

Contents

Preface — V

1 Introduction — 1

- 1.1 The Development of Computer Programming Languages — 1
 - 1.1.1 Machine Language and Assembly Language — 1
 - 1.1.2 High-level Language — 2
 - 1.1.3 Object-oriented Language — 3
- 1.2 Object-oriented Method — 3
 - 1.2.1 The Origin of Object-oriented Method — 3
 - 1.2.2 Basic Concepts of Object-oriented — 5
- 1.3 Object-oriented Software Development — 7
 - 1.3.1 Analysis — 7
 - 1.3.2 Design — 7
 - 1.3.3 Programming — 8
 - 1.3.4 Test — 8
 - 1.3.5 Maintenance — 8
- 1.4 Representation and Storage of Information — 9
 - 1.4.1 Digital System of Computers — 9
 - 1.4.2 Conversions among Numeral Systems — 11
 - 1.4.3 Storage Units of Information — 14
 - 1.4.4 Binary-coded Representation — 14
 - 1.4.5 Fixed Point Number and Floating-Point Number — 19
 - 1.4.6 The Number Range that can be Represented — 20
 - 1.4.7 Representation of Non-numerical Information — 21
- 1.5 The Development Process of Programs — 21
 - 1.5.1 Elementary Terms — 22
 - 1.5.2 The Development Process — 22
- 1.6 Summary — 23
 - Exercises — 24

2 Elementary C++ Programming — 25

- 2.1 An Overview of C++ Language — 25
 - 2.1.1 Origins of C++ — 25
 - 2.1.2 Characteristics of C++ — 26
 - 2.1.3 C++ Programming Examples — 26
 - 2.1.4 Character Set — 28
 - 2.1.5 Lexical Tokens — 28
- 2.2 Basic Data Types and Expressions — 30
 - 2.2.1 Basic Data Types — 31

2.2.2	Constants — 32
2.2.3	Variables — 34
2.2.4	Symbol Constants — 36
2.2.5	Operators and Expressions — 36
2.2.6	Statement — 46
2.3	Data Input and Output — 47
2.3.1	I/O Stream — 47
2.3.2	Predefined Input and Output Operator — 47
2.3.3	Simple I/O Format Control — 48
2.4	The Fundamental Control Structures of Algorithms — 49
2.4.1	Achieving Case Structure Using <i>if</i> Statement — 50
2.4.2	Multiple Selection Structure — 51
2.4.3	Loop Structure — 55
2.4.4	Nestings of Loop Structure and Case Structure — 61
2.4.5	Other Control Statements — 64
2.5	User-Defined Data Type — 65
2.5.1	<i>typedef</i> Declaration — 65
2.5.2	Enumeration Type – enum — 65
2.5.3	Structure — 67
2.5.4	Union — 70
2.6	Summary — 72
	Exercises — 73

3 Functions — 79

3.1	Definition and Use of Function — 79
3.1.1	Definition of Function — 79
3.1.2	Function Calls — 80
3.1.3	Passing Parameters Between Functions — 95
3.2	Inline Functions — 99
3.3	Default Formal Parameters in Functions — 101
3.4	Function Overloading — 104
3.5	Using C++ System Functions — 106
3.6	Summary — 109
	Exercises — 110

4 Class and Object — 113

4.1	Basic Features of Object-Oriented Design — 113
4.1.1	Abstraction — 113
4.1.2	Encapsulation — 114
4.1.3	Inheritance — 115
4.1.4	Polymorphism — 115
4.2	Class and Object — 116

4.2.1	Definition of Class — 117
4.2.2	Access Control to Class Members — 118
4.2.3	Member Function of Class — 120
4.2.4	Object — 121
4.2.5	Program Instance — 122
4.3	Constructor and Destructor — 124
4.3.1	Class Constructors — 124
4.3.2	The Copy Constructor — 127
4.3.3	Class Destructor — 131
4.3.4	Program Instance — 132
4.4	Combination of Classes — 134
4.4.1	Combination — 134
4.4.2	Forward Declaration — 138
4.5	UML — 141
4.5.1	Brief Introduction of UML — 141
4.5.2	UML Class Diagrams — 142
4.6	Program Instance – Personnel Information Management Program — 149
4.6.1	Design of Class — 149
4.6.2	Source Code and Description — 150
4.6.3	Running Result and Analyses — 152
4.7	Summary — 153
	Exercises — 153
5	Data Sharing and Protecting — 157
5.1	Scope and Visibility of Identifiers — 157
5.1.1	Scope — 157
5.1.2	Visibility — 159
5.2	Lifetime of Object — 160
5.2.1	Static Lifetime — 160
5.2.2	Dynamic Lifetime — 161
5.3	Static Members of Class — 164
5.3.1	Static Data Member — 164
5.3.2	Static Function Member — 167
5.4	Friend of Class — 170
5.4.1	Friend Function — 172
5.4.2	Friend Class — 174
5.5	Protection of Shared Data — 175
5.5.1	Constant Reference — 175
5.5.2	Constant Object — 176
5.5.3	Class Members Modified by <i>const</i> — 177
5.6	Multifile Structure and Compilation Preprocessing Directives — 180

5.6.1	General Organization Structure of C++ Program —	180
5.6.2	External Variable and External Function —	183
5.6.3	Standard C++ Library and Namespace —	184
5.6.4	Compilation Preprocessing —	185
5.7	Example – Personnel Information Management Program —	190
5.8	Summary —	194
	Exercises —	194
6	Arrays, Pointers, and Strings —	197
6.1	Arrays —	197
6.1.1	Declaration and Use of Arrays —	198
6.1.2	Storage and Initialization of Arrays —	200
6.1.3	Using Arrays as Function Parameters —	203
6.1.4	Object Arrays —	204
6.1.5	Program Examples —	207
6.2	Pointers —	210
6.2.1	Access Method of Memory Space —	211
6.2.2	Declaration of Pointer Variables —	212
6.2.3	Operations Related to Addresses – ‘*’ and ‘&’ —	213
6.2.4	Assignment of Pointers —	214
6.2.5	Pointer Operations —	217
6.2.6	Using Pointers to Process Array Elements —	219
6.2.7	Pointer Arrays —	220
6.2.8	Using Pointers as Function Parameters —	223
6.2.9	Pointer-Type Functions —	225
6.2.10	Pointers that Point to Functions —	225
6.2.11	Object Pointers —	228
6.3	Dynamic Memory Allocation —	234
6.3.1	<i>new</i> Operation and <i>delete</i> Operation —	234
6.3.2	Dynamic Memory Allocation and Release Functions —	240
6.4	Deep Copy and Shallow Copy —	240
6.5	Strings —	245
6.5.1	Using Character Arrays to Store and Process Strings —	245
6.5.2	The <i>string</i> Class —	248
6.6	Program Example – Personnel Information Management Program —	251
6.7	Summary —	255
	Exercises —	256
7	Inheritance and Derivation —	259
7.1	Inheritance and Derivation of Class —	259
7.1.1	Instances of Inheritance and Derivation —	259

7.1.2	Definition of Derived Class —	261
7.1.3	The Generation Process of Derived Classes —	263
7.2	Access Control —	266
7.2.1	Public Inheritance —	266
7.2.2	Private Inheritance —	269
7.2.3	Protected Inheritance —	271
7.3	Type Compatible Rule —	274
7.4	Constructor and Destructor of Derived Class —	277
7.4.1	Constructor —	277
7.4.2	Copy Constructor —	281
7.4.3	Destructor —	281
7.5	Identification and Access of Derived-Class Member —	284
7.5.1	Scope Resolution —	284
7.5.2	Virtual Base Class —	290
7.5.3	Constructors of Virtual Base Class and Derived Class —	294
7.6	Program Example: Solving Linear Equations using Gaussian Elimination Method —	295
7.6.1	Fundamental Principles —	296
7.6.2	Analysis of the Program Design —	297
7.6.3	Source Code and Explanation —	297
7.6.4	Execution Result and Analysis —	304
7.7	Program Example: Personnel Information Management Program —	305
7.7.1	Problem Description —	305
7.7.2	Class Design —	305
7.7.3	Source Code and Explanation —	306
7.7.4	Execution Result and Analysis —	311
7.8	Summary —	312
	Exercises —	314
8	Polymorphism —	315
8.1	An Overview of Polymorphism —	315
8.1.1	Types of Polymorphism —	315
8.1.2	Implementation of Polymorphism —	316
8.2	Operator Overload —	316
8.2.1	Rules of Operator Overload —	317
8.2.2	Operator Overloaded as Member Function —	318
8.2.3	Operator Overloaded as Friend Function —	323
8.3	Virtual Function —	326
8.3.1	Ordinary Virtual Function Member —	327
8.3.2	Virtual Destructor —	330
8.4	Abstract Classes —	332
8.4.1	Pure Virtual Functions —	332

8.4.2	Abstract Classes —	333
8.5	Program Instance: Variable Stepwise Trapezoid Method to Calculate Functional Definite Integral —	335
8.5.1	Basic Principle —	335
8.5.2	Analysis of Program Design —	337
8.5.3	Source Code and Explanation —	338
8.5.4	Execution Result and Analysis —	341
8.6	Program Instance: Improvement on Staff Information Management System for a Small Corporation —	341
8.7	Summary —	348
	Exercises —	349
9	Collections and Their Organization —	351
9.1	Function Templates and Class Templates —	352
9.1.1	Function Template —	352
9.1.2	Class Template —	355
9.2	Linear Collection —	359
9.2.1	Definition of Linear Collection —	359
9.2.2	Direct Accessible Linear Collection – Array —	362
9.2.3	Sequential Access Collection – Linked List —	372
9.2.4	Stack —	378
9.2.5	Queues —	385
9.3	Organizing Data in Linear Collections —	389
9.3.1	Insertion Sort —	389
9.3.2	Selection Sort —	390
9.3.3	Exchange Sort —	392
9.3.4	Sequential Search —	394
9.3.5	Binary Search —	394
9.4	Application – Improving the HR Management Program of a Small Company —	396
9.5	Summary —	397
	Exercises —	398
10	Generic Programming and the Standard Template Library —	401
10.1	Generic Programming —	401
10.1.1	Introduction —	401
10.1.2	Namespace —	402
10.1.3	Differences of Naming Conventions Between C/C++ —	404
10.1.4	Concepts of STL —	404
10.1.5	Containers —	404
10.1.6	Adapters —	405
10.1.7	Iterators —	405

10.1.8	Algorithms —	405
10.1.9	Interfaces of Containers —	405
10.2	Containers in STL —	406
10.2.1	Sequential Containers —	407
10.2.2	Adapters of Containers —	416
10.3	Iterators —	418
10.3.1	Types of Iterators —	419
10.3.2	Auxiliary Functions in Iterators —	421
10.4	Algorithms in STL —	422
10.4.1	Using the Algorithms —	423
10.4.2	Nonmutating Sequence Algorithms —	424
10.4.3	Mutating Sequence Algorithms —	427
10.4.4	Sorting Related Algorithms —	431
10.4.5	Numerical Algorithms —	436
10.5	Function Objects —	437
10.6	Application – Improving the HR Management Program of a Small Company —	440
10.7	Summary —	442
	Exercises —	442
11	The I/O Stream Library and Input/Output —	445
11.1	I/O Stream's Concept and the Structure of a Stream Library —	445
11.2	Output Stream —	447
11.2.1	Construct Output Object —	448
11.2.2	The Use of Inserter and Manipulator —	449
11.2.3	Output File Stream Member Function —	454
11.2.4	Binary Output File —	457
11.3	Input Stream —	458
11.3.1	Construct Input Stream Object —	458
11.3.2	Extraction Operator —	458
11.3.3	Input Stream Manipulator —	459
11.3.4	Input Stream Member Function —	459
11.4	Input/output Stream —	463
11.5	Example-improve Employee Information Management System —	464
11.6	Summary —	466
	Exercises —	467
12	Exception Handling —	469
12.1	Basic Concepts of Exception Handling —	469
12.2	The Implementation of Exception Handling in C++ —	470
12.2.1	The Syntax of Exception Handling —	470
12.2.2	Exception Interface Declaration —	473

12.3	Destruction and Construction in Exception Handling —	473
12.4	Exception Handling of Standard Library —	476
12.5	Program Example Improvement to Personal Information Administration Program in a Small Company —	478
12.6	Summary —	480
	Exercises —	480
Index —		481