

Ninth Edition

PROBLEM SOLVING with C++

Ninth Edition

PROBLEM SOLVING

with

C++

Walter Savitch

UNIVERSITY OF CALIFORNIA, SAN DIEGO

CONTRIBUTOR

Kenrick Mock

UNIVERSITY OF ALASKA, ANCHORAGE

PEARSON

Boston Columbus Indianapolis New York San Francisco Upper Saddle River
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto
Delhi Mexico City São Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

Editorial Director: *Marcia Horton*
Acquisitions Editor: *Matt Goldstein*
Program Manager: *Kayla Smith-Tarbox*
Editorial Assistant: *Kelsey Loanes*
Marketing Coordinator: *Kathryn Ferranti*
Production Director: *Erin Gregg*
Managing Editor: *Scott Disanno*
Senior Operations Supervisor: *Vincent Scelta*
Operations Specialist: *Linda Sager*
Cover Designer: *Joyce Wells*
Permissions Manager: *Timothy Nicholls*
Image Permissions Manager: *Karen Sanatar*
Media Producer: *Renata Butera*
Media Project Manager: *Wanda Rockwell*
Full-Service Vendor: *Hardik Popli, Cenveo® Publisher Services*
Composition: *Cenveo Publisher Services*
Printer/Binder: *Courier/Westford*
Cover Printer: *Lehigh-Phoenix Color/Hagerstown*

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on appropriate page within text.

Microsoft® and Windows® are registered trademarks of the Microsoft Corporation in the U.S.A. and other countries. Screen shots and icons reprinted with permission from the Microsoft Corporation. This book is not sponsored or endorsed by or affiliated with the Microsoft Corporation.

Copyright © 2015, 2012, 2009, 2007, 2005, 2003 Pearson Education, Inc. All rights reserved. Manufactured in the United States of America. This publication is protected by Copyright, and permission should be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission(s) to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, 501 Boylston Street, Suite 900, Boston, Massachusetts 02116.

Many of the designations by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed in initial caps or all caps.

Library of Congress Cataloging-in-Publication Data

Savitch, Walter J., 1943-
Problem solving with C++ / Walter Savitch ; contributor, Kenrick Mock. -- Ninth edition.
pages cm
Includes index.
ISBN-13: 978-0-13-359174-3 (alkaline paper)
ISBN-10: 0-13-359174-3 (alkaline paper)
1. C++ (Computer program language) 2. Problem solving. I. Mock, Kenrick. II. Title.
QA76.73.C153S29 2014
005.13'3--dc23

2013048487

10 9 8 7 6 5 4 3 2 1—CW—15 14 13 12 11



www.pearsonhighered.com

ISBN 10: 0-13-359174-3

ISBN 13: 978-0-13-359174-3

Preface

This book is meant to be used in a first course in programming and computer science using the C++ language. It assumes no previous programming experience and no mathematics beyond high school algebra.

If you have used the previous edition of this book, you should read the following section that explains the changes to this ninth edition and then you can skip the rest of this preface. If you are new to this book, the rest of this preface will give you an overview of the book.

Changes to the Ninth Edition

This ninth edition presents the same programming philosophy as the eighth edition. All of the material from the eighth edition remains, but with the following enhancements:

- End-of-chapter programs are now split into Practice Programs and Programming Projects. Practice Programs require a direct application of concepts presented in the chapter and solutions are usually short. Practice Programs are appropriate for laboratory exercises. Programming Projects require additional problem solving and solutions are generally longer than Practice Programs. Programming Projects are appropriate for homework problems.
- Introduction to C++11 in the context of C++98. Examples of C++11 content includes new integer types, the auto type, raw string literals, strong enumerations, `nullptr`, `ranged for loop`, conversion between strings and integers, member initializers, and constructor delegation.
- Additional material on sorting, secure programming (e.g., overflow, array out of bounds), and inheritance.
- Correction of errata.
- Twenty-one new Practice Programs and ten new Programming Projects.
- Ten new VideoNotes for a total of sixty-four VideoNotes. These VideoNotes walk students through the process of both problem solving and coding to help reinforce key programming concepts. An icon appears in the margin of the book when a VideoNote is available regarding the topic covered in the text.

If you are an instructor already using the eighth edition, you can continue to teach your course almost without change.

Flexibility in Topic Ordering

This book was written to allow instructors wide latitude in reordering the material. To illustrate this flexibility, we suggest two alternative ways to order

the topics. There is no loss of continuity when the book is read in either of these ways. To ensure this continuity when you rearrange material, you may need to move sections rather than entire chapters. However, only large sections in convenient locations are moved. To help customize a particular order for any class's needs, the end of this preface contains a dependency chart, and each chapter has a "Prerequisites" section that explains what material needs to be covered before each section in that chapter.

Reordering 1: Earlier Classes

To effectively design classes, a student needs some basic tools such as control structures and function definitions. This basic material is covered in Chapters 1 through 6. After completing Chapter 6, students can begin to write their own classes. One possible reordering of chapters that allows for such early coverage of classes is the following:

Basics: Chapters 1, 2, 3, 4, 5, and 6. This material covers all control structures, function definitions, and basic file I/O. Chapter 3, which covers additional control structures, could be deferred if you wish to cover classes as early as possible.

Classes and namespaces: Chapter 10, Sections 11.1 and 11.2 of Chapter 11, and Chapter 12. This material covers defining classes, friends, overloaded operators, and namespaces.

Arrays, strings and vectors: Chapters 7 and 8

Pointers and dynamic arrays: Chapter 9

Arrays in classes: Sections 11.3 and 11.4 of Chapter 11

Inheritance: Chapter 15

Recursion: Chapter 14 (Alternately, recursion may be moved to later in the course.)

Pointers and linked lists: Chapter 13

Any subset of the following chapters may also be used:

Exception handling: Chapter 16

Templates: Chapter 17

Standard Template Library: Chapter 18

Reordering 2: Classes Slightly Later but Still Early

This version covers all control structures and the basic material on arrays before doing classes, but classes are covered later than the previous ordering and slightly earlier than the default ordering.

Basics: Chapters 1, 2, 3, 4, 5, and 6. This material covers all control structures, function definitions, and the basic file I/O.

Arrays and strings: Chapter 7, Sections 8.1 and 8.2 of Chapter 8

Classes and namespaces: Chapter 10, Sections 11.1 and 11.2 of Chapter 11, and Chapter 12. This material covers defining classes, friends, overloaded operators, and namespaces.

Pointers and dynamic arrays: Chapter 9

Arrays in classes: Sections 11.3 and 11.4 of Chapter 11

Inheritance: Chapter 15

Recursion: Chapter 14. (Alternately, recursion may be moved to later in the course.)

Vectors: Chapter 8.3

Pointers and linked lists: Chapter 13

Any subset of the following chapters may also be used:

Exception handling: Chapter 16

Templates: Chapter 17

Standard Template Library: Chapter 18

Accessibility to Students

It is not enough for a book to present the right topics in the right order. It is not even enough for it to be clear and correct when read by an instructor or other experienced programmer. The material needs to be presented in a way that is accessible to beginning students. In this introductory textbook, I have endeavored to write in a way that students find clear and friendly. Reports from the many students who have used the earlier editions of this book confirm that this style makes the material clear and often even enjoyable to students.

ANSI/ISO C++ Standard

This edition is fully compatible with compilers that meet the latest ANSI/ISO C++ standard. At the time of this writing the latest standard is C++11.

Advanced Topics

Many “advanced topics” are becoming part of a standard CS1 course. Even if they are not part of a course, it is good to have them available in the text as enrichment material. This book offers a number of advanced topics that can be integrated into a course or left as enrichment topics. It gives thorough coverage of C++ templates, inheritance (including virtual functions), exception handling, and the STL (Standard Template Library). Although this book uses libraries and teaches students the importance of libraries, it does not require any nonstandard libraries. This book uses only libraries that are provided with essentially all C++ implementations.

Dependency Chart

The dependency chart on the next page shows possible orderings of chapters and subsections. A line joining two boxes means that the upper box must be covered before the lower box. Any ordering that is consistent with this partial ordering can be read without loss of continuity. If a box contains a section number or numbers, then the box refers only to those sections and not to the entire chapter.

Summary Boxes

Each major point is summarized in a boxed section. These boxed sections are spread throughout each chapter.

Self-Test Exercises

Each chapter contains numerous Self-Test Exercises at strategic points. Complete answers for all the Self-Test Exercises are given at the end of each chapter.



VideoNote

VideoNotes

VideoNotes are designed for teaching students key programming concepts and techniques. These short step-by-step videos demonstrate how to solve problems from design through coding. VideoNotes allow for self-paced instruction with easy navigation including the ability to select, play, rewind, fast-forward, and stop within each VideoNote exercise.

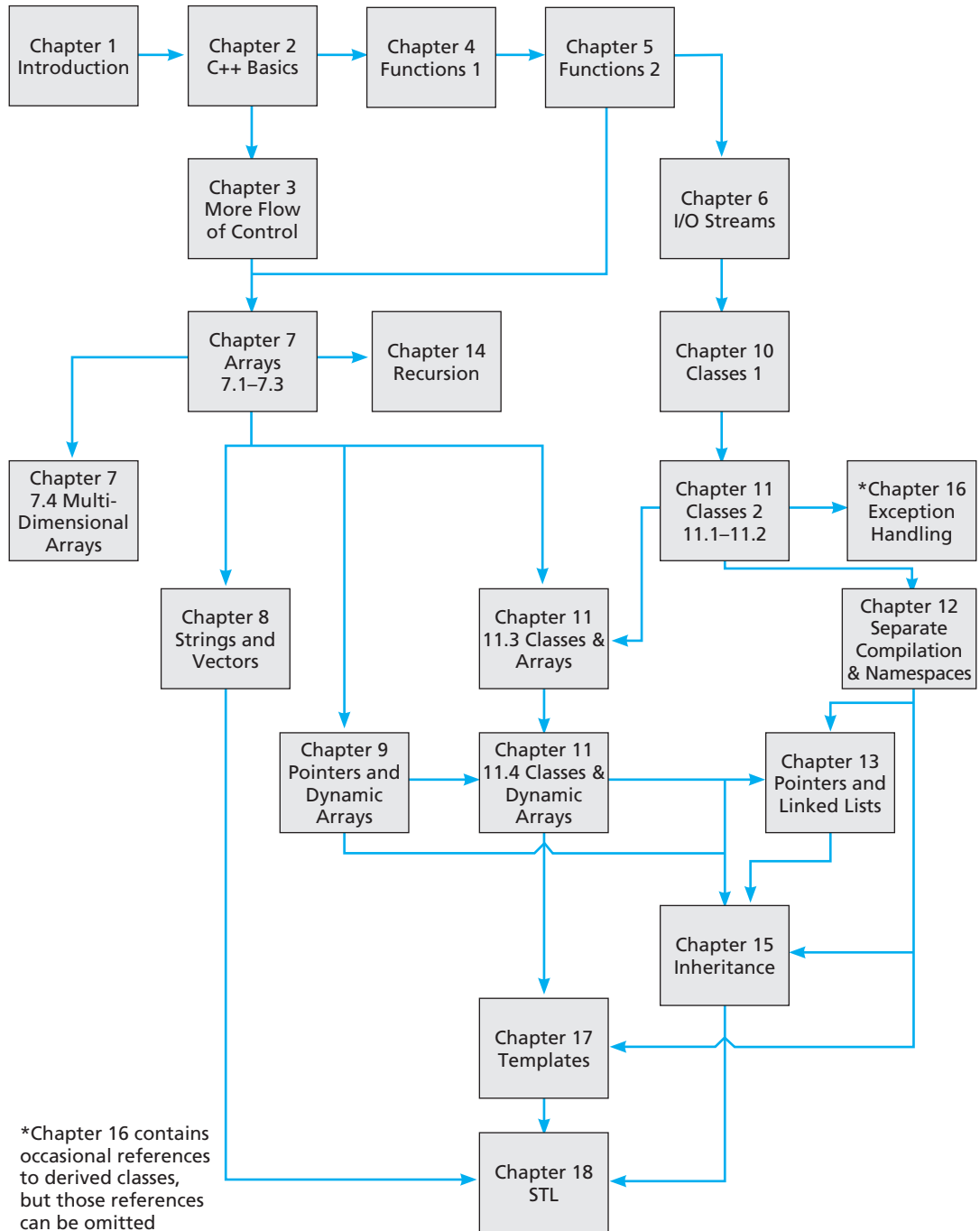
Online Practice and Assessment with MyProgrammingLab

MyProgrammingLab helps students fully grasp the logic, semantics, and syntax of programming. Through practice exercises and immediate, personalized feedback, MyProgrammingLab improves the programming competence of beginning students who often struggle with the basic concepts and paradigms of popular high-level programming languages.

A self-study and homework tool, a MyProgrammingLab course consists of hundreds of small practice problems organized around the structure of this textbook. For students, the system automatically detects errors in the logic and syntax of their code submissions and offers targeted hints that enable students to figure out what went wrong—and why. For instructors, a comprehensive gradebook tracks correct and incorrect answers and stores the code inputted by students for review.

MyProgrammingLab is offered to users of this book in partnership with Turing's Craft, the makers of the CodeLab interactive programming exercise system. For a full demonstration, to see feedback from instructors and students, or to get started using MyProgrammingLab in your course, visit www.myprogramminglab.com.

DISPLAY P.1 Dependency Chart



Support Material

There is support material available to all users of this book and additional material available only to qualified instructors.

Materials Available to All Users of this Book

- Source Code from the book
- PowerPoint slides
- VideoNotes

To access these materials, go to:
www.pearsonhighered.com/savitch

Resources Available to Qualified Instructors Only

Visit Pearson Education's instructor resource center at www.pearsonhighered.com/irc to access the following instructor resources:

- Instructor's Resource Guide—including chapter-by-chapter teaching hints, quiz questions with solutions, and solutions to many programming projects
- Test Bank and Test Generator
- PowerPoint Lectures—including programs and art from the text
- Lab Manual

Integrated Development Environment (IDE) Resource Kits

Instructors who adopt this text can order it for students with a kit containing five popular C++ IDEs (Microsoft® Visual Studio 2013 Express Edition, Dev C++, NetBeans, Eclipse, and CodeLite) and access to a Web site containing written and video tutorials for getting started in each IDE. For ordering information, please contact your campus Pearson Education representative.

Contact Us

Your comments, suggestions, questions, and corrections are always welcome. Please e-mail them to savitch.programming.cpp@gmail.com

Acknowledgments

Numerous individuals and groups have provided me with suggestions, discussions, and other help in preparing this textbook. Much of the first edition of this book was written while I was visiting the Computer Science Department at the University of Colorado in Boulder. The remainder of the writing on the first edition and the work on subsequent editions was done in the Computer Science and Engineering Department at the University of California, San Diego (UCSD). I am grateful to these institutions for providing a conducive environment for teaching this material and writing this book.

I extend a special thanks to all the individuals who have contributed critiques or programming projects for this or earlier editions and drafts of this book. In alphabetical order, they are: Alex Feldman, Amber Settle, Andrew Burt, Andrew Haas, Anne Marchant, Barney MacCabe, Bob Holloway, Bob Matthews, Brian R. King, Bruce Johnston, Carol Roberts, Charles Dowling, Claire Bono, Cynthia Martincic, David Feinstein, David Teague, Dennis Heckman, Donald Needham, Doug Cosman, Dung Nguyen, Edward Carr, Eitan M. Gurari, Ethan Munson, Firooz Khosraviyani, Frank Moore, Gilliean Lee, Huzefa Kagdi, James Stepleton, Jeff Roach, Jeffrey Watson, Jennifer Perkins, Jerry Weltman, Joe Faletti, Joel Cohen, John J. Westman, John Marsaglia, John Russo, Joseph Allen, Joseph D. Oldham, Jerrold Grossman, Jesse Morehouse, Karla Chaveau, Ken Rockwood, Larry Johnson, Len Garrett, Linda F. Wilson, Mal Gunasekera, Marianne Lepp, Matt Johnson, Michael Keenan, Michael Main, Michal Sramka, Naomi Shapiro, Nat Martin, Noah Aydin, Nisar Hundewale, Paul J. Kaiser, Paul Kube, Paulo Franca, Richard Borie, Scot Drysdale, Scott Strong, Sheila Foster, Steve Mahaney, Susanne Sherba, Thomas Judson, Walter A. Manrique, Wei Lian Chen, and Wojciech Komornicki.

I extend a special thanks to the many instructors who used early editions of this book. Their comments provided some of the most helpful reviewing that the book received.

Finally, I thank Kenrick Mock who implemented the changes in this edition. He had the almost impossible task of pleasing me, my editor, and his own sensibilities, and he did a superb job of it.

Walter Savitch

BREAKTHROUGH

To improving results



get with the programming

Through the power of practice and immediate personalized feedback, MyProgrammingLab improves your performance.

MyProgrammingLab[™]

Learn more at www.myprogramminglab.com

Brief Contents

Table of Location of VideoNotes

Inside front cover and inside back cover

Chapter 1	Introduction to Computers and C++ Programming	1
Chapter 2	C++ Basics	39
Chapter 3	More Flow of Control	111
Chapter 4	Procedural Abstraction and Functions That Return a Value	181
Chapter 5	Functions for All Subtasks	251
Chapter 6	I/O Streams as an Introduction to Objects and Classes	305
Chapter 7	Arrays	377
Chapter 8	Strings and Vectors	451
Chapter 9	Pointers and Dynamic Arrays	507
Chapter 10	Defining Classes	541
Chapter 11	Friends, Overloaded Operators, and Arrays in Classes	619

Chapter 12 **Separate Compilation and Namespaces** 703

Chapter 13 **Pointers and Linked Lists** 739

Chapter 14 **Recursion** 789

Chapter 15 **Inheritance** 833

Chapter 16 **Exception Handling** 893

Chapter 17 **Templates** 925

Chapter 18 **Standard Template Library** 957

Appendices

- 1** **C++ Keywords** 1015
- 2** **Precedence of Operators** 1016
- 3** **The ASCII Character Set** 1018
- 4** **Some Library Functions** 1019
- 5** **Inline Functions** 1026
- 6** **Overloading the Array Index**
Square Brackets 1027
- 7** **The `this` Pointer** 1029
- 8** **Overloading Operators as Member**
Operators 1032

Index 1034

Contents

Table of Location of VideoNotes

Inside front cover and inside back cover

Chapter 1 Introduction to Computers and C++ Programming 1

1.1 COMPUTER SYSTEMS 2

Hardware 2
Software 7
High-Level Languages 8
Compilers 9
History Note 12

1.2 PROGRAMMING AND PROBLEM-SOLVING 12

Algorithms 12
Program Design 15
Object-Oriented Programming 16
The Software Life Cycle 17

1.3 INTRODUCTION TO C++ 18

Origins of the C++ Language 18
A Sample C++ Program 19
Pitfall: Using the Wrong Slash in `\n` 23
Programming Tip: Input and Output Syntax 23
Layout of a Simple C++ Program 24
Pitfall: Putting a Space Before the include File Name 26
Compiling and Running a C++ Program 26
Pitfall: Compiling a C++11 program 27
Programming Tip: Getting Your Program to Run 27

1.4 TESTING AND DEBUGGING 29

Kinds of Program Errors 30
Pitfall: Assuming Your Program Is Correct 31

Chapter Summary	32
Answers to Self-Test Exercises	33
Practice Programs	35
Programming Projects	36

Chapter 2 C++ Basics 39

2.1 VARIABLES AND ASSIGNMENTS 40

Variables	40
Names: Identifiers	42
Variable Declarations	44
Assignment Statements	45
<i>Pitfall:</i> Uninitialized Variables	47
<i>Programming Tip:</i> Use Meaningful Names	49

2.2 INPUT AND OUTPUT 50

Output Using <code>cout</code>	50
Include Directives and Namespaces	52
Escape Sequences	53
<i>Programming Tip:</i> End Each Program with a <code>\n</code> or <code>endl</code>	55
Formatting for Numbers with a Decimal Point	55
Input Using <code>cin</code>	56
Designing Input and Output	58
<i>Programming Tip:</i> Line Breaks in I/O	58

2.3 DATA TYPES AND EXPRESSIONS 60

The Types <code>int</code> and <code>double</code>	60
Other Number Types	62
C++11 Types	63
The Type <code>char</code>	64
The Type <code>bool</code>	66
Introduction to the Class <code>string</code>	66
Type Compatibilities	68
Arithmetic Operators and Expressions	69
<i>Pitfall:</i> Whole Numbers in Division	72
More Assignment Statements	74

2.4 SIMPLE FLOW OF CONTROL 74

A Simple Branching Mechanism	75
<i>Pitfall:</i> Strings of Inequalities	80
<i>Pitfall:</i> Using <code>=</code> in place of <code>==</code>	81
Compound Statements	82

Simple Loop Mechanisms 84
Increment and Decrement Operators 87
Programming Example: Charge Card Balance 89
Pitfall: Infinite Loops 90

2.5 PROGRAM STYLE 93

Indenting 93
Comments 93
Naming Constants 95

Chapter Summary 98
Answers to Self-Test Exercises 98
Practice Programs 103
Programming Projects 105

Chapter 3 More Flow of Control 111

3.1 USING BOOLEAN EXPRESSIONS 112

Evaluating Boolean Expressions 112
Pitfall: Boolean Expressions Convert to `int` Values 116
Enumeration Types (*Optional*) 119

3.2 MULTIWAY BRANCHES 120

Nested Statements 120
Programming Tip: Use Braces in Nested Statements 121
Multiway *if-else* Statements 123
Programming Example: State Income Tax 125
The *switch* Statement 128
Pitfall: Forgetting a `break` in a `switch` Statement 132
Using *switch* Statements for Menus 133
Blocks 135
Pitfall: Inadvertent Local Variables 138

3.3 MORE ABOUT C++ LOOP STATEMENTS 139

The *while* Statements Reviewed 139
Increment and Decrement Operators Revisited 141
The *for* Statement 144
Pitfall: Extra Semicolon in a `for` Statement 149
What Kind of Loop to Use 150
Pitfall: Uninitialized Variables and Infinite Loops 152
The *break* Statement 153
Pitfall: The `break` Statement in Nested Loops 154

3.4 DESIGNING LOOPS 155

Loops for Sums and Products 155

Ending a Loop 157

Nested Loops 160

Debugging Loops 162

Chapter Summary 165

Answers to Self-Test Exercises 166

Practice Programs 172

Programming Projects 174

**Chapter 4 Procedural Abstraction and Functions
That Return a Value 181****4.1 TOP-DOWN DESIGN 182****4.2 PREDEFINED FUNCTIONS 183**

Using Predefined Functions 183

Random Number Generation 188

Type Casting 190

Older Form of Type Casting 192

Pitfall: Integer Division Drops the Fractional Part 192**4.3 PROGRAMMER-DEFINED FUNCTIONS 193**

Function Definitions 193

Functions That Return a Boolean Value 199

Alternate Form for Function Declarations 199

Pitfall: Arguments in the Wrong Order 200

Function Definition–Syntax Summary 201

More About Placement of Function Definitions 202

Programming Tip: Use Function Calls in Branching Statements 203**4.4 PROCEDURAL ABSTRACTION 204**

The Black-Box Analogy 204

Programming Tip: Choosing Formal Parameter Names 207*Programming Tip:* Nested Loops 208*Case Study:* Buying Pizza 211*Programming Tip:* Use Pseudocode 217**4.5 SCOPE AND LOCAL VARIABLES 218**

The Small Program Analogy 218

Programming Example: Experimental Pea Patch 221

Global Constants and Global Variables 221
Call-by-Value Formal Parameters Are Local Variables 224
Block Scope 226
Namespaces Revisited 227
Programming Example: The Factorial Function 230

4.6 OVERLOADING FUNCTION NAMES 232

Introduction to Overloading 232
Programming Example: Revised Pizza-Buying Program 235
Automatic Type Conversion 238

Chapter Summary 240
Answers to Self-Test Exercises 240
Practice Programs 245
Programming Projects 247

Chapter 5 Functions for All Subtasks 251

5.1 void FUNCTIONS 252

Definitions of *void* Functions 252
Programming Example: Converting Temperatures 255
return Statements in *void* Functions 255

5.2 CALL-BY-REFERENCE PARAMETERS 259

A First View of Call-by-Reference 259
Call-by-Reference in Detail 262
Programming Example: The swap_values Function 267
Mixed Parameter Lists 268
Programming Tip: What Kind of Parameter to Use 269
Pitfall: Inadvertent Local Variables 270

5.3 USING PROCEDURAL ABSTRACTION 273

Functions Calling Functions 273
Preconditions and Postconditions 275
Case Study: Supermarket Pricing 276

5.4 TESTING AND DEBUGGING FUNCTIONS 281

Stubs and Drivers 282

5.5 GENERAL DEBUGGING TECHNIQUES 287

Keep an Open Mind 287
Check Common Errors 287

Localize the Error	288
The assert Macro	290
Chapter Summary	292
Answers to Self-Test Exercises	293
Practice Programs	296
Programming Projects	299

Chapter 6 I/O Streams as an Introduction to Objects and Classes 305

6.1 STREAMS AND BASIC FILE I/O 306

Why Use Files for I/O?	307
File I/O	308
Introduction to Classes and Objects	312
<i>Programming Tip: Check Whether a File Was Opened Successfully</i>	314
Techniques for File I/O	316
Appending to a File (<i>Optional</i>)	320
File Names as Input (<i>Optional</i>)	321

6.2 TOOLS FOR STREAM I/O 323

Formatting Output with Stream Functions	323
Manipulators	329
Streams as Arguments to Functions	332
<i>Programming Tip: Checking for the End of a File</i>	332
A Note on Namespaces	335
<i>Programming Example: Cleaning Up a File Format</i>	336

6.3 CHARACTER I/O 338

The Member Functions get and put	338
The putback Member Function (<i>Optional</i>)	342
<i>Programming Example: Checking Input</i>	343
<i>Pitfall: Unexpected '\n' in Input</i>	345
<i>Programming Example: Another new_line Function</i>	347
Default Arguments for Functions (<i>Optional</i>)	348
The eof Member Function	353
<i>Programming Example: Editing a Text File</i>	355
Predefined Character Functions	356
<i>Pitfall: toupper and tolower Return Values</i>	358

Chapter Summary	360
Answers to Self-Test Exercises	361
Practice Programs	368
Programming Projects	370

Chapter 7 Arrays 377

7.1 INTRODUCTION TO ARRAYS 378

Declaring and Referencing Arrays	378
<i>Programming Tip:</i> Use <i>for</i> Loops with Arrays	380
<i>Pitfall:</i> Array Indexes Always Start with Zero	380
<i>Programming Tip:</i> Use a Defined <i>Constant</i> for the Size of an Array	380
Arrays in Memory	382
<i>Pitfall:</i> Array Index Out of Range	383
Initializing Arrays	386
<i>Programming Tip:</i> C++11 Range-Based <i>for</i> Statement	386

7.2 ARRAYS IN FUNCTIONS 389

Indexed Variables as Function Arguments	389
Entire Arrays as Function Arguments	391
The <i>const</i> Parameter Modifier	394
<i>Pitfall:</i> Inconsistent Use of <i>const</i> Parameters	397
Functions That Return an Array	397
<i>Case Study:</i> Production Graph	398

7.3 PROGRAMMING WITH ARRAYS 411

Partially Filled Arrays	411
<i>Programming Tip:</i> Do Not Skimp on Formal Parameters	414
<i>Programming Example:</i> Searching an Array	414
<i>Programming Example:</i> Sorting an Array	417
<i>Programming Example:</i> Bubble Sort	421

7.4 MULTIDIMENSIONAL ARRAYS 424

Multidimensional Array Basics	425
Multidimensional Array Parameters	425
<i>Programming Example:</i> Two-Dimensional Grading Program	427
<i>Pitfall:</i> Using Commas Between Array Indexes	431

Chapter Summary	432
Answers to Self-Test Exercises	433
Practice Programs	437
Programming Projects	439

Chapter 8 Strings and Vectors 451

8.1 AN ARRAY TYPE FOR STRINGS 453

C-String Values and C-String Variables	453
<i>Pitfall:</i> Using = and == with C Strings	456
Other Functions in <cstring>	458
<i>Pitfall:</i> Copying past the end of a C-string using strcpy	461
C-String Input and Output	464
C-String-to-Number Conversions and Robust Input	466

8.2 THE STANDARD string CLASS 472

Introduction to the Standard Class string	472
I/O with the Class string	475
<i>Programming Tip:</i> More Versions of getline	478
<i>Pitfall:</i> Mixing cin >> variable; and getline	479
String Processing with the Class string	480
<i>Programming Example:</i> Palindrome Testing	484
Converting Between string Objects and C Strings	487
Converting Between Strings and Numbers	488

8.3 VECTORS 489

Vector Basics	489
<i>Pitfall:</i> Using Square Brackets Beyond the Vector Size	492
<i>Programming Tip:</i> Vector Assignment Is Well Behaved	493
Efficiency Issues	493

Chapter Summary	495
Answers to Self-Test Exercises	495
Practice Programs	497
Programming Projects	498

Chapter 9 Pointers and Dynamic Arrays 507

9.1 POINTERS 508

Pointer Variables	509
Basic Memory Management	516

Pitfall: Dangling Pointers 517
Static Variables and Automatic Variables 518
Programming Tip: Define Pointer Types 518

9.2 DYNAMIC ARRAYS 521

Array Variables and Pointer Variables 521
Creating and Using Dynamic Arrays 522
Pointer Arithmetic (*Optional*) 528
Multidimensional Dynamic Arrays (*Optional*) 530

Chapter Summary 532
Answers to Self-Test Exercises 532
Practice Programs 533
Programming Projects 534

Chapter 10 Defining Classes 541

10.1 STRUCTURES 542

Structures for Diverse Data 542
Pitfall: Forgetting a Semicolon in a Structure Definition 547
Structures as Function Arguments 548
Programming Tip: Use Hierarchical Structures 549
Initializing Structures 551

10.2 CLASSES 554

Defining Classes and Member Functions 554
Public and Private Members 559
Programming Tip: Make All Member Variables Private 567
Programming Tip: Define Accessor and Mutator Functions 567
Programming Tip: Use the Assignment Operator
with Objects 569
Programming Example: BankAccount Class—Version 1 570
Summary of Some Properties of Classes 574
Constructors for Initialization 576
Programming Tip: Always Include a Default Constructor 584
Pitfall: Constructors with No Arguments 585
Member Initializers and Constructor Delegation in C++11 587

10.3 ABSTRACT DATA TYPES 588

Classes to Produce Abstract Data Types 589
Programming Example: Alternative Implementation of a Class 593

10.4 INTRODUCTION TO INHERITANCE 598

Derived Classes 599

Defining Derived Classes 600

Chapter Summary 604

Answers to Self-Test Exercises 605

Practice Programs 611

Programming Projects 612

**Chapter 11 Friends, Overloaded Operators, and
Arrays in Classes 619****11.1 FRIEND FUNCTIONS 620**

Programming Example: An Equality Function 620

Friend Functions 624

Programming Tip: Define Both Accessor Functions and Friend Functions 626

Programming Tip: Use Both Member and Nonmember Functions 628

Programming Example: Money Class (Version 1) 628

Implementation of `digit_to_int` (*Optional*) 635

Pitfall: Leading Zeros in Number Constants 636

The `const` Parameter Modifier 638

Pitfall: Inconsistent Use of `const` 639

11.2 OVERLOADING OPERATORS 643

Overloading Operators 644

Constructors for Automatic Type Conversion 647

Overloading Unary Operators 649

Overloading `>>` and `<<` 650

11.3 ARRAYS AND CLASSES 660

Arrays of Classes 660

Arrays as Class Members 664

Programming Example: A Class for a Partially Filled Array 665

11.4 CLASSES AND DYNAMIC ARRAYS 667

Programming Example: A String Variable Class 668

Destructors 671

Pitfall: Pointers as Call-by-Value Parameters 674

Copy Constructors	675
Overloading the Assignment Operator	680
Chapter Summary	683
Answers to Self-Test Exercises	683
Practice Programs	693
Programming Projects	694

Chapter 12 Separate Compilation and Namespaces 703

12.1 SEPARATE COMPILATION 704

ADTs Reviewed	705
<i>Case Study:</i> DigitalTime —A Class Compiled Separately	706
Using <code>#ifndef</code>	715
<i>Programming Tip:</i> Defining Other Libraries	718

12.2 NAMESPACES 719

Namespaces and <i>using</i> Directives	719
Creating a Namespace	721
Qualifying Names	724
A Subtle Point About Namespaces (<i>Optional</i>)	725
Unnamed Namespaces	726
<i>Programming Tip:</i> Choosing a Name for a Namespace	731
<i>Pitfall:</i> Confusing the Global Namespace and the Unnamed Namespace	732
Chapter Summary	733
Answers to Self-Test Exercises	734
Practice Programs	736
Programming Projects	738

Chapter 13 Pointers and Linked Lists 739

13.1 NODES AND LINKED LISTS 740

Nodes	740
<code>nullptr</code>	745
Linked Lists	746
Inserting a Node at the Head of a List	747
<i>Pitfall:</i> Losing Nodes	750
Searching a Linked List	751

Pointers as Iterators 755
Inserting and Removing Nodes Inside a List 755
*Pitfall: Using the Assignment Operator with Dynamic
Data Structures* 757
Variations on Linked Lists 760
Linked Lists of Classes 762

13.2 STACKS AND QUEUES 765

Stacks 765
Programming Example: A Stack Class 766
Queues 771
Programming Example: A Queue Class 772

Chapter Summary 776
Answers to Self-Test Exercises 776
Practice Programs 779
Programming Projects 780

Chapter 14 Recursion 789

14.1 RECURSIVE FUNCTIONS FOR TASKS 791

Case Study: Vertical Numbers 791
A Closer Look at Recursion 797
Pitfall: Infinite Recursion 799
Stacks for Recursion 800
Pitfall: Stack Overflow 802
Recursion Versus Iteration 802

14.2 RECURSIVE FUNCTIONS FOR VALUES 804

General Form for a Recursive Function That Returns a Value 804
Programming Example: Another Powers Function 804

14.3 THINKING RECURSIVELY 809

Recursive Design Techniques 809
Case Study: Binary Search—An Example of Recursive Thinking 810
Programming Example: A Recursive Member Function 818

Chapter Summary 822
Answers to Self-Test Exercises 822
Practice Programs 827
Programming Projects 827

Chapter 15 Inheritance 833

15.1 INHERITANCE BASICS 834

Derived Classes 837

Constructors in Derived Classes 845

Pitfall: Use of Private Member Variables from the Base Class 848

Pitfall: Private Member Functions Are Effectively Not Inherited 850

The *protected* Qualifier 850

Redefinition of Member Functions 853

Redefining Versus Overloading 856

Access to a Redefined Base Function 858

15.2 INHERITANCE DETAILS 859

Functions That Are Not Inherited 859

Assignment Operators and Copy Constructors in Derived Classes 860

Destructors in Derived Classes 861

15.3 POLYMORPHISM 862

Late Binding 863

Virtual Functions in C++ 864

Virtual Functions and Extended Type Compatibility 869

Pitfall: The Slicing Problem 873

Pitfall: Not Using Virtual Member Functions 874

Pitfall: Attempting to Compile Class Definitions Without Definitions for Every Virtual Member Function 875

Programming Tip: Make Destructors Virtual 875

Chapter Summary 877

Answers to Self-Test Exercises 877

Practice Programs 881

Programming Projects 884

Chapter 16 Exception Handling 893

16.1 EXCEPTION-HANDLING BASICS 895

A Toy Example of Exception Handling 895

Defining Your Own Exception Classes 904

Multiple Throws and Catches 904

Pitfall: Catch the More Specific Exception First 908

Programming Tip: Exception Classes Can Be Trivial 909

Throwing an Exception in a Function 909

Exception Specification	911
<i>Pitfall:</i> Exception Specification in Derived Classes	913

16.2 PROGRAMMING TECHNIQUES FOR EXCEPTION HANDLING 914

When to Throw an Exception	914
<i>Pitfall:</i> Uncaught Exceptions	916
<i>Pitfall:</i> Nested <i>try-catch</i> Blocks	916
<i>Pitfall:</i> Overuse of Exceptions	916
Exception Class Hierarchies	917
Testing for Available Memory	917
Rethrowing an Exception	918
Chapter Summary	918
Answers to Self-Test Exercises	918
Practice Programs	920
Programming Projects	921

Chapter 17 Templates 925

17.1 TEMPLATES FOR ALGORITHM ABSTRACTION 926

Templates for Functions	927
<i>Pitfall:</i> Compiler Complications	931
<i>Programming Example:</i> A Generic Sorting Function	933
<i>Programming Tip:</i> How to Define Templates	937
<i>Pitfall:</i> Using a Template with an Inappropriate Type	938

17.2 TEMPLATES FOR DATA ABSTRACTION 939

Syntax for Class Templates	939
<i>Programming Example:</i> An Array Class	942

Chapter Summary	949
Answers to Self-Test Exercises	949
Practice Programs	953
Programming Projects	953

Chapter 18 Standard Template Library 957

18.1 ITERATORS 959

<i>using</i> Declarations	959
Iterator Basics	960

Programming Tip: Use auto to Simplify Variable Declarations 964

Pitfall: Compiler Problems 964

Kinds of Iterators 966

Constant and Mutable Iterators 970

Reverse Iterators 971

Other Kinds of Iterators 972

18.2 CONTAINERS 973

Sequential Containers 974

Pitfall: Iterators and Removing Elements 978

Programming Tip: Type Definitions in Containers 979

Container Adapters stack and queue 979

Associative Containers set and map 983

*Programming Tip: Use Initialization, Ranged For,
and auto with Containers* 990

Efficiency 990

18.3 GENERIC ALGORITHMS 991

Running Times and Big-*O* Notation 992

Container Access Running Times 995

Nonmodifying Sequence Algorithms 997

Container Modifying Algorithms 1001

Set Algorithms 1003

Sorting Algorithms 1004

Chapter Summary 1005

Answers to Self-Test Exercises 1005

Practice Programs 1007

Programming Projects 1008

APPENDICES

1 C++ Keywords 1015

2 Precedence of Operators 1016

3 The ASCII Character Set 1018

4 Some Library Functions 1019

5 Inline Functions 1026

6 Overloading the Array Index Square Brackets 1027

7 The this Pointer 1029

8 Overloading Operators as Member Operators 1032

INDEX 1034