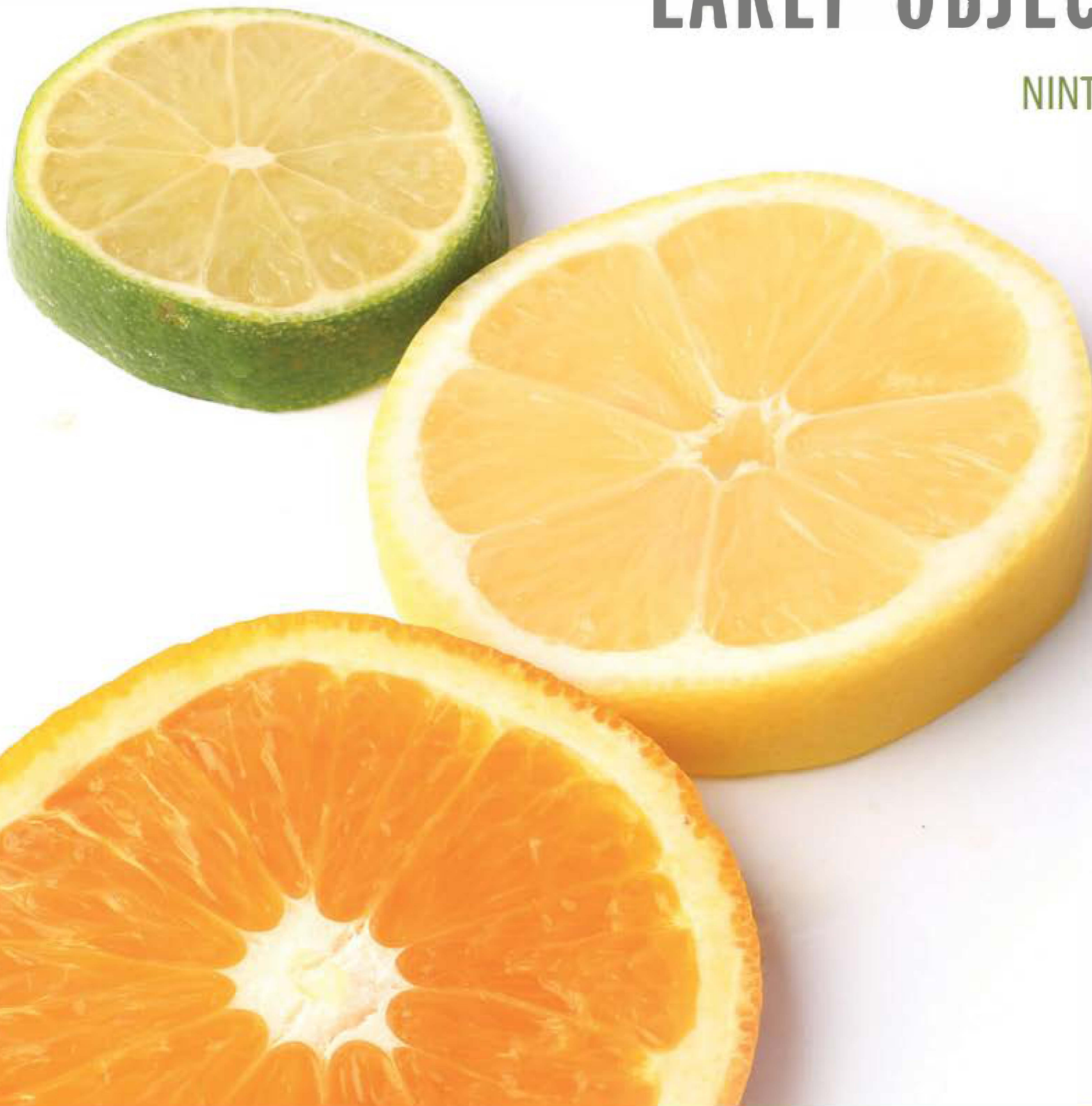


starting out with >>>

# C++

## EARLY OBJECTS

NINTH EDITION



TONY GADDIS • JUDY WALTERS • GODFREY MUGANDA

# Digital Resources for Students

Your new textbook provides 12-month access to digital resources that may include VideoNotes (step-by-step video tutorials on programming concepts), source code, web chapters, quizzes, and more. Refer to the preface in the textbook for a detailed list of resources.

Follow the instructions below to register for the Companion Website for Tony Gaddis, Judy Walters, and Godfrey Muganda's *Staring Out with C++: Early Objects*, Ninth Edition.

1. Go to [www.pearsonhighered.com/cs-resources](http://www.pearsonhighered.com/cs-resources)
2. Enter the title of your textbook or browse by author name.
3. Click Companion Website
4. Click Register and follow the on-screen instructions to create a login name and password.

**Use a coin to scratch off the coating and reveal your access code.  
Do not use a sharp knife or other sharp object as it may damage the code.**

Use the login name and password you created during registration to start using the digital resources that accompany your textbook.

## IMPORTANT:

This prepaid subscription does not include access to MyProgrammingLab, which is available at [www.myprogramminglab.com](http://www.myprogramminglab.com) for purchase.

*This access code can only be used once. This subscription is valid for 12 months upon activation and is not transferrable. If the access code has already been revealed it may no longer be valid. If this is the case you can purchase a subscription on the login page for the Companion Website.*

For technical support go to <http://247pearsoned.custhelp.com>

This page intentionally left blank



## LOCATION OF VIDEONOTES IN THE TEXT



<b>Chapter 1</b>	Designing a Program with Pseudocode, p. 20 Designing the Account Balance Program, p. 25 Predicting the Output of Problem 30, p. 25 Solving the Candy Bar Sales Problem, p. 26
<b>Chapter 2</b>	Using <code>cout</code> to Display Output, p. 32 Assignment Statements, p. 59 Arithmetic Operators, p. 62 Solving the Restaurant Bill Problem, p. 74
<b>Chapter 3</b>	Using <code>cin</code> to Read Input, p. 77 Evaluating Mathematical Expressions, p. 84 Combined Assignment Operators, p. 102 Solving the Stadium Seating Problem, p. 148
<b>Chapter 4</b>	Using an <code>if</code> Statement, p. 161 Using an <code>if/else</code> Statement, p. 170 Using an <code>if/else if</code> Statement, p. 176 Using Logical Operators, p. 189 Solving the Time Calculator Problem, p. 238
<b>Chapter 5</b>	The <code>while</code> Loop, p. 244 The <code>for</code> Loop, p. 271 Nested Loops, p. 279 Solving the Ocean Levels Problem, p. 317
<b>Chapter 6</b>	Defining and Calling Functions, p. 324 Using Function Arguments, p. 333 Value-Returning Functions, p. 343 Solving the Markup Problem, p. 399
<b>Chapter 7</b>	Creating a Class, p. 412 Creating and Using Class Objects, p. 414 Creating and Using Structures, p. 454 Solving the Car Class Problem, p. 500
<b>Chapter 8</b>	Accessing Array Elements, p. 509 Passing an Array to a Function, p. 543 Two-Dimensional Arrays, p. 553 Solving the Chips and Salsa Problem, p. 593
<b>Chapter 9</b>	Performing a Binary Search, p. 606 Sorting a Set of Data, p. 613 Solving the Lottery Winners Problem, p. 641

(continued on next page)

## LOCATION OF VIDEONOTES IN THE TEXT *(continued)*



<b>Chapter 10</b>	Pointer Variables, p. 647 Dynamically Allocating an Array, p. 671 Solving the Days in Current Month Problem, p. 702
<b>Chapter 11</b>	Operator Overloading, p. 730 Aggregation and Composition, p. 776 Overriding Base Class Functions, p. 797 Solving the Number of Days Worked Problem, p. 811
<b>Chapter 12</b>	Converting Strings to Numbers, p. 829 Writing a C-String Handling Function, p. 833 Solving the Case Manipulator Problem, p. 850
<b>Chapter 13</b>	The get Family of Member Functions, p. 869 Rewinding a File, p. 873 Solving the File Encryption Filter Problem, p. 912
<b>Chapter 14</b>	Recursive Binary Search, p. 927 QuickSort, p. 929 Solving the Recursive Multiplication Problem, p. 947
<b>Chapter 15</b>	Polymorphism, p. 955 Composition versus Inheritance, p. 969 Solving the Sequence Sum Problem, p. 985
<b>Chapter 16</b>	Throwing and Handling Exceptions, p. 988 Writing a Function Template, p. 1000 Iterators, p. 1017 Solving the Arithmetic Exceptions Problem, p. 1034
<b>Chapter 17</b>	Adding an Element to a Linked List, p. 1045 Removing an Element from a Linked List, p. 1052 Solving the Member Insertion by Position Problem, p. 1083
<b>Chapter 18</b>	Storing Objects in an STL Stack, p. 1097 Storing Objects in an STL Queue, p. 1111 Solving the File Reverser Problem, p. 1123
<b>Chapter 19</b>	Inserting an Element into a Binary Tree, p. 1132 Removing an Element from a Binary Tree, p. 1136 Solving the Tree Size Problem, p. 1152



Ninth  
Edition

Starting Out with

# C++ Early Objects

**Tony Gaddis**  
**Judy Walters**  
**Godfrey Muganda**

**PEARSON**

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

Vice President, Editorial Director, ECS: Marcia Horton  
Acquisitions Editor: Matt Goldstein  
Editorial Assistant: Kristy Alaura  
Vice President of Marketing: Christy Lesko  
Director of Field Marketing: Tim Galligan  
Product Marketing Manager: Bram Van Kempen  
Field Marketing Manager: Demetrius Hall  
Marketing Assistant: Jon Bryant  
Director of Product Management: Erin Gregg  
Team Lead, Program and Project Management:  
Scott Disanno  
Program Manager: Carole Snyder  
Project Manager: RPK Editorial Services, Inc.

Senior Specialist, Program Planning and Support: Maura Zaldivar-Garcia  
Cover Designer: Joyce Wells  
Cover: Sabyna75/Shutterstock  
Manager, Rights and Permissions: Rachel Youdelman  
Project Manager, Rights and Permissions: William Opaluch  
Inventory Manager: Meredith Maresca  
Media Project Manager: Renata Butera  
Full-Service Project Management: Deepthi Mohan, Aptara® Corporation  
Composition: Aptara® Corporation  
Printer/Binder: Edwards Brothers Malloy, Inc.  
Cover and Insert Printer: Phoenix Color

Microsoft and/or its respective suppliers make no representations about the suitability of the information contained in the documents and related graphics published as part of the services for any purpose. All such documents and related graphics are provided “as is” without warranty of any kind. Microsoft and/or its respective suppliers hereby disclaim all warranties and conditions with regard to this information, including all warranties and conditions of merchantability, whether express, implied or statutory, fitness for a particular purpose, title and non-infringement. In no event shall Microsoft and/or its respective suppliers be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of information available from the services.

The documents and related graphics contained herein could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Microsoft and/or its respective suppliers may make improvements and/or changes in the product(s) and/or the program(s) described herein at any time. Partial screen shots may be viewed in full within the software version specified.

Microsoft® Windows®, and Microsoft Office® are registered trademarks of the Microsoft corporation in the U.S.A. and other countries. This book is not sponsored or endorsed by or affiliated with the Microsoft corporation.

The programs and applications presented in this book have been included for their instructional value. They have been tested with care, but are not guaranteed for any particular purpose. The publisher does not offer any warranties or representations, nor does it accept any liabilities with respect to the programs or applications.

---

Copyright © 2017, 2014 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. For information regarding permissions, request forms and the appropriate contacts within the Pearson Education Global Rights & Permissions department, please visit [www.pearsonhighed.com/permissions/](http://www.pearsonhighed.com/permissions/).

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

Names: Gaddis, Tony. | Walters, Judy. | Muganda, Godfrey.  
Title: Starting out with C++. Early objects / Tony Gaddis, Judy Walters, Godfrey Muganda.  
Description: Ninth edition. | Boston : Pearson, 2017. | Includes index.  
Identifiers: LCCN 2015048847 | ISBN 0134400240  
Subjects: LCSH: C++ (Computer program language)  
Classification: LCC QA76.73.C153 G333 2017 | DDC 005.13/3—dc23 LC record available at <http://lcn.loc.gov/2015048847>

10 9 8 7 6 5 4 3 2 1

**PEARSON**

ISBN 10: 0-13-440024-0  
ISBN 13: 978-0-13-440024-2

# Contents at a Glance

<b>Preface</b>	<b>xv</b>
<b>Chapter 1</b>	<b>Introduction to Computers and Programming 1</b>
<b>Chapter 2</b>	<b>Introduction to C++ 27</b>
<b>Chapter 3</b>	<b>Expressions and Interactivity 77</b>
<b>Chapter 4</b>	<b>Making Decisions 155</b>
<b>Chapter 5</b>	<b>Looping 243</b>
<b>Chapter 6</b>	<b>Functions 323</b>
<b>Chapter 7</b>	<b>Introduction to Classes and Objects 407</b>
<b>Chapter 8</b>	<b>Arrays 507</b>
<b>Chapter 9</b>	<b>Searching, Sorting, and Algorithm Analysis 603</b>
<b>Chapter 10</b>	<b>Pointers 645</b>
<b>Chapter 11</b>	<b>More about Classes and Object-Oriented Programming 703</b>
<b>Chapter 12</b>	<b>More on C-Strings and the <code>string</code> Class 815</b>
<b>Chapter 13</b>	<b>Advanced File and I/O Operations 853</b>
<b>Chapter 14</b>	<b>Recursion 915</b>
<b>Chapter 15</b>	<b>Polymorphism and Virtual Functions 949</b>
<b>Chapter 16</b>	<b>Exceptions, Templates, and the Standard Template Library (STL) 987</b>
<b>Chapter 17</b>	<b>Linked Lists 1037</b>
<b>Chapter 18</b>	<b>Stacks and Queues 1085</b>
<b>Chapter 19</b>	<b>Binary Trees 1125</b>
<b>Appendix A:</b>	<b>The ASCII Character Set 1155</b>
<b>Appendix B:</b>	<b>Operator Precedence and Associativity 1159</b>
<b>Appendix C:</b>	<b>Answers to Checkpoints 1161</b>
<b>Appendix D:</b>	<b>Answers to Odd-Numbered Review Questions 1201</b>
<b>Index</b>	<b>1221</b>



This page intentionally left blank

# Contents

## **Preface xv**

## **CHAPTER 1 Introduction to Computers and Programming 1**

- 1.1 Why Program? 1
- 1.2 Computer Systems: Hardware and Software 3
- 1.3 Programs and Programming Languages 8
- 1.4 What Is a Program Made of? 13
- 1.5 Input, Processing, and Output 17
- 1.6 The Programming Process 18
- 1.7 Tying It All Together: *Hi! It's Me* 23

## **CHAPTER 2 Introduction to C++ 27**

- 2.1 The Parts of a C++ Program 27
- 2.2 The `cout` Object 31
- 2.3 The `#include` Directive 36
- 2.4 Variables and the Assignment Statement 37
- 2.5 Literals 39
- 2.6 Identifiers 41
- 2.7 Integer Data Types 43
- 2.8 Floating-Point Data Types 48
- 2.9 The `char` Data Type 52
- 2.10 The C++ `string` Class 56
- 2.11 The `bool` Data Type 57
- 2.12 Determining the Size of a Data Type 58
- 2.13 More on Variable Assignments and Initialization 59
- 2.14 Scope 61
- 2.15 Arithmetic Operators 62
- 2.16 Comments 66
- 2.17 Programming Style 67
- 2.18 Tying It All Together: *Smile!* 69

**CHAPTER 3 Expressions and Interactivity 77**

- 3.1 The `cin` Object 77
- 3.2 Mathematical Expressions 84
- 3.3 Data Type Conversion and Type Casting 92
- 3.4 Overflow and Underflow 98
- 3.5 Named Constants 99
- 3.6 Multiple and Combined Assignment 102
- 3.7 Formatting Output 106
- 3.8 Working with Characters and Strings 116
- 3.9 More Mathematical Library Functions 130
- 3.10 Random Numbers 132
- 3.11 Focus on Debugging: *Hand Tracing a Program* 136
- 3.12 Green Fields Landscaping Case Study—Part 1 138
- 3.13 Tying It All Together: *Word Game* 141

**CHAPTER 4 Making Decisions 155**

- 4.1 Relational Operators 155
- 4.2 The `if` Statement 161
- 4.3 The `if/else` Statement 170
- 4.4 The `if/else if` Statement 175
- 4.5 Menu-Driven Programs 183
- 4.6 Nested `if` Statements 185
- 4.7 Logical Operators 189
- 4.8 Validating User Input 198
- 4.9 More about Blocks and Scope 200
- 4.10 More about Characters and Strings 203
- 4.11 The Conditional Operator 209
- 4.12 The `switch` Statement 213
- 4.13 Enumerated Data Types 222
- 4.14 Focus on Testing and Debugging: *Validating Output Results* 225
- 4.15 Green Fields Landscaping Case Study—Part 2 227
- 4.16 Tying It All Together: *Fortune Teller* 232

**CHAPTER 5 Looping 243**

- 5.1 Introduction to Loops: The `while` Loop 243
- 5.2 Using the `while` Loop for Input Validation 250
- 5.3 The Increment and Decrement Operators 253
- 5.4 Counters 258
- 5.5 Keeping a Running Total 260
- 5.6 Sentinels 263
- 5.7 The `do-while` Loop 265



5.8	The for Loop	271
5.9	Focus on Software Engineering: <i>Deciding Which Loop to Use</i>	277
5.10	Nested Loops	279
5.11	Breaking Out of a Loop	281
5.12	Using Files for Data Storage	285
5.13	Focus on Testing and Debugging: <i>Creating Good Test Data</i>	301
5.14	Central Mountain Credit Union Case Study	304
5.15	Tying It All Together: <i>What a Colorful World</i>	308

## **CHAPTER 6 Functions 323**

6.1	Modular Programming	323
6.2	Defining and Calling Functions	324
6.3	Function Prototypes	332
6.4	Sending Data into a Function	333
6.5	Passing Data by Value	338
6.6	The return Statement	342
6.7	Returning a Value from a Function	343
6.8	Returning a Boolean Value	349
6.9	Using Functions in a Menu-Driven Program	351
6.10	Local and Global Variables	355
6.11	Static Local Variables	362
6.12	Default Arguments	364
6.13	Using Reference Variables as Parameters	368
6.14	Overloading Functions	378
6.15	The <code>exit()</code> Function	382
6.16	Stubs and Drivers	385
6.17	Little Lotto Case Study	387
6.18	Tying It All Together: <i>Glowing Jack-o-lantern</i>	392

## **CHAPTER 7 Introduction to Classes and Objects 407**

7.1	Abstract Data Types	407
7.2	Object-Oriented Programming	409
7.3	Introduction to Classes	411
7.4	Creating and Using Objects	414
7.5	Defining Member Functions	416
7.6	Constructors	423
7.7	Destructors	429
7.8	Private Member Functions	432
7.9	Passing Objects to Functions	435
7.10	Object Composition	442
7.11	Focus on Software Engineering: <i>Separating Class Specification, Implementation, and Client Code</i>	446

- 7.12 Structures 453
- 7.13 More about Enumerated Data Types 465
- 7.14 Home Software Company OOP Case Study 469
- 7.15 Introduction to Object-Oriented Analysis and Design 476
- 7.16 Screen Control 486
- 7.17 Tying It All Together: *Yoyo Animation* 491

## **CHAPTER 8 Arrays 507**

- 8.1 Arrays Hold Multiple Values 507
- 8.2 Accessing Array Elements 509
- 8.3 Inputting and Displaying Array Data 511
- 8.4 Array Initialization 518
- 8.5 The Range-Based for loop 525
- 8.6 Processing Array Contents 528
- 8.7 Using Parallel Arrays 539
- 8.8 The typedef Statement 543
- 8.9 Arrays as Function Arguments 543
- 8.10 Two-Dimensional Arrays 553
- 8.11 Arrays with Three or More Dimensions 560
- 8.12 Vectors 563
- 8.13 Arrays of Objects 575
- 8.14 National Commerce Bank Case Study 585
- 8.15 Tying It All Together: *Rock, Paper, Scissors* 587

## **CHAPTER 9 Searching, Sorting, and Algorithm Analysis 603**

- 9.1 Introduction to Search Algorithms 603
- 9.2 Searching an Array of Objects 610
- 9.3 Introduction to Sorting Algorithms 613
- 9.4 Sorting an Array of Objects 621
- 9.5 Sorting and Searching Vectors 624
- 9.6 Introduction to Analysis of Algorithms 627
- 9.7 Case Studies 635
- 9.8 Tying It All Together: *Secret Messages* 635

## **CHAPTER 10 Pointers 645**

- 10.1 Pointers and the Address Operator 645
- 10.2 Pointer Variables 647
- 10.3 The Relationship Between Arrays and Pointers 651

- 10.4 Pointer Arithmetic 655
- 10.5 Initializing Pointers 656
- 10.6 Comparing Pointers 659
- 10.7 Pointers as Function Parameters 661
- 10.8 Pointers to Constants and Constant Pointers 665
- 10.9 Focus on Software Engineering: *Dynamic Memory Allocation* 670
- 10.10 Focus on Software Engineering: *Returning Pointers from Functions* 674
- 10.11 Pointers to Class Objects and Structures 680
- 10.12 Focus on Software Engineering: *Selecting Members of Objects* 684
- 10.13 Smart Pointers 686
- 10.14 Tying It All Together: *Pardon Me, Do You Have the Time?* 694

## **CHAPTER 11 More about Classes and Object-Oriented Programming 703**

- 11.1 The `this` Pointer and Constant Member Functions 703
- 11.2 Static Members 708
- 11.3 Friends of Classes 715
- 11.4 Memberwise Assignment 720
- 11.5 Copy Constructors 721
- 11.6 Operator Overloading 730
- 11.7 Rvalue References and Move Operations 751
- 11.8 Function Objects and Lambda Expressions 761
- 11.9 Type Conversion Operators 770
- 11.10 Convert Constructors 773
- 11.11 Aggregation and Composition 776
- 11.12 Inheritance 782
- 11.13 Protected Members and Class Access 787
- 11.14 Constructors, Destructors, and Inheritance 792
- 11.15 Overriding Base Class Functions 797
- 11.16 Tying It All Together: *Putting Data on the World Wide Web* 800

## **CHAPTER 12 More on C-Strings and the `string` Class 815**

- 12.1 C-Strings 815
- 12.2 Library Functions for Working with C-Strings 820
- 12.3 Conversions Between Numbers and Strings 829
- 12.4 Writing Your Own C-String Handling Functions 833
- 12.5 More about the C++ `string` Class 839
- 12.6 Advanced Software Enterprises Case Study 842
- 12.7 Tying It All Together: *Program Execution Environments* 844



**CHAPTER 13 Advanced File and I/O Operations 853**

- 13.1 Input and Output Streams 853
- 13.2 More Detailed Error Testing 861
- 13.3 Member Functions for Reading and Writing Files 865
- 13.4 Binary Files 877
- 13.5 Creating Records with Structures 881
- 13.6 Random-Access Files 886
- 13.7 Opening a File for Both Input and Output 893
- 13.8 Online Friendship Connections Case Study: *Object Serialization* 898
- 13.9 Tying It All Together: *File Merging and Color-Coded HTML* 903

**CHAPTER 14 Recursion 915**

- 14.1 Introduction to Recursion 915
- 14.2 The Recursive Factorial Function 922
- 14.3 The Recursive gcd Function 924
- 14.4 Solving Recursively Defined Problems 925
- 14.5 A Recursive Binary Search Function 927
- 14.6 Focus on Problem Solving and Program Design: *The QuickSort Algorithm* 929
- 14.7 The Towers of Hanoi 933
- 14.8 Focus on Problem Solving: *Exhaustive and Enumeration Algorithms* 936
- 14.9 Focus on Software Engineering: *Recursion versus Iteration* 940
- 14.10 Tying It All Together: *Infix and Prefix Expressions* 941

**CHAPTER 15 Polymorphism and Virtual Functions 949**

- 15.1 Type Compatibility in Inheritance Hierarchies 949
- 15.2 Polymorphism and Virtual Member Functions 955
- 15.3 Abstract Base Classes and Pure Virtual Functions 963
- 15.4 Focus on Object-Oriented Programming: *Composition versus Inheritance* 969
- 15.5 Secure Encryption Systems, Inc., Case Study 973
- 15.6 Tying It All Together: *Let's Move It* 976

**CHAPTER 16 Exceptions, Templates, and the Standard Template Library (STL) 987**

- 16.1 Exceptions 987
- 16.2 Function Templates 999
- 16.3 Class Templates 1007
- 16.4 Class Templates and Inheritance 1012
- 16.5 Introduction to the Standard Template Library 1016
- 16.6 Tying It All Together: *Word Transformers Game* 1029

**CHAPTER 17 Linked Lists 1037**

- 17.1 Introduction to the Linked List ADT 1037
- 17.2 Linked List Operations 1043
- 17.3 A Linked List Template 1055
- 17.4 Recursive Linked List Operations 1059
- 17.5 Variations of the Linked List 1067
- 17.6 The STL list Container 1068
- 17.7 Reliable Software Systems, Inc., Case Study 1071
- 17.8 Tying It All Together: *More on Graphics and Animation* 1074

**CHAPTER 18 Stacks and Queues 1085**

- 18.1 Introduction to the Stack ADT 1085
- 18.2 Dynamic Stacks 1093
- 18.3 The STL stack Container 1097
- 18.4 Introduction to the Queue ADT 1099
- 18.5 Dynamic Queues 1106
- 18.6 The STL deque and queue Containers 1109
- 18.7 Focus on Problem Solving and Program Design: *Eliminating Recursion* 1112
- 18.8 Tying It All Together: *Converting Postfix Expressions to Infix* 1117

**CHAPTER 19 Binary Trees 1125**

- 19.1 Definition and Applications of Binary Trees 1125
- 19.2 Binary Search Tree Operations 1129
- 19.3 Template Considerations for Binary Search Trees 1145
- 19.4 Tying It All Together: *Genealogy Trees* 1145

**Appendix A: The ASCII Character Set 1155****Appendix B: Operator Precedence and Associativity 1159****Appendix C: Answers to Checkpoints 1161****Appendix D: Answers to Odd-Numbered Review Questions 1201****Index 1221**

### **Additional Appendices**

The following appendices are located on the book's companion web site.

**Appendix E: A Brief Introduction to Object-Oriented Programming**

**Appendix F: Using UML in Class Design**

**Appendix G: Multi-Source File Programs**

**Appendix H: Multiple and Virtual Inheritance**

**Appendix I: Header File and Library Function Reference**

**Appendix J: Namespaces**

**Appendix K: C++ Casts and Run-Time Type Identification**

**Appendix L: Passing Command Line Arguments**

**Appendix M: Binary Numbers and Bitwise Operations**

**Appendix N: Introduction to Flowcharting**



# Preface

Welcome to *Starting Out with C++: Early Objects*, 9th Edition. This book is intended for use in a two-term or three-term C++ programming sequence, or an accelerated one-term course. Students new to programming, as well as those with prior course work in other languages, will find this text beneficial. The fundamentals of programming are covered for the novice, while the details, pitfalls, and nuances of the C++ language are explored in-depth for both the beginner and more experienced student. The book is written with clear, easy-to-understand language and it covers all the necessary topics for an introductory programming course. This text is rich in example programs that are concise, practical, and real world oriented, ensuring that the student not only learns how to implement the features and constructs of C++, but why and when to use them.

## What's New in the Ninth Edition

### The New C++11 Standard

C++11 is the latest standard version of the C++ language. In previous years, while the standard was being developed, it was known as C++0x. In August 2011, it was approved by the International Standards Organization (ISO), and the name of the standard was officially changed to C++11. Most of the popular C++ compilers now support this standard.

The new C++11 standard was the primary motivation behind this edition, which introduces many of the new language features. However, a C++11 compiler is not strictly required to use the book. As you progress through the book, you will see C++11 icons in the margins, next to the new features that are introduced. Programs appearing in sections that are not marked with this icon will still compile using an older compiler.

### The C++11 Topics Introduced in This Edition

- The `auto` key word is introduced in Chapter 2 as a way to simplify complex variable definitions. This key word causes the compiler to infer a variable's data type from its initialization value.

- Chapter 2 also introduces the new `long long int` and `unsigned long long int` data types and the `LL` literal suffix.
- Chapter 5 shows how to pass a `string` object directly to a file stream object's `open` member function, without the need to call the `c_str()` member function. A discussion of the `c_str()` function still exists for anyone using a legacy compiler.
- The range-based `for` loop is introduced in Chapter 7. This new looping mechanism automatically iterates over each element of an array, `vector`, or other collection, without the need for a counter variable or a subscript.
- Chapter 7 also introduces strongly typed `enums`.
- Chapter 8 introduces new ways to initialize variables and shows how a `vector` can now be initialized with an initialization list.
- Chapter 10 introduces smart pointers and provides examples of how and why to use the new `unique_ptr` and `shared_ptr` pointers for safely allocating and working with dynamic memory.
- Chapter 10 also introduces the move assignment operator, and the `nullptr` key word, which is now the standard way of representing a null pointer.
- Chapter 11 discusses move constructors, provides more in depth coverage of move assignment operators, and introduces lambda expressions.
- Chapter 12 introduces new functions in the C++11 string library and discusses the new overloaded `to_string` functions for converting numeric values to `string` objects.
- Chapter 15 introduces and demonstrates the use of the new `override` key word that helps prevent subtle overriding errors and the new `final` key word that prevents a virtual member function from being overridden.
- Chapter 16 introduces the new C++11 functions `begin(c)` and `end(c)` to specify positions within a collection `c` where an operation should begin and end.

### What Else is New

This book's pedagogy and clear writing style remain the same as in the previous edition. However, in addition to updating the book to introduce the new C++11 standard, many improvements have been made to make it even more student-friendly.

- **Updated Material**  
Material has been updated throughout the book to reflect changes in technology and in software development environments, as well as to improve clarity and incorporate best practices in teaching introductory programming. As a result, new graphics and new or redesigned figures have been added throughout the book where appropriate and new or improved sample programs have been included in a number of chapters.
- **New Material**  
New material has been added on a number of topics. In addition to introducing and using new C++11 features, this new edition includes new sections on literals, random numbers, and enumerated data types, as well as improved material on designing classes.



- **New Programming Challenges**  
New Programming Challenge problems have been added to every chapter.
- **Reorganized Chapters**  
Several chapters have been redesigned to improve student learning. Chapter 5 (Looping) has been reorganized to give students more practice using the `while` loop before introducing `do-while` and `for` loops. The Chapter 6 (Functions) material on defining and calling functions has been reorganized to introduce function prototypes earlier and allow `main` to always be the first function in a client program.

## Organization of the Text

This text teaches C++ in a step-by-step fashion. Each chapter covers a major set of topics and builds knowledge as the student progresses through the book. Although the chapters can be easily taught in their existing sequence, flexibility is provided. The following dependency diagram (Figure P-1) suggests possible sequences of instruction.

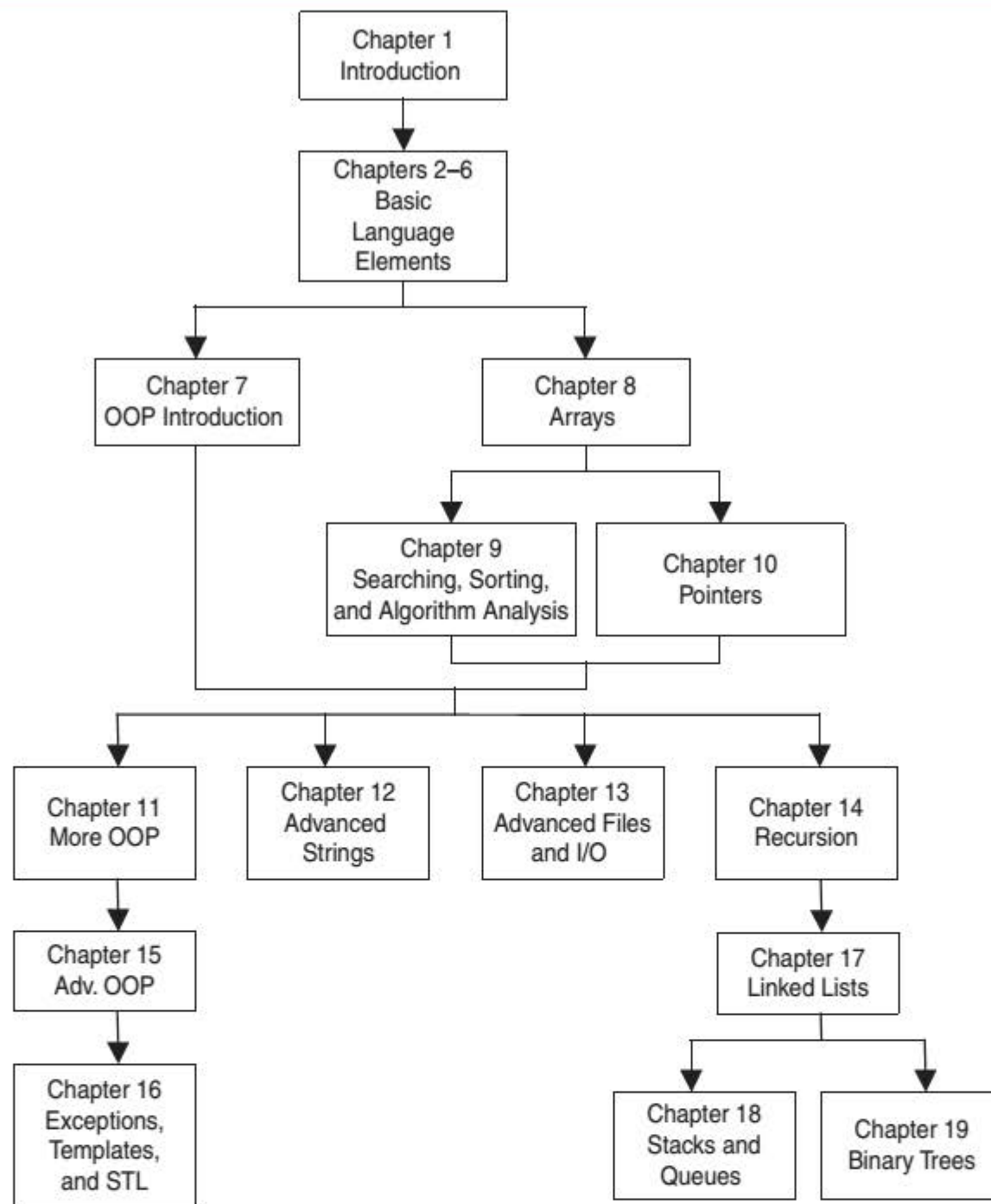
Chapter 1 covers fundamental hardware, software, and programming concepts. The instructor may choose to skip this chapter if the class has already mastered those topics. Chapters 2 through 6 cover basic C++ syntax, data types, expressions, selection structures, repetition structures, and functions. Each of these chapters builds on the previous chapter and should be covered in the order presented.

Chapter 7 introduces object-oriented programming. It can be covered any time after Chapter 6, but before Chapter 11. Instructors who prefer to introduce arrays before classes can cover Chapter 8 before Chapter 7. In this case it is only necessary to postpone Section 8.13 (Arrays of Objects) until Chapter 7 has been covered.

As Figure P-1 illustrates, in the second half of the book Chapters 11, 12, 13, and 14 can be covered in any order. Chapters 11, 15, and 16, however, should be done in sequence. Instructors who wish to introduce data structures at an earlier point in the course, without having first covered advanced C++ and OOP features, can cover Chapter 17 (Linked Lists), followed by Chapters 18 and 19 (Stacks & Queues and Binary Trees), any time after Chapter 14 (Recursion). In this case it is necessary to simply omit the sections in Chapters 17–19 that deal with templates and the Standard Template Library.



Figure P-1



## Brief Overview of Each Chapter

### **Chapter 1: Introduction to Computers and Programming**

This chapter provides an introduction to the field of computer science and covers the fundamentals of hardware, software, operating systems, programming, problem solving, and software engineering. The components of programs, such as key words, variables, operators, and punctuation are covered. The tools of the trade, such as hierarchy charts and pseudocode, are also presented. The *Tying It All Together* section shows students how to use the `cout` statement to create a personalized output message. Programming Challenges at the end of the chapter help students see how the same basic input, processing, and output structure can be used to create multiple programs.

### **Chapter 2: Introduction to C++**

This chapter gets the student started in C++ by introducing the basic parts of a C++ program, data types, the use of variables and literals, assignment statements, simple arithmetic operations, program output, and comments. The C++ `string` class is presented and `string` objects are used from this point on in the book as the primary method of handling strings. Programming style conventions are introduced, and good programming style is modeled here, as it is throughout the text. The *Tying It All Together* section lets the student play with simple text-based graphics.

### **Chapter 3: Expressions and Interactivity**

In this chapter the student learns to write programs that input and handle numeric, character, and string data. The use of arithmetic operators and the creation of mathematical expressions are covered, with emphasis on operator precedence. Debugging is introduced, with a section on hand tracing a program. Sections are also included on using random numbers, on simple output formatting, on data type conversion and type casting, and on using library functions that work with numbers. The *Tying It All Together* section shows students how to create a simple interactive word game.

### **Chapter 4: Making Decisions**

Here the student learns about relational expressions and how to control the flow of a program with `if`, `if/else`, and `if/else if` statements. Logical operators, the conditional operator, and the `switch` statement are also covered. Applications of these constructs, such as menu-driven programs, are illustrated. This chapter also introduces the concepts of blocks and scope and continues the theme of debugging with a section on validating output results. The *Tying It All Together* section uses random numbers and branching statements to create a fortune telling game.

### **Chapter 5: Looping**

This chapter introduces, C++'s repetitive control mechanisms. The `while` loop, `do-while` loop, and `for` loop are presented, along with a variety of methods to control them. These include using counters, user input, end sentinels, and end-of-file testing. Applications utilizing loops, such as keeping a running total and performing data validation, are also covered. The chapter includes an extensive section on working with files and a section on creating good test data, continuing the book's emphasis on testing and debugging. The *Tying It All Together* section introduces students to Windows commands to create colorful output and uses a loop to create a multi-colored display.

### **Chapter 6: Functions**

In this chapter the student learns how and why to modularize programs, using both `void` and value-returning functions. Parameter passing is covered, with emphasis on when arguments should be passed by value versus when they need to be passed by reference. Scope of variables is covered and sections are provided on local versus global variables and on static local variables. Overloaded functions are also introduced and demonstrated. The *Tying It All Together* section includes a modular, menu-driven program that emphasizes the versatility of functions, illustrating how their behavior can be controlled by the arguments sent to them.

### **Chapter 7: Introduction to Classes and Objects**

In this chapter the text begins to focus on the object-oriented paradigm. Students have used provided C++ classes since the beginning of the text, but now they learn how to define their own classes and to create and use objects of these classes. Careful attention is paid to illustrating which functions belong in a class versus which functions belong in a client program that uses the class. Good object-oriented practices are discussed and modeled, such as protecting member data through carefully constructed accessor and mutator functions and hiding class implementation details from client programs. Once students are comfortable working with classes and objects, the chapter provides a brief introduction to the topic of object-oriented analysis and design. The chapter also includes a section on enumerated data types and a section on structures, which are used in the *Tying It All Together* section, where students learn to use screen control techniques to create a yoyo animation.

### **Chapter 8: Arrays**

In this chapter the student learns to create and work with single and multidimensional arrays. Many examples of array processing are provided, including functions to compute the sum, average, highest and lowest values in an array. Students also learn to create tables using two-dimensional arrays, and to analyze array data by row or by column. Programming techniques using parallel arrays are also demonstrated, and the student is shown how to use a data file as an input source to populate an array. The range-based `for` loop is introduced as an easy way to iterate through all the elements of an array, and STL vectors are introduced and compared to arrays. A section on arrays of objects and structures is located at the end of the chapter, so it can be covered now or saved for later if the instructor wishes to cover this chapter before Chapter 7. The *Tying It All Together* section uses arrays to create a game of *Rock, Paper, Scissors* between a human player and the computer.



**Chapter 9: Searching, Sorting, and Algorithm Analysis**

Here the student learns the basics of searching for information stored in arrays and of sorting arrays, including arrays of objects. The chapter covers the Linear Search, Binary Search, Bubble Sort, and Selection Sort algorithms and has an optional section on sorting and searching STL vectors. A brief introduction to algorithm analysis is included, and students are shown how to determine which of two algorithms is more efficient. This chapter's *Tying It All Together* section uses both a table lookup and a searching algorithm to encode and decode secret messages.

**Chapter 10: Pointers**

This chapter explains how to use pointers. Topics include pointer arithmetic, initialization of pointers, comparison of pointers, pointers and arrays, pointers and functions, dynamic memory allocation, the new `nullptr` key word, and more. A new section introduces smart pointers and shows how they can be used to avoid memory leaks. The *Tying It All Together* section demonstrates the use of pointers to access library data structures and functions that return calendar and wall clock time.

**Chapter 11: More About Classes and Object-Oriented Programming**

This chapter continues the study of classes and object-oriented programming, covering more advanced topics such as inheritance and object aggregation and composition. Other topics include constant member functions, static members, friends, memberwise assignment, copy constructors, object type conversion operators, convert constructors, operator overloading, move constructors, and move assignment operators. A new section introduces function objects and the C++11 lambda expressions. The *Tying It All Together* section brings together the concepts of inheritance and convert constructors to build a program that formats the contents of an array to form an HTML table for display on a Web site.

**Chapter 12: More on C-Strings and the `string` Class**

This chapter covers standard library functions for working with characters and C-strings, as well as material on `string` class functions, functions in the new C++11 `string` library, and new overloaded `to_string` functions for converting numeric values to `string` objects. The *Tying It All Together* section shows students how to access string-based program environments to obtain information about the computer and the network on which the program is running.

**Chapter 13: Advanced File and I/O Operations**

This chapter introduces more advanced topics for working with sequential access text files and introduces random access and binary files. Various modes for opening files are discussed, as well as the many methods for reading and writing their contents. The *Tying It All Together* program applies many of the techniques covered in the chapter to merge two text files into an HTML document for display on the Web, with different colors used to illustrate which file each piece of data came from.



**Chapter 14: Recursion**

In this chapter recursion is defined and demonstrated. A visual trace of recursive calls is provided, and recursive applications are discussed. Many recursive algorithms are presented, including recursive functions for computing factorials, finding a greatest common denominator (GCD), performing a binary search, sorting using QuickSort, and solving the famous Towers of Hanoi problem. For students who need more challenge, there is a section on exhaustive and enumeration algorithms. The *Tying It All Together* section uses recursion to evaluate prefix expressions.

**Chapter 15: Polymorphism and Virtual Functions**

The study of classes and object-oriented programming continues in this chapter with the introduction of more advanced concepts such as polymorphism and virtual functions. Information is also presented on abstract base classes, pure virtual functions, type compatibility within an inheritance hierarchy, and virtual inheritance. The *Tying It All Together* section illustrates the use of inheritance and polymorphism to display and animate graphical images.

**Chapter 16: Exceptions, Templates, and the Standard Template Library (STL)**

Here the student learns to develop enhanced error trapping techniques using exceptions. Discussion then turns to using function and class templates to create generic code. Finally, the student is introduced to the containers, iterators, and algorithms offered by the Standard Template Library (STL). The *Tying It All Together* section uses various containers in the Standard Template Library to create an educational children's game.

**Chapter 17: Linked Lists**

This chapter introduces concepts and techniques needed to work with lists. A linked list ADT is developed, and the student learns how to create and destroy a list, as well as to write functions to insert, append, and delete nodes, to traverse the list, and to search for a specific node. A linked list class template is also demonstrated. The *Tying It All Together* section brings together many of the most important concepts of OOP by using objects, inheritance, and polymorphism in conjunction with the STL list class to animate a collection of images.

**Chapter 18: Stacks and Queues**

In this chapter the student learns to create and use static and dynamic stacks and queues. The operations of stacks and queues are defined, and templates for each ADT are demonstrated. The static array-based stack uses exception-handling to handle stack overflow and underflow, providing a realistic and natural example of defining, throwing, and catching exceptions. The *Tying It All Together* section discusses strategies for evaluating postfix expressions and uses a stack to convert a postfix expression to infix.

**Chapter 19: Binary Trees**

This chapter covers the binary tree ADT and demonstrates many binary tree operations. The student learns to traverse a tree, insert, delete, and replace elements, search for a particular element, and destroy a tree. The *Tying It All Together* section introduces a tree structure versatile enough to create genealogy trees.

## Appendices in the Book

**Appendix A: The ASCII Character Set** A list of the ASCII and extended ASCII characters and their codes.

**Appendix B: Operator Precedence and Associativity** A list of the C++ operators with their precedence and associativity.

**Appendix C: Answers to Checkpoints** A tool students can use to assess their understanding by comparing their answers to the Checkpoint exercises found throughout the book. The answers to all Checkpoint exercises are included.

**Appendix D: Answers to Odd-Numbered Review Questions** Another tool students can use to gauge their understanding and progress.

## Additional Appendices on the Book's Companion Website

**Appendix E: A Brief Introduction to Object-Oriented Programming** An introduction to the concepts and terminology of object-oriented programming.

**Appendix F: Using UML in Class Design** A brief introduction to the Unified Modeling Language (UML) class diagrams with examples of their use.

**Appendix G: Multi-Source File Programs** A tutorial on how to create, compile, and link programs with multiple source files. Includes the use of function header files, class specification files, and class implementation files.

**Appendix H: Multiple and Virtual Inheritance** A self-contained discussion of the C++ concepts of multiple and virtual inheritance for anyone already familiar with single inheritance.

**Appendix I: Header File and Library Function Reference** A reference for the C++ library functions and header files used in the book.

**Appendix J: Namespaces** An explanation of namespaces and their purpose, with examples provided on how to define a namespace and access its members.

**Appendix K: C++ Casts and Run-Time Type Identification** An introduction to different ways of doing type casting in C++ and to run-time type identification.





**Appendix L: Passing Command Line Arguments** An introduction to writing C++ programs that accept command-line arguments. This appendix will be useful to students working in a command-line environment, such as UNIX or Linux.

**Appendix M: Binary Numbers and Bitwise Operations** A guide to the binary number system and the C++ bitwise operators, as well as a tutorial on the internal storage of integers.

**Appendix N: Introduction to Flowcharting** A tutorial that introduces flowcharting and its symbols. It includes handling sequence, selection, case, repetition, and calls to other modules. Sample flowcharts for several of the book's example programs are presented.



## Features of the Text

- Concept Statements** Each major section of the text starts with a concept statement. This statement summarizes the key idea of the section.
- Example Programs** The text has over 350 complete example programs, each designed to highlight the topic currently being studied. In most cases, these are practical, real-world examples. Source code for these programs is provided so that students can run the programs themselves.
- Program Output** After each example program there is a sample of its screen output. This immediately shows the student how the program should function.
- Tying It All Together** This special section, found at the end of every chapter, shows the student how to do something clever and fun with the material covered in that chapter.
-  **VideoNotes** A series of online videos, developed specifically for this book, are available for viewing at <http://www.pearsonhighered.com/cs-resources/>. VideoNote icons appear throughout the text, alerting the student to videos about specific topics.
-  **Checkpoints** Checkpoints are questions placed throughout each chapter as a self-test study aid. Answers for all Checkpoint questions are provided in Appendix C at the back of the book so students can check how well they have learned a new topic.
-  **Notes** Notes appear at appropriate places throughout the text. They are short explanations of interesting or often misunderstood points relevant to the topic at hand.
-  **Warnings** Warnings caution the student about certain C++ features, programming techniques, or practices that can lead to malfunctioning programs or lost data.
- Case Studies** Case studies that simulate real-world applications appear in many chapters throughout the text, with complete code provided for each one. Additional case studies are provided on the book's companion website. These case studies are designed to highlight the major topics of the chapter in which they appear.
- Review Questions and Exercises** Each chapter presents a thorough and diverse set of review questions, such as fill-in-the-blank and short answer, that check the student's mastery of the basic material presented in the chapter. These are followed by exercises requiring problem solving and analysis, such as the *Algorithm Workbench*, *Predict the Output*, and *Find the Errors* sections. Each chapter ends with a *Soft Skills* exercise that focuses on communication and group process skills. Answers to the odd numbered review questions and review exercises are provided in Appendix D at the back of the book.

<i>Programming Challenges</i>	Each chapter offers a pool of programming exercises designed to solidify the student's knowledge of the topics currently being studied. In most cases the assignments present real-world problems to be solved.
<i>Group Projects</i>	There are several group programming projects throughout the text, intended to be constructed by a team of students. One student might build the program's user interface, while another student writes the mathematical code, and another designs and implements a class the program uses. This process is similar to the way many professional programs are written and encourages teamwork within the classroom.
<i>C++ Quick Reference Guide</i>	For easy access, a quick reference guide to the C++ language is printed on the inside back cover.

## Supplements

### Student Resources

The following items are available on the Gaddis Series resource page at [www.pearsonhighered.com/cs-resources](http://www.pearsonhighered.com/cs-resources):

- Complete source code for every program included in the book
- Additional case studies, complete with source code
- A full set of appendices (including several tutorials) that accompany the book
- Access to the book's companion VideoNotes
- Links to download numerous programming environments and IDEs, including Visual Studio Community Edition.

### Instructor Resources

The following supplements are available to qualified instructors only.

- Answers to all Review Questions in the text
- Solutions for all Programming Challenges in the text
- PowerPoint presentation slides for every chapter
- A computerized test bank
- A collection of lab materials
- Source code files

Visit the Pearson Education Instructor Resource Center (<http://www.pearsonhighered.com/irc>) for information on how to access them.

### 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, *MyProgrammingLab* consists of hundreds of small practice exercises 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 help them figure out what went wrong. For instructors, a comprehensive gradebook tracks correct and incorrect answers and stores the code input 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](http://www.myprogramminglab.com).

### **Which Gaddis C++ book is right for you?**

The *Starting Out with C++* Series includes three books, one of which is sure to fit your course:

- *Starting Out with C++: Early Objects*
- *Starting Out with C++: From Control Structures through Objects*
- *Starting Out with C++: Brief Version*

## Acknowledgments

There have been many helping hands in the development and publication of this text. We would like to thank the following faculty reviewers for their helpful suggestions and expertise.

### Reviewers of the Ninth Edition or Its Previous Versions

Ahmad Abuhejleh <i>University of Wisconsin, River Falls</i>	Fred M. D'Angelo <i>Pima Community College</i>
David Akins <i>El Camino College</i>	Joseph DeLibero <i>Arizona State University</i>
Steve Allan <i>Utah State University</i>	Dennis Fairclough <i>Utah Valley State College</i>
Ijaz A. Awan <i>Savannah State University</i>	Larry Farrer <i>Guilford Technical Community College</i>
John Bierbauer <i>North Central College</i>	James D. Fitzgerald <i>Golden West College</i>
Don Biggerstaff <i>Fayetteville Technical Community College</i>	Richard Flint <i>North Central College</i>
Paul Bladdek <i>Spokane Falls Community College</i>	Sheila Foster <i>California State University Long Beach</i>
Chuck Boehm <i>Dean Foods, Inc.</i>	David E. Fox <i>American River College</i>
Bill Brown <i>Pikes Peak Community College</i>	Cindy Fry <i>Baylor University</i>
Richard Cacace <i>Pensacola Junior College</i>	Peter Gacs <i>Boston University</i>
Randy Campbell <i>Morningside College</i>	Cristi Gale <i>Sterling College</i>
Stephen P. Carl <i>Wright State University</i>	James Gifford <i>University of Wisconsin, Stevens Point</i>
Wayne Caruolo <i>Red Rocks Community College</i>	Leon Gleiberman <i>Touro College</i>
Thomas Cheatham <i>Middle Tennessee State University</i>	Simon Gray <i>Ashland University—Ohio</i>
James Chegwiddden <i>Tarrant County College</i>	Margaret E. Guertin <i>Tufts University</i>
John Cigas <i>Rockhurst University</i>	Jamshid Haghghi <i>Guilford Technical Community College</i>
John Cross <i>Indiana University of Pennsylvania</i>	Ranette H. Halverson <i>Midwestern State University, Wichita Falls, TX</i>