

## LOCATION OF VIDEONOTES IN THE TEXT



**Chapter 1** Designing a Program with Pseudocode, p. 19  
Designing the Account Balance Program, p. 24  
Predicting the Output of Problem 30, p. 24  
Solving the Candy Bar Sales Problem, p. 25

**Chapter 2** Using `cout` to Display Output, p. 32  
Assignment Statements, p. 59  
Arithmetic Operators, p. 61  
Solving the Restaurant Bill Problem, p. 72

**Chapter 3** Using `cin` to Read Input, p. 75  
Evaluating Mathematical Expressions, p. 81  
Combined Assignment Operators, p. 102  
Solving the Stadium Seating Problem, p. 151

**Chapter 4** Using an `if` Statement, p. 162  
Using an `if/else` Statement, p. 172  
Using an `if/else if` Statement, p. 175  
Solving the Time Calculator Problem, p. 236

**Chapter 5** The `while` Loop, p. 249  
The `for` Loop, p. 263  
Nested Loops, p. 277  
Solving the Ocean Levels Problem, p. 299

**Chapter 6** Defining and Calling Functions, p. 306  
Using Function Arguments, p. 316  
Value-Returning Functions, p. 326  
Solving the Markup Problem, p. 380

**Chapter 7** Creating a Class, p. 391  
Creating and Using Class Objects, p. 393  
Creating and Using Structures, p. 436  
Solving the `car` Class Problem, p. 480

**Chapter 8** Accessing Array Elements, p. 487  
Passing an Array to a Function, p. 517  
Two-Dimensional Arrays, p. 526  
Solving the Chips and Salsa Problem, p. 567

**Chapter 9** Performing a Binary Search, p. 580  
Sorting a Set of Data, p. 587  
Solving the Lottery Winners Problem, p. 616

*(continued on next page)*

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



<b>Chapter 10</b>	Pointer Variables, p. 621 Dynamically Allocating an Array, p. 645 Solving the Days in Current Month Problem, p. 676
<b>Chapter 11</b>	Operator Overloading, p. 704 Aggregation and Composition, p. 734 Overriding Base Class Functions, p. 755 Solving the Number of Days Worked Problem, p. 768
<b>Chapter 12</b>	Converting Strings to Numbers, p. 788 Writing a C-String Handling Function, p. 800 Solving the Backward String Problem, p. 823
<b>Chapter 13</b>	Passing File Stream Objects to Functions, p. 845 Rewinding a File, p. 858 Solving the File Encryption Filter Problem, p. 897
<b>Chapter 14</b>	Recursive Binary Search, p. 913 QuickSort, p. 915 Solving the Recursive Multiplication Problem, p. 933
<b>Chapter 15</b>	Polymorphism, p. 941 Composition Versus Inheritance, p. 952 Solving the Sequence Sum Problem, p. 970
<b>Chapter 16</b>	Throwing and Handling Exceptions, p. 975 Writing a Function Template, p. 986 Iterators, p. 1004 Solving the Arithmetic Exceptions Problem, p. 1020
<b>Chapter 17</b>	Adding an Element to a Linked List, p. 1031 Removing an Element from a Linked List, p. 1038 Solving the Member Insertion by Position Problem, p. 1069
<b>Chapter 18</b>	Storing Objects in an STL Stack, p. 1083 Storing Objects in an STL Queue, p. 1097 Solving the File Reverser Problem, p. 1109
<b>Chapter 19</b>	Inserting an Element into a Binary Tree, p. 1118 Removing an Element from a Binary Tree, p. 1122 Solving the Tree Size Problem, p. 1138

Seventh  
Edition

Starting Out with

# C++ Early Objects

Tony Gaddis  
Judy Walters  
Godfrey Muganda

**Addison-Wesley**

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

**Editor-in-Chief:** Michael Hirsch  
**Editorial Assistant:** Stephanie Sellinger  
**Marketing Coordinator:** Kathryn Ferranti  
**Managing Editor:** Jeffrey Holcomb  
**Production Project Manager:** Heather McNally  
**Senior Manufacturing Buyer:** Carol Melville  
**Media Manufacturing Buyer:** Ginny Michaud  
**Art Director:** Linda Knowles  
**Cover and Interior Designer:** Joyce Cosentino Wells  
**Cover Art:** © 2010 Dmitry Ustyujanin/iStockphoto  
**Media Project Manager:** Katelyn Boller  
**Full-Service Project Management:** Peggy Kellar, Aptara®, Inc.  
**Composition:** Aptara®, Inc.  
**Copyeditor:** Evelyn Perricone

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.

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 © 2011, 2008, 2006, 2005. Pearson Education, Inc., publishing as Addison-Wesley, 501 Boylston Street, Suite 900, Boston, Massachusetts 02116. 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**

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

**Addison-Wesley**  
is an imprint of



[www.pearsonhighered.com](http://www.pearsonhighered.com)

ISBN 10: 0-13-607774-9  
ISBN 13: 978-0-13-607774-9

# Contents at a Glance

	<b>Preface</b>	<b>xv</b>
<b>CHAPTER 1</b>	<b>Introduction to Computers and Programming</b>	<b>1</b>
<b>CHAPTER 2</b>	<b>Introduction to C++</b>	<b>27</b>
<b>CHAPTER 3</b>	<b>Expressions and Interactivity</b>	<b>75</b>
<b>CHAPTER 4</b>	<b>Making Decisions</b>	<b>157</b>
<b>CHAPTER 5</b>	<b>Looping</b>	<b>243</b>
<b>CHAPTER 6</b>	<b>Functions</b>	<b>305</b>
<b>CHAPTER 7</b>	<b>Introduction to Classes and Objects</b>	<b>387</b>
<b>CHAPTER 8</b>	<b>Arrays</b>	<b>485</b>
<b>CHAPTER 9</b>	<b>Searching, Sorting, and Algorithm Analysis</b>	<b>577</b>
<b>CHAPTER 10</b>	<b>Pointers</b>	<b>619</b>
<b>CHAPTER 11</b>	<b>More About Classes and Object-Oriented Programming</b>	<b>677</b>
<b>CHAPTER 12</b>	<b>More About Characters, Strings, and the <code>string</code> Class</b>	<b>771</b>
<b>CHAPTER 13</b>	<b>Advanced File and I/O Operations</b>	<b>829</b>
<b>CHAPTER 14</b>	<b>Recursion</b>	<b>901</b>
<b>CHAPTER 15</b>	<b>Polymorphism and Virtual Functions</b>	<b>935</b>
<b>CHAPTER 16</b>	<b>Exceptions, Templates, and the Standard Template Library (STL)</b>	<b>973</b>
<b>CHAPTER 17</b>	<b>Linked Lists</b>	<b>1023</b>
<b>CHAPTER 18</b>	<b>Stacks and Queues</b>	<b>1071</b>
<b>CHAPTER 19</b>	<b>Binary Trees</b>	<b>1111</b>
	<b>Appendix A: The ASCII Character Set</b>	<b>1141</b>
	<b>Appendix B: Operator Precedence and Associativity</b>	<b>1145</b>
	<b>Index</b>	<b>1147</b>

- Student CD** The following appendices are on the accompanying Student CD.
- Appendix C: A Brief Introduction to Object-Oriented Programming**
  - Appendix D: Using UML in Class Design**
  - Appendix E: Namespaces**
  - Appendix F: Passing Command Line Arguments**
  - Appendix G: Header File and Library Function Reference**
  - Appendix H: Binary Numbers and Bitwise Operations**
  - Appendix I: C++ Casts and Run-Time Type Identification**
  - Appendix J: Multi-Source File Programs**
  - Appendix K: Multiple and Virtual Inheritance**
  - Appendix L: Introduction to the MinGW C++ Compiler and the wxDev-C++ IDE**
  - Appendix M: Introduction to Microsoft Visual C++ 2008 Express Edition**
  - Appendix N: .NET and Managed C++**
  - Appendix O: Introduction to Flowcharting**
  - Appendix P: Answers to Checkpoints**
  - Appendix Q: Answers to Odd-Numbered Review Questions**

# Contents

## Preface xv

### CHAPTER 1 Introduction to Computers and Programming 1

- 1.1 Why Program? 1
- 1.2 Computer Systems: Hardware and Software 2
- 1.3 Programs and Programming Languages 6
- 1.4 What Is a Program Made of? 12
- 1.5 Input, Processing, and Output 16
- 1.6 The Programming Process 17
- 1.7 Tying It All Together: *Hi! It's Me* 22

### 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 35
- 2.4 Standard and Prestandard C++ 37
- 2.5 Variables, Constants, and the Assignment Statement 37
- 2.6 Identifiers 41
- 2.7 Integer Data Types 43
- 2.8 The `char` Data Type 48
- 2.9 The C++ `string` Class 52
- 2.10 Floating-Point Data Types 54
- 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 60
- 2.15 Arithmetic Operators 61
- 2.16 Comments 64
- 2.17 Focus on Software Engineering: *Programming Style* 65
- 2.18 Tying It All Together: *Smile!* 67

**CHAPTER 3 Expressions and Interactivity 75**

- 3.1 The `cin` Object 75
- 3.2 Mathematical Expressions 81
- 3.3 Implicit Type Conversion 89
- 3.4 Explicit Type Conversion 90
- 3.5 Overflow and Underflow 94
- 3.6 Named Constants 97
- 3.7 Multiple and Combined Assignment 101
- 3.8 Formatting Output 105
- 3.9 Working with Characters and String Objects 115
- 3.10 Using C-Strings 120
- 3.11 More Mathematical Library Functions 126
- 3.12 Introduction to Files 130
- 3.13 Focus on Debugging: *Hand Tracing a Program* 138
- 3.14 Green Fields Landscaping Case Study—Part 1 140
- 3.15 Tying It All Together: *Word Game* 142

**CHAPTER 4 Making Decisions 157**

- 4.1 Relational Operators 157
- 4.2 The `if` Statement 162
- 4.3 The `if/else` Statement 172
- 4.4 The `if/else if` Statement 175
- 4.5 Menu-Driven Programs 181
- 4.6 Nested `if` Statements 183
- 4.7 Logical Operators 187
- 4.8 Validating User Input 196
- 4.9 More About Variable Definitions and Scope 197
- 4.10 Comparing Characters and Strings 202
- 4.11 The Conditional Operator 206
- 4.12 The `switch` Statement 210
- 4.13 Enumerated Data Types 219
- 4.14 Testing for File Open Errors 222
- 4.15 Focus on Testing and Debugging: *Validating Output Results* 223
- 4.16 Green Fields Landscaping Case Study—Part 2 225
- 4.17 Tying It All Together: *Fortune Teller* 229

**CHAPTER 5 Looping 243**

- 5.1 The Increment and Decrement Operators 243
- 5.2 Introduction to Loops: The `while` Loop 249
- 5.3 Using the `while` Loop for Input Validation 253
- 5.4 Counters 256
- 5.5 The `do-while` Loop 258
- 5.6 The `for` Loop 263
- 5.7 Keeping a Running Total 269
- 5.8 Sentinels 271
- 5.9 Using a Loop to Read Data from a File 273
- 5.10 Focus on Software Engineering: *Deciding Which Loop to Use* 276
- 5.11 Nested Loops 277
- 5.12 Breaking Out of a Loop 279
- 5.13 The `continue` Statement 281



- 5.14 Focus on Testing and Debugging: *Creating Good Test Data* 284
- 5.15 Central Mountain Credit Union Case Study 287
- 5.16 Tying It All Together: *What a Colorful World* 291

## CHAPTER 6 Functions 305

- 6.1 Modular Programming 305
- 6.2 Defining and Calling Functions 306
- 6.3 Function Prototypes 314
- 6.4 Sending Data into a Function 316
- 6.5 Passing Data by Value 321
- 6.6 The `return` Statement 325
- 6.7 Returning a Value from a Function 326
- 6.8 Returning a Boolean Value 332
- 6.9 Using Functions in a Menu-Driven Program 334
- 6.10 Local and Global Variables 337
- 6.11 Static Local Variables 344
- 6.12 Default Arguments 347
- 6.13 Using Reference Variables as Parameters 350
- 6.14 Overloading Functions 359
- 6.15 The `exit()` Function 363
- 6.16 Stubs and Drivers 366
- 6.17 Little Lotto Case Study 368
- 6.18 Tying It All Together: *Glowing Jack-o-lantern* 373

## CHAPTER 7 Introduction to Classes and Objects 387

- 7.1 Abstract Data Types 387
- 7.2 Object-Oriented Programming 389
- 7.3 Introduction to Classes 391
- 7.4 Introduction to Objects 393
- 7.5 Defining Member Functions 395
- 7.6 Constructors 402
- 7.7 Destructors 407
- 7.8 Private Member Functions 410
- 7.9 Passing Objects to Functions 413
- 7.10 Object Composition 420
- 7.11 Focus on Software Engineering: *Separating Class Specification, Implementation, and Client Code* 424
- 7.12 Input Validation Objects 431
- 7.13 Structures 435
- 7.14 Home Software Company OOP Case Study 449
- 7.15 Introduction to Object-Oriented Analysis and Design 455
- 7.16 Screen Control 464
- 7.17 Tying It All Together: *Yoyo Animation* 469

## CHAPTER 8 Arrays 485

- 8.1 Arrays Hold Multiple Values 485
- 8.2 Accessing Array Elements 487
- 8.3 Inputting and Displaying Array Contents 489
- 8.4 Array Initialization 496
- 8.5 Processing Array Contents 502

- 8.6 Using Parallel Arrays 513
- 8.7 The `typedef` Statement 516
- 8.8 Arrays as Function Arguments 517
- 8.9 Two-Dimensional Arrays 526
- 8.10 Arrays with Three or More Dimensions 534
- 8.11 Vectors 537
- 8.12 Arrays of Class Objects 549
- 8.13 National Commerce Bank Case Study 559
- 8.14 Tying It All Together: *Rock, Paper, Scissors* 561

**CHAPTER 9 Searching, Sorting, and Algorithm Analysis 577**

- 9.1 Introduction to Search Algorithms 577
- 9.2 Searching an Array of Objects 584
- 9.3 Introduction to Sorting Algorithms 587
- 9.4 Sorting an Array of Objects 596
- 9.5 Sorting and Searching Vectors 599
- 9.6 Introduction to Analysis of Algorithms 601
- 9.7 Case Studies 609
- 9.8 Tying It All Together: *Secret Messages* 610

**CHAPTER 10 Pointers 619**

- 10.1 Pointers and the Address Operator 619
- 10.2 Pointer Variables 621
- 10.3 The Relationship Between Arrays and Pointers 625
- 10.4 Pointer Arithmetic 629
- 10.5 Initializing Pointers 630
- 10.6 Comparing Pointers 632
- 10.7 Pointers as Function Parameters 635
- 10.8 Pointers to Constants and Constant Pointers 639
- 10.9 Focus on Software Engineering: *Dynamic Memory Allocation* 643
- 10.10 Focus on Software Engineering: *Returning Pointers from Functions* 648
- 10.11 Pointers to Class Objects and Structures 652
- 10.12 Focus on Software Engineering: *Selecting Members of Objects* 659
- 10.13 United Cause Relief Agency Case Study 661
- 10.14 Tying It All Together: *Pardon Me, Do You Have the Time?* 669

**CHAPTER 11 More About Classes and Object-Oriented Programming 677**

- 11.1 The `this` Pointer and Constant Member Functions 677
- 11.2 Static Members 681
- 11.3 Friends of Classes 689
- 11.4 Memberwise Assignment 694
- 11.5 Copy Constructors 695
- 11.6 Operator Overloading 704
- 11.7 Type Conversion Operators 728
- 11.8 Convert Constructors 731
- 11.9 Aggregation and Composition 734
- 11.10 Inheritance 740
- 11.11 Protected Members and Class Access 745

- 11.12 Constructors, Destructors, and Inheritance 750
- 11.13 Overriding Base Class Functions 755
- 11.14 Tying It All Together: *Putting Data on the World Wide Web* 757

## **CHAPTER 12 More About Characters, Strings, and the `string` Class 771**

- 12.1 C-Strings 771
- 12.2 Library Functions for Working with C-Strings 776
- 12.3 Conversions Between Numbers and Strings 787
- 12.4 Character Testing 793
- 12.5 Character Case Conversion 797
- 12.6 Writing Your Own C-String Handling Functions 800
- 12.7 More About the C++ `string` Class 805
- 12.8 Creating Your Own `string` Class 810
- 12.9 Advanced Software Enterprises Case Study 817
- 12.10 Tying It All Together: *Program Execution Environments* 819

## **CHAPTER 13 Advanced File and I/O Operations 829**

- 13.1 Files 829
- 13.2 Output Formatting 838
- 13.3 Passing File Stream Objects to Functions 845
- 13.4 More Detailed Error Testing 847
- 13.5 Member Functions for Reading and Writing Files 849
- 13.6 Binary Files 862
- 13.7 Creating Records with Structures 866
- 13.8 Random-Access Files 871
- 13.9 Opening a File for Both Input and Output 878
- 13.10 Online Friendship Connections Case Study 883
- 13.11 Tying It All Together: *File Merging and Color-Coded HTML* 888

## **CHAPTER 14 Recursion 901**

- 14.1 Introduction to Recursion 901
- 14.2 The Recursive Factorial Function 908
- 14.3 The Recursive gcd Function 910
- 14.4 Solving Recursively Defined Problems 911
- 14.5 A Recursive Binary Search Function 913
- 14.6 Focus on Problem Solving and Program Design: *The QuickSort Algorithm* 915
- 14.7 The Towers of Hanoi 919
- 14.8 Focus on Problem Solving: *Exhaustive and Enumeration Algorithms* 922
- 14.9 Focus on Software Engineering: *Recursion Versus Iteration* 926
- 14.10 Tying It All Together: *Infix and Prefix Expressions* 927

## **CHAPTER 15 Polymorphism and Virtual Functions 935**

- 15.1 Type Compatibility in Inheritance Hierarchies 935
- 15.2 Polymorphism and Virtual Member Functions 941
- 15.3 Abstract Base Classes and Pure Virtual Functions 946
- 15.4 Focus on Object-Oriented Programming: *Composition Versus Inheritance* 952
- 15.5 Secure Encryption Systems, Inc., Case Study 957
- 15.6 Tying It All Together: *Let's Move It* 961

**CHAPTER 16 Exceptions, Templates, and the Standard Template Library (STL) 973**

- 16.1 Exceptions 973
- 16.2 Function Templates 985
- 16.3 Class Templates 993
- 16.4 Class Templates and Inheritance 999
- 16.5 Introduction to the Standard Template Library 1002
- 16.6 Tying It All Together: *Word Transformers Game* 1015

**CHAPTER 17 Linked Lists 1023**

- 17.1 Introduction to the Linked List ADT 1023
- 17.2 Linked List Operations 1029
- 17.3 A Linked List Template 1041
- 17.4 Recursive Linked List Operations 1045
- 17.5 Variations of the Linked List 1054
- 17.6 The STL `list` Container 1054
- 17.7 Reliable Software Systems, Inc., Case Study 1056
- 17.8 Tying It All Together: *More on Graphics and Animation* 1060

**CHAPTER 18 Stacks and Queues 1071**

- 18.1 Introduction to the Stack ADT 1071
- 18.2 Dynamic Stacks 1079
- 18.3 The STL `stack` Container 1082
- 18.4 Introduction to the Queue ADT 1084
- 18.5 Dynamic Queues 1092
- 18.6 The STL `deque` and `queue` Containers 1096
- 18.7 Focus on Problem Solving and Program Design: *Eliminating Recursion* 1098
- 18.8 Tying It All Together: *Converting Postfix Expressions to Infix* 1103

**CHAPTER 19 Binary Trees 1111**

- 19.1 Definition and Applications of Binary Trees 1111
- 19.2 Binary Search Tree Operations 1115
- 19.3 Template Considerations for Binary Search Trees 1131
- 19.4 Tying It All Together: *Genealogy Trees* 1131

**Appendix A: The ASCII Character Set 1141**

**Appendix B: Operator Precedence and Associativity 1145**

**Index 1147**

- Student CD** The following appendices are on the accompanying Student CD.
- Appendix C: A Brief Introduction to Object-Oriented Programming**
  - Appendix D: Using UML in Class Design**
  - Appendix E: Namespaces**
  - Appendix F: Passing Command Line Arguments**
  - Appendix G: Header File and Library Function Reference**
  - Appendix H: Binary Numbers and Bitwise Operations**
  - Appendix I: C++ Casts and Run-Time Type Identification**
  - Appendix J: Multi-Source File Programs**
  - Appendix K: Multiple and Virtual Inheritance**
  - Appendix L: Introduction to the MinGW C++ Compiler and the wxDev-C++ IDE**
  - Appendix M: Introduction to Microsoft Visual C++ 2008 Express Edition**
  - Appendix N: .NET and Managed C++**
  - Appendix O: Introduction to Flowcharting**
  - Appendix P: Answers to Checkpoints**
  - Appendix Q: Answers to Odd-Numbered Review Questions**



# Preface

Welcome to *Starting Out with C++: Early Objects*, 7th 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 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 Seventh Edition

This book's pedagogy, organization, and clear writing style remain the same as in the previous edition. However, many improvements have been made to make it even more student-friendly and to keep it state of the art for introductory programming using the C++ programming language.

- **Updated Material**  
Material has been updated throughout the book to reflect changes in technology, operating systems, and software development environments, as well as to improve clarity and incorporate best practices in object-oriented programming.
- **New Material**  
New material has been added on a number of topics including embedding operating system calls in program code, using object composition and aggregation, and creating text-based graphics.
- **Completely Revised Chapter 7**  
Chapter 7, *Introduction to Classes and Objects*, has been reorganized and almost entirely rewritten to start right in with classes and objects, instead of introducing structures first.

- **Greater Focus on Object-Oriented Programming**  
Many examples throughout the text have been rewritten to incorporate appropriate use of classes and objects.
- **Reusability**  
Material has been added illustrating how to create general classes that can be appropriately reused in multiple applications.
- **Improved Diagrams**  
Many diagrams have been improved and new diagrams added to better illustrate important concepts.
- **Online VideoNotes**  
An extensive set of online videos have been developed to accompany this text. Throughout the book, *VideoNotes* icons alert the student to videos covering specific topics they are studying. Additionally, one Programming Challenge at the end of each chapter now has an accompanying video explaining how to develop the problem's solution. The videos are available at <http://www.pearsonhighered.com/gaddis/>
- **New *Tying It All Together* Sections**  
A new *Tying It All Together* section has been added at the end of every chapter that shows the student how to do something clever and fun with the material covered in that chapter.
- **New Programming Challenges**  
New Programming Challenges have been added to every chapter, including a number of Challenges that ask students to develop object-oriented solutions and to create solutions that reuse, modify, and build on previously written code.
- **New Compiler and IDE Bundled with the Book**  
The MinGW C++ Compiler and wxDev-C++ Software Development Environment now come bundled, for free, with the book.
- **New Appendices**  
An Appendix has been added on using the MinGW C++ Compiler and wxDev-C++ IDE that accompany the book. Additional new appendices cover the Microsoft Visual C++ 2008 Express Edition IDE and Multiple and Virtual Inheritance.

## 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.

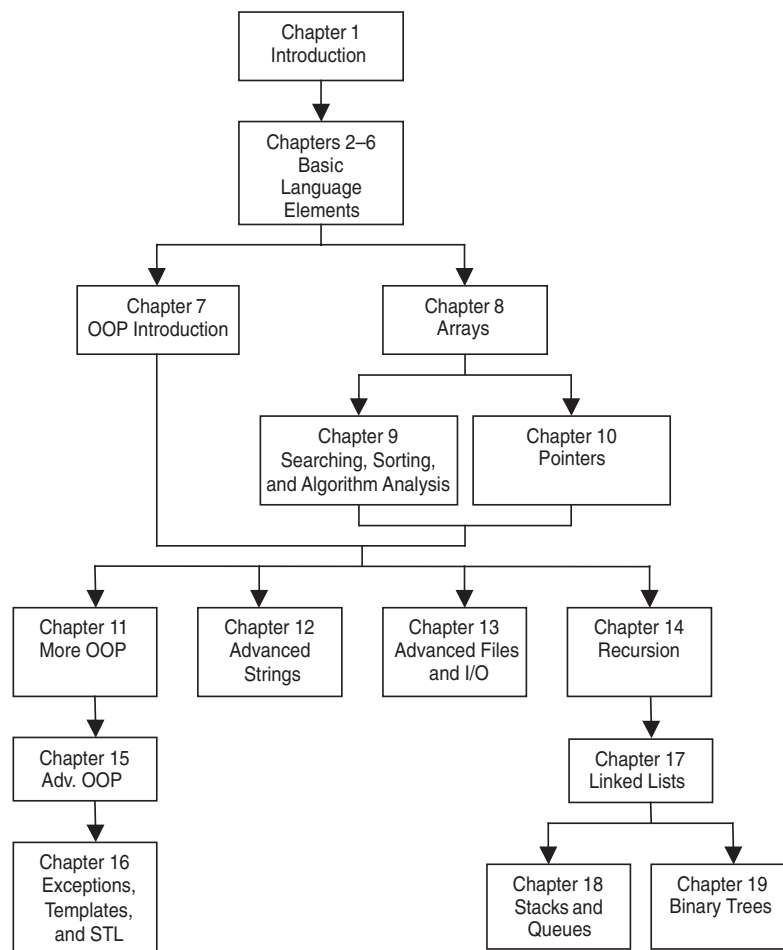
Chapter 1 covers fundamental hardware, software, and programming concepts. The instructor may choose to skip this chapter if the class has already mastered those topics. Chapters 2 through 6 cover basic C++ syntax, data types, expressions, selection structures, repetition structures, and functions. Each 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 8.12 (Arrays of Class 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 new *Tying It All Together* section shows students how to use the `cout` statement to create a personalized output message. Two new Programming Challenges 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, variable definitions, assignment statements, constants, comments, program output, and simple arithmetic operations. 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. An optional section explains the difference between ANSI standard and prestandard C++ programs. The new *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 reading and writing sequential files, on simple output formatting, on data type conversion and type casting, and on using library functions that work with numbers. For those who wish to cover them, there is also a section on C-strings. The new *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 the `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 continues the theme of debugging with a section on validating output results. The new *Tying It All Together* section uses random numbers and branching statements to create a fortune telling game.

### Chapter 5: Looping

This chapter covers C++'s repetitive control mechanisms. The `while` loop, `do-while` loop, and `for` loop are taught, 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 covered. The emphasis on testing and debugging continues, with a section on creating good test data. The new *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 new *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 learn how to define classes and to create and use objects. 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 introduces structures and uses them in this chapter's *Tying It All Together* section, where students learn to use screen control techniques to create an animation that simulates the motion of a yoyo.

## **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 the 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. STL vectors are introduced and compared to arrays. Sections on arrays of objects and structures are located at the end of the chapter, so they can be covered now or saved for later if the instructor wishes to cover this chapter before Chapter 7. The new *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. The new *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. The topics include pointer arithmetic, initialization of pointers, comparison of pointers, pointers and arrays, pointers and functions, dynamic memory allocation, and more. The new *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. It covers object aggregation and composition, as well as inheritance, and illustrates the difference between is-a and has-a relations. Constant member functions, static members, friends, memberwise assignment, copy constructors, object type conversion operators, convert constructors, and a newly rewritten section on operator overloading are also included. The new *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 about Characters, Strings, and the `string` Class**

This chapter covers standard library functions for working with characters and C-strings, covering topics such as passing C-strings to functions and using the C++ `sstream` classes to convert between numeric and string forms of numbers. Additional material about the C++ string class and its member functions and operators is presented and a new, improved program illustrates how to write your own string class. The new *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 covers sequential access, random access, text, and binary files. Various modes for opening files are discussed, as well as the many methods for reading and writing file contents. Advanced output formatting is also covered. The new *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 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 new *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 material on multiple inheritance previously in the chapter has been rewritten and moved to an appendix. The new *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 function and class templates as a method for writing generic code. Finally, the student is introduced to the containers, iterators, and algorithms offered by the Standard Template Library (STL). The new *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 is taught to code operations such as creating a linked list, appending a node, traversing the list, searching for a node, inserting a node, deleting a node, and destroying a list. A linked list class template is also demonstrated. The new *Tying It All Together* section brings together many of the most important concepts of OOP by using objects, inheritance, and polymorphism 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 new *Tying It All Together* section discusses strategies for evaluating postfix expressions and for converting them 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 an element, delete an element, replace an element, test for an element, and destroy a tree. The new *Tying It All Together* section introduces a tree structure versatile enough to create genealogy trees.

## **Appendices**

**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.

### **The following appendices are on the accompanying student CD**

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

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

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

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

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

**Appendix J: 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 K: 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 L: Introduction to the MinGW C++ Compiler and the wxDev-C++ IDE** A tutorial on how to start a wxDev-C++ project, compile and run a program, save source files, and more.

**Appendix M: Introduction to Microsoft Visual C++ 2008 Express Edition** A tutorial on how to start a project using Microsoft Visual C++ 2008, compile and run a program, save source files, and more.





**Appendix N: .NET and Managed C++** A short introduction to Microsoft .NET and managed C++.

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

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

## 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/gaddis/>. *VideoNotes* 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 on the student CD 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 student CD. 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 on the student CD.
- Programming Challenges* 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. When applicable, these exercises include input validation rules.



**Group Projects**

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 team work within the classroom.

**C++ Quick Reference Guide**

For easy access, a quick reference guide to the C++ language is printed on the inside front and back covers.

## Supplements

**Student CD**

This CD includes:

- MinGW C++ Compiler
- wxDev-C++ IDE
- Answers to all Checkpoint questions (Appendix P)
- Answers to all odd-numbered Review Questions and Exercises (Appendix Q)
- Complete source code for every program included in the book
- Additional case studies, complete with source code
- Serendipity Booksellers ongoing software development project
- A full set of appendices (including several tutorials) that accompany the book

If a CD did not come with your book or you can't locate your CD, you can access most of these items at <http://www.pearsonhighered.com/cssupport>

**Other CDs Upon Request** Professors should contact their campus Pearson Education/Addison-Wesley representative for the specific ISBN to order this book packaged with Microsoft Visual C++.

### MyCodeMate—Your Own T.A. Just a Click Away

Addison-Wesley's *MyCodeMate* is a book-specific Web resource that provides tutorial help and evaluation of student programs. Example programs throughout the book and selected Programming Challenges from every chapter have been integrated into *MyCodeMate*. Using this tool, a student is able to write and compile programs from any computer with Internet access and receive guidance and feedback on how to proceed and on how to address compiler error messages. Instructors can track each student's progress on Programming Challenges from the text or can develop projects of their own. **A complimentary subscription to *MyCodeMate* is offered when the access code is ordered in a package with a new copy of this text.** Subscriptions can also be purchased online. For more information visit [www.mycodemate.com](http://www.mycodemate.com), or contact your campus Pearson Education/Addison-Wesley representative.

**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>) or send an email to [computing@aw.com](mailto:computing@aw.com) for information on how to access them.

### Textbook Web Site

A Web site for the *Starting Out with C++* series of books is located at the following URL:

<http://www.pearsonhighered.com/gaddis/>

## Get This Book the Way You Want It!

This book is part of the Pearson Custom Computer Science Library. Use our online PubSelect system to select just the chapters you need from this, and other, Pearson Education CS textbooks. You can edit the sequence to exactly match your course organization and teaching approach. Visit [www.pearsoncustom.com/cs](http://www.pearsoncustom.com/cs) for details.

## 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++: Control Structures through Objects;*
- *Starting Out with C++: Early Objects;*
- *Starting Out with C++: Brief Version.*

The following chart will help you determine which book is right for your course.

<ul style="list-style-type: none"> <li>▪ FROM CONTROL STRUCTURES THROUGH OBJECTS</li> <li>▪ BRIEF VERSION</li> </ul>	<ul style="list-style-type: none"> <li>▪ EARLY OBJECTS</li> </ul>
<p><b>LATE INTRODUCTION OF OBJECTS</b> Classes are introduced in Chapter 13 of the standard text and Chapter 11 of the brief text, after control structures, functions, arrays, and pointers. Advanced OOP topics, such as inheritance and polymorphism, are covered in the following two chapters.</p> <p><b>USE OF C-STRINGS</b> Null-terminated C-strings are used throughout, with the C++ <code>string</code> class covered briefly.</p> <p><b>INTRODUCTION OF DATA STRUCTURES AND RECURSION</b> Linked lists, stacks and queues, and binary trees are introduced in the final chapters of the standard text. Recursion is covered after stacks and queues, but before binary trees. These topics are not covered in the brief text, though it does have appendices dealing with linked lists and recursion.</p>	<p><b>EARLIER INTRODUCTION OF OBJECTS</b> Classes are introduced in Chapter 7, after control structures and functions, but before arrays and pointers. Their use is then integrated into the remainder of the text. Advanced OOP topics, such as inheritance and polymorphism, are covered in Chapters 11 and 15.</p> <p><b>USE OF <code>string</code> OBJECTS</b> Standard library <code>string</code> class objects are used throughout, with C-strings covered briefly.</p> <p><b>INTRODUCTION OF DATA STRUCTURES AND RECURSION</b> Linked lists, stacks and queues, and binary trees are introduced in the final chapters of the text, after the chapter on recursion.</p>

## 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 Seventh Edition or Its Previous Versions

Ahmad Abuhejleh <i>University of Wisconsin, River Falls</i>	Richard Flint <i>North Central College</i>
David Akins <i>El Camino College</i>	Sheila Foster <i>California State University Long Beach</i>
Steve Allan <i>Utah State University</i>	David E. Fox <i>American River College</i>
Ijaz A. Awan <i>Savannah State University</i>	Cindy Fry <i>Baylor University</i>
John Bierbauer <i>North Central College</i>	Peter Gacs <i>Boston University</i>
Don Biggerstaff <i>Fayetteville Technical Community College</i>	Cristi Gale <i>Sterling College</i>
Paul Bladek <i>Spokane Falls Community College</i>	James Gifford <i>University of Wisconsin, Stevens Point</i>
Chuck Boehm <i>Dean Foods, Inc.</i>	Leon Gleiberman <i>Touro College</i>
Bill Brown <i>Pikes Peak Community College</i>	Simon Gray <i>Ashland University—Ohio</i>
Richard Cacace <i>Pensacola Junior College</i>	Margaret E. Guertin <i>Tufts University</i>
Randy Campbell <i>Morningside College</i>	Jamshid Haghighi <i>Guilford Technical Community College</i>
Stephen P. Carl <i>Wright State University</i>	Ranette H. Halverson <i>Midwestern State University, Wichita Falls, TX</i>
Wayne Caruolo <i>Red Rocks Community College</i>	Dennis Heckman <i>Portland Community College</i>
Thomas Cheatham <i>Middle Tennessee State University</i>	Ric Heishman <i>Northern Virginia Community College</i>
James Chegwidan <i>Tarrant County College</i>	Patricia Hines <i>Brookdale Community College</i>
John Cigas <i>Rockhurst University</i>	Mike Holland <i>Northern Virginia Community College</i>
John Cross <i>Indiana University of Pennsylvania</i>	Lister Wayne Horn <i>Pensacola Junior College</i>
Joseph DeLibero <i>Arizona State University</i>	Richard Hull <i>Lenoir-Rhyne College</i>
Dennis Fairclough <i>Utah Valley State College</i>	Norman Jacobson <i>University of California, Irvine</i>
Larry Farrer <i>Guilford Technical Community College</i>	Eric Jiang <i>San Diego State University</i>

- David Kaeli  
*Northeastern University*
- Chris Kardaras  
*North Central College*
- Eugene Katzen  
*Montgomery College—Rockville*
- Willard Keeling  
*Blue Ridge Community College*
- A. J. Krygeris  
*Houston Community College*
- Ray Larson  
*Inver Hills Community College*
- Stephen Leach  
*Florida State University*
- Parkay Louie  
*Houston Community College*
- Zhu-qu Lu  
*University of Maine, Presque Isle*
- Tucjer Maney  
*George Mason University*
- Bill Martin  
*Central Piedmont Community College*
- Debbie Mathews  
*J. Sargeant Reynolds*
- Ron McCarty  
*Penn State Erie, The Behrend College*
- Robert McDonald  
*East Stroudsburg University*
- James McGuffee  
*Austin Community College*
- M. Dee Medley  
*Augusta State University*
- Cathi Chambley-Miller  
*Aiken Technical College*
- Sandeep Mitra  
*SUNY Brockport*
- Frank Paiano  
*Southwestern Community College*
- Theresa Park  
*Texas State Technical College*
- Mark Parker  
*Shoreline Community College*
- Robert Plantz  
*Sonoma State University*
- Tino Posillico  
*SUNY Farmingdale*
- M. Padmaja Rao  
*Francis Marion University*
- Timothy Reeves  
*San Juan College*
- Ronald Robison  
*Arkansas Tech University*
- Caroline St. Clair  
*North Central College*
- Dolly Samson  
*Weber State University*
- Kate Sanders  
*Rhode Island College*
- Lalchand Shimpi  
*Saint Augustine's College*
- Sung Shin  
*South Dakota State University*
- Garth Sorenson  
*Snow College*
- Daniel Spiegel  
*Kutztown University*
- Ray Springston  
*University of Texas at Arlington*
- Kirk Stephens  
*Southwestern Community College*
- Cherie Stevens  
*South Florida Community College*
- Hong Sung  
*University of Central Oklahoma*
- Mark Swanson  
*Red Wing Technical College*
- Martha Tillman  
*College of San Mateo*
- Delores Tull  
*Itawamba Community College*
- Rober Tureman  
*Paul D. Camp Community College*
- Jane Turk  
*LaSalle University*
- Sylvia Unwin  
*Bellevue Community College*
- Stewart Venit  
*California State University, Los Angeles*
- Doug White  
*University of Northern Colorado*
- Chris Wild  
*Old Dominion University*
- Catherine Wyman  
*DeVry Institute of Technology, Phoenix*
- Sherali Zeadally  
*University of the District of Columbia*

The authors would like to thank their students at Haywood Community College and North Central College for inspiring them to write student-friendly books. They would also like to thank their families for their tremendous support throughout this project, as well as North Central College for providing Prof. Walters and Muganda with the sabbatical term during which they worked on this book. An especially big thanks goes to our terrific editorial, production, and marketing team at Addison-Wesley. In particular we want to thank our editor Michael Hirsch and our production project manager Heather McNally, who have been instrumental in guiding the production of this book. We also want to thank our project manager, Peggy Kellar, who helped everything run smoothly, and our meticulous and knowledgeable copyeditor, Evelyn Perricone, who dedicated many hours to making this book the best book it could be. You are great people to work with!

## About the Authors

Tony Gaddis is the principal author of the *Starting Out With . . .* series of textbooks. He is a highly acclaimed instructor with twenty years of experience teaching computer science courses at Haywood Community College. Tony was previously selected as the North Carolina Community College “Teacher of the Year” and has received the Teaching Excellence award from the National Institute for Staff and Organizational Development. The *Starting Out With . . .* series includes introductory books covering C++, Java™, Microsoft® Visual Basic®, Microsoft® C#, and Alice, all published by Addison-Wesley.

Judy Walters is an Associate Professor of Computer Science at North Central College in Naperville, Illinois. In addition to her many computer science courses, she also teaches two film-related courses she developed for the college’s interdisciplinary freshman seminar program. She recently returned from her second semester teaching in Costa Rica, where she hopes to retire some day.

Godfrey Muganda is an Associate Professor of Computer Science at North Central College. He teaches a wide variety of courses at both the undergraduate and graduate levels, including courses in Algorithms, Computer Organization, Web Applications, and Web Services. His primary research interests are in the area of Fuzzy Sets and Systems.