

Contents

Preface Before You Begin

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 <i>C++ How to Program, 5/e</i>	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 Contents

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 I 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	209
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

4 Contents

7.10	Case Study: Class <code>GradeBook</code> Using a Two-Dimensional Array	365
7.11	Introduction to C++ Standard Library Class Template <code>vector</code>	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 <code>const</code> with Pointers	411
8.6	Selection Sort Using Pass-by-Reference	418
8.7	<code>sizeof</code> 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 I 480

9.1	Introduction	481
9.2	<code>Time</code> 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	<code>Time</code> Class Case Study: Constructors with Default Arguments	493
9.7	Destructors	499
9.8	When Constructors and Destructors Are Called	500
9.9	<code>Time</code> Class Case Study: A Subtle Trap—Returning a Reference to a <code>private</code> 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	<code>const</code> (Constant) Objects and <code>const</code> Member Functions	524
10.3	Composition: Objects as Members of Classes	534

10.4	friend Functions and friend Classes	541
10.5	Using the <code>this</code> Pointer	545
10.6	Dynamic Memory Management with Operators <code>new</code> and <code>delete</code>	550
10.7	<code>static</code> 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 <code>++</code> and <code>--</code>	607
11.12	Case Study: A Date Class	609
11.13	Standard Library Class <code>string</code>	613
11.14	<code>explicit</code> 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	<code>protected</code> Members	638
12.4	Relationship between Base Classes and Derived Classes	638
12.4.1	Creating and Using a <code>CommissionEmployee</code> Class	639
12.4.2	Creating a <code>BasePlusCommissionEmployee</code> Class Without Using Inheritance	644
12.4.3	Creating a <code>CommissionEmployee–BasePlusCommissionEmployee</code> Inheritance Hierarchy	650
12.4.4	<code>CommissionEmployee–BasePlusCommissionEmployee</code> Inheritance Hierarchy Using <code>protected</code> Data	655
12.4.5	<code>CommissionEmployee–BasePlusCommissionEmployee</code> Inheritance Hierarchy Using <code>private</code> Data	662

6 Contents

12.5	Constructors and Destructors in Derived Classes	670
12.6	<code>public</code> , <code>protected</code> and <code>private</code> 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 <code>switch</code> Statements	708
13.5	Abstract Classes and Pure <code>virtual</code> Functions	708
13.6	Case Study: Payroll System Using Polymorphism	711
13.6.1	Creating Abstract Base Class <code>Employee</code>	712
13.6.2	Creating Concrete Derived Class <code>SalariedEmployee</code>	715
13.6.3	Creating Concrete Derived Class <code>HourlyEmployee</code>	718
13.6.4	Creating Concrete Derived Class <code>CommissionEmployee</code>	720
13.6.5	Creating Indirect Concrete Derived Class <code>BasePlusCommissionEmployee</code>	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, <code>dynamic_cast</code> , <code>typeid</code> and <code>type_info</code>	732
13.9	Virtual Destructors	736
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 <code>static</code> 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	<code>iostream</code> Library Header Files	772
15.2.3	Stream Input/Output Classes and Objects	772
15.3	Stream Output	775
15.3.1	Output of <code>char *</code> Variables	775
15.3.2	Character Output using Member Function <code>put</code>	775
15.4	Stream Input	776
15.4.1	<code>get</code> and <code>getline</code> Member Functions	777
15.4.2	<code>istream</code> Member Functions <code>peek</code> , <code>putback</code> and <code>ignore</code>	780
15.4.3	Type-Safe I/O	780
15.5	Unformatted I/O using <code>read</code> , <code>write</code> and <code>gcount</code>	780
15.6	Introduction to Stream Manipulators	781
15.6.1	Integral Stream Base: <code>dec</code> , <code>oct</code> , <code>hex</code> and <code>setbase</code>	782
15.6.2	Floating-Point Precision (<code>precision</code> , <code>setprecision</code>)	783
15.6.3	Field Width (<code>width</code> , <code>setw</code>)	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 (<code>showpoint</code>)	788
15.7.2	Justification (<code>left</code> , <code>right</code> and <code>internal</code>)	789
15.7.3	Padding (<code>fill</code> , <code>setfill</code>)	791
15.7.4	Integral Stream Base (<code>dec</code> , <code>oct</code> , <code>hex</code> , <code>showbase</code>)	792
15.7.5	Floating-Point Numbers; Scientific and Fixed Notation (<code>scientific</code> , <code>fixed</code>)	793
15.7.6	Uppercase/Lowercase Control (<code>uppercase</code>)	793
15.7.7	Specifying Boolean Format (<code>boolalpha</code>)	795
15.7.8	Setting and Resetting the Format State via Member Function <code>flags</code>	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

8 Contents

16.10	Exceptions and Inheritance	825
16.11	Processing new Failures	825
16.12	Class <code>auto_ptr</code> 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	<code>string</code> Assignment and Concatenation	885
18.3	Comparing strings	887
18.4	Substrings	890
18.5	Swapping strings	891
18.6	<code>string</code> Characteristics	892
18.7	Finding Strings and Characters in a <code>string</code>	894
18.8	Replacing Characters in a <code>string</code>	896
18.9	Inserting Characters into a <code>string</code>	898
18.10	Conversion to C-Style Pointer-Based <code>char *</code> 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	967
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

10 Contents

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	<code>const_cast</code> Operator	1201
24.3	<code>namespaces</code>	1203
24.4	Operator Keywords	1207
24.5	<code>mutable</code> Class Members	1209
24.6	Pointers to Class Members (<code>.</code> , <code>*</code> and <code>->*</code>)	1211
24.7	Multiple Inheritance	1213
24.8	Multiple Inheritance and <code>virtual</code> 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	1248
E.1	Introduction	1249
E.2	Redirecting Input/Output on UNIX and DOS Systems	1249
E.3	Variable-Length Argument Lists	1250
E.4	Using Command-Line Arguments	1253
E.5	Notes on Compiling Multiple-Source-File Programs	1254
E.6	Program Termination with <code>exit</code> and <code>atexit</code>	1256
E.7	The <code>volatile</code> Type Qualifier	1257
E.8	Suffixes for Integer and Floating-Point Constants	1258
E.9	Signal Handling	1258
E.10	Dynamic Memory Allocation with <code>calloc</code> and <code>realloc</code>	1261
E.11	The Unconditional Branch: <code>goto</code>	1261
E.12	Unions	1264
E.13	Linkage Specifications	1266
E.14	Wrap-Up	1268

12 Contents**F Preprocessor 1293**

F.1	Introduction	1294
F.2	The <code>#include</code> Preprocessor Directive	1294
F.3	The <code>#define</code> Preprocessor Directive: Symbolic Constants	1295
F.4	The <code>#define</code> Preprocessor Directive: Macros	1296
F.5	Conditional Compilation	1297
F.6	The <code>#error</code> and <code>#pragma</code> Preprocessor Directives	1299
F.7	The <code>#</code> and <code>##</code> Operators	1299
F.8	Line Numbers	1300
F.9	Predefined Symbolic Constants	1300
F.10	Assertions	1300
F.11	Wrap-Up	1301

G ATM Case Study Code 1323

G.1	ATM Case Study Implementation	1323
G.2	Class <code>ATM</code>	1324
G.3	Class <code>Screen</code>	1331
G.4	Class <code>Keypad</code>	1332
G.5	Class <code>CashDispenser</code>	1333
G.6	Class <code>DepositSlot</code>	1335
G.7	Class <code>Account</code>	1336
G.8	Class <code>BankDatabase</code>	1338
G.9	Class <code>Transaction</code>	1342
G.10	Class <code>BalanceInquiry</code>	1344
G.11	Class <code>Withdrawal</code>	1346
G.12	Class <code>Deposit</code>	1351
G.13	Test Program <code>ATMCaseStudy.cpp</code>	1354
G.14	Wrap-Up	1355

H UML 2: Additional Diagram Types 1356

H.1	Introduction	1356
H.2	Additional Diagram Types	1356

I C++ Internet and Web Resources 1358

I.1	Resources	1358
I.2	Tutorials	1360
I.3	FAQs	1360
I.4	Visual C++	1360
I.5	Newsgroups	1361
I.6	Compilers and Development Tools	1361
I.7	Standard Template Library	1362

J	Introduction to XHTML	1363
J.1	Introduction	1364
J.2	Editing XHTML	1364
J.3	First XHTML Example	1365
J.4	Headers	1368
J.5	Linking	1369
J.6	Images	1372
J.7	Special Characters and More Line Breaks	1377
J.8	Unordered Lists	1379
J.9	Nested and Ordered Lists	1379
J.10	Basic XHTML Tables	1380
J.11	Intermediate XHTML Tables and Formatting	1386
J.12	Basic XHTML Forms	1388
J.13	More Complex XHTML Forms	1391
J.14	Internet and World Wide Web Resources	1398
K	XHTML Special Characters	1403
L	Using the Visual StudioQ® .NET Debugger	1404
L.1	Introduction	1405
L.2	Breakpoints and the Continue Command	1405
L.3	The Locals and Watch Windows	1411
L.4	Controlling Execution Using the Step Into , Step Over , Step Out and Continue Commands	1414
L.5	The Autos Window	1417
L.6	Wrap-Up	1419
M	Using the GNU C++ Debugger	1421
M.1	Introduction	1422
M.2	Breakpoints and the run , stop , continue and print Commands	1422
M.3	The print and set Commands	1429
M.4	Controlling Execution Using the step , finish and next Commands	1431
M.5	The watch Command	1433
M.6	Wrap-Up	1437

Bibliography

Index