

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



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

English reprint edition copyright © 2005 by PEARSON EDUCATION ASIA LIMITED and TSINGHUA UNIVERSITY PRESS.

Original English language title from Proprietor's edition of the Work.

Original English language title: Object-Oriented Programming in C++, Second Edition by Richard Johnsonbaugh, Martin Kalin, Copyright © 2000

All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Prentice Hall, Inc.

This edition is authorized for sale and distribution only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong, Macao SAR and Taiwan).

本书影印版由 Pearson Education(培生教育出版集团)授权给清华大学出版社出版发行。

For sale and distribution in the People's Republic of China exclusively (except Taiwan, Hong Kong SAR and Macao SAR).

仅限于中华人民共和国境内(不包括中国香港、澳门特别行政区和中国台湾地区)销售发行。

北京市版权局著作权合同登记号 图字: 01-2005-0468

版权所有, 翻印必究。举报电话: 010-62782989 13501256678 13801310933

本书封面贴有 Pearson Education(培生教育出版集团)激光防伪标签, 无标签者不得销售。

#### 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

<http://www.tup.com.cn>

社总机: 010 62770175

地 址: 北京清华大学学研大厦

邮 编: 100084

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:

## **x PREFACE**

- 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**

---

- 1 OBJECT-ORIENTED PROGRAMMING 1**
- 2 FROM C TO C++ 21**
- 3 CLASSES 98**
- 4 INHERITANCE 175**
- 5 POLYMORPHISM 229**
- 6 OPERATOR OVERLOADING 285**
- 7 TEMPLATES AND THE STANDARD TEMPLATE LIBRARY 340**
- 8 THE C++ INPUT/OUTPUT CLASS HIERARCHY 402**
- 9 OBJECT-ORIENTED PROGRAMMING IN THE MICROSOFT  
FOUNDATION CLASSES 479**
- A ASCII TABLE 527**
- B SELECTED C++ FUNCTIONS AND METHODS 531**
- HINTS AND SOLUTIONS TO ODD-NUMBERED  
EXERCISES 573**
- INDEX 601**

# Contents

---

## Preface v

## 1 OBJECT-ORIENTED PROGRAMMING 1

- 1.1 Object-Oriented and Procedural Programming 2
  - Relationships 4
- 1.2 Classes and Abstract Data Types 5
  - Information Hiding 5
  - Encapsulation 6
  - Abstract Data Types 6
- 1.3 The Client/Server Model and Message Passing 9
  - The Client/Server Model 9
  - Message Passing and Method Invocation 10
- 1.4 Inheritance and Polymorphism 12
  - Inheritance 12
  - Polymorphism 13
  - Polymorphism and Recursion 14
- 1.5 Interfaces and Components 17
  - Component Technology 19

## 2 FROM C TO C++ 21

- 2.1 Namespaces 22
- 2.2 Introduction to C++ Input/Output 27
  - Manipulators 29
  - Mixing C and C++ Input/Output 34
- 2.3 Files 35
  - Testing Whether Files Are Open 37
- 2.4 C++ Features 38
  - Casts 38
    - `static_cast` 38
    - `const_cast` 39
    - `reinterpret_cast` 40
    - `dynamic_cast` 40
  - Constants 40
  - The Data Type `bool` 40
  - Enumerated Types 41
  - Defining Variables 42
  - Structures 43

- 2.5 The Type **string** 44
  - Defining **string** Variables 45
  - Conversion to C-Style Strings 45
  - String Length 45
  - Writing and Reading **strings** 45
  - Assignment 47
  - Concatenation 48
  - Modifying Strings 48
  - Extracting a Substring 51
  - Searching 51
  - Comparing Strings 53
- 2.6 Functions 56
  - Prototypes 56
  - The **main** Function 57
  - References 57
  - Call by Reference 58
  - Return by Reference 59
  - Inline Functions 62
  - Default Arguments 63
  - Overloading Functions 64
  - Function Signatures 65
- 2.7 The **new** and **delete** Operators 70
- 2.8 Exception Handling 73
  - C++ Postscript 78
  - Keywords 78
  - Unnamed Namespaces 78
  - Anonymous Unions 78
  - The Member Selector Operators 79
  - Common Programming Errors 83
  - Programming Exercises 93

## 3 CLASSES 98

- 3.1 Classes and Objects 99
  - Class Declarations 99
  - Information Hiding in C++ 100
  - The Member Selector Operator 101
  - Class Scope 103
  - The Difference between the Keywords **class** and **struct** 103
  - Defining Class Methods 104
  - Using Classes in a Program 106
- 3.2 Sample Application: A Stack Class 109
- 3.3 Efficiency and Robustness Issues for Classes and Objects 112
  - Passing and Returning Objects by Reference 113
  - Object References as **const** Parameters 113
  - const** Methods 114

- Overloading Methods to Handle Two Types of Strings 116
- 3.4 Sample Application: A Time Stamp Class 117
- 3.5 Constructors and the Destructor 124
  - Constructors 124
  - Arrays of Class Objects and the Default Constructor 127
  - Restricting Object Creation Through Constructors 127
  - The Copy Constructor 129
  - Defining a Copy Constructor 129
  - Disabling Passing and Returning by Value for Class Objects 135
  - Convert Constructors 136
  - The Convert Constructor and Implicit Type Conversion 137
  - Constructor Initializers 138
  - Constructors and the Operators `new` and `new[]` 139
  - The Destructor 139
- 3.6 Sample Application: A Task Class 145
- 3.7 Class Data Members and Methods 151
  - static** Variables Defined Inside Methods 155
- 3.8 Pointers to Objects 157
  - The Pointer Constant `this` 159
  - Common Programming Errors 161
  - Programming Exercises 169

## 4 INHERITANCE 175

- 4.1 Introduction 176
- 4.2 Basic Concepts and Syntax 178
  - private** Members in Inheritance 180
  - Adjusting Access 181
  - Name Hiding 182
  - Indirect Inheritance 183
- 4.3 Sample Application: Tracking Films 185
- 4.4 **protected** Members 190
- 4.5 Constructors and Destructors Under Inheritance 195
  - Constructors Under Inheritance 195
  - Derived Class Constructor Rules 198
  - Destructors Under Inheritance 202
- 4.6 Sample Application: A Sequence Hierarchy 205
- 4.7 Multiple Inheritance 216
  - Inheritance and Access 217
  - Virtual Base Classes 218
  - C++ Postscript 221
  - protected** Inheritance 221
  - private** Inheritance 222
  - Common Programming Errors 223
  - Programming Exercises 226

## 5 POLYMORPHISM 229

- 5.1 Run-Time versus Compile-Time Binding in C++ 230
  - Requirements for C++ Polymorphism 231
  - Inheriting **virtual** Methods 235
  - Run-Time Binding and the **Vtable** 236
  - Constructors and the Destructor 237
  - virtual** Destructors 237
  - Object Methods and Class Methods 240
- 5.2 Sample Application: Tracking Films Revisited 242
- 5.3 Name Overloading, Name Overriding, and Name Hiding 252
  - Name Overloading 252
  - Name Overriding 253
  - Name Hiding 255
  - Name Sharing in C++ Programming 257
- 5.4 Abstract Base Classes 260
  - Abstract Base Classes and Pure **virtual** Methods 260
  - Restrictions on Pure Functions 262
  - Uses of Abstract Base Classes 262
  - Microsoft's **Unknown** Interface 263
- 5.5 Run-Time Type Identification 265
  - The **dynamic\_cast** Operator 265
  - The Rules for **dynamic\_casts** 271
  - Summary of **dynamic\_cast** and **static\_cast** 272
  - The **typeid** Operator 272
  - Extending RTTI 274
  - C++ Postscript 275
  - Strong and Weak Polymorphism 275
  - Common Programming Errors 276
  - Programming Exercises 280

## 6 OPERATOR OVERLOADING 285

- 6.1 Basic Operator Overloading 286
  - Operator Precedence and Syntax 289
- 6.2 Sample Application: A Complex Number Class 291
- 6.3 Operator Overloading Using Top-Level Functions 296
- 6.4 **friend** Functions 302
- 6.5 Overloading the Input and Output Operators 305
- 6.6 Overloading the Assignment Operator 307
- 6.7 Overloading Some Special Operators 311
  - Overloading the Subscript Operator 312
  - Overloading the Function Call Operator 315
  - Overloading the Increment and Decrement Operators 318
  - Type Conversions 321
- 6.8 Sample Application: An Associative Array 324



	<b>basic ifstream</b>	434
	<b>basic fstream</b>	436
8.6	Sample Application: A Random Access File Class	438
8.7	The Character Stream Input/Output Classes	457
	<b>basic ostringstream</b>	457
	<b>basic istream</b>	458
	<b>basic stringstream</b>	460
8.8	Sample Application: A High-Level Copy Function	461
8.9	The Buffer Classes	463
	<b>basic streambuf</b>	463
	<b>basic filebuf</b>	465
	<b>basic stringbuf</b>	472
	C++ Postscript	474
	Common Programming Errors	475
	Programming Exercises	477

## **9 OBJECT-ORIENTED PROGRAMMING IN THE MICROSOFT FOUNDATION CLASSES 479**

9.1	Windows Programming in MFC	481
	Code Generators for MFC Programming	482
9.2	The Document/View Architecture in MFC	483
	Document Serialization	486
9.3	Sample Application: Document Serialization	489
9.4	The Common Object Model	503
	Changeable Servers and Unchangeable Interfaces	505
	The Interface Hierarchy	505
	The <b>IDispatch</b> Interface	506
	Types of COM Applications	507
	VC++ Support for COM	508
	COM and OLE	508
9.5	Sample Application: An Automation Server and Controller	510
	The Challenge of Reference Counting	522
	C++ Postscript	523
	Acronyms Used in Chapter 9	523
	Programming Exercises	524

## **APPENDICES**

<b>A</b>	<b>ASCII TABLE</b>	527
<b>B</b>	<b>SELECTED C++ FUNCTIONS AND METHODS</b>	531

## **HINTS AND SOLUTIONS TO ODD-NUMBERED EXERCISES 573**

## **INDEX 601**