Table of Contents

1 Object-Oriented Programming **1**

1.1 Object-Oriented Problem Solving 2
1.2 Classes, Objects, and Encapsulation 4
1.3 Subclasses—Inheritance and Polymorphism 7
1.4 Challenges in Object-Oriented Programming 7
1.4.1 Partitioning Software into Classes 8
1.4.2 Adding Functionality to an Existing Software System 8
1.4.3 Hierarchical Structure of Types and Subtypes 8
Exercises 10
References 10

2 From C to Shining C++ **11**

2.1 The Language and Its History 11
2.2 How C++ Enhances C in Small Ways 13
2.2.1 Comments 14
2.2.2 Enumeration Names 14
2.2.3 Struct or Class Names 14
2.2.4 Declarations within Blocks 15
2.2.5 Scope Qualifier Operator 15
2.2.6 The Const Specifier 15
2.2.7 Anonymous Unions 15
2.2.8 Explicit Type Conversion 15
2.2.9 Function Prototypes 16
2.2.10 Overloading of Function Names 16

2.2.11 Default Value for Function Parameters 16
2.2.12 Functions with an Unspecified Number of Parameters 16
2.2.13 Reference Parameters in a Function 17
2.2.14 The Inline Specifier 17
2.2.15 The New and Delete Operators 17
2.2.16 Pointers to Void and Functions That Return Void 17
2.3 How C++ Enhances C in Large Ways 18
2.3.1 The Class Construct and Data Encapsulation 18
2.3.2 Struct as a Special Case of a Class 18
2.3.3 Constructors and Destructors 18
2.3.4 Private, Protected, and Public Sections 19
2.3.5 Objects and Messages 19
2.3.6 Friends 19
2.3.7 Overloading of Operators and Function Names in Classes 20
2.3.8 Derived Classes 20
2.3.9 Virtual Functions 20
2.3.10 The Stream Library 21
Exercises 21

3 Getting Up to Speed with C++ 22

3.1 Comments 22
3.2 Constants, Types, and Declarations 23
3.3 C++ Operators 30
3.4 Pass-by-Reference 31
3.5 Pointers 33
3.6 The Const Specifier 42
3.7 Enumeration Types 43
3.8 Anonymous Unions 44
3.9 Explicit Type Conversion 44
3.10 Functions 46
3.10.1 Function Prototypes 46
3.10.2 Inline Functions 46
3.10.3 Default Arguments 47
3.10.4 Overloading Function Names 47
3.10.5 Functions with an Unspecified Number of Arguments 49
3.10.6 Pointers to Functions and Generics 49
3.11 Files and the Physical Organization of C++ Systems 54
Exercises 55

4 Data Encapsulation and Data Hiding Using Classes **58**

4.1 Procedural Languages, Data Abstraction, Encapsulation, and Data Hiding 58
4.2 An Introduction to Classes in C++ 60
4.3 Self-Reference in Classes 67
4.4 Constructors and Destructors 69
4.4.1 Stack Abstract Data Type Implemented as a C++ Class 70
4.4.2 Stack Abstract Data Type Implemented in Modula-2 72
4.5 Class Objects as Members 75
4.6 Vectors of Objects 77
4.7 Friends 78
4.8 Static Members of a Class 80
4.9 Overloading of Operators 81
4.9.1 Binary and Unary Operators 82
4.9.2 Some Examples of Operator Overloading 84
4.9.3 The <stream.h> Library 95
4.10 Some Baseline Classes 100
4.10.1 Generic Lists 101
4.10.2 A Generic Search Table Implemented as a Binary Search Tree 106
4.10.2-1 Modula-2 Encapsulation of Search_Table Abstraction 107
4.10.2-2 C++ Encapsulation of Search_Table Abstraction 110
Exercises 121

5 Inheritance and Derived Classes **124**

5.1 The Derived Class Construct 125
5.2 Derived Classes with Parent Class Constructors 130
5.3 Some Examples of Derived Classes 132
5.3.1 Derived Counter Class 132
5.3.2 The Class System at a University 134
5.3.3 A Stack and Queue Derived from a Generic List 139
Exercises 142

6 Polymorphism and Virtual Functions 144

6.1 Virtual Functions 145
6.2 An Object-Oriented Solution to Generating a Linked List 150
6.2.1 Nonpolymorphic Solution to Heterogeneous Linked List 150
6.2.2 Object-Oriented Solution to Heterogeneous Linked List 157
6.2.3 Maintenance on the Non–Object-Oriented and Object-Oriented Systems 165
6.3 A Heterogeneous Search Tree Using Polymorphism 170
6.4 Finite-State Machine Using Polymorphism 176
Exercises 181

7 Case Studies in Object-Oriented Programming 182

7.1 A "Super Fast" Spelling Checker 182
7.1.1 Specifications for Spelling Checker 182
7.1.2 High-Level Design of Spelling Checker 183
7.1.3 Low-Level Design of Spelling Checker 186
7.1.4 Implementation of Spelling Checker 187
7.2 Bank Teller Discrete Event Simulation 199
7.2.1 Specifications for Queue Simulation 199
7.2.2 High-Level Design of Queue Simulation 200
7.2.3 Low-Level Design of Queue Simulation 206
7.2.4 Implementation of Queue Simulation 206
7.2.5 Simulation Output 223
7.2.6 Maintenance of Queue Simulation 224
7.3 Interactive Function Evaluator 229
7.3.1 Specifications for Function Evaluator 230
7.3.2 Review of Expression Trees 233
7.3.3 High-Level Design of Function Evaluator 239
7.3.4 Low-Level Design of Function Evaluator 246
7.3.5 Full Implementation of Function Evaluator 253
Exercises 268

Index 269