

# Contents

---

## **Chapter 1 Introduction to Computers and C++ Programming 31**

### **1.1 COMPUTER SYSTEMS 32**

Hardware 32

Software 37

High-Level Languages 38

Compilers 39

*History Note* 42

### **1.2 PROGRAMMING AND PROBLEM-SOLVING 42**

Algorithms 42

Program Design 45

Object-Oriented Programming 46

The Software Life Cycle 47

### **1.3 INTRODUCTION TO C++ 48**

Origins of the C++ Language 48

A Sample C++ Program 49

*Pitfall:* Using the Wrong Slash in `\n` 53

*Programming Tip:* Input and Output Syntax 53

Layout of a Simple C++ Program 54

*Pitfall:* Putting a Space Before the include File Name 56

Compiling and Running a C++ Program 56

*Pitfall:* Compiling a C++11 program 57

*Programming Tip:* Getting Your Program to Run 57

### **1.4 TESTING AND DEBUGGING 59**

Kinds of Program Errors 60

*Pitfall:* Assuming Your Program Is Correct 61

Chapter Summary	62
Answers to Self-Test Exercises	63
Practice Programs	65
Programming Projects	66

## **Chapter 2 C++ Basics 69**

### **2.1 VARIABLES AND ASSIGNMENTS 70**

Variables	70
Names: Identifiers	72
Variable Declarations	74
Assignment Statements	75
<i>Pitfall: Uninitialized Variables</i>	77
<i>Programming Tip: Use Meaningful Names</i>	79

### **2.2 INPUT AND OUTPUT 80**

Output Using <code>cout</code>	80
Include Directives and Namespaces	82
Escape Sequences	83
<i>Programming Tip: End Each Program with a <code>\n</code> or <code>endl</code></i>	85
Formatting for Numbers with a Decimal Point	85
Input Using <code>cin</code>	86
Designing Input and Output	88
<i>Programming Tip: Line Breaks in I/O</i>	88

### **2.3 DATA TYPES AND EXPRESSIONS 90**

The Types <code>int</code> and <code>double</code>	90
Other Number Types	92
C++11 Types	93
The Type <code>char</code>	94
The Type <code>bool</code>	96
Introduction to the Class <code>string</code>	96
Type Compatibilities	98
Arithmetic Operators and Expressions	99
<i>Pitfall: Whole Numbers in Division</i>	102
More Assignment Statements	104

### **2.4 SIMPLE FLOW OF CONTROL 104**

A Simple Branching Mechanism	105
<i>Pitfall: Strings of Inequalities</i>	110
<i>Pitfall: Using <code>=</code> in place of <code>==</code></i>	111
Compound Statements	112

Simple Loop Mechanisms 114  
Increment and Decrement Operators 117  
*Programming Example: Charge Card Balance* 119  
*Pitfall: Infinite Loops* 120

## **2.5 PROGRAM STYLE 123**

Indenting 123  
Comments 123  
Naming Constants 125  
  
Chapter Summary 128  
Answers to Self-Test Exercises 128  
Practice Programs 133  
Programming Projects 135

# **Chapter 3 More Flow of Control 141**

## **3.1 USING BOOLEAN EXPRESSIONS 142**

Evaluating Boolean Expressions 142  
*Pitfall: Boolean Expressions Convert to int Values* 146  
Enumeration Types (*Optional*) 149

## **3.2 MULTIWAY BRANCHES 150**

Nested Statements 150  
*Programming Tip: Use Braces in Nested Statements* 151  
Multiway *if-else* Statements 153  
*Programming Example: State Income Tax* 155  
The *switch* Statement 158  
*Pitfall: Forgetting a break in a switch Statement* 162  
Using *switch* Statements for Menus 163  
Blocks 165  
*Pitfall: Inadvertent Local Variables* 168

## **3.3 MORE ABOUT C++ LOOP STATEMENTS 169**

The *while* Statements Reviewed 169  
Increment and Decrement Operators Revisited 171  
The *for* Statement 174  
*Pitfall: Extra Semicolon in a for Statement* 179  
What Kind of Loop to Use 180  
*Pitfall: Uninitialized Variables and Infinite Loops* 182  
The *break* Statement 183  
*Pitfall: The break Statement in Nested Loops* 184

**3.4 DESIGNING LOOPS 185**

Loops for Sums and Products 185

Ending a Loop 187

Nested Loops 190

Debugging Loops 192

Chapter Summary 195

Answers to Self-Test Exercises 196

Practice Programs 202

Programming Projects 204

**Chapter 4 Procedural Abstraction and Functions  
That Return a Value 211****4.1 TOP-DOWN DESIGN 212****4.2 PREDEFINED FUNCTIONS 213**

Using Predefined Functions 213

Random Number Generation 218

Type Casting 220

Older Form of Type Casting 222

*Pitfall:* Integer Division Drops the Fractional Part 222**4.3 PROGRAMMER-DEFINED FUNCTIONS 223**

Function Definitions 223

Functions That Return a Boolean Value 229

Alternate Form for Function Declarations 229

*Pitfall:* Arguments in the Wrong Order 230

Function Definition–Syntax Summary 231

More About Placement of Function Definitions 232

*Programming Tip:* Use Function Calls in Branching Statements 233**4.4 PROCEDURAL ABSTRACTION 234**

The Black-Box Analogy 234

*Programming Tip:* Choosing Formal Parameter Names 237*Programming Tip:* Nested Loops 238*Case Study:* Buying Pizza 241*Programming Tip:* Use Pseudocode 247**4.5 SCOPE AND LOCAL VARIABLES 248**

The Small Program Analogy 248

*Programming Example:* Experimental Pea Patch 251

Global Constants and Global Variables 251  
Call-by-Value Formal Parameters Are Local Variables 254  
Block Scope 256  
Namespaces Revisited 257  
*Programming Example: The Factorial Function* 260

#### **4.6 OVERLOADING FUNCTION NAMES 262**

Introduction to Overloading 262  
*Programming Example: Revised Pizza-Buying Program* 265  
Automatic Type Conversion 268  
Chapter Summary 270  
Answers to Self-Test Exercises 270  
Practice Programs 275  
Programming Projects 277

### **Chapter 5 Functions for All Subtasks 281**

#### **5.1 void FUNCTIONS 282**

Definitions of *void* Functions 282  
*Programming Example: Converting Temperatures* 285  
*return* Statements in *void* Functions 285

#### **5.2 CALL-BY-REFERENCE PARAMETERS 289**

A First View of Call-by-Reference 289  
Call-by-Reference in Detail 292  
*Programming Example: The swap\_values Function* 297  
Mixed Parameter Lists 298  
*Programming Tip: What Kind of Parameter to Use* 299  
*Pitfall: Inadvertent Local Variables* 300

#### **5.3 USING PROCEDURAL ABSTRACTION 303**

Functions Calling Functions 303  
Preconditions and Postconditions 305  
*Case Study: Supermarket Pricing* 306

#### **5.4 TESTING AND DEBUGGING FUNCTIONS 311**

Stubs and Drivers 312

#### **5.5 GENERAL DEBUGGING TECHNIQUES 317**

Keep an Open Mind 317  
Check Common Errors 317

Localize the Error	318
The assert Macro	320
Chapter Summary	322
Answers to Self-Test Exercises	323
Practice Programs	326
Programming Projects	329

## **Chapter 6 I/O Streams as an Introduction to Objects and Classes 335**

### **6.1 STREAMS AND BASIC FILE I/O 336**

Why Use Files for I/O?	337
File I/O	338
Introduction to Classes and Objects	342
<i>Programming Tip: Check Whether a File Was Opened Successfully</i>	344
Techniques for File I/O	346
Appending to a File ( <i>Optional</i> )	350
File Names as Input ( <i>Optional</i> )	351

### **6.2 TOOLS FOR STREAM I/O 353**

Formatting Output with Stream Functions	353
Manipulators	359
Streams as Arguments to Functions	362
<i>Programming Tip: Checking for the End of a File</i>	362
A Note on Namespaces	365
<i>Programming Example: Cleaning Up a File Format</i>	366

### **6.3 CHARACTER I/O 368**

The Member Functions get and put	368
The putback Member Function ( <i>Optional</i> )	372
<i>Programming Example: Checking Input</i>	373
<i>Pitfall: Unexpected '\n' in Input</i>	375
<i>Programming Example: Another new_line Function</i>	377
Default Arguments for Functions ( <i>Optional</i> )	378
The eof Member Function	383
<i>Programming Example: Editing a Text File</i>	385
Predefined Character Functions	386
<i>Pitfall: toupper and tolower Return Values</i>	388

Chapter Summary	390
Answers to Self-Test Exercises	391
Practice Programs	398
Programming Projects	400

## **Chapter 7 Arrays 407**

### **7.1 INTRODUCTION TO ARRAYS 408**

Declaring and Referencing Arrays	408
<i>Programming Tip: Use for Loops with Arrays</i>	410
<i>Pitfall: Array Indexes Always Start with Zero</i>	410
<i>Programming Tip: Use a Defined Constant for the Size of an Array</i>	410
Arrays in Memory	412
<i>Pitfall: Array Index Out of Range</i>	413
Initializing Arrays	416
<i>Programming Tip: C++11 Range-Based for Statement</i>	416

### **7.2 ARRAYS IN FUNCTIONS 419**

Indexed Variables as Function Arguments	419
Entire Arrays as Function Arguments	421
The <i>const</i> Parameter Modifier	424
<i>Pitfall: Inconsistent Use of const Parameters</i>	427
Functions That Return an Array	427
<i>Case Study: Production Graph</i>	428

### **7.3 PROGRAMMING WITH ARRAYS 441**

Partially Filled Arrays	441
<i>Programming Tip: Do Not Skimp on Formal Parameters</i>	444
<i>Programming Example: Searching an Array</i>	444
<i>Programming Example: Sorting an Array</i>	447
<i>Programming Example: Bubble Sort</i>	451

### **7.4 MULTIDIMENSIONAL ARRAYS 454**

Multidimensional Array Basics	455
Multidimensional Array Parameters	455
<i>Programming Example: Two-Dimensional Grading Program</i>	457
<i>Pitfall: Using Commas Between Array Indexes</i>	461

Chapter Summary	462
Answers to Self-Test Exercises	463
Practice Programs	467
Programming Projects	469

## **Chapter 8 Strings and Vectors 481**

### **8.1 AN ARRAY TYPE FOR STRINGS 483**

C-String Values and C-String Variables	483
<i>Pitfall:</i> Using = and == with C Strings	486
Other Functions in <code>&lt;cstring&gt;</code>	488
<i>Pitfall:</i> Copying past the end of a C-string using <code>strcpy</code>	491
C-String Input and Output	494
C-String-to-Number Conversions and Robust Input	496

### **8.2 THE STANDARD string CLASS 502**

Introduction to the Standard Class <code>string</code>	502
I/O with the Class <code>string</code>	505
<i>Programming Tip:</i> More Versions of <code>getline</code>	508
<i>Pitfall:</i> Mixing <code>cin &gt;&gt; variable;</code> and <code>getline</code>	509
String Processing with the Class <code>string</code>	510
<i>Programming Example:</i> Palindrome Testing	514
Converting Between <code>string</code> Objects and C Strings	517
Converting Between Strings and Numbers	518

### **8.3 VECTORS 519**

Vector Basics	519
<i>Pitfall:</i> Using Square Brackets Beyond the Vector Size	522
<i>Programming Tip:</i> Vector Assignment Is Well Behaved	523
Efficiency Issues	523

Chapter Summary	525
Answers to Self-Test Exercises	525
Practice Programs	527
Programming Projects	528

## **Chapter 9 Pointers and Dynamic Arrays 537**

### **9.1 POINTERS 538**

Pointer Variables	539
Basic Memory Management	546



*Pitfall: Dangling Pointers* 547  
Static Variables and Automatic Variables 548  
*Programming Tip: Define Pointer Types* 548

## **9.2 DYNAMIC ARRAYS 551**

Array Variables and Pointer Variables 551  
Creating and Using Dynamic Arrays 552  
Pointer Arithmetic (*Optional*) 558  
Multidimensional Dynamic Arrays (*Optional*) 560  
Chapter Summary 562  
Answers to Self-Test Exercises 562  
Practice Programs 563  
Programming Projects 564

# **Chapter 10 Defining Classes 571**

## **10.1 STRUCTURES 572**

Structures for Diverse Data 572  
*Pitfall: Forgetting a Semicolon in a Structure Definition* 577  
Structures as Function Arguments 578  
*Programming Tip: Use Hierarchical Structures* 579  
Initializing Structures 581

## **10.2 CLASSES 584**

Defining Classes and Member Functions 584  
Public and Private Members 589  
*Programming Tip: Make All Member Variables Private* 597  
*Programming Tip: Define Accessor and Mutator Functions* 597  
*Programming Tip: Use the Assignment Operator with Objects* 599  
*Programming Example: BankAccount Class—Version 1* 600  
Summary of Some Properties of Classes 604  
Constructors for Initialization 606  
*Programming Tip: Always Include a Default Constructor* 614  
*Pitfall: Constructors with No Arguments* 615  
Member Initializers and Constructor Delegation in C++11 617

## **10.3 ABSTRACT DATA TYPES 618**

Classes to Produce Abstract Data Types 619  
*Programming Example: Alternative Implementation of a Class* 623

**10.4 INTRODUCTION TO INHERITANCE 628**

Derived Classes 629

Defining Derived Classes 630

Chapter Summary 634

Answers to Self-Test Exercises 635

Practice Programs 641

Programming Projects 642

**Chapter 11 Friends, Overloaded Operators, and  
Arrays in Classes 649****11.1 FRIEND FUNCTIONS 650***Programming Example: An Equality Function* 650

Friend Functions 654

*Programming Tip: Define Both Accessor Functions and Friend  
Functions* 656*Programming Tip: Use Both Member and Nonmember  
Functions* 658*Programming Example: Money Class (Version 1)* 658Implementation of `digit_to_int` (*Optional*) 665*Pitfall: Leading Zeros in Number Constants* 666The `const` Parameter Modifier 668*Pitfall: Inconsistent Use of `const`* 669**11.2 OVERLOADING OPERATORS 673**

Overloading Operators 674

Constructors for Automatic Type Conversion 677

Overloading Unary Operators 679

Overloading `>>` and `<<` 680**11.3 ARRAYS AND CLASSES 690**

Arrays of Classes 690

Arrays as Class Members 694

*Programming Example: A Class for a Partially Filled Array* 695**11.4 CLASSES AND DYNAMIC ARRAYS 697***Programming Example: A String Variable Class* 698

Destructors 701

*Pitfall: Pointers as Call-by-Value Parameters* 704

Copy Constructors	705
Overloading the Assignment Operator	710
Chapter Summary	713
Answers to Self-Test Exercises	713
Practice Programs	723
Programming Projects	724

## **Chapter 12 Separate Compilation and Namespaces 733**

### **12.1 SEPARATE COMPILATION 734**

ADTs Reviewed	735
<i>Case Study: DigitalTime</i> —A Class Compiled Separately	736
Using <code>#ifndef</code>	745
<i>Programming Tip: Defining Other Libraries</i>	748

### **12.2 NAMESPACES 749**

Namespaces and <i>using</i> Directives	749
Creating a Namespace	751
Qualifying Names	754
A Subtle Point About Namespaces ( <i>Optional</i> )	755
Unnamed Namespaces	756
<i>Programming Tip: Choosing a Name for a Namespace</i>	761
<i>Pitfall: Confusing the Global Namespace and the Unnamed Namespace</i>	762

Chapter Summary	763
Answers to Self-Test Exercises	764
Practice Programs	766
Programming Projects	768

## **Chapter 13 Pointers and Linked Lists 769**

### **13.1 NODES AND LINKED LISTS 770**

Nodes	770
<code>nullptr</code>	775
Linked Lists	776
Inserting a Node at the Head of a List	777
<i>Pitfall: Losing Nodes</i>	780
Searching a Linked List	781

Pointers as Iterators 785  
Inserting and Removing Nodes Inside a List 785  
*Pitfall: Using the Assignment Operator with Dynamic Data Structures* 787  
Variations on Linked Lists 790  
Linked Lists of Classes 792

### **13.2 STACKS AND QUEUES 795**

Stacks 795  
*Programming Example: A Stack Class* 796  
Queues 801  
*Programming Example: A Queue Class* 802  
Chapter Summary 806  
Answers to Self-Test Exercises 806  
Practice Programs 809  
Programming Projects 810

## **Chapter 14 Recursion 819**

### **14.1 RECURSIVE FUNCTIONS FOR TASKS 821**

*Case Study: Vertical Numbers* 821  
A Closer Look at Recursion 827  
*Pitfall: Infinite Recursion* 829  
Stacks for Recursion 830  
*Pitfall: Stack Overflow* 832  
Recursion Versus Iteration 832

### **14.2 RECURSIVE FUNCTIONS FOR VALUES 834**

General Form for a Recursive Function That Returns a Value 834  
*Programming Example: Another Powers Function* 834

### **14.3 THINKING RECURSIVELY 839**

Recursive Design Techniques 839  
*Case Study: Binary Search—An Example of Recursive Thinking* 840  
*Programming Example: A Recursive Member Function* 848

Chapter Summary 852  
Answers to Self-Test Exercises 852  
Practice Programs 857  
Programming Projects 857

## **Chapter 15 Inheritance 863**

### **15.1 INHERITANCE BASICS 864**

Derived Classes 867

Constructors in Derived Classes 875

*Pitfall:* Use of Private Member Variables from the Base Class 878

*Pitfall:* Private Member Functions Are Effectively Not Inherited 880

The *protected* Qualifier 880

Redefinition of Member Functions 883

Redefining Versus Overloading 886

Access to a Redefined Base Function 888

### **15.2 INHERITANCE DETAILS 889**

Functions That Are Not Inherited 889

Assignment Operators and Copy Constructors in Derived Classes 890

Destructors in Derived Classes 891

### **15.3 POLYMORPHISM 892**

Late Binding 893

Virtual Functions in C++ 894

Virtual Functions and Extended Type Compatibility 899

*Pitfall:* The Slicing Problem 903

*Pitfall:* Not Using Virtual Member Functions 904

*Pitfall:* Attempting to Compile Class Definitions Without

Definitions for Every Virtual Member Function 905

*Programming Tip:* Make Destructors Virtual 905

Chapter Summary 907

Answers to Self-Test Exercises 907

Practice Programs 911

Programming Projects 914

## **Chapter 16 Exception Handling 923**

### **16.1 EXCEPTION-HANDLING BASICS 925**

A Toy Example of Exception Handling 925

Defining Your Own Exception Classes 934

Multiple Throws and Catches 934

*Pitfall:* Catch the More Specific Exception First 938

*Programming Tip:* Exception Classes Can Be Trivial 939

Throwing an Exception in a Function 939

Exception Specification 941  
*Pitfall: Exception Specification in Derived Classes* 943

## **16.2 PROGRAMMING TECHNIQUES FOR EXCEPTION HANDLING 944**

When to Throw an Exception 944  
*Pitfall: Uncaught Exceptions* 946  
*Pitfall: Nested try-catch Blocks* 946  
*Pitfall: Overuse of Exceptions* 946  
Exception Class Hierarchies 947  
Testing for Available Memory 947  
Rethrowing an Exception 948

Chapter Summary 948  
Answers to Self-Test Exercises 948  
Practice Programs 950  
Programming Projects 951

## **Chapter 17 Templates 955**

### **17.1 TEMPLATES FOR ALGORITHM ABSTRACTION 956**

Templates for Functions 957  
*Pitfall: Compiler Complications* 961  
*Programming Example: A Generic Sorting Function* 963  
*Programming Tip: How to Define Templates* 967  
*Pitfall: Using a Template with an Inappropriate Type* 968

### **17.2 TEMPLATES FOR DATA ABSTRACTION 969**

Syntax for Class Templates 969  
*Programming Example: An Array Class* 972

Chapter Summary 979  
Answers to Self-Test Exercises 979  
Practice Programs 983  
Programming Projects 983

## **Chapter 18 Standard Template Library 987**

### **18.1 ITERATORS 989**

*using* Declarations 989  
Iterator Basics 990

<i>Programming Tip: Use auto to Simplify Variable Declarations</i>	994
<i>Pitfall: Compiler Problems</i>	994
Kinds of Iterators	996
Constant and Mutable Iterators	1000
Reverse Iterators	1001
Other Kinds of Iterators	1002

## **18.2 CONTAINERS 1003**

Sequential Containers	1004
<i>Pitfall: Iterators and Removing Elements</i>	1008
<i>Programming Tip: Type Definitions in Containers</i>	1009
Container Adapters <i>stack</i> and <i>queue</i>	1009
Associative Containers <i>set</i> and <i>map</i>	1013
<i>Programming Tip: Use Initialization, Ranged For, and auto with Containers</i>	1020
Efficiency	1020

## **18.3 GENERIC ALGORITHMS 1021**

Running Times and Big- <i>O</i> Notation	1022
Container Access Running Times	1025
Nonmodifying Sequence Algorithms	1027
Container Modifying Algorithms	1031
Set Algorithms	1033
Sorting Algorithms	1034
Chapter Summary	1035
Answers to Self-Test Exercises	1035
Practice Programs	1037
Programming Projects	1038

## **APPENDICES**

<b>1 C++ Keywords</b>	1045
<b>2 Precedence of Operators</b>	1046
<b>3 The ASCII Character Set</b>	1048
<b>4 Some Library Functions</b>	1049
<b>5 Inline Functions</b>	1056
<b>6 Overloading the Array Index Square Brackets</b>	1057
<b>7 The <i>this</i> Pointer</b>	1059
<b>8 Overloading Operators as Member Operators</b>	1062

<b>INDEX</b>	1064
--------------	------

<b>CREDITS</b>	1085
----------------	------