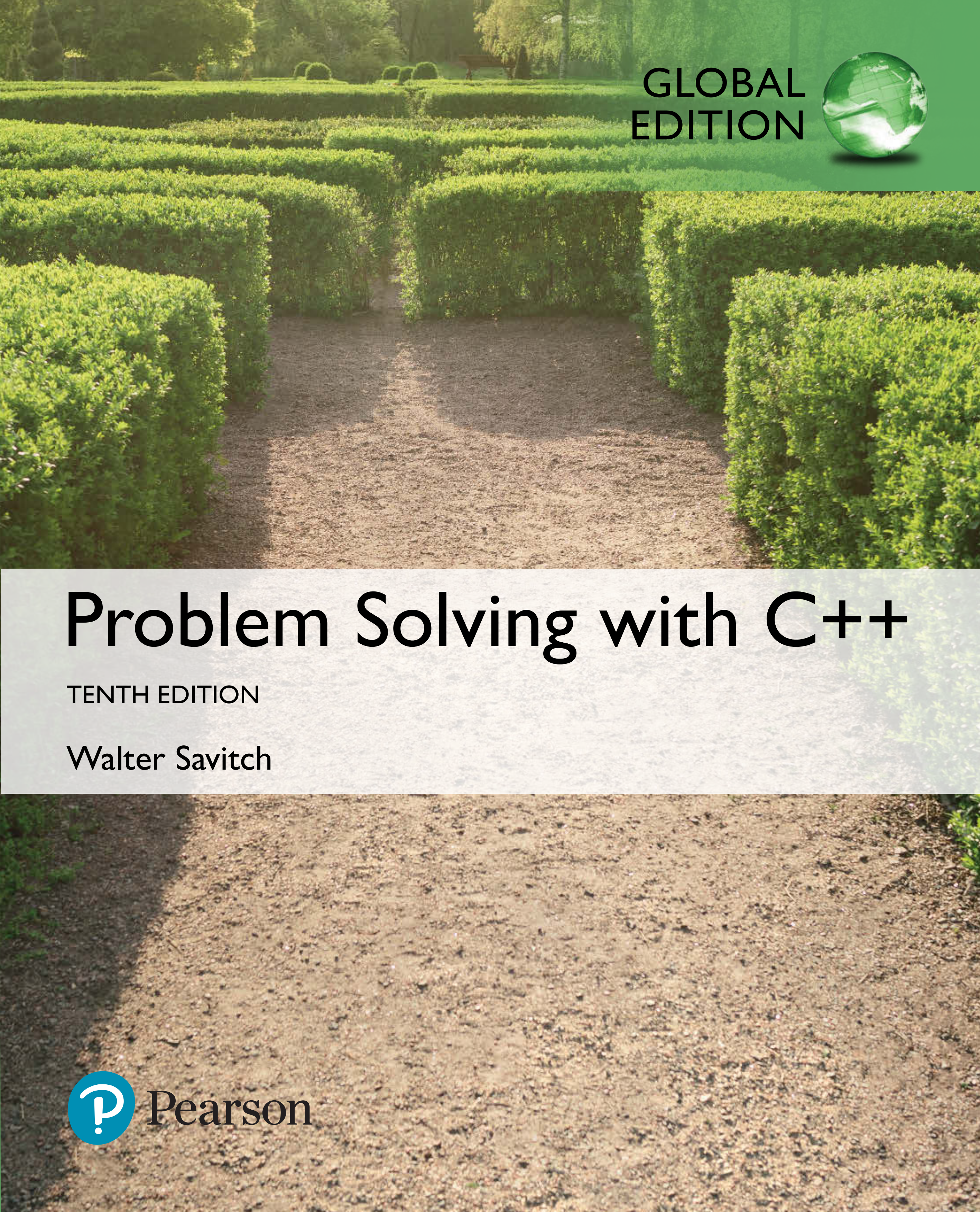

GLOBAL EDITION

Problem Solving with C++

TENTH EDITION

Walter Savitch

Pearson

Digital Resources for Students

Your new textbook provides 12-month access to digital resources that may include VideoNotes (step-by-step video tutorials on programming concepts), source code, web chapters, quizzes, and more. Refer to the preface in the textbook for a detailed list of resources.

Follow the instructions below to register for the Companion Website for Walter Savitch's *Problem Solving with C++,* Tenth Edition, Global Edition.

1. Go to www.pearsonglobaleditions.com/savitch.
2. Enter the title of your textbook or browse by author name.
3. Click Companion Website.
4. Click Register and follow the on-screen instructions to create a login name and password.

Use a coin to scratch off the coating and reveal your access code.
Do not use a sharp knife or other sharp object as it may damage the code.

Use the login name and password you created during registration to start using the online resources that accompany your textbook.

IMPORTANT:

This prepaid subscription does not include access to MyProgrammingLab, which is available at www.myprogramminglab.com for purchase.

This access code can only be used once. This subscription is valid for 12 months upon activation and is not transferable. If the access code has already been revealed it may no longer be valid.

For technical support go to https://support.pearson.com/getsupport/

Problem Solving with C++, Global Edition

Table of Contents

Cover

Inside Front Cover

Title Page

Copyright Page

Preface

Acknowledgments

Brief Contents

Contents

Chapter 1 Introduction to Computers and C++ Programming

- 1.1 COMPUTER SYSTEMS
  - Hardware
  - Software
  - High-Level Languages
  - Compilers
  - History Note
- 1.2 PROGRAMMING AND PROBLEM-SOLVING
  - Algorithms
  - Program Design
  - Object-Oriented Programming
  - The Software Life Cycle
- 1.3 INTRODUCTION TO C++
  - Origins of the C++ Language
  - A Sample C++ Program

Table of Contents

Pitfall: Using the Wrong Slash in

Programming Tip: Input and Output Syntax

Layout of a Simple C++ Program

Pitfall: Putting a Space Before the include File Name

Compiling and Running a C++ Program

Pitfall: Compiling a C++11 Program

Programming Tip: Getting Your Program to Run

1.4 TESTING AND DEBUGGING

Kinds of Program Errors

Pitfall: Assuming Your Program Is Correct

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 2 C++ Basics

2.1 VARIABLES AND ASSIGNMENTS

Variables

Names: Identifiers

Variable Declarations

Assignment Statements

Pitfall: Uninitialized Variables

Programming Tip: Use Meaningful Names

2.2 INPUT AND OUTPUT

Output Using cout

Include Directives and Namespaces

Escape Sequences

Programming Tip: End Each Program with a
or end1

Table of Contents

Formatting for Numbers with a Decimal Point
Input Using cin
Designing Input and Output
Programming Tip: Line Breaks in I/O

2.3 DATA TYPES AND EXPRESSIONS
The Types int and double
Other Number Types
C++11 Types
The Type char
The Type bool
Introduction to the Class string
Type Compatibilities
Arithmetic Operators and Expressions
Pitfall: Whole Numbers in Division
More Assignment Statements

2.4 SIMPLE FLOW OF CONTROL
A Simple Branching Mechanism
Pitfall: Strings of Inequalities
Pitfall: Using = in place of ==
Compound Statements
Simple Loop Mechanisms
Increment and Decrement Operators
Programming Example: Charge Card Balance
Pitfall: Infinite Loops

2.5 PROGRAM STYLE
Indenting
Comments
Naming Constants
Chapter Summary

Table of Contents

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 3 More Flow of Control

3.1 USING BOOLEAN EXPRESSIONS

Evaluating Boolean Expressions

Pitfall: Boolean Expressions Convert to int Values

Enumeration Types (Optional)

3.2 MULTIWAY BRANCHES

Nested Statements

Programming Tip: Use Braces in Nested Statements

Multiway if-else Statements

Programming Example: State Income Tax

The switch Statement

Pitfall: Forgetting a break in a switch Statement

Using switch Statements for Menus

Blocks

Pitfall: Inadvertent Local Variables

3.3 MORE ABOUT C++ LOOP STATEMENTS

The while Statements Reviewed

Increment and Decrement Operators Revisited

The for Statement

Pitfall: Extra Semicolon in a for Statement

What Kind of Loop to Use

Pitfall: Uninitialized Variables and Infinite Loops

The break Statement

Pitfall: The break Statement in Nested Loops

3.4 DESIGNING LOOPS

Loops for Sums and Products

Table of Contents

Ending a Loop
Nested Loops
Debugging Loops
Chapter Summary
Answers to Self-Test Exercises
Practice Programs
Programming Projects
Chapter 4 Procedural Abstraction and Functions That Return a Value
4.1 TOP-DOWN DESIGN
4.2 PREDEFINED FUNCTIONS
Using Predefined Functions
Random Number Generation
Type Casting
Older Form of Type Casting
Pitfall: Integer Division Drops the Fractional Part
4.3 PROGRAMMER-DEFINED FUNCTIONS
Function Definitions
Functions That Return a Boolean Value
Alternate Form for Function Declarations
Pitfall: Arguments in the Wrong Order
Function DefinitionSyntax Summary
More About Placement of Function Definitions
Programming Tip: Use Function Calls in Branching Statements
4.4 PROCEDURAL ABSTRACTION
The Black-Box Analogy
Programming Tip: Choosing Formal Parameter Names
Programming Tip: Nested Loops
Case Study: Buying Pizza
Programming Tip: Use Pseudocode

Table of Contents

4.5 SCOPE AND LOCAL VARIABLES

The Small Program Analogy

Programming Example: Experimental Pea Patch

Global Constants and Global Variables

Call-by-Value Formal Parameters Are Local Variables

Block Scope

Namespaces Revisited

Programming Example: The Factorial Function

4.6 OVERLOADING FUNCTION NAMES

Introduction to Overloading

Programming Example: Revised Pizza-Buying Program

Automatic Type Conversion

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 5 Functions for All Subtasks

5.1 VOID FUNCTIONS

Definitions of void Functions

Programming Example: Converting Temperatures

return Statements in void Functions

5.2 CALL-BY-REFERENCE PARAMETERS

A First View of Call-by-Reference

Call-by-Reference in Detail

Programming Example: The swapValues Function

Mixed Parameter Lists

Programming Tip: What Kind of Parameter to Use

Pitfall: Inadvertent Local Variables

5.3 USING PROCEDURAL ABSTRACTION

Table of Contents

Functions Calling Functions

Preconditions and Postconditions

Case Study: Supermarket Pricing

5.4 TESTING AND DEBUGGING FUNCTIONS

Stubs and Drivers

5.5 GENERAL DEBUGGING TECHNIQUES

Keep an Open Mind

Check Common Errors

Localize the Error

The assert Macro

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 6 I/O Streams as an Introduction to Objects and Classes

6.1 STREAMS AND BASIC FILE I/O

Why Use Files for I/O?

File I/O

Introduction to Classes and Objects

Programming Tip: Check Whether a File Was Opened Successfully

Techniques for File I/O

Appending to a File (Optional)

File Names as Input (Optional)

6.2 TOOLS FOR STREAM I/O

Formatting Output with Stream Functions

Manipulators

Streams as Arguments to Functions

Programming Tip: Checking for the End of a File

A Note on Namespaces

Table of Contents

Programming Example: Cleaning Up a File Format

6.3 CHARACTER I/O

The Member Functions get and put

The putback Member Function (Optional)

Programming Example: Checking Input

Pitfall: Unexpected '
' in Input

Programming Example: Another newLine Function

Default Arguments for Functions (Optional)

The eof Member Function

Programming Example: Editing a Text File

Predefined Character Functions

Pitfall: toupper and tolower Return Values

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 7 Arrays

7.1 INTRODUCTION TO ARRAYS

Declaring and Referencing Arrays

Programming Tip: Use for Loops with Arrays

Pitfall: Array Indexes Always Start with Zero

Programming Tip: Use a Defined Constant for the Size of an Array

Arrays in Memory

Pitfall: Array Index Out of Range

Initializing Arrays

Programming Tip: C++11 Range-Based for Statement

7.2 ARRAYS IN FUNCTIONS

Indexed Variables as Function Arguments

Table of Contents

Entire Arrays as Function Arguments

The const Parameter Modifier

Pitfall: Inconsistent Use of const Parameters

Functions That Return an Array

Case Study: Production Graph

7.3 PROGRAMMING WITH ARRAYS

Partially Filled Arrays

Programming Tip: Do Not Skimp on Formal Parameters

Programming Example: Searching an Array

Programming Example: Sorting an Array

Programming Example: Bubble Sort

7.4 MULTIDIMENSIONAL ARRAYS

Multidimensional Array Basics

Multidimensional Array Parameters

Programming Example: Two-Dimensional Grading Program

Pitfall: Using Commas Between Array Indexes

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 8 Strings and Vectors

8.1 AN ARRAY TYPE FOR STRINGS

C-String Values and C-String Variables

Pitfall: Using = and == with C Strings

Other Functions in <cstring>

Pitfall: Copying past the end of a C-string using strcpy

C-String Input and Output

C-String-to-Number Conversions and Robust Input

8.2 THE STANDARD STRING CLASS

Table of Contents

Introduction to the Standard Class string

I/O with the Class string

Programming Tip: More Versions of getline

Pitfall: Mixing cin >> variable; and getline

String Processing with the Class string

Programming Example: Palindrome Testing

Converting between string Objects and C Strings

Converting Between Strings and Numbers

8.3 VECTORS

Vector Basics

Pitfall: Using Square Brackets Beyond the Vector Size

Programming Tip: Vector Assignment Is Well Behaved

Efficiency Issues

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 9 Pointers and Dynamic Arrays

9.1 POINTERS

Pointer Variables

Basic Memory Management

Pitfall: Dangling Pointers

Static Variables and Automatic Variables

Programming Tip: Define Pointer Types

9.2 DYNAMIC ARRAYS

Array Variables and Pointer Variables

Creating and Using Dynamic Arrays

Pointer Arithmetic (Optional)

Multidimensional Dynamic Arrays (Optional)

Table of Contents

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 10 Defining Classes

10.1 STRUCTURES

Structures for Diverse Data

Pitfall: Forgetting a Semicolon in a Structure Definition

Structures as Function Arguments

Programming Tip: Use Hierarchical Structures

Initializing Structures

10.2 CLASSES

Defining Classes and Member Functions

Public and Private Members

Programming Tip: Make All Member Variables Private

Programming Tip: Define Accessor and Mutator Functions

Programming Tip: Use the Assignment Operator with Objects

Programming Example: BankAccount ClassVersion 1

Summary of Some Properties of Classes

Constructors for Initialization

Programming Tip: Always Include a Default Constructor

Pitfall: Constructors with No Arguments

Member Initializers and Constructor Delegation in C++11

10.3 ABSTRACT DATA TYPES

Classes to Produce Abstract Data Types

Programming Example: Alternative Implementation of a Class

10.4 INTRODUCTION TO INHERITANCE

Derived Classes

Defining Derived Classes

Table of Contents

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 11 Friends, Overloaded Operators, and Arrays in Classes

11.1 FRIEND FUNCTIONS

Programming Example: An Equality Function

Friend Functions

Programming Tip: Define Both Accessor Functions and Friend Functions

Programming Tip: Use Both Member and Nonmember Functions

Programming Example: Money Class (Version 1)

Implementation of digitToInt (Optional)

Pitfall: Leading Zeros in Number Constants

The const Parameter Modifier

Pitfall: Inconsistent Use of const

11.2 OVERLOADING OPERATORS

Overloading Operators

Constructors for Automatic Type Conversion

Overloading Unary Operators

Overloading >> and <<

11.3 ARRAYS AND CLASSES

Arrays of Classes

Arrays as Class Members

Programming Example: A Class for a Partially Filled Array

11.4 CLASSES AND DYNAMIC ARRAYS

Programming Example: A String Variable Class

Destructors

Pitfall: Pointers as Call-by-Value Parameters

Table of Contents

Copy Constructors

Overloading the Assignment Operator

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 12 Separate Compilation and Namespaces

12.1 SEPARATE COMPILATION

ADTs Reviewed

Case Study: DigitalTimeA Class Compiled Separately

Using #ifndef

Programming Tip: Defining Other Libraries

12.2 NAMESPACES

Namespaces and using Directives

Creating a Namespace

Qualifying Names

A Subtle Point About Namespaces (Optional)

Unnamed Namespaces

Programming Tip: Choosing a Name for a Namespace

Pitfall: Confusing the Global Namespace and the Unnamed Namespace

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 13 Pointers and Linked Lists

13.1 NODES AND LINKED LISTS

Nodes

nullptr

Linked Lists

Table of Contents

Inserting a Node at the Head of a List

Pitfall: Losing Nodes

Searching a Linked List

Pointers as Iterators

Inserting and Removing Nodes Inside a List

Pitfall: Using the Assignment Operator with Dynamic Data Structures

Variations on Linked Lists

Linked Lists of Classes

13.2 STACKS AND QUEUES

Stacks

Programming Examples: A Stack Class

Queues

Programming Examples: A Queue Class

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 14 Recursion

14.1 RECURSIVE FUNCTIONS FOR TASKS

Case Study: Vertical Numbers

A Closer Look at Recursion

Pitfall: Infinite Recursion

Stacks for Recursion

Pitfall: Stack Overflow

Recursion Versus Iteration

14.2 RECURSIVE FUNCTIONS FOR VALUES

General Form for a Recursive Function That Returns a Value

Programming Example: Another Powers Function

14.3 THINKING RECURSIVELY

Table of Contents

Recursive Design Techniques

Case Study: Binary SearchAn Example of Recursive Thinking

Programming Example: A Recursive Member Function

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 15 Inheritance

15.1 INHERITANCE BASICS

Derived Classes

Constructors in Derived Classes

Pitfall: Use of Private Member Variables from the Base Class

Pitfall: Private Member Functions Are Effectively Not Inherited

The protected Qualifier

Redefinition of Member Functions

Redefining Versus Overloading

Access to a Redefined Base Function

15.2 INHERITANCE DETAILS

Functions That Are Not Inherited

Assignment Operators and Copy Constructors in Derived Classes

Destructors in Derived Classes

15.3 POLYMORPHISM

Late Binding

Virtual Functions in C++

Virtual Functions and Extended Type Compatibility

Pitfall: The Slicing Problem

Pitfall: Not Using Virtual Member Functions

Pitfall: Attempting to Compile Class Definitions Without Definitions for Every Virtual Member Function

Table of Contents

Programming Tip: Make Destructors Virtual

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 16 Exception Handling

16.1 EXCEPTION-HANDLING BASICS

A Toy Example of Exception Handling

Defining Your Own Exception Classes

Multiple Throws and Catches

Pitfall: Catch the More Specific Exception First

Programming Tip: Exception Classes Can Be Trivial

Throwing an Exception in a Function

Exception Specification

Pitfall: Exception Specification in Derived Classes

16.2 PROGRAMMING TECHNIQUES FOR EXCEPTION HANDLING

When to Throw an Exception

Pitfall: Uncaught Exceptions

Pitfall: Nested try-catch Blocks

Pitfall: Overuse of Exceptions

Exception Class Hierarchies

Testing for Available Memory

Rethrowing an Exception

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 17 Templates

17.1 TEMPLATES FOR ALGORITHM ABSTRACTION

Table of Contents

Templates for Functions

Pitfall: Compiler Complications

Programming Example: A Generic Sorting Function

Programming Tip: How to Define Templates

Pitfall: Using a Template with an Inappropriate Type

17.2 TEMPLATES FOR DATA ABSTRACTION

Syntax for Class Templates

Programming Example: An Array Class

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

Chapter 18 Standard Template Library and C++11

18.1 ITERATORS

using Declarations

Iterator Basics

Programming Tip: Use auto to Simplify Variable Declarations

Pitfall: Compiler Problems

Kinds of Iterators

Constant and Mutable Iterators

Reverse Iterators

Other Kinds of Iterators

18.2 CONTAINERS

Sequential Containers

Pitfall: Iterators and Removing Elements

Programming Tip: Type Definitions in Containers

Container Adapters stack and queue

Associative Containers set and map

Programming Tip: Use Initialization, Ranged for, and auto with Containers

Table of Contents

Efficiency

18.3 GENERIC ALGORITHMS

Running Times and Big-O Notation

Container Access Running Times

Nonmodifying Sequence Algorithms

Container Modifying Algorithms

Set Algorithms

Sorting Algorithms

18.4 C++ IS EVOLVING

std::array

Regular Expressions

Threads

Smart Pointers

Chapter Summary

Answers to Self-Test Exercises

Practice Programs

Programming Projects

APPENDICES

1 C++ Keywords

2 Precedence of Operators

3 The ASCII Character Set

4 Some Library Functions

5 Inline Functions

6 Overloading the Array Index Square Brackets

7 The this Pointer

8 Overloading Operators as Member Operators

CREDITS

INDEX

Table of Contents