atexit p. 923
- 18.7 The volatile Type Qualifier p. 924
- 18.8 Suffixes for Integer and Floating-Point Constants p. 925
- 18.9 Signal Handling p. 925
- 18.10 Dynamic Memory Allocation with calloc and realloc p. 927
- 18.11 The Unconditional Branch: goto p. 928
- 18.12 Unions p. 929
- 18.13 Linkage Specifications p. 933
- 19 Class string and String Stream Processing p. 940
- 19.1 Introduction p. 941
- 19.2 string Assignment and Concatenation p. 943
- 19.3 Comparing strings p. 945
- 19.4 Substrings p. 947
- 19.5 Swapping strings p. 948
- 19.6 string Characteristics p. 949
- 19.7 Finding Characters in a string p. 951
- 19.8 Replacing Characters in a string p. 954
- 19.9 Inserting Characters into a string p. 955

- 19.10 Conversion to C-Style char \* Strings p. 957
- 19.11 Iterators p. 959
- 19.12 String Stream Processing p. 960
- 20 Standard Template Library (STL) p. 970
- 20.1 Introduction to the Standard Template Library (STL) p. 972
- 20.1.1 Introduction to Containers p. 974
- 20.1.2 Introduction to Iterators p. 978
- 20.1.3 Introduction to Algorithms p. 983
- 20.2 Sequence Containers p. 985
- 20.2.1 vector Sequence Container p. 986
- 20.2.2 list Sequence Container p. 993
- 20.2.3 deque Sequence Container p. 997
- 20.3 Associative Containers p. 999
- 20.3.1 multiset Associative Container p. 1000
- 20.3.2 set Associative Container p. 1003
- 20.3.3 multimap Associative Container p. 1004
- 20.3.4 map Associative Container p. 1006
- 20.4 Container Adapters p. 1008
- 20.4.1 stack Adapter p. 1008
- 20.4.2 queue Adapter p. 1010
- 20.4.3 priority\_queue Adapter p. 1012
- 20.5 Algorithms p. 1013
- 20.5.1 fill, fill\_n, generate and generate\_n p. 1014
- 20.5.2 equal, mismatch and lexicographical\_compare p. 1016
- 20.5.3 remove, remove\_if, remove\_copy and remove\_copy\_if p. 1019
- 20.5.4 replace, replace\_if, replace\_copy and replace\_copy\_if p. 1022
- 20.5.5 Mathematical Algorithms p. 1025
- 20.5.6 Basic Searching and Sorting Algorithms p. 1028
- 20.5.7 swap, iter\_swap and swap\_ranges p. 1031
- 20.5.8 copy\_backward, merge, unique and reverse p. 1032
- 20.5.9 inplace\_merge, unique\_copy and reverse\_copy p. 1035
- 20.5.10 Set Operations p. 1037
- 20.5.11 lower\_bound, upper\_bound and equal\_range p. 1040
- 20.5.12 Heapsort p. 1043
- 20.5.13 min and max p. 1046
- 20.5.14 Algorithms Not Covered in This Chapter p. 1046
- 20.6 Class bitset p. 1048
- 20.7 Function Objects p. 1052
- 21 Standard C++ Language Additions p. 1067
- 21.1 Introduction p. 1068
- 21.2 bool Data Type p. 1068
- 21.3 static\_cast Operator p. 1070
- 21.4 const\_cast Operator p. 1072
- 21.5 reinterpret\_cast Operator p. 1073
- 21.6 namespaces p. 1074
- 21.7 Run-Time Type Information (RTTI) p. 1078
- 21.8 Operator Keywords p. 1082
- 21.9 explicit Constructors p. 1083
- 21.10 mutable Class Members p. 1089
- 21.11 Pointers to Class Members (. \* and -]\*) p. 1090
- 21.12 Multiple Inheritance and virtual Base Classes p. 1092
- 21.13 Closing Remarks p. 1097

- A Operator Precedence Chart p. 1102
- B ASCII Character Set p. 1104
- C Number Systems p. 1105
  - C.1 Introduction p. 1106
  - C.2 Abbreviating Binary Numbers as Octal Numbers and Hexadecimal Numbers p. 1109
  - C.3 Converting Octal Numbers and Hexadecimal Numbers to Binary Numbers p. 1110
  - C.4 Converting from Binary, Octal, or Hexadecimal to Decimal p. 1110
  - C.5 Converting from Decimal to Binary, Octal, or Hexadecimal p. 1111
  - C.6 Negative Binary Numbers: Two's Complement Notation p. 1112
- D C++ Internet and Web Resources p. 1118
  - D.1 Resources p. 1118
  - D.2 Tutorials p. 1119
  - D.3 FAQs p. 1119
  - D.4 Visual C++ p. 1120
  - D.5 comp.lang.c++ p. 1120
  - D.6 Compilers p. 1122
  - D.7 Development Tools p. 1122
  - D.8 Standard Template Library p. 1123
- Bibliography p. 1125
- Index p. 1131