Instructor's Summary for *Murach's C++ Programming*

The instructor's materials for *Murach's C++ Programming* will help any college instructor or corporate trainer run an effective course based on the book. This summary introduces you to these materials and helps you get started using them.

At the least, we recommend that you read the topics under "What's included in the instructor's materials" because they not only describe the components but also our underlying instructional philosophy. We also recommend that you read "How to get started with our materials" because that provides the installation procedures that you'll need and gives you charts that summarize the components at a glance.

But first, some thoughts about the modular structure of this book that you should be aware of. This structure is important because it gives you instructional options that you just don't have with other books.

Mike Murach & Associates, Inc.
1-800-221-5528 • (559) 440-9071 • Fax: (559) 440-0963
murachbooks@murach.com • www.MurachForInstructors.com

About the structure of the book

To present the C++ skills that your students need in a manageable progression, *Murach’s C++ Programming* is divided into four sections.

Section 1: Essential skills for modern C++

Section 1 presents an 8-chapter course in C++ programming that gets your students off to a great start. By the time they finish chapter 8, they’ll be able to develop, test, debug, and deploy C++ procedural programs. In these chapters, your students will learn how to use strings and vectors, how to use files for persistent data storage, and how to code functions. This completes a section that by itself is an excellent first course in programming, and this will put your students far ahead of what they can accomplish with competing books.

Section 2: More skills as you need them

The 5 chapters in this section present other skills that every C++ programmer should have. That includes how to work with structures, enumerations, Standard Template Library (STL) containers, STL iterators, STL algorithms, built-in arrays, C strings, and exceptions.

Since all of the chapters in this section are written as independent modules, you can assign them in whatever sequence you prefer, and you don’t have to assign all of them. This makes it easy for you to adapt this book to the time constraints and requirements of your course. The only exception is that chapters 10 (STL containers and iterators) and 11 (STL algorithms) are a pair. Because of that, if you want to teach chapter 11, you should teach chapter 10 first.

Section 3: Object-oriented programming

The 3 chapters in this section present the most important concepts of object-oriented programming (OOP): encapsulation, object composition, inheritance, and polymorphism. These concepts are the same in all modern programming languages, so once your students master them, they’ll be able to apply them in any other language they need to learn.

Depending on the emphasis of your course, you can skip to this section right after you complete chapters 1-9 and 13. Then, you can return to the chapters that you skipped at a later time.

Section 4: Skills for legacy and generic programming

The first 2 chapters in this section are a solid introduction to some skills that are necessary for legacy programming and generic programming. Then, the last chapter brings together all of the skills covered in this book by showing how to code custom containers and algorithms that work like the containers and algorithms of the STL that’s presented in section 2.

About the modularity of the book

For most courses, *Murach’s C++ Programming* will present more concepts and skills than you can cover in a single course. So please keep in mind that our book has a modular design, which means you don't have to teach the chapters in sequence, and you can choose which ones to cover in your course. With that in mind, we offer these thoughts on how you can use our book:

- When your students complete section 1, you've already taught an excellent first course in programming.
- Except for chapters 10 and 11, which should be taught sequentially, you can teach the chapters in section 2 in whatever sequence you prefer.
- You can skip to section 3 after you finish section 1 plus chapters 9 and 13.

So if your students are completely new to programming, you can focus on section 1, teaching it at a pace that won't leave anyone behind. But if they're able to move more quickly through that section, you have a number of options as to what to cover next, depending on your own interests and those of your students.

Beyond that, instructors often tell us that their students keep our books for reference on the job later on. So don't worry if you don't have the time to teach all the chapters…your students will still get their money's worth out of our book!

What's included in the student download

To help your students get the most from our book, our retail website at www.murach.com lets them download (1) the book applications, (2) the starting code for the exercises in the book, and (3) *the solutions to the book exercises*. Appendixes A and B in the book show your students how to download and set up these materials on their own systems.

Chapter 1 in the book introduces your students to using Visual Studio or Xcode to develop C++ programs. So the downloadable code is stored in folders named *vs* and *xcode*. In Visual Studio, any data files needed for a project are included in the VS project, but with Xcode, those files are separate and are stored in a folder called *files*. There's also a folder named *dist* that contains the files needed to deploy an application as described in chapter 8.

Book applications

All of the programs in this book are included in what we refer to as the *book applications,* and they're stored in a folder named *book_apps* within the *vs* and *xcode* folders. Once your students have set up the book applications on their own systems, they can run them to see how they work. They can review all of the code in any application when the book only presents the coding highlights. And they can copy and paste code from the book applications into their own C++ programs.

Exercise starts

Each chapter in the book ends with exercises to help your students master the skills covered in the chapter. But unlike other exercises you've seen, these are designed to give your students the most practice in the least time. That's why your students will start most of the exercises from program files that contain some of the routine code that the exercise requires. That way, your students can focus on the new skills that they're learning. These exercise starts are stored starting in a folder named *ex_starts* within the *vs* and *xcode* folders.

And just so you know, the chapter exercises also differ from the norm in that they don't focus on trivial busywork. Instead, they guide students through the process of building and enhancing a variety of programs that show how new skills are used together. Yes, the exercises go step-by-step, but your students will be practicing with the kind of programs they’ll encounter in the real world, instead of dealing with single, isolated skills that provide no perspective. In fact, if your students can successfully do all of the exercises, they will be well on their way to a professional level of competence.

Exercise solutions

To help students get over any learning obstacles when they’re working on their own, the download also provides the solutions to the book exercises in a folder named *ex_solutions* within the *vs* and *xcode* folders. That way, students can check the solutions to see how something is done whenever they’re wasting time on what is likely to be a trivial coding mistake. We think that providing the solutions is the right approach didactically because it helps students learn faster and better.

We realize, however, that this makes it difficult for you to use the book exercises to test your students. That’s why the instructor’s materials include a set of chapter-by-chapter projects as well as two extensive case studies that can be used for testing. The instructor’s materials also include solutions for these projects and case studies.

What’s included in the instructor’s materials

The instructor’s materials for our C++ book are designed to save you time in preparing and running an effective course based on the text, so that your students gain the programming skills they’ll need on the job. Besides the materials in the student download, these resources include instructional objectives, test banks, projects, case studies, and PowerPoint slides. A summary of these materials follows.

Book applications, exercise starts, and solutions

These are the same materials that your students can download from our retail website. We’ve included them in the instructor’s files so you can demonstrate and review the book applications and exercise solutions in class, without having to download them yourself.

One note: *If you use the Xcode versions of the programs*, you’ll need to get the path to the data files set correctly before you can demonstrate the code. You can usually do that by moving the *files* folder into the *cpp* folder.

Objectives

Since we believe that instructional objectives should be the start of any educational methodology, we provide a set of objectives for each chapter in the book. We prepared these objectives based on the principles presented by Robert F. Mager in his classic book, *Preparing Instructional Objectives*. As a result, our objectives describe the skills that your students should have when they complete a chapter, and you should be able to test whether they have those skills.

If you review the objectives, you’ll see that the first objectives for each chapter are what we refer to as *applied objectives*. These ask the students to apply what they’ve learned as they develop C++ programs. These of course are the critical objectives of a programming course, and they are best tested by having the students develop the projects or case studies that we provide.

After the applied objectives for each chapter, you’ll find what we refer to as *knowledge objectives*. These objectives define skills like identifying, describing, and explaining the required concepts, terms, and procedures. These objectives determine whether your students are able to talk intelligently about the topics that are presented. And these objectives can be tested by the test banks that we provide.

To help you get the most from the instructional objectives, we include them at the start of the PowerPoint slides for each chapter. As we see it, if you can convince your students that they only need to have the skills that are described by the objectives, their study becomes more focused and efficient.

Test banks

To test comprehension, you can use the test banks that we’ve created; there’s one for each chapter in the book. We developed these test banks in ExamView, and we provide them in multiple formats: ExamView, Rich Text (RTF), the current Blackboard formats, Respondus, and IMS QTI (which can be used by multiple LMS’s).

Each test bank provides questions that are designed to test the skills described by the objectives for that chapter, and each test question is designed to test the skill described by one objective. This keeps the promise to the students that they will only be expected to have the skills that are described by the objectives.

In our test banks, we use only multiple-choice test questions because they not only are the easiest to score but also have the highest validity. To us, that means that the students who get the best scores are also the ones with the best knowledge and skills. In contrast, matching and true/false questions have low validity, so we don’t use them.

Projects and solutions

To give your students practice and to test whether they can develop their own programs, the instructor’s materials include over 50 projects, broken down by chapter. For each chapter, the projects range from simple to complex so you can assign the ones that are appropriate for your students. If your students can do the more difficult projects for each chapter, that’s proof that they’re developing the skills that are needed on the job.

The instructor’s materials also include any starting files that are needed for the projects as well as the project solutions. That way, you can demonstrate the projects in class. You can also show the code for the solutions, and you can compare it to the solutions that the students develop.

Case studies and solutions

To provide a more extensive way to test the programming skills of your students, the instructor’s materials also include two case studies. One case study has the students build a Shopping Cart program that lets the user add or remove items from a virtual shopping cart. The other has the students build a Blackjack game that lets the user play blackjack against a computer dealer. The instructor’s materials also include the solutions for both case studies.

These case studies can be assigned on a section-by-section basis. That means that the students build one version of the program for the case study for section 1 of the book, an enhanced version for section 2, an object-oriented version for section 3, and a version that uses a custom container and algorithm for section 4. To facilitate that, the instructor’s materials provide section-by-section write-ups for the case studies, as well as the solutions for each section.

PowerPoint slides

The PowerPoint slides present all of the critical information from the figures of the book. That includes all of the screenshots, diagrams, tables, and code that you may want to review in class. As a result, these slides make it easy for you to review any of the skills that your students have difficulty with. Beyond the book information, the slides for each chapter start with the instructional objectives so you can review them in class.

How to get started with our materials

Once you have an instructor account at our instructor website (www.murachforinstructors.com), you can request the instructor’s materials for our book and download them from your account page. The download is available as an executable file (for Windows systems only) or as a zip file. Then, you can install the materials on your computer as described below.

Once the installation is done, you can do a thorough review of the materials that are provided. In particular, you’ll want to run some of the book applications, as well as some of the solutions for the book exercises, projects, and case studies, to see the level of competence that our book develops. You’ll also want to click through some of the PowerPoint slides to see how they can help you review and reinforce the information that’s presented in the book. To help you find what you’re looking for, the entire file structure for the instructor’s materials is shown on the next page.

How to install the executable file on a Windows system

1. Find the .exe file that you downloaded from your account page at our instructor site.
2. Double-click on the file and respond to the dialog boxes that follow. This will install the folders and files onto your C drive in a folder structure that starts with

   ```
   c:\murach\cpp
   ```

How to install the zip file on a macOS (or any) system

1. Find the .zip file that you downloaded from your account page at our instructor site.
2. Move the .zip file to the folder where you want to keep the files, if needed. On macOS, that will be the Documents folder.
3. Double-click on the .zip file to unzip the files and folders into a folder structure that starts with

   ```
   murach/cpp
   ```

Note for Xcode users

To get the code for student projects and case studies to work correctly, you will need to set the full path to the file. You’ll probably want to do that by changing the working directory for the Xcode project to the directory that contains the text file:

1. Open the project and select the Product→Scheme→Edit Scheme item.
2. Select the Run category.
3. Check the Use Custom Working Directory box, and specify the working directory.

The student download files that get installed

murach\cpp\student_download\	Contents
dist\	The executable files for a console application that you can run as described in chapter 8.
files\	The files used by the projects presented throughout the book (for Xcode only; not used by Visual Studio).
vs\	The Visual Studio projects for the book applications, exercise starts, and exercise solutions.
xcode\	The Xcode projects for the book applications, exercise starts, and exercise solutions.

The instructor materials that get installed

murach\cpp\instructors\	Contents
Instructor's summary.pdf	This instructor's summary in PDF.
Objectives.docx Objectives.pdf	The instructional objectives for all chapters in both Word and PDF (the individual chapter objectives are repeated in the chapter slides).
student_projects\	
Projects.docx Projects.pdf	The specifications for the chapter-by-chapter Student Projects in both Word and PDF.
vs\	
project_starts	The project starts for Visual Studio.
product_solutions	The project solutions for Visual Studio.
xcode\	
project_starts	The project starts for Xcode.
project_solutions	The project solutions for Xcode.
case_studies\	
Case study – Shopping cart.docx Case study – Shopping cart.pdf	The section-by-section specifications for the Shopping Cart case study in both Word and PDF.
Case study – Blackjack.docx Case study – Blackjack.pdf	The section-by-section specifications for the Blackjack case study in both Word and PDF.
vs\case_study_solutions	The case study solutions for Visual Studio.
xcode\case_study_solutions	The case study solutions for Xcode.
slides\	One PowerPoint file for each chapter.
test_banks\	One test bank for each chapter in various formats, including ExamView, RTF (Word), Blackboard, Respondus, and IMS QTI.

Any comments?

If you have any comments about our book or its instructor’s materials, we would be delighted to hear from you. If you discover any errors in our applications or solutions, we would appreciate hearing about them. And if you want to let us know that you’re going to adopt our book for your course, that would make our day.

Just e-mail us at the addresses below. But whether or not we hear from you, we want to thank you for your interest in our C++ book.

Joel Murach, Author
joel@murach.com

Judy Taylor, Educational Liaison
judy@murach.com