

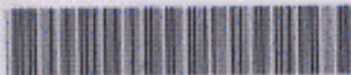


SECOND EDITION

COMPUTER
SCIENCE
TAPESTRY

EXPLORING PROGRAMMING AND
COMPUTER SCIENCE WITH C++

Suranaree University of Technology



31051000647053

DOWEN L. ASTRACHAN



McGRAW-HILL INTERNATIONAL EDITIONS

Computer Science Series

Contents

<i>List of Programs</i>	xvii
<i>Preface</i>	xxiii
I Computer Science and Programming	3
1.1 What Is Computer Science?	3
1.1.1 <i>The Tapestry of Computer Science</i>	4
1.2 Algorithms	5
1.2.1 <i>Arranging 13 Cards</i>	6
1.2.2 <i>Arranging 100,000 exams</i>	7
1.3 Computer Science Themes and Concepts	8
1.3.1 <i>Theory, Language, and Architecture</i>	8
1.3.2 <i>Abstractions, Models, and Complexity</i>	9
1.4 Language, Architecture, and Programs	12
1.4.1 <i>High- and Low-Level Languages</i>	12
1.5 Creating and Developing Programs	15
1.6 Language and Program Design	18
1.6.1 <i>Off-the-Shelf Components</i>	19
1.6.2 <i>Using Components</i>	20
1.7 Chapter Review	20
1.8 Exercises	21

I FOUNDATIONS OF C++ PROGRAMMING

2 C++ Programs: Form and Function	29
2.1 Simple C++ Programs	30
2.1.1 <i>Syntax and Semantics</i>	31
2.2 How a Program Works	35
2.2.1 <i>Flow of Control</i>	36
2.3 What Can Be Output?	37
2.4 Using Functions	40
2.5 Functions with Parameters	44
2.5.1 <i>What Is a Parameter?</i>	44
2.5.2 <i>An Example of Parameterization: Happy Birthday</i>	45
2.5.3 <i>Passing Parameters</i>	48
2.6 Functions with Several Parameters	51
2.7 Program Style	60
2.7.1 <i>Identifiers</i>	61
2.8 Chapter Review	61
2.9 Exercises	63

3	Program Design and Implementation	67
3.1	The Input Phase of Computation	68
3.1.1	<i>The Input Stream, cin</i>	69
3.1.2	<i>Variables</i>	69
3.2	Processing Numbers	73
3.2.1	<i>Numeric Data</i>	75
3.2.2	<i>Arithmetic Operators</i>	77
3.2.3	<i>Evaluating Expressions</i>	79
3.2.4	<i>The Type char</i>	82
3.3	Case Study: Pizza Slices	83
3.3.1	<i>Pizza Statistics</i>	83
3.4	Classes and Types: An Introduction	86
3.4.1	<i>Member Functions</i>	88
3.4.2	<i>Reading Programs</i>	89
3.4.3	<i>Private and Public</i>	91
3.5	Compiling and Linking	93
3.6	Chapter Review	94
3.7	Exercises	95
4	Control, Functions, and Classes	99
4.1	The Assignment Operator	100
4.2	Choices and Conditional Execution	103
4.2.1	<i>The if/else Statement</i>	105
4.3	Operators	107
4.3.1	<i>Relational Operators</i>	108
4.3.2	<i>Logical Operators</i>	111
4.3.3	<i>Short-Circuit Evaluation</i>	112
4.3.4	<i>Arithmetic Assignment Operators</i>	113
4.4	Block Statements and Defensive Programming	114
4.4.1	<i>Defensive Programming Conventions</i>	115
4.4.2	<i>Cascaded if/else Statements</i>	118
4.5	Functions That Return Values	123
4.5.1	<i>The Math Library <cmath></i>	126
4.5.2	<i>Pre- and Postconditions</i>	128
4.5.3	<i>Function Return Types</i>	129
4.6	Class Member Functions	137
4.6.1	<i>string Member Functions</i>	137
4.6.2	<i>Calling and Writing Functions</i>	140
4.6.3	<i>The Date Class</i>	143
4.7	Using Boolean Operators: De Morgan's Law	144
4.8	Chapter Review	146
4.9	Exercises	148
5	Iteration with Programs and Classes	153
5.1	The while Loop	153

5.1.1	<i>Infinite Loops</i>	156
5.1.2	<i>Loops and Mathematical Functions</i>	157
5.1.3	<i>Computing Factorials</i>	158
5.1.4	<i>Computing Prime Numbers</i>	162
5.1.5	<i>Kinds of Loops</i>	166
5.1.6	<i>Efficiency Considerations</i>	166
5.1.7	<i>Exponentiation: A Case Study in Loop Development</i>	167
5.1.8	<i>Numbers Written in English</i>	173
5.1.9	<i>Fence Post Problems</i>	175
5.2	Alternative Looping Statements	177
5.2.1	<i>The for Loop</i>	178
5.2.2	<i>The Operators ++ and --</i>	179
5.2.3	<i>The do-while Loop</i>	180
5.2.4	<i>Pseudo-Infinite Loops</i>	181
5.2.5	<i>Choosing a Looping Statement</i>	183
5.2.6	<i>Nested Loops</i>	183
5.2.7	<i>Defining Constants</i>	188
5.3	Variable Scope	189
5.4	Using Classes	191
5.4.1	<i>The Date Class</i>	191
5.4.2	<i>The Dice Class</i>	195
5.4.3	<i>Testing the Dice Class</i>	198
5.5	Chapter Review	202
5.6	Exercises	203

2

PROGRAM AND CLASS CONSTRUCTION: EXTENDING THE FOUNDATION

6	Classes, Iterators, and Patterns	213
6.1	Classes: From Use to Implementation	213
6.1.1	<i>Class Documentation: The Interface (.h File)</i>	213
6.1.2	<i>Comments in .h Files</i>	214
6.1.3	<i>Class Documentation: The Implementation or .cpp File</i>	216
6.1.4	<i>Member Function Implementation</i>	219
6.1.5	<i>Scope of Private Variables</i>	220
6.2	Program Design with Functions	222
6.2.1	<i>Evaluating Classes and Code: Coupling and Cohesion</i>	224
6.2.2	<i>Toward a Class-Based Quiz Program</i>	225
6.2.3	<i>Reference Parameters</i>	226
6.2.4	<i>Pass by Value and Pass by Reference</i>	229
6.2.5	<i>const Reference Parameters</i>	231
6.3	Reading Words: Stream Iteration	234
6.3.1	<i>Recommended Problem-Solving and Programming Steps</i>	235
6.3.2	<i>A Pseudocode Solution</i>	235

6.3.3	<i>Solving a Related Problem</i>	238
6.3.4	<i>The Final Program: Counting Words</i>	240
6.3.5	<i>Streams Associated with Files</i>	243
6.3.6	<i>Type Casting</i>	245
6.3.7	<i>A Word-Reading Class Using ifstream</i>	247
6.4	Finding Extreme Values	249
6.4.1	<i>Largest/Smallest Values</i>	251
6.4.2	<i>Initialization: Another Fence Post Problem</i>	252
6.4.3	<i>Word Frequencies</i>	254
6.4.4	<i>Using the CTimer class</i>	256
6.5	Case Study: Iteration and String Sets	259
6.5.1	<i>Iterators and the strutils.h Library</i>	261
6.5.2	<i>The Type ofstream</i>	261
6.5.3	<i>Sets and Word Counting</i>	263
6.6	Chapter Review	266
6.7	Exercises	267
7	Class Interfaces, Design, and Implementation	277
7.1	Designing Classes: From Requirements to Implementation	277
7.1.1	<i>Requirements</i>	278
7.1.2	<i>Nouns as Classes</i>	278
7.1.3	<i>Verbs as Member Functions (Methods)</i>	279
7.1.4	<i>Finding Verbs Using Scenarios</i>	279
7.1.5	<i>Assigning Responsibilities</i>	281
7.1.6	<i>Implementing and Testing Classes</i>	282
7.1.7	<i>Implementing the Class Quiz</i>	285
7.1.8	<i>Implementing the Class Question</i>	287
7.1.9	<i>Sidebar: Converting int and double Values to strings</i>	289
7.2	A Conforming Interface: A New Question Class	300
7.2.1	<i>Using the New Question Class</i>	300
7.2.2	<i>Creating a Program</i>	301
7.2.3	<i>The Preprocessor</i>	302
7.2.4	<i>The Compiler</i>	304
7.2.5	<i>The Linker</i>	305
7.2.6	<i>A New Question Class</i>	306
7.3	Random Walks	309
7.3.1	<i>One-Dimensional Random Walks</i>	310
7.3.2	<i>Selection with the switch Statement</i>	312
7.3.3	<i>A RandomWalk Class</i>	314
7.3.4	<i>A Two-Dimensional Walk Class</i>	321
7.3.5	<i>The Common Interface in RandomWalk and RandomWalk2D</i>	327
7.4	structs as Data Aggregates	329
7.4.1	<i>structs for Storing Points</i>	331
7.4.2	<i>Operators for structs</i>	333

7.5	Chapter Review	334
7.6	Exercises	335
8	Arrays, Data, and Random Access	339
8.1	Arrays and Vectors as Counters	340
8.1.1	<i>An Introduction to the Class <code>tvector</code></i>	343
8.1.2	<i>Counting with <code>tvector</code>s</i>	344
8.2	Defining and Using <code>tvector</code> s	347
8.2.1	<i><code>tvector</code> Definition</i>	347
8.2.2	<i><code>tvector</code> Initialization</i>	348
8.2.3	<i><code>tvector</code> Parameters</i>	348
8.2.4	<i>A <code>tvector</code> Case Study: Shuffling CD Tracks</i>	352
8.3	Collections and Lists Using <code>tvector</code> s	357
8.3.1	<i>Size and Capacity</i>	358
8.3.2	<i>Using <code>push_back</code>, <code>resize</code>, and <code>reserve</code></i>	358
8.3.3	<i>Vector Idioms: Insertion, Deletion, and Searching</i>	363
8.3.4	<i>Insertion into a Sorted Vector</i>	366
8.3.5	<i>Deleting an Element Using <code>pop_back</code></i>	368
8.3.6	<i>Searching a Vector</i>	369
8.3.7	<i>Binary Search</i>	374
8.3.8	<i>Comparing Sequential and Binary Search</i>	375
8.4	Built-in Arrays	380
8.4.1	<i>Defining an Array</i>	380
8.4.2	<i>Initializing an Array</i>	381
8.4.3	<i>Arrays as Parameters</i>	382
8.5	Chapter Review	386
8.6	Exercises	388

3

DESIGN, USE, AND ANALYSIS: EXTENDING THE FOUNDATION

9	Strings, Streams, and Operators	397
9.1	Characters: Building Blocks for Strings	398
9.1.1	<i>The Type <code>char</code> as an Abstraction</i>	398
9.1.2	<i>The Library <code><cctype></code></i>	401
9.1.3	<i>Strings as <code>char</code> Sequences</i>	403
9.2	Streams and Files as Lines and Characters	406
9.2.1	<i>Input Using <code>getline()</code></i>	407
9.2.2	<i>Parsing Line-Oriented Data Using <code>istringstream</code></i>	411
9.2.3	<i>Output Using <code>ostringstream</code></i>	413
9.2.4	<i>Strings, Streams, and Characters</i>	414
9.3	Case Study: Removing Comments with State Machines	417
9.3.1	<i>Counting Words</i>	417
9.3.2	<i>Problem Specification: What Is a Comment?</i>	419
9.3.3	<i>A State Machine Approach to I/O</i>	419

9.3.4	<i>Enumerated Types</i>	424
9.4	Case Study: Overloaded Operators and the <code>ClockTime</code> Class	426
9.4.1	<i>Throw-Away Code vs. Class Design</i>	428
9.4.2	<i>Implementing the <code>ClockTime</code> Class</i>	429
9.4.3	<i>Class of Data Invariants</i>	432
9.4.4	<i>Overloaded Operators</i>	433
9.4.5	<i>Friend Classes</i>	433
9.4.6	<i>Overloaded operator <<</i>	434
9.4.7	<i>Overloaded Relational Operators</i>	435
9.4.8	<i>Overloaded operator + and +=</i>	435
9.4.9	<i>Testing the <code>ClockTime</code> Class</i>	436
9.4.10	<i>The Final Program</i>	438
9.5	Chapter Review	440
9.6	Exercises	441
10	Recursion, Lists, and Matrices	449
10.1	Recursive Functions	449
10.1.1	<i>Similar and Simpler Functions</i>	449
10.1.2	<i>General Rules for Recursion</i>	454
10.1.3	<i>Infinite Recursion</i>	456
10.2	Recursion and Directories	458
10.2.1	<i>Classes for Traversing Directories</i>	459
10.2.2	<i>Recursion and Directory Traversal</i>	460
10.2.3	<i>Properties of Recursive Functions</i>	466
10.3	Comparing Recursion and Iteration	467
10.3.1	<i>The Factorial Function</i>	467
10.3.2	<i>Fibonacci Numbers</i>	471
10.3.3	<i>Permutation Generation</i>	474
10.4	Scope and Lifetime	478
10.4.1	<i>Global Variables</i>	479
10.4.2	<i>Hidden Identifiers</i>	481
10.4.3	<i>Static Definitions</i>	483
10.4.4	<i>Static or Class Variables and Functions</i>	484
10.5	Case Study: Lists and the Class <code>CList</code>	486
10.5.1	<i>What Is a <code>CList</code> Object?</i>	487
10.5.2	<i>Tail-ing Down a List</i>	489
10.5.3	<i>cons-ing Up a List</i>	491
10.5.4	<i>Append, Reverse, and Auxiliary Functions</i>	493
10.5.5	<i>Polynomials Implemented with Lists</i>	499
10.5.6	<i><code>CList</code> and Sparse, Sequential Structures</i>	500
10.6	The Class <code>tmatrix</code>	504
10.6.1	<i>A Simple <code>tmatrix</code> Program</i>	504
10.6.2	<i>Case Study: Finding Blobs</i>	506
10.7	Chapter Review	515
10.8	Exercises	516

11	Sorting, Templates, and Generic Programming	525
11.1	Sorting an Array	525
11.1.1	<i>Selection Sort</i>	526
11.1.2	<i>Insertion Sort</i>	530
11.2	Function Templates	535
11.2.1	<i>Printing a tvector with a Function Template</i>	536
11.2.2	<i>Function Templates and Iterators</i>	539
11.2.3	<i>Function Templates, Reuse, and Code Bloat</i>	543
11.3	Function Objects	543
11.3.1	<i>The Function Object Comparer</i>	544
11.3.2	<i>Predicate Function Objects</i>	549
11.4	Analyzing Sorts	553
11.4.1	<i>O Notation</i>	556
11.4.2	<i>Worst Case and Average Case</i>	556
11.4.3	<i>Analyzing Insertion Sort</i>	557
11.5	Quicksort	559
11.5.1	<i>The Partition/Pivot Function</i>	561
11.5.2	<i>Analysis of Quicksort</i>	564
11.6	Chapter Review	567
11.7	Exercises	568
12	Dynamic Data, Lists, and Class Templates	571
12.1	Pointers as Indirect References	571
12.1.1	<i>What Is a Pointer?</i>	571
12.1.2	<i>Heap Objects</i>	575
12.1.3	<i>Sharing Objects</i>	578
12.1.4	<i>Reference Variables</i>	581
12.1.5	<i>Pointers for Sharing</i>	582
12.1.6	<i>Interdependencies, Class Declarations, and Header Files</i>	583
12.1.7	<i>delete and Destructors</i>	589
12.2	Linked Lists	595
12.2.1	<i>Creating Nodes with Linked Lists</i>	598
12.2.2	<i>Iterating over a Linked List</i>	599
12.2.3	<i>Adding a Last Node to a Linked List</i>	600
12.2.4	<i>Deleting Nodes in a Linked List</i>	600
12.2.5	<i>Splicing Nodes into a Linked List</i>	602
12.2.6	<i>Doubly and Circularly Linked Lists, Header Nodes</i>	606
12.3	A Templated Class for Sets	610
12.3.1	<i>Sets of Strings with Linked Lists</i>	611
12.3.2	<i>Searching, Clearing, Helper Functions</i>	612
12.3.3	<i>Iterators and Friend Functions</i>	614
12.3.4	<i>Interactive Testing</i>	616
12.3.5	<i>Deep Copy, Assignment, and Destruction</i>	620
12.3.6	<i>A Templated Version of LinkStringSet</i>	625

12.4 Chapter Review	632
12.5 Exercises	634
13 Inheritance for Object-Oriented Design	641
13.1 Essential Aspects of Inheritance	641
13.1.1 <i>The Inheritance Hierarchy for Streams</i>	642
13.1.2 <i>An Inheritance Hierarchy: Math Quiz Questions</i>	644
13.1.3 <i>Implementing Inheritance</i>	647
13.1.4 <i>Public Inheritance</i>	649
13.1.5 <i>Virtual Functions</i>	650
13.1.6 <i>Protected Data Members</i>	654
13.2 Using an Abstract Base Class	655
13.2.1 <i>Abstract Classes and Pure Virtual Functions</i>	657
13.2.2 <i>When Is a Method virtual?</i>	661
13.3 Advanced Case Study: Gates, Circuits, and Design Patterns	668
13.3.1 <i>An Introduction to Gates and Circuits</i>	668
13.3.2 <i>Wires, Gates, and Probes</i>	670
13.3.3 <i>Composite Gates and Connectors</i>	672
13.3.4 <i>Implementation of the Wire and Gate Classes</i>	680
13.3.5 <i>Gates and Wires: Observers and Observables</i>	682
13.3.6 <i>Encapsulating Construction in WireFactory</i>	684
13.3.7 <i>Refactoring: Creating a BinaryGate Class</i>	686
13.3.8 <i>Interactive Circuit Building</i>	690
13.3.9 <i>SimpleMap: Mapping Names to Gates</i>	695
13.4 Chapter Review	696
13.5 Exercises	697
A How to: Use Basic C++ Syntax and Operators	705
A.1 Syntax	705
A.1.1 <i>The Function main</i>	705
A.1.2 <i>Built-in and Other Types</i>	705
A.1.3 <i>Variable Definition and Assignment</i>	706
A.1.4 <i>C++ Keywords</i>	708
A.1.5 <i>Control Flow</i>	708
A.2 Functions and Classes	711
A.2.1 <i>Defining and Declaring Functions and Classes</i>	711
A.2.2 <i>Importing Classes and Functions: #include</i>	713
A.2.3 <i>Namespaces</i>	713
A.2.4 <i>Operators</i>	716
A.2.5 <i>Characters</i>	716
A.2.6 <i>Command-Line Parameters</i>	716
B How to: Format Output and Use Streams	719
B.1 Formatting Output	719
B.1.1 <i>General and Floating-Point Formatting</i>	719
B.1.2 <i>Manipulators</i>	720

B.1.3	<i>Stream Functions</i>	726
B.2	Random Access Files	727
B.3	I/O Redirection	730
C	How to: Use the Class string	731
C.1	The Class string	731
C.1.1	<i>Basic Operations</i>	731
C.1.2	<i>Conversion to/from C-Style Strings</i>	732
C.2	String Member Functions	732
C.2.1	<i>Adding Characters or Strings</i>	732
C.2.2	<i>Using Substrings</i>	733
C.2.3	<i>Finding (Sub)strings and Characters</i>	735
D	How to: Understand and Use const	737
D.1	Why const?	737
D.1.1	<i>Literal Arguments</i>	738
D.2	const Member Functions	738
D.2.1	<i>Overloading on const</i>	740
D.3	mutable Data	741
D.4	Pointers and const	743
D.5	Summary	744
E	How to: Overload Operators	745
E.1	Overloading Overview	745
E.2	Arithmetic Operators	745
E.2.1	<i>Binary Operators</i>	746
E.2.2	<i>Arithmetic Assignment Operators</i>	748
E.3	Relational Operators	750
E.4	I/O Operators	753
E.4.1	<i>The Function tostring()</i>	754
E.5	Constructors and Conversions	755
F	How to: Understand and Use Standard Libraries	757
F.1	Functions	757
F.1.1	<i>The Library <cmath></i>	757
F.1.2	<i>The Library <cctype></i>	758
F.2	Constants and Limits	758
F.2.1	<i>Limits in <climits></i>	759
F.2.2	<i>double Limits in <cfloat></i>	760
F.2.3	<i>Limits in <limits></i>	761
F.2.4	<i>ASCII Values</i>	763
G	How to: Understand and Use Tapestry Classes	765
G.1	A Library of Useful Classes	765
G.1.1	<i>Summary of Classes and Functions</i>	765
G.1.2	<i>Implementations of Tapestry Classes</i>	766
G.2	Header Files for Tapestry Classes	767
G.2.1	<i>Prompting Functions in prompt.h</i>	767

G.2.2	<i>The Class Date</i>	769
G.2.3	<i>The Class Dice</i>	771
G.2.4	<i>The Class RandGen</i>	772
G.2.5	<i>The Class CTimer</i>	773
G.2.6	<i>The Class WordStreamIterator</i>	773
G.2.7	<i>The Class StringSet</i>	775
G.2.8	<i>The String Functions in strutils.h</i>	776
G.2.9	<i>The Math Helper Functions in mathutils.h</i>	776
G.2.10	<i>The struct Point</i>	777
G.2.11	<i>The Classes in directory.h</i>	778
G.2.12	<i>The Class CList</i>	781
G.2.13	<i>The Class Poly</i>	784
G.2.14	<i>The Sorting Functions in sortall.h</i>	785
G.2.15	<i>Header Files from Circuit Simulation</i>	787
G.2.16	<i>The Map Class SimpleMap</i>	792
H	How to: Use the Graphics Classes in canvas.h	795
H.1	<i>The Graphics Library: TOOGL 1.0</i>	795
H.2	<i>Using the Canvas Class</i>	796
H.2.1	<i>Canvas Basics</i>	796
H.2.2	<i>Drawing Styles, and Colors</i>	796
H.2.3	<i>Drawing Shapes and Text</i>	798
H.3	<i>Using the AnimatedCanvas Class</i>	801
H.3.1	<i>The Shape Hierarchy</i>	801
H.3.2	<i>Properties of Shape Objects</i>	802
H.3.3	<i>Using Shapes: addShape and clone</i>	803
H.3.4	<i>The CompositeShape Class</i>	805
H.3.5	<i>Processing Mouse and Key Events</i>	807
H.3.6	<i>Animations with Bouncer and Mover</i>	810
H.3.7	<i>Canvas Iterator</i>	813
H.3.8	<i>Specifying Color with Class CanvasColor</i>	815
H.3.9	<i>The Class Key in key.h</i>	816
I	How to: Cope with C++ Environments	817
I.1	<i>Coping with Compilers</i>	817
I.1.1	<i>Keeping Current</i>	818
I.2	<i>Creating a C++ Program</i>	818
I.2.1	<i>The Preprocessor</i>	818
I.2.2	<i>The Compiler</i>	820
I.2.3	<i>The Linker</i>	820
	<i>Bibliography</i>	821
	<i>Photo Credits</i>	825
	<i>Index</i>	826