

Bestseller Since 1986

Completely Rewritten for the New C++11 Standard



Fifth Edition

C++ Primer

Stanley B. Lippman
Josée Lajoie
Barbara E. Moo

C++ Primer, Fifth Edition

Stanley B. Lippman
Josée Lajoie
Barbara E. Moo

◆◆ Addison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sidney • Tokyo • Singapore • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U. S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the U. S., please contact:

International Sales
international@pearsoned.com

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data

Lippman, Stanley B.

C++ primer / Stanley B. Lippman, Josée Lajoie, Barbara E. Moo. – 5th ed.

p. cm.

Includes index.

ISBN 0-321-71411-3 (pbk. : alk. paper) 1. C++ (Computer program language) I.

Lajoie, Josée. II.

Moo, Barbara E. III. Title.

QA76.73.C153L57697 2013

005.13'3–dc23

2012020184

Copyright © 2013 Objectwrite Inc., Josée Lajoie and Barbara E. Moo

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to

Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle River, New Jersey 07458, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-321-71411-4

ISBN-10: 0-321-71411-3

Text printed in the United States on recycled paper at Courier in Westford, Massachusetts.

First printing, August 2012

To Beth, who makes this, and all things, possible.

—

To Daniel and Anna, who contain virtually all possibilities.

—SBL

To Mark and Mom, for their unconditional love and support.

—JL

To Andy, who taught me to program and so much more.

—BEM

Contents

Preface

Chapter 1 Getting Started

1.1 Writing a Simple C++ Program

1.1.1 Compiling and Executing Our Program

1.2 A First Look at Input/Output

1.3 A Word about Comments

1.4 Flow of Control

1.4.1 The while Statement

1.4.2 The for Statement

1.4.3 Reading an Unknown Number of Inputs

1.4.4 The if Statement

1.5 Introducing Classes

1.5.1 The Sales_item Class

1.5.2 A First Look at Member Functions

1.6 The Bookstore Program

Chapter Summary

Defined Terms

Part I The Basics

Chapter 2 Variables and Basic Types

2.1 Primitive Built-in Types

2.1.1 Arithmetic Types

2.1.2 Type Conversions

2.1.3 Literals

2.2 Variables

2.2.1 Variable Definitions

2.2.2 Variable Declarations and Definitions

2.2.3 Identifiers

[2.2.4 Scope of a Name](#)

[2.3 Compound Types](#)

[2.3.1 References](#)

[2.3.2 Pointers](#)

[2.3.3 Understanding Compound Type Declarations](#)

[2.4 const Qualifier](#)

[2.4.1 References to const](#)

[2.4.2 Pointers and const](#)

[2.4.3 Top-Level const](#)

[2.4.4 constexpr and Constant Expressions](#)

[2.5 Dealing with Types](#)

[2.5.1 Type Aliases](#)

[2.5.2 The auto Type Specifier](#)

[2.5.3 The decltype Type Specifier](#)

[2.6 Defining Our Own Data Structures](#)

[2.6.1 Defining the Sales_data Type](#)

[2.6.2 Using the Sales_data Class](#)

[2.6.3 Writing Our Own Header Files](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 3 Strings, Vectors, and Arrays](#)

[3.1 Namespace using Declarations](#)

[3.2 Library string Type](#)

[3.2.1 Defining and Initializing strings](#)

[3.2.2 Operations on strings](#)

[3.2.3 Dealing with the Characters in a string](#)

[3.3 Library vector Type](#)

[3.3.1 Defining and Initializing vectors](#)

[3.3.2 Adding Elements to a vector](#)

[3.3.3 Other vector Operations](#)

[3.4 Introducing Iterators](#)

[3.4.1 Using Iterators](#)

[3.4.2 Iterator Arithmetic](#)

[3.5 Arrays](#)

[3.5.1 Defining and Initializing Built-in Arrays](#)

[3.5.2 Accessing the Elements of an Array](#)

[3.5.3 Pointers and Arrays](#)

[3.5.4 C-Style Character Strings](#)

[3.5.5 Interfacing to Older Code](#)

[3.6 Multidimensional Arrays](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 4 Expressions](#)

[4.1 Fundamentals](#)

[4.1.1 Basic Concepts](#)

[4.1.2 Precedence and Associativity](#)

[4.1.3 Order of Evaluation](#)

[4.2 Arithmetic Operators](#)

[4.3 Logical and Relational Operators](#)

[4.4 Assignment Operators](#)

[4.5 Increment and Decrement Operators](#)

[4.6 The Member Access Operators](#)

[4.7 The Conditional Operator](#)

[4.8 The Bitwise Operators](#)

[4.9 The sizeof Operator](#)

[4.10 Comma Operator](#)

[4.11 Type Conversions](#)

[4.11.1 The Arithmetic Conversions](#)

[4.11.2 Other Implicit Conversions](#)

[4.11.3 Explicit Conversions](#)

[4.12 Operator Precedence Table](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 5 Statements](#)

[5.1 Simple Statements](#)

[5.2 Statement Scope](#)

[5.3 Conditional Statements](#)

[5.3.1 The if Statement](#)

[5.3.2 The switch Statement](#)

[5.4 Iterative Statements](#)

[5.4.1 The while Statement](#)

[5.4.2 Traditional for Statement](#)

[5.4.3 Range for Statement](#)

[5.4.4 The do while Statement](#)

[5.5 Jump Statements](#)

[5.5.1 The break Statement](#)

[5.5.2 The continue Statement](#)

[5.5.3 The goto Statement](#)

[5.6 try Blocks and Exception Handling](#)

[5.6.1 A throw Expression](#)

[5.6.2 The try Block](#)

[5.6.3 Standard Exceptions](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 6 Functions](#)

[6.1 Function Basics](#)

[6.1.1 Local Objects](#)

[6.1.2 Function Declarations](#)

[6.1.3 Separate Compilation](#)

[6.2 Argument Passing](#)

[6.2.1 Passing Arguments by Value](#)

[6.2.2 Passing Arguments by Reference](#)

[6.2.3 const Parameters and Arguments](#)
[6.2.4 Array Parameters](#)
[6.2.5 main: Handling Command-Line Options](#)
[6.2.6 Functions with Varying Parameters](#)
[6.3 Return Types and the return Statement](#)
[6.3.1 Functions with No Return Value](#)
[6.3.2 Functions That Return a Value](#)
[6.3.3 Returning a Pointer to an Array](#)
[6.4 Overloaded Functions](#)
[6.4.1 Overloading and Scope](#)
[6.5 Features for Specialized Uses](#)
[6.5.1 Default Arguments](#)
[6.5.2 Inline and constexpr Functions](#)
[6.5.3 Aids for Debugging](#)
[6.6 Function Matching](#)
[6.6.1 Argument Type Conversions](#)
[6.7 Pointers to Functions](#)
[Chapter Summary](#)
[Defined Terms](#)

[Chapter 7 Classes](#)

[7.1 Defining Abstract Data Types](#)
[7.1.1 Designing the Sales_data Class](#)
[7.1.2 Defining the Revised Sales_data Class](#)
[7.1.3 Defining Nonmember Class-Related Functions](#)
[7.1.4 Constructors](#)
[7.1.5 Copy, Assignment, and Destruction](#)
[7.2 Access Control and Encapsulation](#)
[7.2.1 Friends](#)
[7.3 Additional Class Features](#)
[7.3.1 Class Members Revisited](#)
[7.3.2 Functions That Return *this](#)

[7.3.3 Class Types](#)
[7.3.4 Friendship Revisited](#)
[7.4 Class Scope](#)
[7.4.1 Name Lookup and Class Scope](#)
[7.5 Constructors Revisited](#)
[7.5.1 Constructor Initializer List](#)
[7.5.2 Delegating Constructors](#)
[7.5.3 The Role of the Default Constructor](#)
[7.5.4 Implicit Class-Type Conversions](#)
[7.5.5 Aggregate Classes](#)
[7.5.6 Literal Classes](#)
[7.6 static Class Members](#)
[Chapter Summary](#)
[Defined Terms](#)

Part II The C++ Library

Chapter 8 The IO Library

[8.1 The IO Classes](#)
[8.1.1 No Copy or Assign for IO Objects](#)
[8.1.2 Condition States](#)
[8.1.3 Managing the Output Buffer](#)
[8.2 File Input and Output](#)
[8.2.1 Using File Stream Objects](#)
[8.2.2 File Modes](#)
[8.3 string Streams](#)
[8.3.1 Using an istream](#)
[8.3.2 Using ostream](#)
[Chapter Summary](#)
[Defined Terms](#)

Chapter 9 Sequential Containers

[9.1 Overview of the Sequential Containers](#)

[9.2 Container Library Overview](#)

[9.2.1 Iterators](#)

[9.2.2 Container Type Members](#)

[9.2.3 begin and end Members](#)

[9.2.4 Defining and Initializing a Container](#)

[9.2.5 Assignment and swap](#)

[9.2.6 Container Size Operations](#)

[9.2.7 Relational Operators](#)

[9.3 Sequential Container Operations](#)

[9.3.1 Adding Elements to a Sequential Container](#)

[9.3.2 Accessing Elements](#)

[9.3.3 Erasing Elements](#)

[9.3.4 Specialized forward_list Operations](#)

[9.3.5 Resizing a Container](#)

[9.3.6 Container Operations May Invalidate Iterators](#)

[9.4 How a vector Grows](#)

[9.5 Additional string Operations](#)

[9.5.1 Other Ways to Construct strings](#)

[9.5.2 Other Ways to Change a string](#)

[9.5.3 string Search Operations](#)

[9.5.4 The compare Functions](#)

[9.5.5 Numeric Conversions](#)

[9.6 Container Adaptors](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 10 Generic Algorithms](#)

[10.1 Overview](#)

[10.2 A First Look at the Algorithms](#)

[10.2.1 Read-Only Algorithms](#)

[10.2.2 Algorithms That Write Container Elements](#)

[10.2.3 Algorithms That Reorder Container Elements](#)

[10.3 Customizing Operations](#)
[10.3.1 Passing a Function to an Algorithm](#)
[10.3.2 Lambda Expressions](#)
[10.3.3 Lambda Captures and Returns](#)
[10.3.4 Binding Arguments](#)
[10.4 Revisiting Iterators](#)
[10.4.1 Insert Iterators](#)
[10.4.2 iostream Iterators](#)
[10.4.3 Reverse Iterators](#)
[10.5 Structure of Generic Algorithms](#)
[10.5.1 The Five Iterator Categories](#)
[10.5.2 Algorithm Parameter Patterns](#)
[10.5.3 Algorithm Naming Conventions](#)
[10.6 Container-Specific Algorithms](#)
[Chapter Summary](#)
[Defined Terms](#)

[Chapter 11 Associative Containers](#)

[11.1 Using an Associative Container](#)
[11.2 Overview of the Associative Containers](#)
[11.2.1 Defining an Associative Container](#)
[11.2.2 Requirements on Key Type](#)
[11.2.3 The pair Type](#)
[11.3 Operations on Associative Containers](#)
[11.3.1 Associative Container Iterators](#)
[11.3.2 Adding Elements](#)
[11.3.3 Erasing Elements](#)
[11.3.4 Subscripting a map](#)
[11.3.5 Accessing Elements](#)
[11.3.6 A Word Transformation Map](#)
[11.4 The Unordered Containers](#)
[Chapter Summary](#)

[Defined Terms](#)

[Chapter 12 Dynamic Memory](#)

[12.1 Dynamic Memory and Smart Pointers](#)

[12.1.1 The `shared_ptr` Class](#)

[12.1.2 Managing Memory Directly](#)

[12.1.3 Using `shared_ptr`s with `new`](#)

[12.1.4 Smart Pointers and Exceptions](#)

[12.1.5 `unique_ptr`](#)

[12.1.6 `weak_ptr`](#)

[12.2 Dynamic Arrays](#)

[12.2.1 `new` and Arrays](#)

[12.2.2 The `allocator` Class](#)

[12.3 Using the Library: A Text-Query Program](#)

[12.3.1 Design of the Query Program](#)

[12.3.2 Defining the Query Program Classes](#)

[Chapter Summary](#)

[Defined Terms](#)

[Part III Tools for Class Authors](#)

[Chapter 13 Copy Control](#)

[13.1 Copy, Assign, and Destroy](#)

[13.1.1 The Copy Constructor](#)

[13.1.2 The Copy-Assignment Operator](#)

[13.1.3 The Destructor](#)

[13.1.4 The Rule of Three/Five](#)

[13.1.5 Using `= default`](#)

[13.1.6 Preventing Copies](#)

[13.2 Copy Control and Resource Management](#)

[13.2.1 Classes That Act Like Values](#)

[13.2.2 Defining Classes That Act Like Pointers](#)

[13.3 Swap](#)

[13.4 A Copy-Control Example](#)

[13.5 Classes That Manage Dynamic Memory](#)

[13.6 Moving Objects](#)

[13.6.1 Rvalue References](#)

[13.6.2 Move Constructor and Move Assignment](#)

[13.6.3 Rvalue References and Member Functions](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 14 Overloaded Operations and Conversions](#)

[14.1 Basic Concepts](#)

[14.2 Input and Output Operators](#)

[14.2.1 Overloading the Output Operator <<](#)

[14.2.2 Overloading the Input Operator >>](#)

[14.3 Arithmetic and Relational Operators](#)

[14.3.1 Equality Operators](#)

[14.3.2 Relational Operators](#)

[14.4 Assignment Operators](#)

[14.5 Subscript Operator](#)

[14.6 Increment and Decrement Operators](#)

[14.7 Member Access Operators](#)

[14.8 Function-Call Operator](#)

[14.8.1 Lambdas Are Function Objects](#)

[14.8.2 Library-Defined Function Objects](#)

[14.8.3 Callable Objects and function](#)

[14.9 Overloading, Conversions, and Operators](#)

[14.9.1 Conversion Operators](#)

[14.9.2 Avoiding Ambiguous Conversions](#)

[14.9.3 Function Matching and Overloaded Operators](#)

[Chapter Summary](#)

[Defined Terms](#)

Chapter 15 Object-Oriented Programming

[15.1 OOP: An Overview](#)

[15.2 Defining Base and Derived Classes](#)

[15.2.1 Defining a Base Class](#)

[15.2.2 Defining a Derived Class](#)

[15.2.3 Conversions and Inheritance](#)

[15.3 Virtual Functions](#)

[15.4 Abstract Base Classes](#)

[15.5 Access Control and Inheritance](#)

[15.6 Class Scope under Inheritance](#)

[15.7 Constructors and Copy Control](#)

[15.7.1 Virtual Destructors](#)

[15.7.2 Synthesized Copy Control and Inheritance](#)

[15.7.3 Derived-Class Copy-Control Members](#)

[15.7.4 Inherited Constructors](#)

[15.8 Containers and Inheritance](#)

[15.8.1 Writing a Basket Class](#)

[15.9 Text Queries Revisited](#)

[15.9.1 An Object-Oriented Solution](#)

[15.9.2 The Query_base and Query Classes](#)

[15.9.3 The Derived Classes](#)

[15.9.4 The eval Functions](#)

[Chapter Summary](#)

[Defined Terms](#)

Chapter 16 Templates and Generic Programming

[16.1 Defining a Template](#)

[16.1.1 Function Templates](#)

[16.1.2 Class Templates](#)

[16.1.3 Template Parameters](#)

[16.1.4 Member Templates](#)

[16.1.5 Controlling Instantiations](#)

[16.1.6 Efficiency and Flexibility](#)
[16.2 Template Argument Deduction](#)
[16.2.1 Conversions and Template Type Parameters](#)
[16.2.2 Function-Template Explicit Arguments](#)
[16.2.3 Trailing Return Types and Type Transformation](#)
[16.2.4 Function Pointers and Argument Deduction](#)
[16.2.5 Template Argument Deduction and References](#)
[16.2.6 Understanding `std::move`](#)
[16.2.7 Forwarding](#)
[16.3 Overloading and Templates](#)
[16.4 Variadic Templates](#)
[16.4.1 Writing a Variadic Function Template](#)
[16.4.2 Pack Expansion](#)
[16.4.3 Forwarding Parameter Packs](#)
[16.5 Template Specializations](#)
[Chapter Summary](#)
[Defined Terms](#)

Part IV Advanced Topics

Chapter 17 Specialized Library Facilities

[17.1 The tuple Type](#)
[17.1.1 Defining and Initializing tuples](#)
[17.1.2 Using a tuple to Return Multiple Values](#)
[17.2 The bitset Type](#)
[17.2.1 Defining and Initializing bitsets](#)
[17.2.2 Operations on bitsets](#)
[17.3 Regular Expressions](#)
[17.3.1 Using the Regular Expression Library](#)
[17.3.2 The Match and Regex Iterator Types](#)
[17.3.3 Using Subexpressions](#)
[17.3.4 Using `regex_replace`](#)

[17.4 Random Numbers](#)

[17.4.1 Random-Number Engines and Distribution](#)

[17.4.2 Other Kinds of Distributions](#)

[17.5 The IO Library Revisited](#)

[17.5.1 Formatted Input and Output](#)

[17.5.2 Unformatted Input/Output Operations](#)

[17.5.3 Random Access to a Stream](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 18 Tools for Large Programs](#)

[18.1 Exception Handling](#)

[18.1.1 Throwing an Exception](#)

[18.1.2 Catching an Exception](#)

[18.1.3 Function try Blocks and Constructors](#)

[18.1.4 The noexcept Exception Specification](#)

[18.1.5 Exception Class Hierarchies](#)

[18.2 Namespaces](#)

[18.2.1 Namespace Definitions](#)

[18.2.2 Using Namespace Members](#)

[18.2.3 Classes, Namespaces, and Scope](#)

[18.2.4 Overloading and Namespaces](#)

[18.3 Multiple and Virtual Inheritance](#)

[18.3.1 Multiple Inheritance](#)

[18.3.2 Conversions and Multiple Base Classes](#)

[18.3.3 Class Scope under Multiple Inheritance](#)

[18.3.4 Virtual Inheritance](#)

[18.3.5 Constructors and Virtual Inheritance](#)

[Chapter Summary](#)

[Defined Terms](#)

[Chapter 19 Specialized Tools and Techniques](#)

[19.1 Controlling Memory Allocation](#)

[19.1.1 Overloading new and delete](#)
[19.1.2 Placement new Expressions](#)
[19.2 Run-Time Type Identification](#)
[19.2.1 The dynamic_cast Operator](#)
[19.2.2 The typeid Operator](#)
[19.2.3 Using RTTI](#)
[19.2.4 The type_info Class](#)
[19.3 Enumerations](#)
[19.4 Pointer to Class Member](#)
[19.4.1 Pointers to Data Members](#)
[19.4.2 Pointers to Member Functions](#)
[19.4.3 Using Member Functions as Callable Objects](#)
[19.5 Nested Classes](#)
[19.6 union: A Space-Saving Class](#)
[19.7 Local Classes](#)
[19.8 Inherently Nonportable Features](#)
[19.8.1 Bit-fields](#)
[19.8.2 volatile Qualifier](#)
[19.8.3 Linkage Directives: extern "C"](#)
[Chapter Summary](#)
[Defined Terms](#)

[Appendix A The Library](#)

[A.1 Library Names and Headers](#)
[A.2 A Brief Tour of the Algorithms](#)
[A.2.1 Algorithms to Find an Object](#)
[A.2.2 Other Read-Only Algorithms](#)
[A.2.3 Binary Search Algorithms](#)
[A.2.4 Algorithms That Write Container Elements](#)
[A.2.5 Partitioning and Sorting Algorithms](#)
[A.2.6 General Reordering Operations](#)
[A.2.7 Permutation Algorithms](#)

[A.2.8 Set Algorithms for Sorted Sequences](#)

[A.2.9 Minimum and Maximum Values](#)

[A.2.10 Numeric Algorithms](#)

[A.3 Random Numbers](#)

[A.3.1 Random Number Distributions](#)

[A.3.2 Random Number Engines](#)

[**Index**](#)