Preface

"The chief merit of language is clearness …"
—Galen

Welcome to the world of C++ programming and *C++ How to Program, Late Objects Version, Seventh Edition*! This book is appropriate for introductory programming courses (commonly known as CS1 and CS2). It also provides substantial depth for those who already know how to program in another language and want to learn C++.

At the heart of this book is the Deitel signature "live-code approach." Concepts are presented in the context of complete working C++ programs. Each code example is immediately followed by one or more sample executions. All the source code is available at `www.deitel.com/books/cpphtp7LOV/`.

Motivation for C++ *How to Program, Late Objects Version, 7/e*

The two most popular approaches for teaching first courses in C++ programming are the *late objects approach* and the *early objects approach*. To meet these needs, we've published two versions of the seventh edition of *C++ How to Program*:

- *C++ How to Program, Late Objects Version, 7/e* (this book), and
- *C++ How to Program, 7/e* (which offers an early objects approach).

Chapters 1–8 in this book form the core of a procedural programming CS1 course that covers operators, data types, input/output, control statements, methods, arrays, strings, pointers and files. Instructors who want to introduce *some* object-oriented programming in a first course can include portions of the next several chapters.

The key differences between the two books are the order in which topics are presented in Chapters 1–8, the front portion of Chapter 9 and the coverage of files in Chapter 17. The books are virtually identical from the middle of Chapter 9 through Chapter 27 and the appendices, except that the late objects book moves the treatment of sequential files from Chapter 17 forward to Chapter 8. In *C++ How to Program, Late Objects Version, 7/e*, Chapters 22–27 are available only in PDF format at the book's companion website.

Features of this "Late Objects" Version

Here are some key features of *C++ How to Program, Late Objects Version, 7/e:*

- ***Late-Objects Approach:*** We briefly introduce the basic concepts and terminology of object technology in Chapter 1—this section can be deferred to the beginning of Chapter 9 by instructors who wish to teach only procedural programming in the early chapters. You'll develop your first *customized* classes and objects in Chapter 9.

- *Sequential File Processing.* We introduce sequential file processing in Chapter 8. This enables instructors to teach these topics in a first course if they wish. We cover random-access file processing in Chapter 17.
- *VideoNotes.* On the book's Companion Website (`www.pearsonhighered.com/deitel/`), we provide extensive *VideoNotes* that walk you through *all* of the code examples in Chapters 2–13 and a portion of Chapter 16. The companion website also includes solutions to many of the book's exercises, bonus chapters, and more (see the Companion Website section later in this Preface).
- *Making a Difference Exercises.* The dozens of Making a Difference exercises explore complex and challenging social issues, including global warming, population growth, gender neutrality, computerization of health records, alternative tax plans, hybrid vehicles, accessibility for people with disabilities and more. A complete list of the Making a Difference exercises is available at `www.deitel.com/books/cpphtp7LOV/`.
- *Prefer **`string`** Objects to C Strings.* C++ offers two types of strings—`string` class objects (which we use starting in Chapter 6) and C-style, pointer-based strings. We include some early discussions of C strings to give you practice with pointer manipulations, to illustrate dynamic memory allocation with `new` and `delete` and to prepare you for working with C strings in the "legacy code" that you'll encounter in industry. In new development, you should favor `string` class objects, so we use instances of C++ class `string` to make programs more robust and eliminate many of the security problems that can be caused by manipulating C strings.
- *Prefer **`vector`**s to C Arrays.* Similarly, C++ offers two types of arrays—`vector` class objects (which we use starting in Chapter 6) and C-style, pointer-based arrays. As appropriate, we use class template `vector` instead of C arrays throughout the book. However, we begin discussing C arrays in Chapter 6 to prepare you for working with legacy code and to use them as a basis for building your own customized `Array` class in Chapter 11, Operator Overloading.
- *Titled Programming Exercises.* We've titled the programming exercises. This helps instructors select assignments for their classes.
- *Consistent Use of **`const`**.* Keyword `const` is used throughout to encourage better software engineering.
- *Function Pointer Exercises.* We include several real-world function-pointers exercises. These are available at the Companion Website.
- *Terminology Sections.* We include page numbers for the defining occurrences of all terms in the terminology lists for easy reference.

Other Features

Other features of *C++ How to Program, Late Objects Version, 7/e*, include:

- *Game Programming.* The computer-game industry's revenues are already greater than those of the first-run movie business, creating lots of career opportunities. Chapter 27, Game Programming with Ogre, introduces game programming and graphics with the open source Ogre 3D graphics engine. We discuss basic issues involved in game programming, then demonstrate how to create a scene with 3D color graphics, smoothly animate moving objects, use timers to control animation speed, detect collisions between objects and add sound.

- *Future of C++.* Chapter 23 considers the future of C++—we introduce the Boost C++ Libraries, Technical Report 1 (TR1) and C++0x. The free Boost open source libraries are created by members of the C++ community. Technical Report 1 describes the proposed changes to the C++ Standard Library, many of which are based on current Boost libraries. The C++ Standards Committee is revising the C++ Standard. The main goals for the new standard are to make C++ easier to learn, improve library building capabilities, and increase compatibility with the C programming language. The new standard will include changes to the core language and many of the libraries in TR1. We overview the Boost libraries and provide code examples for the "regular expression" and "smart pointer" libraries. Regular expressions are used to match specific character patterns in text. They can be used, for example, to validate data to ensure that it's in a particular format, to replace parts of one string with another or to split a string. Many common bugs in C and C++ code are related to pointers, a powerful programming capability you'll study in Chapter 7, Pointers. Smart pointers help you avoid errors by providing additional functionality beyond that of standard pointers.

- *Integrated Case Studies.* We provide many case studies, including the `Time` class in Chapters 9–10, the `Employee` class in Chapters 12–13, and the optional OOD/UML ATM case study in Chapters 25–26.

- *Function Call Stack Explanation.* In Chapter 5, we provide a detailed discussion (with illustrations) of the function call stack and activation records to explain how C++ is able to keep track of which function is currently executing, how automatic variables of functions are maintained in memory and how a function knows where to return after it completes execution.

- *Compilation and Linking Process for Multiple-Source-File Programs.* Chapter 9 includes a detailed diagram and discussion of the compilation and linking process that produces an executable program.

- *Inheritance and Polymorphism.* Chapters 12–13 have been carefully developed using an `Employee` class hierarchy to make the treatment of inheritance and polymorphism accessible for students who are new to OOP.

- *Discussion and Illustration of How Polymorphism Works "Under the Hood."* Chapter 13 contains a detailed diagram and explanation of how C++ can implement polymorphism, `virtual` functions and dynamic binding internally. This gives students a solid understanding of how these capabilities really work—a key "light bulb" moment in the book.

- *Standard Template Library (STL).* This might be one of the most important topics in the book in terms of your appreciation of software reuse. The STL defines powerful, template-based, reusable components that implement many common data structures and algorithms used to process those data structures. Chapter 21 introduces the STL and discusses its three key components—containers, iterators and algorithms. The STL provides tremendous expressive power, often reducing many lines of code to a single statement.

- ***ISO/IEC C++ Standard Compliance.*** We've audited our presentation against the ISO/IEC C++ standard document.
- ***Debugger Appendices.*** We provide two Using the Debugger appendices on the book's Companion Website—Appendix H, Using the Visual Studio Debugger, and Appendix I, Using the GNU C++ Debugger.
- ***Code Testing on Multiple Platforms.*** We tested the code examples on various popular C++ platforms including GNU C++ on Linux and Microsoft Visual C++ on Windows. For the most part, the book's examples port to popular standard-compliant compilers.

We believe that this book and its support materials will give you an informative, challenging and entertaining C++ educational experience. If you have questions, send an e-mail to `deitel@deitel.com`; we'll respond promptly. For updates on this book and the status of all supporting C++ software, and for the latest news on all Deitel publications and services, visit `www.deitel.com`.

Optional Case Study: Designing an Object-Oriented ATM

The UML has become the preferred graphical modeling language for designing object-oriented systems. We use industry standard UML activity diagrams (in preference to the older flowcharts) to demonstrate the flow of control in each of C++'s control statements, and we use UML class diagrams to visually represent classes and their inheritance relationships.

The optional Software Engineering Case Study in Chapters 25 and 26 presents a carefully paced introduction to object-oriented design using the UML—we design and fully implement the software for a simple automated teller machine (ATM). The case study has been reviewed by academics and industry professionals, including leaders in the field from Rational (the creators of the UML; Rational is now part of IBM) and the Object Management Group (an industry group now responsible for evolving the UML).

This case study helps prepare you for the kinds of substantial projects you'll encounter in industry. We employ a carefully developed, incremental object-oriented design process to produce a UML model for the ATM system. From this design, we produce a substantial working C++ implementation using key object-oriented programming concepts, including classes, objects, encapsulation, visibility, composition, inheritance and polymorphism.

In Chapter 25, undertake the challenging ATM problem with the techniques of OOD. We analyze a typical requirements document that specifies a system to be built, determine the objects needed to implement that system, determine the attributes these objects need to have, determine the behaviors these objects need to exhibit, and specify how the objects must interact with one another to meet the system requirements. In Chapter 26, we include a complete C++ code implementation of the object-oriented ATM system that we designed in Chapter 25.

The ATM is a nice business example that most people can relate to. In our experience, teaching these two chapters in a second programming course helps students tie together the object-oriented concepts they learn in Chapters 9–13. A key concept in object-oriented programming is the interactions among objects. In most programming textbooks, each code example creates and uses only one or two objects. The ATM case study gives students the opportunity to study interactions of *many* objects that provide the functionality of a substantial system.

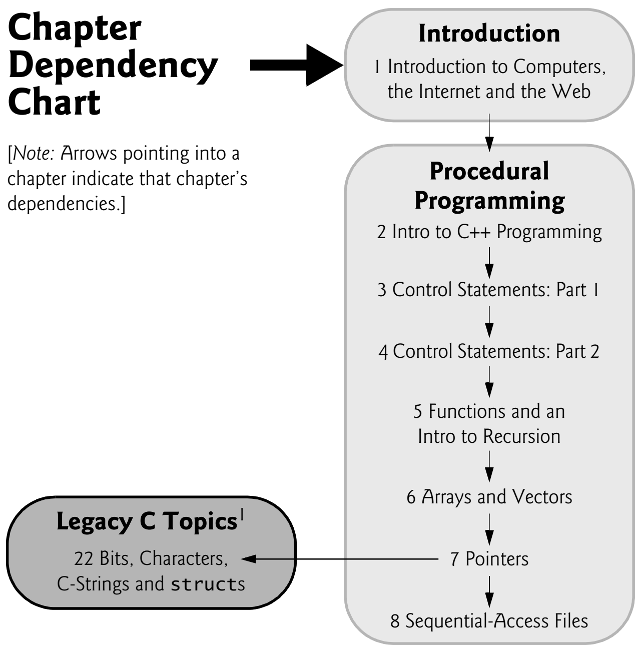
Dependency Charts

The charts in Figs. 1–2 show the dependencies among the chapters to help instructors plan their syllabi. *C++ How to Program, Late Objects Version, 7/e* is appropriate for a variety of programming courses at various levels, most notably CS1 and CS2 courses, and introductory course sequences in related disciplines. The book has a convenient modular organization.

Teaching Approach

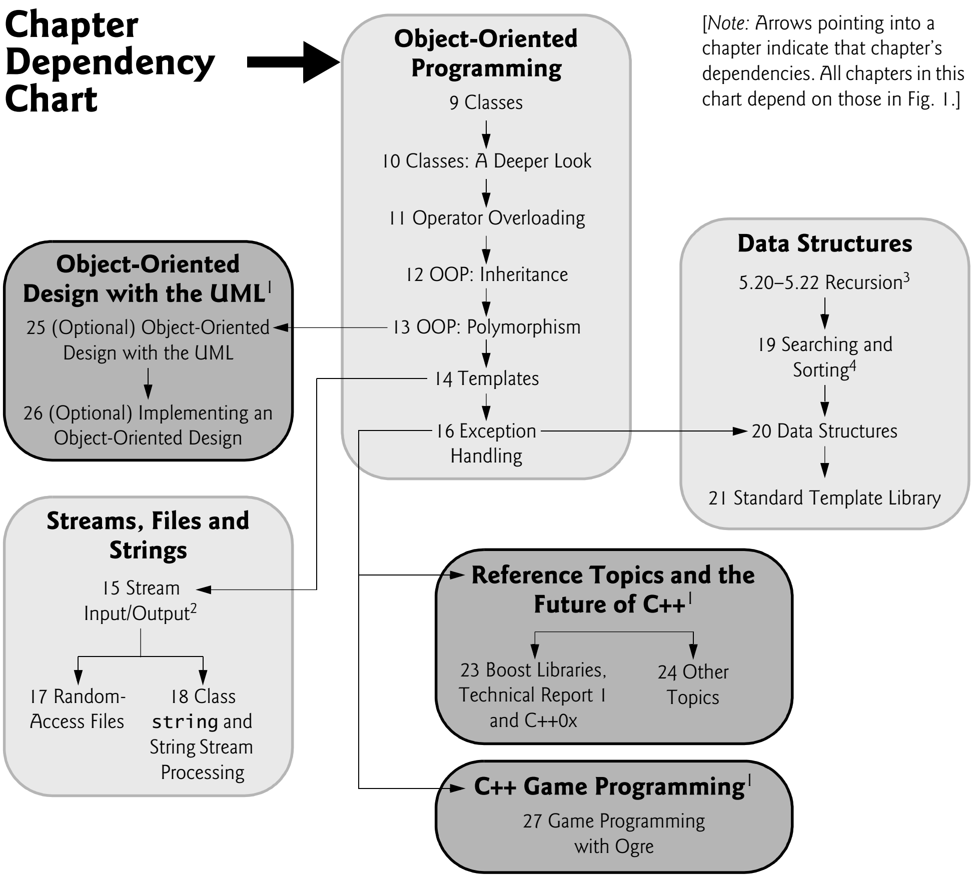
C++ How to Program, Late Objects Version, 7/e, contains a rich collection of examples. The book concentrates on the principles of good software engineering and stresses program clarity. We teach by example.

Live-Code Approach. *C++ How to Program, Late Objects Version, 7/e,* is loaded with "live-code" examples. By this we mean that most new concepts are presented in the context of complete working C++ applications, followed immediately by one or more actual executions showing program inputs and outputs.


Chapter Dependency Chart
[*Note:* Arrows pointing into a chapter indicate that chapter's dependencies.]
Introduction
1 Introduction to Computers, the Internet and the Web
Procedural Programming
2 Intro to C++ Programming
3 Control Statements: Part 1
4 Control Statements: Part 2
5 Functions and an Intro to Recursion
6 Arrays and Vectors
7 Pointers
8 Sequential-Access Files
Legacy C Topics[1]
22 Bits, Characters, C-Strings and structs

1. Online as a PDF document at the book's Companion Website (www.pearsonhighered.com/deitel).

Fig. 1 | Procedural programming in C++.



Chapter Dependency Chart
[Note: Arrows pointing into a chapter indicate that chapter's dependencies. All chapters in this chart depend on those in Fig. 1.]
Object-Oriented Programming
9 Classes
10 Classes: A Deeper Look
11 Operator Overloading
12 OOP: Inheritance
13 OOP: Polymorphism
14 Templates
16 Exception Handling
Object-Oriented Design with the UML1
25 (Optional) Object-Oriented Design with the UML
26 (Optional) Implementing an Object-Oriented Design
Data Structures
5.20–5.22 Recursion3
19 Searching and Sorting4
20 Data Structures
21 Standard Template Library
Streams, Files and Strings
15 Stream Input/Output2
17 Random-Access Files
18 Class string and String Stream Processing
Reference Topics and the Future of C++1
23 Boost Libraries, Technical Report 1 and C++0x
24 Other Topics
C++ Game Programming1
27 Game Programming with Ogre

1. Chapters in the dark gray boxes are available online as PDF documents at the book's Companion Website (`www.pearsonhighered.com/deitel`).
2. Most of Chapter 15 can be read after Chapter 6. A small portion requires Chapters 12 and 14.
3. Requires Chapter 5.
4. Requires Chapter 6.

Fig. 2 | Object-oriented programming and other topics in C++.

Syntax Shading. For readability, we syntax shade all the C++ code, similar to the way most C++ integrated-development environments and code editors syntax color code. Our syntax-shading conventions are as follows:

```
comments appear like this
keywords appear like this
constants and literal values appear like this
all other code appears like this
```

Code Highlighting. We place gray rectangles around key code segments.

Using Fonts for Emphasis. We place the key terms and the index's page reference for each defining occurrence in **bold** text for easier reference. We emphasize on-screen components in the **bold Helvetica** font (e.g., the **File** menu) and emphasize C++ program text in the `Lucida` font (for example, `int x = 5;`).

Web Access. All of the source-code examples are available for download from:

```
www.deitel.com/books/cpphtp7LOV/
```

Quotations. Each chapter begins with quotations. We hope that you enjoy relating these to the chapter material.

Objectives. The quotes are followed by a list of chapter objectives.

Illustrations/Figures. Abundant charts, tables, line drawings, programs and program output are included. We model the flow of control in control statements with UML activity diagrams. UML class diagrams model the fields, constructors and methods of classes. We use six major UML diagram types in the optional OOD/UML 2 ATM case study in Chapters 25 and 26.

Programming Tips. We include programming tips to help you focus on important aspects of program development. These tips and practices represent the best we've gleaned from a combined seven decades of programming and teaching experience.

Good Programming Practices
The Good Programming Practices *call attention to techniques that will help you produce programs that are clearer, more understandable and more maintainable.*

Common Programming Errors
Pointing out these Common Programming Errors *reduces the likelihood that you'll make them.*

Error-Prevention Tips
These tips contain suggestions for exposing and removing bugs from your programs; many describe aspects of C++ that prevent bugs from getting into programs in the first place.

Performance Tips
These tips highlight opportunities for making your programs run faster or minimizing the amount of memory that they occupy.

Portability Tips
The Portability Tips *help you write code that will run on a variety of platforms.*

Software Engineering Observations
The Software Engineering Observations *highlight architectural and design issues that affect the construction of software systems, especially large-scale systems.*

Wrap-Up Section. Each chapter ends with a "wrap-up" section that recaps the chapter content and transitions to the next chapter.

Summary Bullets. We present a section-by-section, bullet-list summary of the chapter.

Terminology. We include an alphabetized list of the important terms defined in each chapter with the page number of the term's defining occurrence for quick reference.

Self-Review Exercises and Answers. Extensive self-review exercises *and* answers are included for self-study.

Exercises. Each chapter concludes with a substantial set of exercises, including simple recall of important terminology and concepts; identifying the errors in code samples; writing individual C++ statements; writing small portions of functions and classes; writing complete C++ functions, classes and programs; and major projects. [*NOTE:* **Please do not write to us requesting access to the Pearson Instructor's Resource Center which contains the book's instructor supplements, including the exercise solutions. Access is limited strictly to college instructors teaching from the book. Instructors may obtain access only through their Pearson representatives.**] Check out our Programming Projects Resource Center (`www.deitel.com/ProgrammingProjects/`) for lots of additional exercise and project possibilities.

Thousands of Index Entries. We have included an extensive index. Defining occurrences of key terms are highlighted with a **bold** page number for quick reference.

Student Resources

Many C++ development tools are available. We wrote *C++ How to Program, Late Objects Version, 7/e* primarily using Microsoft's free Visual C++ Express Edition (which is available free for download at `www.microsoft.com/express`) and the free GNU C++ (`gcc.gnu.org/install/binaries.html`), which is already installed on most Linux systems and can be installed on Mac OS X and Windows systems as well. You can learn more about GNU C++ at `gcc.gnu.org`. Apple includes GNU C++ in their Xcode development tools, which Mac OS X users can download from `developer.apple.com/tools/xcode`.

For additional resources and software downloads see our C++ Resource Center:

```
www.deitel.com/cplusplus/
```

For other C++ compilers that are available free for download, visit:

```
www.thefreecountry.com/developercity/ccompilers.shtml
www.compilers.net/Dir/Compilers/CCpp.htm
```

Companion Website

We include a set of free, web-based student supplements to the book—the Companion Website—available with *new* books purchased from Pearson (see the scratch card at the front of the book for your access code). To access the Companion Website, visit `www.pearsonhighered.com/deitel/` and select the Companion Website link in the section for this book. *If the access code in front of your book is already redeemed, you can purchase access to this material directly from the Companion Website.*

The Companion Website contains the following chapters and appendices in searchable PDF format:

- Chapter 22, Bits, Characters, Strings and `structs`
- Chapter 23, Boost Libraries, Technical Report 1 and C++0x

- Chapter 24, Other Topics
- Chapter 25, ATM Case Study, Part 1: Object-Oriented Design with the UML
- Chapter 26, ATM Case Study, Part 2: Implementing an Object-Oriented Design
- Chapter 27, Game Programming with Ogre
- Appendix D, Number Systems
- Appendix E, Preprocessor
- Appendix F, C Legacy Code Topics
- Appendix G, UML 2: Additional Diagram Types
- Appendix H, Using the Visual Studio Debugger
- Appendix I, Using the GNU C++ Debugger

The Companion Website also includes:

- VideoNotes in which you can watch and listen as Paul Deitel walks you through *all* the code examples in Chapters 2–13 and a portion of Chapter 16.
- Two true/false questions per section with answers for self-review.
- Solutions to approximately half of the solved exercises in the book.

The following additional materials are posted at both the Companion Website and at `www.deitel.com/books/cpphtp7LOV/`:

- An array of pointers-to-functions example and additional function pointer exercises (from Chapter 7).
- `String` Class Operator Overloading Case Study (from Chapter 11).
- Building Your Own Compiler exercise descriptions (from Chapter 20).

CourseSmart Web Books

Today's students and instructors have increasing demands on their time and money. Pearson has responded to that need by offering digital texts and course materials online through CourseSmart. CourseSmart allows faculty to review course materials online saving time and costs. It is also environmentally sound and offers students a high-quality digital version of the text for as much as 50% off the cost of a print copy of the text. Students receive the same content offered in the print textbook enhanced by search, note-taking, and printing tools. For more information, visit `www.coursesmart.com`.

Instructor Supplements

The following supplements are available to qualified instructors only through Pearson Education's Instructor Resource Center (`www.pearsonhighered.com/irc`):

- *Solutions Manual* with solutions to most of the end-of-chapter exercises
- *Test Item File* of multiple-choice questions (approximately two per book section)
- Customizable *PowerPoint® Slides* containing all the code and figures in the text, plus bulleted items that summarize the key points in the text

If you are not already a registered faculty member, contact your Pearson representative or visit `www.pearsonhighered.com/educator/replocator/`.

Deitel® Buzz Online Free E-mail Newsletter

The *Deitel® Buzz Online* e-mail newsletter will keep you posted about issues related to *C++ How to Program, Late Objects Version, 7/e*. It also includes commentary on industry trends and developments, links to free articles and resources from our published books and upcoming publications, product-release schedules, errata, challenges, anecdotes, information on our corporate instructor-led training courses and more. To subscribe, visit

```
www.deitel.com/newsletter/subscribe.html
```

The Deitel Resource Centers

Our website `www.deitel.com` provides more than 100 Resource Centers on various programming languages and software development topics, Web 2.0, Internet business and open-source projects—see the list of Resource Centers in the first few pages of this book and visit `www.deitel.com/ResourceCenters.html`. The Resource Centers evolve out of the research we do to support our publications and business endeavors. We've found many exceptional resources online, including tutorials, documentation, software downloads, articles, blogs, podcasts, videos, code samples, books, e-books and more—most of them are free. Each week we announce our latest Resource Centers on Facebook® and Twitter™, and in our newsletter, the *Deitel® Buzz Online*. Some of the Resource Centers you might find helpful while studying this book are C++, C++ Boost Libraries, C++ Game Programming and Regular Expressions, Code Search Engines and Code Sites, Game Programming and Programming Projects.

Follow Deitel on Twitter and Facebook

To receive updates on Deitel publications, Resource Centers, training courses, partner offers and more, follow us on Twitter

```
@deitel
```

and join the Deitel & Associates group on Facebook

```
www.facebook.com/deitelfan
```

Acknowledgments

It's a pleasure to acknowledge the efforts of people whose names do not appear on the cover, but whose hard work, cooperation, friendship and understanding were crucial to the book's production. Many people at Deitel & Associates, Inc., devoted long hours to this project—thanks especially to Abbey Deitel and Barbara Deitel.

We would also like to thank the participants of our Honors Internship program who contributed to this publication—Matthew Pearson, a Computer Science graduate of Cornell University; and Christine Chen, an Operations Research and Information Engineering major at Cornell University.

We are fortunate to have worked on this project with the dedicated publishing professionals at Pearson. We appreciate the extraordinary efforts of Marcia Horton, Edi-

torial Director of Pearson's Engineering and Computer Science Division, and Michael Hirsch, Editor-in-Chief of Computer Science. Carole Snyder did an outstanding job recruiting the book's review team and managing the review process. Kristine Carney of Pearson did a wonderful job designing the book's cover—we provided the concept, and she made it happen. Robert Engelhardt did a marvelous job managing the book's production. Margaret Waples did a great job marketing the book through academic and professional channels.

Reviewers
We wish to acknowledge the efforts of our reviewers, whose comments had a great impact on *C++ How to Program, Late Objects Version, 7/e*. They scrutinized the text and the programs, and provided suggestions for improving the accuracy and completeness of the presentation:

***C++ How to Program, 7/e* Academic Reviewers:**

- Thomas J. Borrelli (Rochester Institute of Technology)
- Peter J. DePasquale (The College of New Jersey)
- Jack Hagemeister (Washington State University)
- Williams M. Higdon (University of Indiana)
- Dean Mathias (Utah State University)
- Robert A. McLain (Tidewater Community College)
- Dave Topham (Ohlone College)

Industry Reviewers:

- Chris Cox (Adobe Systems)
- Gregory Dai (Kernel Development)
- Doug Gregor (Apple, Inc.)
- April Reagan (Microsoft)
- José Antonio González Seco (Parliament of Andalusia, Spain)

Well, there you have it! Welcome to the exciting world of C++ and object-oriented programming. We hope you enjoy this look at contemporary computer programming. As you read the book, we would sincerely appreciate your comments, criticisms, corrections and suggestions for improving the text. Please address all correspondence to:

```
    deitel@deitel.com
```

We'll respond promptly, and post corrections and clarifications on:

```
    www.deitel.com/books/cpphtp7LOV/
```

We hope you enjoy reading *C++ How to Program, Late Objects Version, 7/e* as much as we enjoyed writing it!

Paul Deitel
Harvey Deitel
Sudbury, Massachusetts
July 2010

About the Authors

Paul J. Deitel, CEO and Chief Technical Officer of Deitel & Associates, Inc., is a graduate of MIT's Sloan School of Management, where he studied Information Technology. Through Deitel & Associates, Inc., he has delivered C++, C, Java, C#, Visual Basic and Internet programming courses to industry clients, including Cisco, IBM, Sun Microsystems, Dell, Lucent Technologies, Fidelity, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, White Sands Missile Range, Rogue Wave Software, Boeing, SunGard Higher Education, Stratus, Cambridge Technology Partners, Open Environment Corporation, One Wave, Hyperion Software, Adra Systems, Entergy, CableData Systems, Nortel Networks, Puma, iRobot, Invensys and many more. He has also lectured on C++ and Java for the Boston Chapter of the Association for Computing Machinery. He has been designated by Sun Microsystems as a Java Champion. He and his co-author, Dr. Harvey M. Deitel, are the world's best-selling programming-language textbook authors.

Dr. Harvey M. Deitel, Chairman and Chief Strategy Officer of Deitel & Associates, Inc., has 49 years of academic and industry experience in the computer field. Dr. Deitel earned B.S. and M.S. degrees from MIT and a Ph.D. from Boston University. He has 20 years of college teaching experience, including earning tenure and serving as the Chairman of the Computer Science Department at Boston College before founding Deitel & Associates, Inc., with his son, Paul J. Deitel. He and Paul are the co-authors of dozens of books and multimedia packages and they are writing many more. With translations published in Traditional Chinese, Simplified Chinese, Japanese, German, Russian, Spanish, Korean, French, Polish, Italian, Portuguese, Greek, Urdu and Turkish, the Deitels' texts have earned international recognition. Dr. Deitel has delivered hundreds of professional seminars to major corporations, academic institutions, government organizations and the military.

About Deitel & Associates, Inc.

Deitel & Associates, Inc., is an internationally recognized corporate training and authoring organization specializing in computer programming languages, Internet and web software technology, object-technology education and Android™ and iPhone® app development. The company provides instructor-led courses delivered at client sites worldwide on major programming languages and platforms, such as C++, Visual C++®, C, Java™, Visual C#®, Visual Basic®, XML®, Python®, object technology, Internet and web programming, Android and iPhone app development, and a growing list of additional programming and software-development-related courses. The founders of Deitel & Associates, Inc., are Paul J. Deitel and Dr. Harvey M. Deitel. The company's clients include many of the world's largest companies, government agencies, branches of the military, and academic institutions. Through its 34-year publishing partnership with Prentice Hall/Pearson, Deitel & Associates, Inc., publishes leading-edge programming textbooks, professional books, interactive multimedia *Cyber Classrooms*, and *LiveLessons* DVD-based and web-based video courses. Deitel & Associates, Inc., and the authors can be reached via e-mail at:

```
    deitel@deitel.com
```

To learn more about Deitel & Associates, Inc., its publications and its *Dive Into® Series* Corporate Training curriculum delivered at client locations worldwide, visit:

```
    www.deitel.com/training/
```

and subscribe to the free *Deitel® Buzz Online* e-mail newsletter at:

```
www.deitel.com/newsletter/subscribe.html
```

Individuals wishing to purchase Deitel books, and *LiveLessons* DVD and web-based training courses can do so through `www.deitel.com`. Bulk orders by corporations, the government, the military and academic institutions should be placed directly with Pearson. For more information, visit `www.prenhall.com/mischtm/support.html#order`.