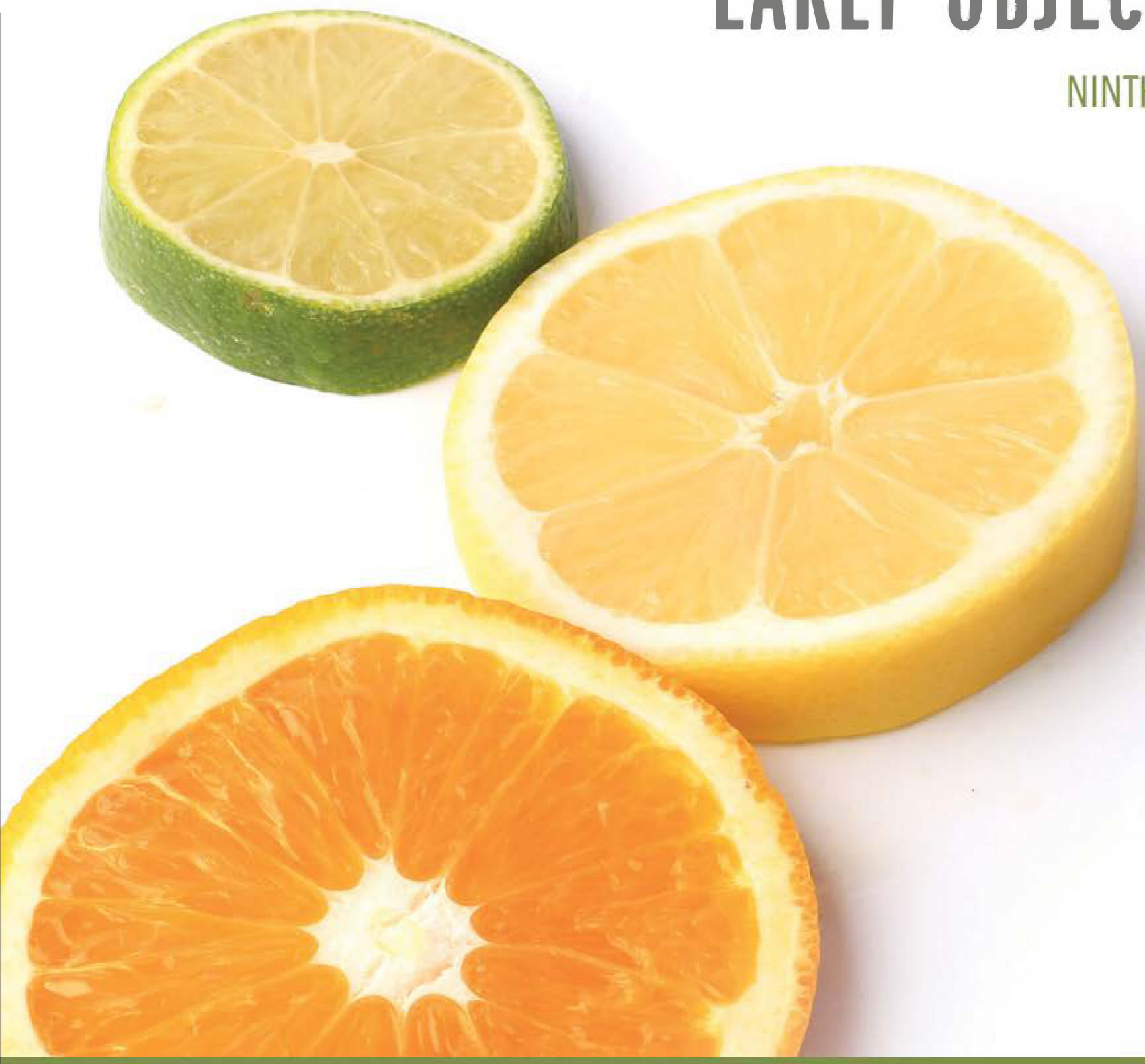
starting out with >>>

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
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 MyProgammingLab, which is available at **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

This page intentionally left blank

LOCATION OF VIDEONOTES IN THE TEXT

Chapter 1	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
Chapter 2	Using `cout` to Display Output, p. 32 Assignment Statements, p. 59 Arithmetic Operators, p. 62 Solving the Restaurant Bill Problem, p. 74
Chapter 3	Using `cin` to Read Input, p. 77 Evaluating Mathematical Expressions, p. 84 Combined Assignment Operators, p. 102 Solving the Stadium Seating Problem, p. 148
Chapter 4	Using an `if` Statement, p. 161 Using an `if/else` Statement, p. 170 Using an `if/else if` Statement, p. 176 Using Logical Operators, p. 189 Solving the Time Calculator Problem, p. 238
Chapter 5	The `while` Loop, p. 244 The `for` Loop, p. 271 Nested Loops, p. 279 Solving the Ocean Levels Problem, p. 317
Chapter 6	Defining and Calling Functions, p. 324 Using Function Arguments, p. 333 Value-Returning Functions, p. 343 Solving the Markup Problem, p. 399
Chapter 7	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
Chapter 8	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
Chapter 9	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)*

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

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://lccn.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

This page intentionally left blank

This page intentionally left blank

Contents

Additional Appendices

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

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

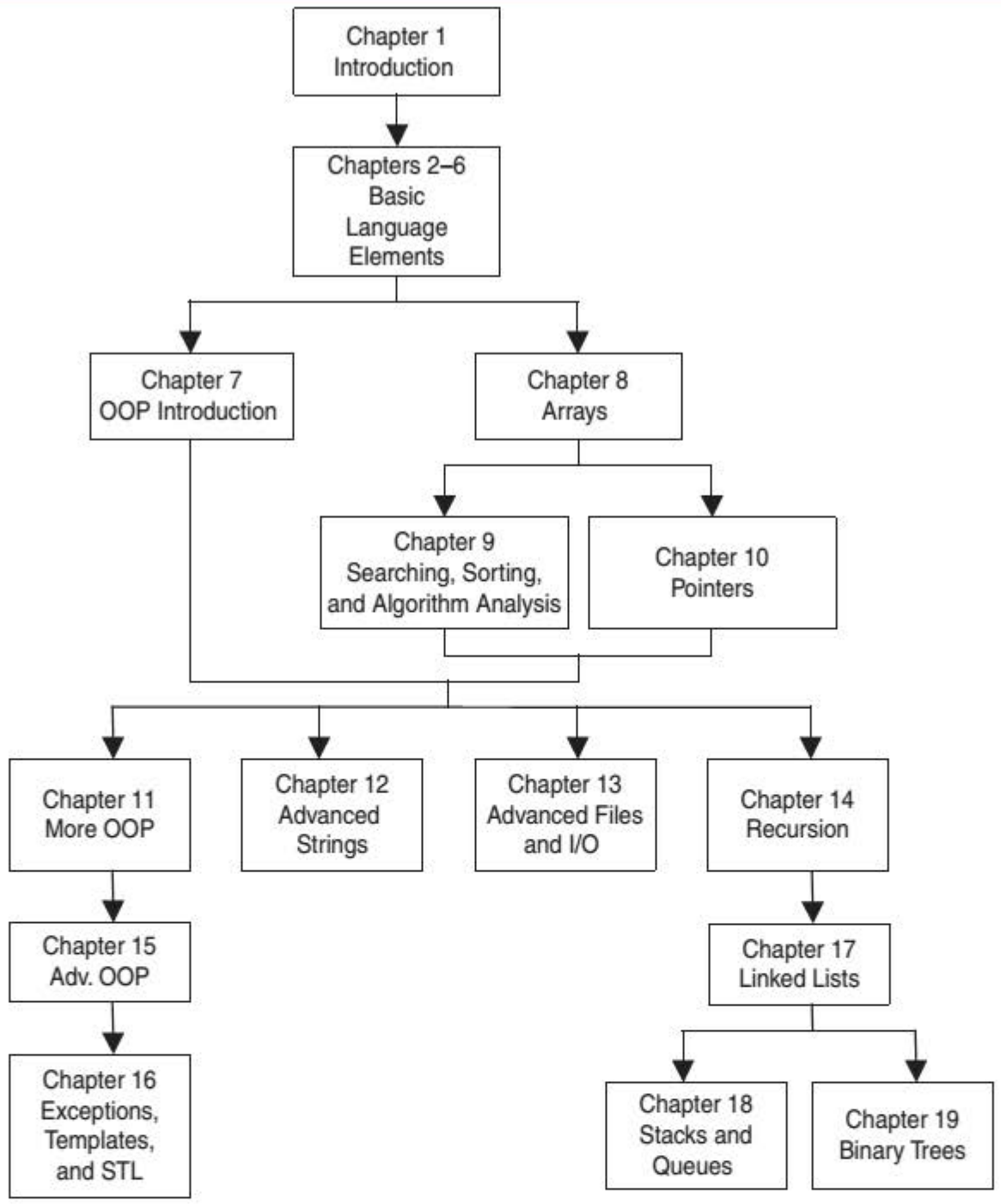
This text teaches C++ in a step-by-step fashion. Each chapter covers a major set of topics and builds knowledge as the student progresses through the book. Although the chapters can be easily taught in their existing sequence, flexibility is provided. The following dependency diagram (Figure P-1) suggests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


Chapter 1
Introduction
Chapters 2–6
Basic
Language
Elements
Chapter 7
OOP Introduction
Chapter 8
Arrays
Chapter 9
Searching, Sorting,
and Algorithm Analysis
Chapter 10
Pointers
Chapter 11
More OOP
Chapter 12
Advanced
Strings
Chapter 13
Advanced Files
and I/O
Chapter 14
Recursion
Chapter 15
Adv. OOP
Chapter 16
Exceptions,
Templates,
and STL
Chapter 17
Linked Lists
Chapter 18
Stacks and
Queues
Chapter 19
Binary Trees

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 selftest 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.

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.
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 teamwork within the classroom.
C++ Quick Reference Guide	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:

- 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.

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
University of Wisconsin, River Falls

David Akins
El Camino College

Steve Allan
Utah State University

Ijaz A. Awan
Savannah State University

John Bierbauer
North Central College

Don Biggerstaff
Fayetteville Technical Community College

Paul Bladek
Spokane Falls Community College

Chuck Boehm
Dean Foods, Inc.

Bill Brown
Pikes Peak Community College

Richard Cacace
Pensacola Junior College

Randy Campbell
Morningside College

Stephen P. Carl
Wright State University

Wayne Caruolo
Red Rocks Community College

Thomas Cheatham
Middle Tennessee State University

James Chegwidden
Tarrant County College

John Cigas
Rockhurst University

John Cross
Indiana University of Pennsylvania

Fred M. D'Angelo
Pima Community College

Joseph DeLibero
Arizona State University

Dennis Fairclough
Utah Valley State College

Larry Farrer
Guilford Technical Community College

James D. Fitzgerald
Golden West College

Richard Flint
North Central College

Sheila Foster
California State University Long Beach

David E. Fox
American River College

Cindy Fry
Baylor University

Peter Gacs
Boston University

Cristi Gale
Sterling College

James Gifford
University of Wisconsin, Stevens Point

Leon Gleiberman
Touro College

Simon Gray
Ashland University—Ohio

Margaret E. Guertin
Tufts University

Jamshid Haghighi
Guilford Technical Community College

Ranette H. Halverson
Midwestern State University, Wichita Falls, TX