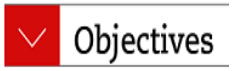
Note to Students

Welcome to Revel *Introduction to Programming with C++,* by Y. Daniel Liang.

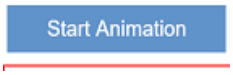
Revel was built to work on your laptop and tablet, so you can study whenever and wherever you like. Interactive animations, LiveExamples and VideoNotes are designed to help you immediately apply and practice what you've read. An assignment calendar shows exactly when every assignment is due, helping you stay on track throughout the course. Highlighting, note taking, and a glossary personalize the learning experience. Instructors can add notes as well, including reminders or study tips.

Pedagogical Features

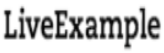


The **Objectives** at the beginning of each chapter lists what students should learn from the chapter. This concise list will help them determine whether they have met the objectives after completing the chapter.

Key Points highlight the important concepts covered in each section.

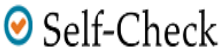


Animated Listings step students through the code line-by-line, showing and explaining what is happening in the program.



LiveExamples enable students to practice what they've learned by modifying and running live code and receiving immediate feedback on their submission.

VideoNotes simulate the "office hours experience" through narrated video tutorials that show how to solve problems completely, from design through coding.



CheckPoints, Order Statements, Word Match, Freestyle Exercises, and Multiple-Choice Questions enable students to evaluate their learning.

Worth 9 pts

Embedded assessments, which are assignable, gradable programming exercises, are presented as quizzes at the end of sections and chapters. These assessments give students the opportunity to practice and demonstrate their programming skills, and offer instructors the ability to assess student understanding of key concepts. Individual student and class performance data report to the Performance tracker in Revel.

C++ Development Tools

You can use a text editor, such as Windows Notepad or WordPad, to create C++ programs, then you can compile and run the programs from the command window. You can also use a C++ development tool such as Visual C++ or Dev-C++. These tools support an integrated development environment (IDE) for rapidly developing C++ programs. Editing, compiling, building, executing, and debugging programs are integrated in one graphical user interface. Using these tools effectively will greatly increase your programming productivity. How to create, compile, and run programs is introduced in Chapter 1, and how to use Visual C++ and Dev-C++ is introduced in the Supplemental Material section at the bottom of the Table of Contents in Revel. The programs in this book have been tested on Visual C++ 2015 and 2017 and on the GNU C++ compiler 5.1.

Student Supplemental Resources

Pearson is pleased to offer the following resources to students using Revel for Liang C++. These valuable resources are available in the Supplemental Material section at the bottom of the Table of Contents in Revel.

Solutions

- Solutions to Even-Numbered Programming Exercises From the Book (presented as downloadable zip files)
- Solutions to UML Diagrams (Chapters 9-15)

Software Downloads and Supplements

- Part I General Supplements
- Part II IDE Supplements
- Part III C++ Preprocessors Directives
- Part IV Advanced C++ Topics
- Part V Legacy Topics
- Part VI Case Studies
- Part VII C++ Resources

Errata

Note to Instructors

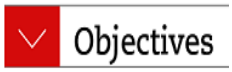
This course takes a “fundamentals first” approach by introducing basic programming concepts and techniques before designing custom classes. The fundamental concepts and techniques of selection statements, loops, methods, and arrays are the foundation of programming. Building this strong foundation prepares students to learn object-oriented programming and advanced C++ programming.

We know students learn more deeply when they engage with content: when they are able to read and immediately apply and practice what they’ve learned. This “read-a-little, do-a-little” approach is the inspiration behind Revel for Introduction to Programming with C++.

Programming is presented in a problem-driven manner, focusing on problem solving rather than syntax and using examples from many different areas including math, science, business, finance, gaming, animation, and multimedia. Basic programming concepts are explained by examples. and support a variety of interactive learning activities to keep students engaged. Well-paced assignable, gradable programming assignments show instructors when and where individual students, and the class as a whole, might be struggling.

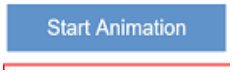
Our goal is to teach problem solving and programming using proven pedagogy and content, engaging interactives, and self-paced and assignable assessments, designed for the way today’s students read, think, and learn.

Pedagogical Features

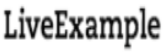


The **Objectives** at the beginning of each chapter lists what students should learn from the chapter. This concise list will help them determine whether they have met the objectives after completing the chapter.

Key Points highlight the important concepts covered in each section.



Animated Listings step students through the code line-by-line, showing and explaining what is happening in the program.



LiveExamples enable students to practice what they've learned by modifying and running live code and receiving immediate feedback on their submission.

VideoNotes simulate the "office hours experience" through narrated video tutorials that show how to solve problems completely, from design through coding.

CheckPoints, Order Statements, Word Match, Freestyle Exercises, and Multiple-Choice Questions enable students to evaluate their learning.

Embedded assessments, which are assignable, gradable programming exercises, are presented as quizzes at the end of sections and chapters. These assessments give students the opportunity to practice and demonstrate their programming skills, and offer instructors the ability to assess student understanding of key concepts. Individual student and class performance data report to the Performance tracker in Revel.

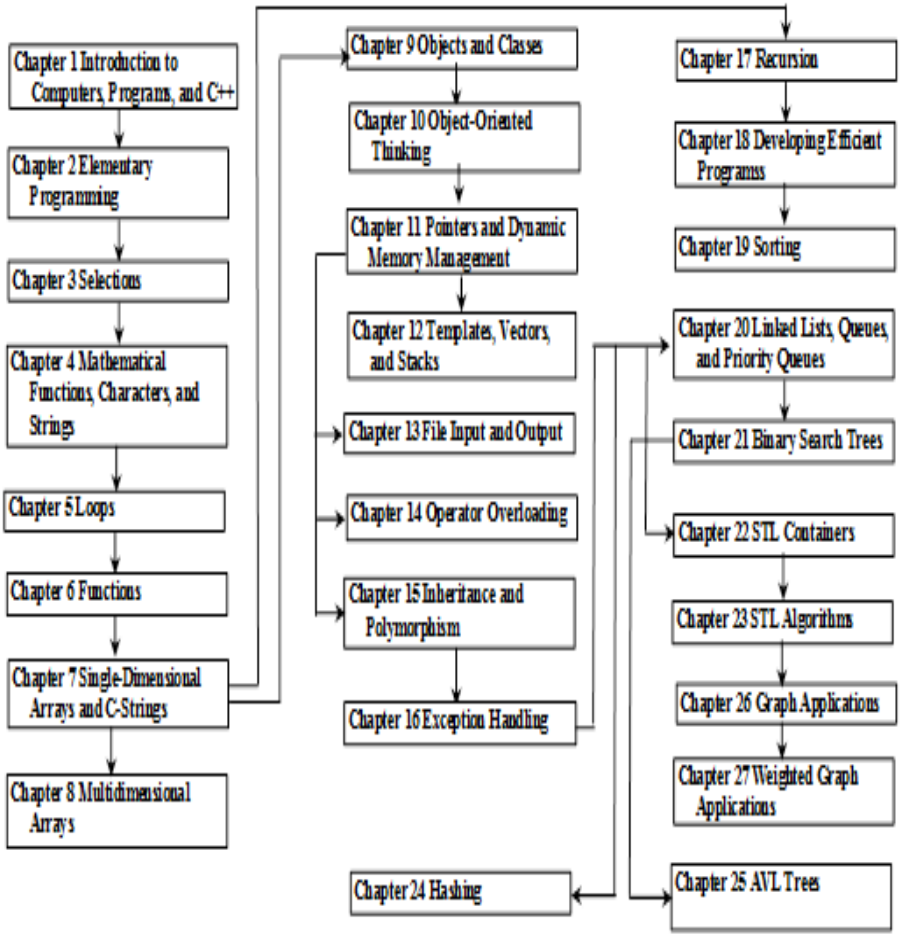
New Features

Since C++11 compilers are widely available now, this edition incorporates the following C++11 features to develop better programs throughout the book.

- The **long long int** and **unsigned long long** types for declaring a signed and unsigned long integer of 64 bits are covered Chapter 2.
- Chapter 7 introduces traversing array elements using a **foreach** loop. The **foreach** can also be used to traverse elements in a container such as vector, list, and set. Chapter 7 also introduces converting numbers to strings using the **to_string** function.
- Chapter 10 introduces member initializers for initializing data fields in an object.
- The **string** class's **back()** and **front()** functions are introduced in Chapter 10.
- You can use the **nullptr** keyword to represent a null pointer in Chapter 11, which is better than using **NULL** or **0**.
- Chapter 12 introduces vector initializers to simplify coding for initializing vectors. The smart pointer class **unique_ptr** is also introduced in this chapter for performing automatic object destruction.
- Chapter 13 shows that you can pass a file name as a string or C-String, whichever is convenient.
- Chapter 15 introduces to use the **override** keyword to prevent errors for overriding functions, and the **final** keyword to prevent a virtual function from being further overridden.
- Chapter 22 introduces the **auto** keyword for type inference. This is useful to simplify coding for variables of complex data types that can be automatically discovered by the compiler in the context. The **auto** keyword should only be used for complex types, not simple types for improving the readability of the code. We introduce this new feature later in the book to avoid misusing or abusing it.
- Chapter 23 introduces lambda functions and new STL algorithms.
- Appendix H discusses regular expressions.

Flexible Chapter Orderings

The book provides flexible chapter orderings, as shown in the following diagram:



Instructor Supplements

Make more time for your students with instructor resources that offer effective learning assessments and classroom engagement. Pearson's partnership with educators does not end with the delivery of course materials; Pearson is there with you on the first day of class and beyond. A dedicated team of local Pearson representatives will work with you not only to choose course materials, but also to integrate them into your class and assess their effectiveness. Our goal is your goal: to improve instruction with each semester.

Pearson is pleased to offer the following resources to qualified adopters of Revel Introduction to Programming with C++. Several of these supplements are available to download on the

Instructor Resource Center (IRC); please visit the IRC at www.pearsonhighered.com/irc to register for access.

- **ACM/IEEE Curricular 2013.** The new ACM/IEEE Computer Science Curricular 2013 defines the Body of Knowledge organized into 18 Knowledge Areas. To help instructors design the courses based on Introduction to Programming with C++, we provide sample syllabi to identify the Knowledge Areas and Knowledge Units. The sample syllabi are for a three-semester course sequence, and serve as an example for institutional customization.
- **ABET Course Assessment.** Many of our users are from the ABET-accredited programs. A key component of the ABET accreditation is to identify weaknesses through continuous course assessment against the course outcomes. To help instructors, we provide sample course outcomes and sample exams for measuring course outcomes.
- **Microsoft PowerPoint Lecture Presentations** with interactive buttons to view full-color, syntax-highlighted source code with code animation, and to run programs without leaving the slides.
- **Solutions to all programming exercises.** Students have access to the solutions of majority of even-numbered programming exercises, and instructors have access to the solutions of majority of even- and odd-numbered programming exercises. Additionally, instructors can access extra programming exercises and their solutions that are not printed in the book.
- **Sample exams.** Most exams have four parts:
  - Multiple-choice questions or short answers
  - Correct programming errors
  - Trace programs
  - Write programs
- Project. In general, each project gives a description and asks students to analyze, design, and implement the project.