THE WAITE GROUP'S

C++® Primer Plus®

Third Edition

Over 100,000 Copies Sold

STEPHEN PRATA

FINAL ANSI/ISO STANDARD C++

SAMS

THE WAITE GROUP'S

C++ Primer Plus, Third Edition

Stephen Prata

SAMS
201 West 103rd St., Indianapolis, Indiana, 46290

EXECUTIVE EDITOR: Tracy Dunkelberger
ACQUISITIONS EDITOR: Tracy Dunkelberger
DEVELOPMENT EDITOR: Sean Dixon
MANAGING EDITOR: Jodi Jensen
PROJECT EDITOR: Tonya Simpson
INDEXER: Joy Dean Lee
TECHNICAL EDITORS: Steven Mitchell, Dallas Releford, Rabi Satler
TEAM COORDINATOR: Carol Ackerman
COVER DESIGNER: Karen Ruggles
INTERIOR DESIGNER: Gary Adair
PRODUCTION: Michael Henry, Linda Knose, Tim Osborn, Staci Somers, Mark Walchle

The Waite Group's C++ Primer Plus, Third Edition

International Standard Book Number: 1-57169-162-6

Library of Congress Catalog Card Number: 98-85890

Printed in the United States of America

First Printing: September, 1998

01 8 7 6

Trademarks

Warning and Disclaimer

About the Author

Stephen Prata teaches astronomy, physics, and computer science at the College of Marin in Kentfield, California. He received his B.S. from the California Institute of Technology and his Ph.D. from the University of California, Berkeley. Stephen has authored or coauthored over a dozen books for The Waite Group, including *Artificial Life Playhouse* and *Certified Course in Visual Basic 4*. He also wrote The Waite Group's *New C Primer Plus*, which received the Computer Press Association's 1990 Best How-to Computer Book Award and The Waite Group's *C++ Primer Plus*, nominated for the Computer Press Association's Best How-to Computer Book Award in 1991.

Dedication

To my colleagues and students at the College of Marin,
with whom it is a pleasure to work.

-Stephen Prata

Acknowledgments

ACKNOWLEDGMENTS TO THE THIRD EDITION
I'd like to thank the editors from Sams Publishing and the Waite Group for the roles they played in putting this book together: Tracy Dunkelberger, Susan Walton, and Andrea Rosenberg. Thanks, too, to Russ Jacobs for his content and technical editing. From Metrowerks, I'd like to thank Dave Mark, Alex Harper, and especially Ron Liechty, for their help and cooperation.

ACKNOWLEDGMENTS TO THE SECOND EDITION
I'd like to thank Mitchell Waite and Scott Calamar for supporting a second edition and Joel Fugazzotto and Joanne Miller for guiding the project to completion. Thanks to Michael Marcotty of Metrowerks for dealing with my questions about their beta version CodeWarrior compiler. I'd also like to thank the following instructors for taking the time to give us feedback on the first edition: Jeff Buckwalter, Earl Brynner, Mike Holland, Andy Yao, Larry Sanders, Shahin Momtazi, and Don Stephens. Finally, I wish to thank Heidi Brumbaugh for her helpful content editing of new and revised material.

ACKNOWLEDGMENTS TO THE FIRST EDITION
Many people have contributed to this book. In particular, I wish to thank Mitch Waite for his work in developing, shaping, and reshaping this book, and for reviewing the manuscript. I appreciate Harry Henderson's work in reviewing the last few chapters and in testing programs with the Zortech C++ compiler. Thanks to David Gerrold for reviewing the entire manuscript and for championing the needs of less experienced readers. Also thanks to Hank Shiffman for testing programs using Sun C++ and to Kent Williams for testing programs with AT&T cfront and with G++. Thanks to Nan Borreson of Borland International for her responsive and cheerful assistance with Turbo C++ and Borland C++. Thank you, Ruth Myers and Christine Bush, for handling the relentless paper flow involved with this kind of project. Finally, thanks to Scott Calamar for keeping everything on track.

Tell Us What You Think!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

As the Executive Editor for the Advanced Programming and Distributed Architechture team at Sams Publishing, I welcome your comments. You can fax, email, or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books stronger.

Please note that I cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail I receive, I might not be able to reply to every message.

When you write, please be sure to include this book's title and author as well as your name and phone or fax number. I will carefully review your comments and share them with the author and editors who worked on the book.

Fax:	317-817-7070
E-mail:	`feedback@samspublishing.com`
Mail:	Executive Editor Advanced Programming and Distributed Architecture Team 201 West 103rd Street Indianapolis, IN 46290 USA

Preface to the Third Edition

Learning C++ is an adventure of discovery, particularly because C++ is a moving target. Since the second edition of this book, the C++ language has continued to evolve under the guidance of the ISO/ANSI committee, and it has continued to mature as programmers explore the language's features. But now the C++ standard is in place, and no further major changes are expected for a while. Thus, it is a good time to summarize the state of the language, and that is the goal of this third edition of *C++ Primer Plus.*

The main changes in content are these:

- The book reflects changes in and additions to the language since the previous edition, including the expanding role of templates and generic programming.
- A new chapter explores the string class and the Standard Template Library (STL). One of the main thrusts of C++ is reusable code, and these class libraries provide useful and efficient examples.
- Some of the discussions have been revised to further simplify and clarify the presentation of ideas.

Like the previous editions, this book practices generic C++ so that it is not tied to any particular kind of computer, operating system, or compiler. All the programs were tested with CodeWarrior Pro 2 (Macintosh and Windows) and Microsoft Visual C++ 5.0 and most were tested with Borland C++Builder 1.0, Symantic C++ 8.0, Release 5 for the Macintosh, Watcom C++ 10.6 for IBM PC compatibles, and Gnu g++ 2.7.1 running under Linux. None of the implementations were completely consistent with the standard, but that state of affairs is to be expected, for these implementations all preceded the acceptance of the standard.

C++ offers a lot to the programmer; have fun as you harvest its riches.

Preface to the Second Edition

Learning C++ is not a simple task. Not only is it a very full-featured language, but it also supports a programming style (object-oriented programming) that may require you to learn new ways of thinking about programming. Furthermore, C++ has rules of practice that aren't built into the language. For example, to use the language feature called inheritance correctly, you have to learn the proper language rules so that the compiler will accept your program, but you also have to learn conceptual rules about when it is and isn't appropriate to use inheritance. Also, C++ is a moving target, and it has evolved significantly since the first edition of this book.

This book aims to make learning C++ manageable, even pleasurable. It follows the precepts outlined in the Preface to the First Edition. In addition, the new edition does the following:

- It presents additions to C++, such as templates, exceptions, RTTI, and namespaces.
- It tracks changes in C++, such as in the rules governing reference arguments.
- It reflects the developing draft ANSI/ISO C++ standard.
- It provides more conceptual guidance about when to use particular features, such as using public inheritance to model what are known as *is-a* relationships.
- It illustrates common C++ programming idioms and techniques.
- It has programming exercises at the end of each chapter to provide practice in applying new ideas.
- It devotes greater attention to organizing and explaining C++ classes, dividing the original presentation into more chapters, and revising and expanding the discussion.

Like the first edition, this book practices generic C++. That means you should be able to use it with any contemporary C++ implementation. Toward the end we tested the examples with a variety of compilers, including Borland C++ 3.1, Borland C++ 4.0, GNU C++ 2.0, Metrowerks CodeWarrior CW 3.5, Microsoft Visual C++ 1.0, Symantec C++ 6.0 (PC), and Symantec C++ 7.0 (Mac). Ideally, C++ is C++ is C++, but compilers do differ in how closely they track the draft standard. For example, many of these compilers don't yet support templates or exceptions; naturally, they won't run examples using those features. Aside from that, however, we ran into only a few minor differences, which the book notes. In general, C++ implementations are more consistent with each other now than they were at the time of the first edition, which is good news for programmers.

Learn and enjoy!

Preface to the First Edition

When the Waite Group first released *C Primer Plus* in 1984, the C language had been around for about a decade but was just beginning to boom. We take pride in the important role that our book played in introducing programmers to C. Today the C++ language, which derives from C, has reached a similar stage in its evolution. It's booming because it offers a new paradigm—object-oriented programming, or OOP—well-suited to modern programming needs. Thus, AT&T is rewriting UNIX in C++ because C++ improves the reliability, maintainability, and reusability of the code. Apple is using C++ to develop system software for its Macintosh line for the same reasons and because OOP techniques are a natural match to program features such as windows and dialog boxes. Individual programmers are turning to C++ because its new features bring the thrill back to programming. Naturally, it's time to release *C++ Primer Plus* and help this boom along.

One difference between now and then is that many more books have been written about C++ than were written about C when it was new. However, none of the new C++ books plays the role that a Waite Group "primer" does. Many C++ books assume that you already know C and know it well. That's of little help to those who wish to move to C++ from, say, Pascal or BASIC, or to those who have enjoyed C recreationally without acquiring expert status. Most of the other C++ books present the full language, not just the new elements, but still assume you are fairly knowledgeable in C and in programming in general. Some C++ books make excellent references but can be rather tough sledding for learning the language. A few C++ titles were rushed out the door. They merely tack on a few new chapters to an old C book and don't fully integrate the new material or really do justice to C++'s exciting new object-oriented features.

Enter the Waite Group's *C++ Primer Plus*. We don't assume you know C, and we integrate discussing the basic C language with presenting the C++ features. We do assume you've had some programming experience, but we don't skip over the basics. We've tried to present C++ in a book instilled with traditional Waite Group primer virtues:

- A primer should be an easy-to-use, friendly guide.
- A primer doesn't assume that you already are familiar with all relevant programming concepts.
- A primer emphasizes hands-on learning with brief, easily typed examples that develop your understanding a concept or two at a time.
- A primer clarifies concepts with illustrations.
- A primer provides exercises to let you test your understanding, making the book suitable for self-learning or for the classroom.

C++ Primer Plus presents C++ fundamentals and illustrates them with short, to-the-point programs that are easy to copy and to experiment with. The book is not intended to provide encyclopedic coverage of all features and nuances of the C++ language, but it does present the most important aspects while laying the foundation for further study. You'll learn about input and output, how to make programs perform repetitive tasks and make choices, the many ways to handle the data, and how to use functions. You'll learn about the important object-oriented programming concepts of information hiding (lots of fun), polymorphism (not as bad as it sounds), and inheritance. Besides learning basic techniques, you'll learn about the OOP philosophy. Meanwhile, we'll do our best to keep the presentation short, simple, and fun. Our goal is that by the end you'll be able to write solid, effective programs and enjoy yourself doing so.

Note to Instructors

One of the goals of the third edition is to provide a book that can be used either as a teach-yourself book or as a textbook. Here are some of the features that support using *C++ Primer Plus*, Third Edition as a textbook:

- This book describes generic C++, so it isn't dependent upon some particular implementation.
- The contents track the ISO/ANSI C++ standards committee's work and include discussions of templates, the Standard Template Library, the string class, exceptions, RTTI, and namespaces.
- It doesn't assume prior knowledge of C, so it can be used without a C prerequisite. (Some programming background is desirable, however.)
- Topics are arranged so that the early chapters can be covered rapidly as review chapters for courses that do have a C prerequisite.
- Chapters have review questions and programming exercises.
- The book introduces several topics appropriate for computer science courses, including abstract data types, stacks, queues, simple lists, simulations, generic programming, and using recursion to implement a divide-and-conquer strategy.
- Most chapters are short enough to cover in a week or less.
- The book discusses *when* to use certain features as well as *how* to use them. For example, it links public inheritance to *is-a* relationships and composition and private inheritance to *has-a* relationships, and it discusses when to use virtual functions and when not to.

How This Book Is Organized

This book is divided into 16 chapters and 10 appendices summarized here.

Chapter 1: Getting Started

This chapter relates how Bjarne Stroustrup created the C++ programming language by adding object-oriented programming support to the C language. You'll learn the distinctions between procedural languages, such as C, and object-oriented languages, such as C++. You'll read about the joint ANSI/ISO work to develop a C++ standard. The chapter discusses the mechanics of creating a C++ program, outlining the approach for several current C++ compilers. Finally, it describes the conventions used in this book.

Chapter 2: Setting Out to C++

Chapter 2 guides you through the process of creating simple C++ programs. You'll learn about the role of the `main()` function and about some of the kinds of statements that C++ programs use. You'll use the predefined `cout` and `cin` objects for program output and input, and you'll learn about creating and using variables. Finally, you'll be introduced to functions, C++'s programming modules.

Chapter 3: Dealing with Data

C++ provides built-in types for storing two kinds of data: integers (numbers with no fractional parts) and floating-point numbers (numbers with fractional parts). To meet the diverse requirements of programmers, C++ offers several types in each category. This chapter discusses these types, including creating variables and writing constants of various types. You'll also learn how C++ handles implicit and explicit conversions from one type to another.

Chapter 4: Derived Types

C++ lets you construct more elaborate types from the basic built-in types. The most advanced form is the class, discussed in Chapters 9, 10, 11, 12, and 13. This chapter discusses other forms, including arrays, which hold several values of a single type; structures, which hold several values of unlike types; and pointers, which identify locations in memory. You'll also learn how to create and store text strings and to handle text input and output. Finally, you'll learn some of the ways C++ handles memory allocation, including the `new` and `delete` operators for managing memory explicitly.

Chapter 5: Loops and Relational Expressions

Programs often must perform repetitive actions, and C++ provides three looping structures for that purpose: the `for` loop, the `while` loop, and the `do while` loop. Such loops must know when they should terminate, and the C++ relational operators enable you to create tests to guide such loops. You'll also learn how to create loops that read and process input character-by-character. Finally, you'll learn how to create two-dimensional arrays and how to use nested loops to process them.

Chapter 6: Branching Statements and Logical Operators

Programs can behave intelligently if they can tailor their behavior to circumstances. In this chapter you'll learn how to control program flow by using the `if`, `if else`, and `switch` statements and the conditional operator. You'll learn how to use logical operators to help express decision-making tests. Also, you'll meet the `cctype` library of functions for evaluating character relations, such as testing whether a character is a digit or a nonprinting character.

Chapter 7: Functions—C++'s Programming Modules

Functions are the basic building blocks of C++ programming. This chapter concentrates on features that C++ functions share with C functions. In particular, you'll review the general format of a function definition and examine how function prototypes increase the reliability of programs. Also, you'll investigate how to write functions to process arrays, character strings, and structures. Next you'll learn about recursion, which is when a function calls itself, and see how it can be used to implement a divide-and-conquer strategy. Finally, you'll meet pointers to functions, which enable you to use a function argument to tell one function to use a second function.

Chapter 8: Adventures in Functions

This chapter explores the new features C++ adds to functions. You'll learn about inline functions, which can speed program execution at the cost of additional program size. You'll work with reference variables, which provide an alternative way to pass information to functions. Default arguments let a function automatically supply values for function arguments that you omit from a function call. Function overloading lets you create functions having the same name but taking different argument lists. All these features have frequent use in class design. Also, you'll learn about function templates, which allow you to specify the design of a family of related functions. You'll learn about putting together multifile programs. Finally, you'll examine storage classes, scope, linkage, and namespaces, which determine what parts of a program know about a variable.

Chapter 9: Objects and Classes

A class is a user-defined type, and an object is an instance of a class, such as a variable. This chapter introduces you to object-oriented programming and to class design. A class declaration describes the information stored in a class object and also the operations (class methods) allowed for class objects. Some parts of an object are visible to the outside world (the public portion), and some are hidden (the private portion). Special class methods (constructors and destructors) come into play when objects are created and destroyed. You will learn about all this and other class details in this chapter, and you'll see how classes can be used to implement abstract data types (ADTs), such as a stack.

Chapter 14: Friends, Exceptions, and More

This chapter extends the discussion of friends to include friend classes and friend member functions. Then it presents several new developments in C++, beginning with exceptions, which provide a mechanism for dealing with unusual program occurrences, such an inappropriate function argument values or running out of memory. Then you'll learn about RTTI (routine type information), a mechanism for identifying object types. Finally, you'll learn about the safer alternatives to unrestricted typecasting.

Chapter 15: The `string` Class and the Standard Template Library

This chapter discusses some useful class libraries recently added to the language. The `string` class is a convenient and powerful alternative to traditional C-style strings. The `auto_ptr` class helps manage dynamically allocated memory. The Standard Template Library (STL) provides several generic containers, including template representations of arrays, queues, lists, sets, and maps. It also provides an efficient library of generic algorithms that can be used with STL containers and also with ordinary arrays.

Chapter 16: Input, Output, and Files

This chapter reviews C++ I/O and discusses how to format output. You'll learn how to use class methods to determine the state of an input or output stream and to see, for example, if there has been a type mismatch on input or if end-of-file has been detected. C++ uses inheritance to derive classes for managing file input and output. You'll learn how to open files for input and output, how to append data to a file, how to use binary files, and how to get random access to a file. Finally, you'll learn how to apply standard I/O methods to read from and write to strings.

Appendix A: Number Bases

This appendix discusses octal, hexadecimal, and binary numbers.

Appendix B: C++ Keywords

This appendix lists C++ keywords.

Appendix C: The ASCII Character Set

This appendix lists the ASCII character set along with decimal, octal, hexadecimal, and binary representations.

Appendix D: Operator Precedence

This appendix lists the C++ operators in order of decreasing precedence.

Appendix E: Other Operators

This appendix summarizes those C++ operators, such as the bitwise operators, not covered in the main body of the text.

Appendix F: The `string` Template Class

This appendix summarizes `string` class methods and functions.

Appendix G: The STL Methods and Functions

This appendix summarizes the STL container methods and the general STL algorithm functions.

Appendix H: Selected Readings

This appendix lists some books that can further your understanding of C++.

Appendix I: Converting to ANSI/ISO Standard C++

This appendix provides guidelines for moving from C and older C++ implementations to Standard C++.

Appendix J: Answers to Review Questions

This appendix contains the answers to the review questions posed at the end of each chapter.

Table of Contents

Contents