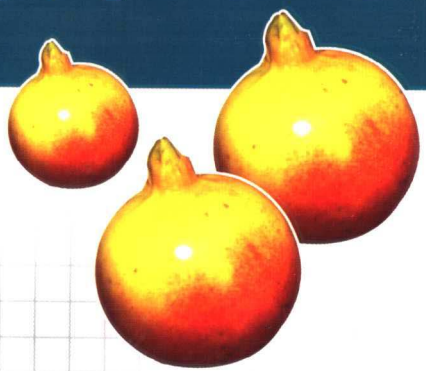
大学计算机教育国外著名教材系列（影印版）

PEARSON Prentice Hall

# OBJECT-ORIENTED PROGRAMMING IN C++

SECOND EDITION

# C++面向对象程序设计（第2版）



Richard Johnsonbaugh
Martin Kalin

著

PEARSON Education

清华大学出版社

大学计算机教育国外著名教材系列(影印版)

# Object-Oriented Programming in C++

Second Edition

# C++面向对象程序设计

(第2版)

Richard Johnsonbaugh
Martin Kalin
*DePaul University*

清华大学出版社
北京

**图书在版编目(CIP)数据**

C++面向对象程序设计=Object-Oriented Programming in C++/(美)约翰逊鲍(Johnsonbaugh, R.),(美)马丁卡林(Kalin, M.)著. —影印本. —北京：清华大学出版社，2005.8
(大学计算机教育国外著名教材系列)
ISBN 7-302-11240-1

Ⅰ.C… Ⅱ.①约…②马… Ⅲ.C语言—程序设计—高等学校—教材—英文 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2005)第 066689 号

**出 版 者**：清华大学出版社 **地 址**：北京清华大学学研大厦
http://www.tup.com.cn **邮 编**：100084
**社总机**：010 62770175 **客户服务**：010-62776969
**印 刷 者**：北京市世界知识印刷厂
**装 订 者**：三河市化甲屯小学装订二厂
**发 行 者**：新华书店总店北京发行所
**开 本**：148×210 **印张**：20
**版 次**：2005 年 8 月第 1 版 2005 年 8 月第 1 次印刷
**书 号**：ISBN 7-302-11240-1/TP·7415
**印 数**：1～3000
**定 价**：38.00 元

# 出 版 说 明

进入21世纪，世界各国的经济、科技以及综合国力的竞争将更加激烈。竞争的中心无疑是对人才的竞争。谁拥有大量高素质的人才，谁就能在竞争中取得优势。高等教育，作为培养高素质人才的事业，必然受到高度重视。目前我国高等教育的教材更新较慢，为了加快教材的更新频率，教育部正在大力促进我国高校采用国外原版教材。

清华大学出版社从1996年开始，与国外著名出版公司合作，影印出版了“大学计算机教育丛书(影印版)”等一系列引进图书，受到国内读者的欢迎和支持。跨入21世纪，我们本着为我国高等教育教材建设服务的初衷，在已有的基础上，进一步扩大选题内容，改变图书开本尺寸，一如既往地请有关专家挑选适用于我国高等本科及研究生计算机教育的国外经典教材或著名教材，组成本套“大学计算机教育国外著名教材系列(影印版)”，以飨读者。深切期盼读者及时将使用本系列教材的效果和意见反馈给我们。更希望国内专家、教授积极向我们推荐国外计算机教育的优秀教材，以利我们把“大学计算机教育国外著名教材系列(影印版)”做得更好，更适合高校师生的需要。

清华大学出版社

# Preface

This book is based on C++ courses given by the authors at DePaul University and can be used for self-study or for a course on object-oriented programming in C++. We assume no prior knowledge of C++, but we do assume knowledge of C. Coverage of C at the level provided in R. Johnsonbaugh and M. Kalin, *Applications Programming in ANSI C*, 3rd ed. (Upper Saddle River, N.J.: Prentice Hall, 1996) provides sufficient background for this book. The book and its supplements—an *Instructor's Guide*, and a World Wide Web site—provide a comprehensive support system to help the reader master C++. In this book, as in our other C and C++ books, we make extensive use of examples, figures, self-study exercises, sample applications, lists of common programming errors, and programming exercises. We strive for clarity throughout the book and also illustrate a variety of good programming practices.

This book treats object-oriented principles (see Chapter 1); emphasizes sound programming practices; introduces templates and the standard template library (see Chapter 7); presents in depth the C++ input/output class hierarchy (Chapter 8); features major, useful examples (e.g., a stack class, Sections 3.2 and 7.2; and a random access file class, Section 8.6); and introduces object-oriented programming in the Microsoft Foundation Classes in Chapter 9.

The C++ language presented here is based on the approved standard. As such it contains the latest additions and changes to the language including

- The logical type **`bool`**.
- The **`string`** class.
- New-style headers.
- Namespaces and the namespace **`std`**.
- New-style casts.
- STL (Standard Template Library).
- Exception handling.
- Run-time type identification.
- The operator **`new[ ]`**.
- The template input/output classes.
- The **`stringstream`** classes.

## Overview

During the 1980s and early 1990s, C became the language of choice for many applications and systems programmers. Most major software available for personal computers was written in C: spreadsheets, word processors, databases, communications packages, statistical software, graphics packages, and so on. Virtually all software written for the UNIX

environment was likewise written in C, and many mainframe systems meant to be ported from one platform to another were also coded in C. In the early 1980s, Bjarne Stroustrup of AT&T Bell Labs developed C++ as an extension of C that supports object-oriented programming, a type of programming that is well suited to the large, complex software systems now written for all platforms, from the cheapest personal computers to the most expensive mainframes. C++ also corrects some shortcomings in C, which C++ includes as a subset, and it supports abstract data types and generic functions through templates.

C++ is a highly complex language. Fortunately, most C++ programmers can benefit from its power without mastering each and every one of its features. We focus on the most useful aspects of the language, but we place some of the more esoteric and specialized parts of the language in end-of-chapter sections labeled *C++ Postscript*. We focus on *using* C++ to write practical programs based on sound design techniques, rather than on tricks and surprises in C++.

## About This Book

This book includes

- Examples and exercises that cover a wide range of applications.
- Motivating real-world applications.
- A broad variety of programming exercises. The book contains over 100 programming exercises.
- End-of-chapter lists of common programming errors.
- Coverage of STL (the Standard Template Library) (Chapter 7).
- Discussion of the standard C++ input/output class library (Chapter 8).
- Coverage of object-oriented programming in the Microsoft Foundation Classes (Chapter 9).
- Exercises at the ends of sections so that readers can check their mastery of the sections. The book contains over 500 such exercises. Answers to the odd-numbered section exercises are given in the back of the book, and answers to the even-numbered section exercises are given in the *Instructor's Guide*.
- Figures to facilitate the learning process.
- The latest changes and additions to the C++ language.
- Major data structures, including stacks (Sections 3.2 and 7.2) and files (Section 8.6), implemented in C++.
- Understandable code. We have opted for clarity rather than subterfuges based on obscure C++ features.

## Changes from the First Edition

- Recent changes and additions to C++ are incorporated throughout the book.

- Namespaces are introduced earlier (Section 2.1) because namespace **std** is needed for the new-style headers.
- Exception handling is also introduced earlier (Section 2.8).
- An entire chapter is devoted to polymorphism (Chapter 5).
- Inheritance (Chapter 4) and polymorphism (Chapter 5) are introduced earlier to underscore their importance to the object-oriented programming paradigm. Operator overloading therefore comes later (Chapter 6).
- Run-time type identification is integrated into the main body of the text (Section 5.5).
- An entire chapter (Chapter 7) is devoted to templates and the standard template library.
- Chapter 8 (Chapter 7 in the first edition) on the C++ input/output class hierarchy is considerably revised to incorporate significant changes to this hierarchy.
- Chapter 9 is added to cover object-oriented programming in the Microsoft Foundation Classes.
- Some of the more recondite parts of the language are reserved for *C++ Postscript* sections at ends of chapters.
- The number of worked examples has been increased to nearly 300.
- The number of exercises has been increased to over 500.
- We have extensively revised the sample applications.
- The figures are boxed to separate them visually from the text.
- A World Wide Web site has been established to provide up-to-date support for the book.

## Organization of the Book

Chapter 1 introduces key concepts associated with object-oriented design and programming: classes, abstract data types, objects, encapsulation, the client/server model, message passing, inheritance, polymorphism, and others. The chapter contrasts object-oriented design with top-down functional decomposition and offers examples to illustrate the differences between the approaches.

Important changes and additions to C++ are detailed in Chapter 2. Chapter 2 also introduces namespaces, type **string**, the **new** and **delete** operators, exception handling, and basic C++ input/output. The reader can begin using these important C++ features right away.

Chapter 3 covers the basics of classes so that the reader can begin using classes at once. The chapter explains how to declare classes; how to write constructors, destructors, and other methods; **static** data members and methods; and pointers to objects. Chapter 3 relies heavily on examples to explain how classes may be used to implement abstract data types and to meet the object-oriented goal of encapsulation.

Inheritance (including multiple inheritance) is the topic of Chapter 4. Through examples and sample applications (e.g., a sequence hierarchy), the chapter illustrates basic programming techniques.

Polymorphism is covered in Chapter 5. Section 5.1 carefully explains the distinction between run-time and compile-time binding. Sections 5.4 and 5.5 explain abstract base classes and run-time type identification.

Chapter 6 is devoted to operator overloading. The chapter shows how to overload common operators (e.g., +, /) as well as the subscript, function call, memory management, preincrement, and postincrement operators among others. Many examples and sample applications highlight the power of operator overloading.

Templates and STL (standard template library) are explained in Chapter 7. A template stack class (Section 7.2) shows templates in action, and a sample application (stock performance reports in Section 7.4) demonstrates how to use STL.

Chapter 8 serves several purposes. First, the chapter examines the C++ input/output library in detail so that the interested reader can exploit the powerful classes contained therein. This treatment culminates in a sample application that builds a random access file class through inheritance from a system file class. Second, the chapter uses the input/output library as a major, sophisticated example of object-oriented design realized in C++. Third, the hierarchy provides an excellent example of the use of templates. Chapter 8 pays close attention to manipulators, which are powerful ways to do sophisticated input/output in C++. We believe that Chapter 8 offers an unrivaled examination of C++'s input/output.

Chapter 9 covers object-oriented programming in the Microsoft Foundation Classes (MFC). We clarify the relationship between MFC and the Win32 Applications Programmer Interface, Microsoft's C libraries for accessing systems services. We also introduce the basic concepts and constructs of event-driven programming. The chapter focuses on object persistence through serialization and interapplication communication under Microsoft's Common Object Model. We provide two sample applications together with an overview of MFC and Visual C++.

Two appendices are provided for reference. Appendix A contains the ASCII table. Appendix B contains a list of some of the most useful C++ functions and class methods. We describe the parameters and return values, the header to include, and what the function or method does.

We rely heavily on short examples, figures, and tables to illustrate specific points about the syntax and semantics of C++. From our experience teaching C++ and other languages, we are convinced that no single method is appropriate for clarifying every aspect about a language.

Most of our students agree with us that learning and using C++ is exciting. We have tried to incorporate this view by using engaging examples, sample applications, programming exercises, and short segments of code.

## Chapter Structure

The basic chapter organization is as follows:

Contents
Overview
Section
Section Exercises
Section
Section Exercises
⋮

C++ Postscript

Common Programming Errors

Programming Exercises

In every chapter except 1 and 2, several sections are devoted to sample applications. Each of these sections contains a statement of a problem, sample input and output, a solution to the problem, and a well-documented implementation of a solution to the problem in C++. Most of these sections conclude with an extended discussion.

The sample applications include the following:

- A stack class (Sections 3.2 and 7.2)
- Tracking films (Sections 4.3 and 5.2)
- A sequence hierarchy (Section 4.6)
- A complex number class (Section 6.2)
- An associative array (Section 6.8)
- Stock performance reports (Section 7.4)
- A random access file class (Section 8.6)
- Interapplication communication under Microsoft's Common Object Model (Section 9.5)

The *C++ Postscript* sections discuss less-used parts of the language and give additional technical details about certain parts of the language.

The *Common Programming Errors* sections highlight those aspects of the language that are easily misunderstood.

The book contains over 100 programming exercises drawn from a wide variety of applications.

## Examples

The book contains nearly 300 examples, which clarify particular facets of C++ for the reader and show the purpose of various C++ features. A box ■ marks the end of each example.

## Exercises

The book contains over 500 section review exercises, the answers to which are short answers, code segments, and, in a few cases, entire programs. These exercises are suitable as homework problems or as self-tests. The answers to the odd-numbered exercises are given in the back of the book, and the answers to the even-numbered exercises are given in the *Instructor's Guide*. Our experience in teaching C++ has convinced us of the importance of these exercises.

The applications covered in the programming exercises at the ends of the chapters include the following:

- Simulation (Programming Exercise 2.9)
- Queues (Programming Exercises 3.8 and 7.4)
- Process synchronization (Programming Exercise 3.10)
- Databases (Programming Exercise 3.15)
- Local area networks (Programming Exercises 3.17 and 6.7)
- Array hierarchy (Programming Exercise 4.4)
- Dating services (Programming Exercise 5.10)
- Iterators (Programming Exercises 7.6, 7.7, and 7.8)
- Scheduling (Programming Exercise 7.14)
- An indexed file class (Programming Exercise 8.5)
- A dialog-based application with a graphical-user interface for a system administrator (Programming Exercise 9.9)

Not every reader will be interested in all of these applications; however, we think that it is important to show the variety of problems that C++ can address.

## Instructor Supplement

An *Instructor's Guide* is available from the publisher at no cost to adopters of this book. The *Instructor's Guide* contains solutions to even-numbered section exercises, sample syllabi, and transparency masters.

## World Wide Web Site

The World Wide Web site

```
http://condor.depaul.edu/~mkalin
```

contains the source code, header files, and data files for all of the book's sample applications; the source code for some of the longer examples; sample syllabi; transparencies; a sample chapter; information about using Microsoft Visual C++; additional technical details about the Microsoft Foundation Classes; and an errata list.

## Acknowledgments

We thank the following reviewers: Adair Dingle, Seattle University; Rex Jaeschke, independent consultant; Glenn Lancaster, DePaul University; and Winnie Y. Yu, Southern Connecticut State University.

We are again grateful to our friendly copy editor, Patricia Johnsonbaugh, for checking numerous details and suggesting changes that improved the book.

We are indebted to the School of Computer Science, Telecommunications and Information Systems at DePaul University and its dean, Helmut Epp, for providing time and encouragement for the development of this book.

We received consistent support from the people at Prentice Hall. Special thanks go to Alan R. Apt, publisher; Petra Recter, senior acquisitions editor; and Scott Disanno, production editor.

R.J.
M.K.

# Brief Table of Contents

# Contents