Welcome to the C programming language and to *C How to Program, Eighth Edition!* This book presents leading-edge computing technologies for college students, instructors and software-development professionals.

At the heart of the book is the Deitel signature “live-code approach”—we present concepts in the context of complete working programs, rather than in code snippets. Each code example is followed by one or more sample executions. Read the online Before You Begin section at

http://www.deitel.com/books/cht8/cht8_BYB.pdf

to learn how to set up your computer to run the hundreds of code examples. All the source code is available at

http://www.deitel.com/books/cht8

Use the source code we provide to run every program as you study it.

We believe that this book and its support materials will give you an informative, challenging and entertaining introduction to C. As you read the book, if you have questions, send an e-mail to deitel@deitel.com—we’ll respond promptly. For book updates, visit [www.deitel.com/books/cht8](http://www.deitel.com/books/cht8), join our social media communities:

- Facebook®—http://facebook.com/DeitelFan
- Twitter®—@deitel
- LinkedIn®—http://linkedin.com/company/deitel-&-associates
- YouTube™—http://youtube.com/DeitelTV
- Google+™—http://google.com/+DeitelFan

and register for the *Deitel® Buzz Online* e-mail newsletter at:

http://www.deitel.com/newsletter/subscribe.html

**New and Updated Features**

Here are some key features of *C How to Program, 8/e*:

- **Integrated More Capabilities of the C11 and C99 standards.** Support for the C11 and C99 standards varies by compiler. Microsoft Visual C++ supports a subset of the features that were added to C in C99 and C11—primarily the features that are also required by the C++ standard. We incorporated several widely supported C11 and C99 features into the book’s early chapters, as appropriate for introduc-
Preface

This book is for introductory courses and for the compilers we used in this book. Appendix E, Multi-threading and Other C11 and C99 Topics, presents more advanced features (such as multithreading for today’s increasingly popular multi-core architectures) and various other features that are not widely supported by today’s C compilers.

- **All Code Tested on Linux, Windows and OS X.** We retested all the example and exercise code using GNU gcc on Linux, Visual C++ on Windows (in Visual Studio 2013 Community Edition) and LLVM in Xcode on OS X.

- **Updated Chapter 1.** The new Chapter 1 engages students with updated intriguing facts and figures to get them excited about studying computers and computer programming. The chapter includes current technology trends and hardware discussions, the data hierarchy, social networking and a table of business and technology publications and websites that will help you stay up to date with the latest technology news and trends. We’ve included updated test-drives that show how to run a command-line C program on Linux, Microsoft Windows and OS X. We also updated the discussions of the Internet and web, and the introduction to object technology.

- **Updated Coverage of C++ and Object-Oriented Programming.** We updated Chapters 15–23 on object-oriented programming in C++ with material from our textbook *C++ How to Program, 9/e*, which is up-to-date with the C++11 standard.

- **Updated Code Style.** We removed the spacing inside parentheses and square brackets, and toned down our use of comments a bit. We also added parentheses to certain compound conditions for clarity.

- **Variable Declarations.** Because of improved compiler support, we were able to move variable declarations closer to where they’re first used and define for-loop counter-control variables in each for’s initialization section.

- **Summary Bullets.** We removed the end-of-chapter terminology lists and updated the detailed section-by-section, bullet-list summaries with bolded key terms and, for most, page references to their defining occurrences.

- **Use of Standard Terminology.** To help students prepare to work in industry worldwide, we audited the book against the C standard and upgraded our terminology to use C standard terms in preference to general programming terms.

- **Online Debugger Appendices.** We’ve updated the online GNU gdb and Visual C++® debugging appendices, and added an Xcode® debugging appendix.

- **Additional Exercises.** We updated various exercises and added some new ones, including one for the Fisher-Yates unbiased shuffling algorithm in Chapter 10.

**Other Features**

Other features of *C How to Program, 8/e* include:

- **Secure C Programming Sections.** Many of the C chapters end with a Secure C Programming Section. We’ve also posted a Secure C Programming Resource Center at [www.deitel.com/SecureC/](http://www.deitel.com/SecureC/). For more details, see the section “A Note About Secure C Programming” on the next page.

* C How to Program, 8/e. © Copyright 1992–2015 by Pearson Education, Inc. All Rights Reserved.
• **Focus on Performance Issues.** C (and C++) are favored by designers of performance-intensive systems such as operating systems, real-time systems, embedded systems and communications systems, so we focus intensively on performance issues.

• **“Making a Difference” Contemporary Exercises.** We encourage you to use computers and the Internet to research and solve significant problems. These exercises are meant to increase awareness of important issues the world is facing. We hope you’ll approach them with your own values, politics and beliefs.

• **Sorting: A Deeper Look.** Sorting places data in order, based on one or more sort keys. We begin our sorting presentation in Chapter 6 with a simple algorithm—in Appendix D, we present a deeper look. We consider several algorithms and compare them with regard to their memory consumption and processor demands. For this purpose, we present a friendly introduction to Big O notation, which indicates how hard an algorithm may have to work to solve a problem. Through examples and exercises, Appendix D discusses the selection sort, insertion sort, recursive merge sort, recursive selection sort, bucket sort and recursive Quicksort. Sorting is an intriguing problem because different sorting techniques achieve the same final result but they can vary hugely in their consumption of memory, CPU time and other system resources.

• **Titled Programming Exercises.** Most of the programming exercises are titled to help instructors conveniently choose assignments appropriate for their students.

• **Order of Evaluation.** We caution the reader about subtle order of evaluation issues.

• **C++-Style // Comments.** We use the newer, more concise C++-style // comments in preference to C’s older style /*...*/ comments.

### A Note About Secure C Programming

Throughout this book, we focus on C programming *fundamentals*. When we write each *How to Program* book, we search the corresponding language’s standards document for the features that we feel novices need to learn in a first programming course, and features that professional programmers need to know to begin working in that language. We also cover computer-science and software-engineering fundamentals for novices—our core audience.

*Industrial-strength* coding techniques in any programming language are beyond the scope of an introductory textbook. For that reason, our Secure C Programming sections present some key issues and techniques, and provide links and references so you can continue learning.

Experience has shown that it’s difficult to build industrial-strength systems that stand up to attacks from viruses, worms, etc. Today, via the Internet, such attacks can be instantaneous and global in scope. Software vulnerabilities often come from simple programming issues. Building security into software from the start of the development cycle can greatly reduce costs and vulnerabilities.

The CERT® Coordination Center ([www.cert.org](http://www.cert.org)) was created to analyze and respond promptly to attacks. CERT—the Computer Emergency Response Team—publishes and promotes secure coding standards to help C programmers and others implement industrial-strength systems that avoid the programming practices that leave systems vulnerable to attacks. The CERT standards evolve as new security issues arise.
We’ve upgraded our code (as appropriate for an introductory book) to conform to various CERT recommendations. If you’ll be building C systems in industry, consider reading *The CERT C Secure Coding Standard, 2/e* (Robert Seacord, Addison-Wesley Professional, 2014) and *Secure Coding in C and C++, 2/e* (Robert Seacord, Addison-Wesley Professional, 2013). The CERT guidelines are available free online at

[https://www.securecoding.cert.org/confluence/display/seccode/CERT+C+Coding+Standard](https://www.securecoding.cert.org/confluence/display/seccode/CERT+C+Coding+Standard)

Mr. Seacord, a technical reviewer for the C portion of the last edition of this book, provided specific recommendations on each of our Secure C Programming sections. Mr. Seacord is the Secure Coding Manager at CERT at Carnegie Mellon University’s Software Engineering Institute (SEI) and an adjunct professor in the Carnegie Mellon University School of Computer Science.

The Secure C Programming sections at the ends of Chapters 2–13 discuss many important topics, including:

- testing for arithmetic overflows
- using unsigned integer types
- the more secure functions in the C standard’s Annex K
- the importance of checking the status information returned by standard-library functions
- range checking
- secure random-number generation
- array bounds checking
- preventing buffer overflows
- input validation
- avoiding undefined behaviors
- choosing functions that return status information vs. using similar functions that do not
- ensuring that pointers are always NULL or contain valid addresses
- using C functions vs. using preprocessor macros, and more.

**Web-Based Materials**

The book’s open access Companion Website ([http://www.pearsonhighered.com/deitel](http://www.pearsonhighered.com/deitel)) contains source code for all the code examples and the following appendices in PDF format:

- Appendix F, Using the Visual Studio Debugger
- Appendix G, Using the GNU gdb Debugger
- Appendix H, Using the Xcode Debugger

**Dependency Charts**

Figures 1 and 2 on the next two pages show the dependencies among the chapters to help instructors plan their syllabi. *C How to Program, 8/e* is appropriate for CS1 and many CS2 courses, and for intermediate-level C and C++ programming courses. The C++ part of the book assumes that you’ve studied C Chapters 1–10.

**Teaching Approach**

*C How to Program, 8/e*, contains a rich collection of examples. We focus on good software engineering, program clarity, preventing common errors, program portability and performance issues.
**Syntax Shading.** For readability, we syntax shade the code, similar to the way most IDEs and code editors syntax color code. Our syntax-shading conventions are:

- comments appear like this in gray
- *keywords appear like this in dark blue*
- constants and literal values appear like this in light blue
- all other code appears in black

---

Preface

**Object-Based Programming**
- 15 C++ as a Better C: Intro to Object Technology
- 16 Intro to Classes and Objects
- 17 Classes: A Deeper Look; Throwing Exceptions
- 18 Operator Overloading; Class string

**Object-Oriented Programming**
- 19 OOP: Inheritance
- 20 OOP: Polymorphism
- 21 Stream Input/Output
- 22 Exception Handling: A Deeper Look
- 23 Intro to Custom Templates

---

**Fig. 2** C++ chapter dependency chart.

**Code Highlighting.** We place gray rectangles around the key code in each program.

**Using Fonts for Emphasis.** We place the key terms and the index’s page reference for each defining occurrence in **bold colored** text for easy reference. We emphasize C program text in the Lucida font (for example, `int x = 5;`).

**Objectives.** Each chapter begins with a list of objectives.

**Illustrations/Figures.** Abundant flowcharts, tables, line drawings, UML diagrams (in the C++ chapters), programs and program outputs are included.

**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 eight 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 and for avoiding bugs in the first place.
Summary Bullets. We present a detailed section-by-section, bullet-list summary of each chapter with bolded key terms. For easy reference, most of the key terms are followed by the page number of their defining occurrences.

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 program statements
- writing small portions of C functions (and C++ member functions and classes)
- writing complete programs
- implementing major projects

Index. We’ve included an extensive index, which is especially helpful when you use the book as a reference. Defining occurrences of key terms are highlighted in the index with a bold colored page number.

Software Used in C How to Program, 8/e

We tested the programs in C How to Program, 8/e using the following free compilers:

- GNU C and C++ (http://gcc.gnu.org/install/binaries.html), which are already installed on most Linux systems and can be installed on OS X and Windows systems.
- LLVM in Apple’s Xcode IDE, which OS X users can download from the Mac App Store.

For other free C and C++ compilers, visit:

http://www.thefreecountry.com/compilers/cpp.shtml
http://www.compilers.net/Dir/Compilers/CCpp.htm
http://www.freebyte.com/programming/cpp/#cppcompilers

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.
Preface

CourseSmart Web Books
Today’s students and instructors have increasing demands on their time and money. Pearson has responded to that need by offering various digital texts and course materials online through CourseSmart. Faculty can review course materials online, saving time and costs. It offers students a high-quality digital version of the text for less than the cost of a print copy. Students receive the same content offered in the print textbook enhanced by search, note-taking and printing tools. For more information, visit http://www.coursesmart.com.

Instructor Resources
The following supplements are available to qualified instructors only through Pearson Education’s password-protected Instructor Resource Center (www.pearsonhighered.com/irc):

- PowerPoint® slides containing all the code and figures in the text, plus bulleted items that summarize key points.
- Test Item File of multiple-choice questions (approximately two per top-level book section)
- Solutions Manual with solutions to most (but not all) of the end-of-chapter exercises. Please check the Instructor Resource Center to determine which exercises have solutions.

Please do not write to us requesting access to the Instructor Resource Center. Access is restricted to college instructors teaching from the book. Instructors may obtain access only through their Pearson representatives. If you’re not a registered faculty member, contact your Pearson representative or visit http://www.pearsonhighered.com/replocator/.

Solutions are not provided for “project” exercises. Check out our Programming Projects Resource Center for lots of additional exercise and project possibilities (http://www.deitel.com/ProgrammingProjects/).

Acknowledgments
We’d like to thank Abbey Deitel and Barbara Deitel for long hours devoted to this project. Abbey co-authored Chapter 1. We’re fortunate to have worked with the dedicated team of publishing professionals at Pearson. We appreciate the guidance, savvy and energy of Tracy Johnson, Executive Editor, Computer Science. Kelsey Loanes and Bob Engelhardt did a marvelous job managing the review and production processes, respectively.

C How to Program, 8/e Reviewers
We wish to acknowledge the efforts of our reviewers. Under tight deadlines, they scrutinized the text and the programs and provided countless suggestions for improving the presentation: Dr. Brandon Invergo (GNU/European Bioinformatics Institute), Danny Kalev (A Certified System Analyst, C Expert and Former Member of the C++ Standards Committee), Jim Hogg (Program Manager, C/C++ Compiler Team, Microsoft Corporation), José Antonio González Seco (Parliament of Andalusia), Sebnem Onsay (Special Instructor, Oakland University School of Engineering and Computer Science), Alan Bunning (Purdue University), Paul Clingan (Ohio State University), Michael Geiger (University of Massachusetts, Lowell), Jeonghwa Lee (Shippensburg University), Susan Mengel (Texas Tech University), Judith O’Rourke (SUNY at Albany) and Chen-Chi Shin (Radford University).
Other Recent Editions Reviewers
William Albrecht (University of South Florida), Ian Barland (Radford University), Ed James Beckham (Altera), John Benito (Blue Pilot Consulting, Inc. and Convener of ISO WG14—the Working Group responsible for the C Programming Language Standard), Dr. John F. Doyle (Indiana University Southeast), Alireza Fazelpour (Palm Beach Community College), Mahesh Hariharan (Microsoft), Hemanth H.M. (Software Engineer at SonicWALL), Kevin Mark Jones (Hewlett Packard), Lawrence Jones, (UGS Corp.), Don Kostuch (Independent Consultant), Vytautus Leonavicius (Microsoft), Xiaolong Li (Indiana State University), William Mike Miller (Edison Design Group, Inc.), Tom Rethard (The University of Texas at Arlington), Robert Seacord (Secure Coding Manager at SEI/CERT, author of The CERT C Secure Coding Standard and technical expert for the international standardization working group for the programming language C), José Antonio González Seco (Parliament of Andalusia), Benjamin Seyfarth (University of Southern Mississippi), Gary Sibbitts (St. Louis Community College at Meramec), William Smith (Tulsa Community College) and Douglas Walls (Senior Staff Engineer, C compiler, Sun Microsystems—now part of Oracle).

A Special Thank You to Brandon Invergo and Jim Hogg
We were privileged to have Brandon Invergo (GNU/European Bioinformatics Institute) and Jim Hogg (Program Manager, C/C++ Compiler Team, Microsoft Corporation) do full-book reviews. They scrutinized the C portion of the book, providing numerous insights and constructive comments. The largest part of our audience uses either the GNU gcc compiler or Microsoft’s Visual C++ compiler (which also compiles C). Brandon and Jim helped us ensure that our content was accurate for the GNU and Microsoft compilers, respectively. Their comments conveyed a love of software engineering, computer science and education that we share.

Well, there you have it! C is a powerful programming language that will help you write high-performance, portable programs quickly and effectively. It scales nicely into the realm of enterprise systems development to help organizations build their business-critical and mission-critical information systems. As you read the book, we would sincerely appreciate your comments, criticisms, corrections and suggestions for improving the text. Please address all correspondence—including questions—to:

deitel@deitel.com

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

www.deitel.com/books/chtp8/

We hope you enjoy working with C How to Program, Eighth Edition as much as we enjoyed writing it!

Paul Deitel
Harvey Deitel

About the Authors
Paul Deitel, CEO and Chief Technical Officer of Deitel & Associates, Inc., is a graduate of MIT, where he studied Information Technology. Through Deitel & Associates, Inc.,
he has delivered hundreds of programming courses to industry clients, including Cisco, IBM, Siemens, Sun Microsystems, Dell, Lucent Technologies, Fidelity, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, White Sands Missile Range, Hospital Sisters Health System, Rogue Wave Software, Boeing, SunGard Higher Education, Stratus, Cambridge Technology Partners, One Wave, Hyperion Software, Adra Systems, Entergy, CableData Systems, Nortel Networks, Puma, iRobot, Invensys and many more. He and his co-author, Dr. Harvey M. Deitel, are the world’s best-selling programming-language textbook/professional book/video authors.

Dr. Harvey M. Deitel, Chairman and Chief Strategy Officer of Deitel & Associates, Inc., has 54 years of experience in the computer field. Dr. Deitel earned B.S. and M.S. degrees in electrical engineering from MIT and a Ph.D. in mathematics from Boston University (all with a focus on computing). He has extensive college teaching experience, including earning tenure and serving as the Chairman of the Computer Science Department at Boston College before founding Deitel & Associates in 1991 with his son, Paul Deitel. The Deitels’ publications have earned international recognition, with translations published in Chinese, Korean, Japanese, German, Russian, Spanish, French, Polish, Italian, Portuguese, Greek, Urdu and Turkish. Dr. Deitel has delivered hundreds of programming courses to academic institutions, major corporations, government organizations and the military.

About Deitel & Associates, Inc.
Deitel & Associates, Inc., founded by Paul Deitel and Harvey Deitel, is an internationally recognized authoring and corporate training organization, specializing in computer programming languages, object technology, mobile app development and Internet and web software technology. The company’s training clients include many of the world’s largest companies, government agencies, branches of the military, and academic institutions. The company offers instructor-led training courses delivered at client sites worldwide on major programming languages and platforms, including C, C++, Java™, Android app development, Swift™ and iOS® app development, Visual C#®, Visual Basic®, Visual C++, Python®, object technology, Internet and web programming and a growing list of additional programming and software development courses.

Through its 40-year publishing partnership with Pearson/Prentice Hall, Deitel & Associates, Inc., publishes leading-edge programming textbooks and professional books in print and popular e-book formats, and LiveLessons video courses (available on Safari Books Online and other video platforms). Deitel & Associates, Inc. and the authors can be reached at:

deitel@deitel.com

To learn more about Deitel’s Dive-Into® Series Corporate Training curriculum delivered to groups of software engineers at client sites worldwide, visit:

http://www.deitel.com/training

To request a proposal for on-site, instructor-led training at your organization, e-mail deitel@deitel.com.

Individuals wishing to purchase Deitel books and LiveLessons video training 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
