C++ for You++

An Introduction to Programming and Computer Science

Maria Litvin
Phillips Academy, Andover, Massachusetts

Gary Litvin
Skylight Software, Inc.

Skylight Publishing
Andover, Massachusetts

Copyright © 1998 by
Maria Litvin and Gary Litvin

C++ for You++, AP Edition, by Maria Litvin and Gary Litvin is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License.

You are free:

- to Share — to copy, distribute and transmit the work
- to Remix — to adapt the work

Under the following conditions:

- Attribution — You must attribute the work to Maria Litvin and Gary Litvin (but not in any way that suggests that they endorse you or your use of the work). On the title page of your copy or adaptation place the following statement:

> Adapted from *C++ for You++* by Maria Litvin and Gary Litvin, Skylight Publishing, 1998, available at http://www.skylit.com.

- Noncommercial — You may not use this work for commercial purposes.
- Share Alike — If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

See http://creativecommons.org/licenses/by-nc-sa/3.0/ for details.

Skylight Publishing
9 Bartlet Street, Suite 70
Andover, MA 01810
(978) 475-1431
e-mail: support@skylit.com
web: http://www.skylit.com

Library of Congress Catalog Card Number: 97–091209

ISBN 0-9654853-9-0

To Marg and Aaron

Brief Contents

Part One. Programs: Syntax and Style

Chapter 1. Introduction to Hardware and Software 3

Chapter 2. A First Look at a C++ Program 23

Chapter 3. Variables and Constants 49

Chapter 4. Arithmetic Expressions 73

Chapter 5. Arrays, `apvector` and `apmatrix` Classes 85

Chapter 6. Logical Expressions and `if-else` Statements 99

Chapter 7. Iterative Statements: `while`, `for`, `do-while` 121

Chapter 8. The `switch` Statement 143

Chapter 9. Algorithms 157

Chapter 10. Monte Carlo Methods 171

Chapter 11. Pointers, References, Dynamic Memory Allocation 179

Chapter 12. Strings 199

Chapter 13. Structures 223

Part Two. Classes and Data Structures

Chapter 14. Modularity 243
Chapter 15. Classes 259
Chapter 16. Templates 279
Chapter 17. Linked Lists 289
Chapter 18. Stacks 313
Chapter 19. Recursion 327
Chapter 20. Queues 345
Chapter 21. Classes: More Advanced Features 363
Chapter 22. Trees 399
Chapter 23. Expression Trees 435
Chapter 24. Heaps 447
Chapter 25. Analysis of Algorithms 461
Chapter 26. Searching and Hashing 475
Chapter 27. Sorting 489
Chapter 28. Inheritance 509
Appendix A: Bit-Wise Logical Operators 531
Appendix B: Pointers and Arrays 539
Appendix C: Stream I/O Classes 545
Index 553

Contents

Preface xv

Part One. Programs: Syntax and Style

Chapter 1. Introduction to Hardware and Software 3

1.1 Discussion 4
1.2 Hardware Overview 6
  1.2.1 The CPU 6
  1.2.2 Memory 7
  1.2.3 Secondary Storage Devices 8
  1.2.4 Input and Output Devices 9
1.3 Representation of Information in Computer Memory 10
  1.3.1 Numbers 11
  1.3.2 Characters 15
1.4 Software Overview 17
1.5 Software Development 18
1.6 Suggested Reading 21

Chapter 2. A First Look at a C++ Program 23

2.1 Discussion 24
2.2 *Case Study:* Dictionary Program 24
2.3 Use of Comments 28
2.4 Functions 29
2.5 Class Libraries and Header Files 34
2.6 The Preprocessor 35
2.7 Reserved Words and Programmer-Defined Names 38
2.8 Syntax and Style 40
2.9 Statements, Blocks, Indentation 43
2.10 Input and Output 45
2.11 *Lab:* Compile and Run 47
2.12 Summary 47

Chapter 3. Variables and Constants 49

3.1 Discussion 50
3.2 *Case Study:* Fastfood, a Point-of-Sale Program 51
3.3 Declarations of Variables 54
3.4 Data Types 55
3.5 Renaming Data Types with `typedef` 57
3.6 Constants 58
3.7 Initialization of Variables 60
3.8 *Case Study:* Fastfood Continued 61
3.9 Output Formatting 64
3.10 Scope of Variables and Constants 65
3.11 Advanced Scope Rules 68
3.12 *Lab:* Statistics for Fastfood 70
3.13 `enum` Data Types 70
3.14 Summary 72

Chapter 4. Arithmetic Expressions 73

4.1 Discussion 74
4.2 Data Types in Expressions 74
4.3 Type Conversions with the Cast Operator 76
4.4 Compound Assignment Operators 77
4.5 Increment and Decrement Operators 78
4.6 The Modulo Division Operator 80
4.7 *Lab:* Three Means 81
4.8 Summary 83

Chapter 5. Arrays, `apvector` and `apmatrix` Classes 85

5.1 One-Dimensional Arrays 86
5.2 The `apvector` Class 87
5.3 Declaring and Using `apvector` Variables 89
5.4 Passing `apvector` Arguments to Functions 91
5.5 Two-Dimensional Arrays 93
5.6 The `apmatrix` Class 94
5.7 *Lab:* Reverse an Array 95
5.8 Summary 97

Chapter 6. Logical Expressions and `if-else` Statements 99

6.1 Discussion 100
6.2 `if-else` Statements 102
6.3 *True* and *False* Values 102
6.4 Relational Operators 103
6.5 Logical Operators 104
6.6 Order of Operators 106
6.7 Short-Circuit Evaluation 107
6.8 *Case Study:* Day of the Week Program 108
6.9 *Lab:* Holidays 114
6.10 `if-else if` and Nested `if-else` 115
6.11 Common `if-else` Errors 118
6.12 Summary 119

Chapter 7. Iterative Statements: `while`, `for`, `do-while` 121

7.1 Discussion 122
7.2 `while` and `for` Loops 122
7.3 *Lab:* Fibonacci Numbers 127
7.4 The `do-while` Loop 127
7.5 `break` and `continue` 128
7.6 A Word About `goto` 132
7.7 Iterations and Arrays 132
7.8 *Lab:* Students' Grades 134
7.9 Iterations and Two-Dimensional Arrays 137
7.10 *Lab:* John Conway's Game of Life 138
7.11 Summary 142

Chapter 8. The `switch` Statement 143

8.1 Discussion 144
8.2 *Case Study:* The Calculator Program 146
8.3 *Case Study:* Menu 147
8.4 *Lab: One of Each* Inventory System 151
8.5 Details of the `switch` Statement 154
8.6 Breaks in Nested Loops and Switches 155
8.7 Summary 156

Chapter 9. Algorithms 157

9.1 Discussion 158
9.2 Selection Sort 160
9.3 Binary Search 161
9.4 Euclid's Algorithm for Finding GCF 164
9.5 *Lab:* Practice in Algorithms 167
9.6 Summary 169

Chapter 10. Monte Carlo Methods 171

10.1 Discussion 172
10.2 *Case Study:* Estimating the Perimeter of an Ellipse 174
10.3 *Lab:* Estimating π Using the Monte Carlo Method 178

Chapter 11. Pointers, References, Dynamic Memory Allocation 179

11.1 Discussion 180
11.2 Pointers and References: Declarations and Assignments 181
11.3 Passing Arguments to Functions by Reference 186
11.4 *Lab:* Quadratic Formula 190
11.5 The Null Pointer 191
11.6 Dynamic Memory Allocation: `new` and `delete` 192
11.7 Returning Pointers or References from Functions 195
11.8 Summary 197

Chapter 12. Strings 199

12.1 Discussion 200
12.2 Literal Strings 201
12.3 Standard Library Functions for Strings 204
12.4 Input and Output for Strings 206
12.5 The `apstring` Class 211
12.6 *Lab:* Palindromes 216
12.7 *Lab:* GREP 216
12.8 Formatted Output to a Character Array 219
12.9 Summary 221

Chapter 13. Structures 223

13.1 User-Defined Types 224
13.2 Initialization and Assignments 226
13.3 Accessing Structure Members 228
13.4 Passing and Returning Structures to and from Functions 232
13.5 Input/Output for User-Defined Types 235
13.6 *Lab:* Updating Your Inventory 237
13.7 *Programming Project:* Enhancements to the Dictionary Program 238

Part Two. Classes and Data Structures

Chapter 14. Modularity 243

14.1 Discussion 244
14.2 Example: Dates Revisited 245
14.3 Program Modules and Header Files 246
14.4 Module Hierarchies 251
14.5 Linking 253
14.6 Global and Static Variables 254
14.7 Inline Functions 256
14.8 Summary 257

Chapter 15. Classes 259

15.1 Discussion 260
15.2 Public and Private Members, Encapsulation 261
15.3 Implementation of a Class 266
15.4 Syntax for Accessing Class Members 269
15.5 Constructors and Destructors 270
15.6 *Lab:* Add a Constructor to the `apstring` Class 275
15.7 *Lab:* Vending Machine Class 276
15.8 Summary 277

Chapter 16. Templates 279

16.1 Discussion 280
16.2 Syntax for Templates 281
16.3 Classes with Parameterized Types 282
16.4 How to Use Templates 284
16.5 *Lab:* Adding Functions to the `apvector` Class 286
16.6 Summary 286

Chapter 17. Linked Lists 289

17.1 Data Structures and Abstract Data Types 290
17.2 Linked List Data Structure 291
17.3 Linked List Traversal 293
17.4 The Insert Function 296
17.5 *Lab:* Creating, Traversing and Destroying a Linked List 301
17.6 The Remove Function 302
17.7 *Lab:* Maintaining a List 304
17.8 Linked Lists vs. Arrays 305
17.9 Linked Lists with a Tail and Doubly Linked Lists 306
17.10 *Lab:* Doubly Linked List 309
17.11 Summary 310

Chapter 18. Stacks 313

18.1 Discussion 314
18.2 Array Implementation of Stack 315
18.3 The `apstack` Class 318
18.4 *Case Study and Lab:* Music 319
18.5 The Hardware Stack 323
18.6 Summary 326

Chapter 19. Recursion 327

19.1 Discussion 328
19.2 Examples of Recursive Functions 329
19.3 Base Case and Recursive Case 332
19.4 When Not to Use Recursion 333
19.5 Understanding and Debugging Recursive Functions 338
19.6 *Lab:* The Towers of Hanoi 341
19.7 *Lab:* Area Fill 341
19.8 Summary 343

Chapter 20. Queues 345

20.1 Discussion 346
20.2 Ring Buffer and Linked List Queue Implementations 347
20.3 The `apqueue` Class 353
20.4 *Case Study:* Application of Queues 354
20.5 *Lab:* e-Mail 357
20.6 Summary 361

Chapter 21. Classes: More Advanced Features 363

21.1 Discussion 364
21.2 Initializer Lists 365
21.3 Operator Overloading 370
21.4 Canonical Features of a Class 376
21.5 Constructors as Casts 381
21.6 Friend Classes and Functions 383
21.7 Iterators 384
21.8 Static Class Members 390
21.9 Efficiency and Design Considerations 395
21.10 Summary 396

Chapter 22. Trees 399

22.1 Discussion 400
22.2 Binary Search Trees 404
22.3 BST's Destroy, Find, and Insert Functions 408
22.4 BST's Remove Function 412
22.5 Binary Tree Traversals 417
22.6 Implementing Tree as a Class 419
22.7 *Lab:* Morse Code 423
22.8 *Programming Project:* e-Mail Revisited 430
22.9 Summary 433

Chapter 23. Expression Trees 435

23.1 Discussion 436
23.2 Evaluating Expressions Represented by Trees 437
23.3 Prefix and Postfix Notations 440
23.4 Summary 444

Chapter 24. Heaps 447

24.1 Discussion 448
24.2 Binary Trees: Non-Linked Representation 449
24.3 Implementation of a Heap 450
24.4 *Programming Project:* Stock Exchange 458
24.5 Summary 459

Chapter 25. Analysis of Algorithms 461

25.1 Discussion 462
25.2 Asymptotics: Big-O Notation 464
25.3 Summary 473

Chapter 26. Searching and Hashing 475

26.1 Discussion 476
26.2 Sequential and Binary Search 477
26.3 Lookup Tables 479
26.4 *Lab:* Cryptography 480
26.5 Hashing 483
26.6 Summary 487

Chapter 27. Sorting 489

27.1 Discussion 490
27.2 $O(n^2)$ Sorts 492
  27.2.1 Selection Sort 492
  27.2.2 Insertion Sort 492
  27.2.3 Bubble Sort 494
27.3 $O(n \log n)$ Sorts 495
  27.3.1 Mergesort 496
  27.3.2 Quicksort 499
  27.3.3 Treesort and Heapsort 504
27.4 Radix Sort 506
27.5 Summary 507

Chapter 28. Inheritance 509

28.1 Discussion 510
28.2 Inheritance vs. Embedding 512
28.3 Member Access in Derived Classes 514
28.4 Redefining Member Functions 517
28.5 Base and Derived Class Constructors 519
28.6 Assignments and Pointer Conversions 521
28.7 Virtual Functions, Polymorphism 525
28.8 Inheritance and Sound Software Design 529
28.9 Summary 530

Appendix A: Bit-Wise Logical Operators 531

Appendix B: Pointers and Arrays 539

Appendix C: Stream I/O Classes 545

Index 553

Preface

C++ is becoming the language of choice for introducing college students across the country to computer science and programming. In high schools, the Advanced Placement[*] (AP) examination in Computer Science will be administered in C++ for the first time at the end of the 1998-99 academic year. While Maria was teaching an experimental year-long AP computer science course in C++ in 1995-96, we both saw the need for a manageable and concise textbook that would cover programming, algorithms, and data structures in a style indigenous to C++. Maria's students at Phillips Academy embraced the opportunity to take the AP course in C++ (even though they had to switch to Pascal in the final weeks before the AP exam) and, with their support, *C++ for You++* was born.

We have designed this book for a two- or three-semester high school or college introductory course in programming and data structures, with the choice of topics guided by a typical first-year college course as described in the College Board's Advanced Placement curriculum. Part 1 covers C++ programming (excluding classes), with the emphasis on effective programming practices and good style. Part 2 introduces C++ classes and covers the usual data structures as well as searching and sorting algorithms.

This *Special AP Edition* introduces the five AP classes, *apvector*, *apmatrix*, *apstring*, *apstack*, and *apqueue*, and explains how to use them. These classes were developed by the College Board's C++ AP Development Committee and are required for the APCS exam. This book follows the Committee's recommendations that students always use the *apvector* and *apmatrix* classes instead of built-in one- and two-dimensional arrays, and that the *apstring* class always be used instead of null-terminated strings. The *apstack* and *apqueue*

[*]Advanced Placement is a registered trademark of the College Entrance Examination Board which is not responsible for the contents of this text.

classes provide standard implementations of the stack and queue data structures. Students who take the A- or AB- level AP exam are expected to know how to use the *apvector*, *apmatrix*, and *apstring* classes in programs. Students who take the AB-level exam are also expected to use and re-implement the *apstack* and *apqueue* classes.

Computer science is an applied discipline, not just a set of academic theories. Therefore, the main thrust of *C++ for You++* is to teach students to write effective programs. Combining our experience as a teacher and a professional software engineer, we have sought to include modern, realistic examples and present them in a format teachers and their students will find accessible. Our labs and case studies aim to demonstrate the most appropriate uses of the programming techniques and data structures we cover.

We assume that at least one or two classes each week will be spent in a computer lab with students working independently or in small groups. The accompanying disk contains all the labs and case studies, and the teacher's edition disk provides complete solutions to all labs. To simplify some of the lab exercises, teachers can share hints or fragments of code from their solution disk. Meanwhile, "extra credit" tasks can make the lab exercises more challenging for more advanced students. The book also proposes several independent programming projects that can stretch over a couple of weeks. The *Workbook to Accompany C++ for You++* provides many additional questions, exercises, and projects.

C++ for You++ does not require prior knowledge of programming. For beginners seeking a primer on C++ programming, our book includes many code fragments and "cookbook" recipes (in the text and on the accompanying disk) for writing reliable programs. Our lab exercises ask students to modify or enhance existing code before having them write programs from scratch—a "training wheels" approach that turns out confident, competent programmers.

For those already familiar with C++ (including structures, but not necessarily classes), Part 2 can serve as an independent introduction to data structures. After a brief discussion of how to create modular programs, we introduce C++ classes and templates and learn how to implement and use them. Then we begin a serious discussion of some software development topics and techniques not specific to C++ that are important to any computer programmer. We discuss data structures (linked lists, stacks, queues, trees) and their uses, recursion, and common algorithms for searching, hashing, and sorting. We also describe the *apstack* and *apqueue* classes and their use.

C++ for You++ seeks to accommodate different learning styles and aptitudes. In general, we have tried to reveal the underlying concepts of C++, where possible, and emphasize the programming choices that have become part of the C++ culture. Straightforward "cookbook" examples are followed by more detailed explanations of how and why they work. Throughout the book, less important technical details are grouped in sections that can be skipped on a first reading. For instance, Chapter 10, "Monte Carlo Methods," is optional; Chapter 11, "Pointers, References, Dynamic Memory Allocation," can be skipped almost entirely (with the exception of Section 11.3, which explains how to pass arguments to functions by reference). Some more advanced topics, in particular friends, iterators, static class members (Sections 21.6 - 21.8) and inheritance (Chapter 28) are not part of the AP subset required for the AP exam and can be skipped or covered partially, as time permits. Stream input and output classes are summarized in more detail in an appendix.

Without further delay, let us begin learning to program in C++!

⌘ ⌘ ⌘

Our sincere thanks to Doug Kuhlmann, the chairman of the Mathematics Department at Phillips Academy, for suggesting that Maria switch her Advanced Placement computer science course to C++ three years ahead of the national requirement; his support was most valuable in this effort. We thank George Best for encouraging us to write this book. Thanks to Bill Adams of Concord Academy and Kathy Larson of Kingston High School who read a preliminary draft of the book and suggested some important improvements. We are very grateful to Deborah Roudebush of Potomac Falls High School for inspiring this *AP Edition*, encouragement, and help with converting programs from built-in arrays to *apstring*, *apvector*, and *apmatrix* classes. And our special thanks to Margaret Litvin for her thoughtful and thorough editing.

⌘ ⌘ ⌘

The student files are available at http://www.skylit.com/cpp4ap/.

Complete answers and solutions are available to teachers — for access please
e-mail from your school email account to support@skylit.com.

About the Authors

Maria Litvin has taught computer science and mathematics at Phillips Academy in Andover, Massachusetts, since 1987. She is an Advanced Placement Computer Science exam reader and Table Leader and, as a consultant for The College Board, provides AP training for high school computer science teachers. Maria has received the 1999 Siemens Award for Advanced Placement for Mathematics, Science, and Technology for New England and the 2003 RadioShack National Teacher Award. Prior to joining Phillips Academy, Maria taught computer science at Boston University. Maria is a co-author of *C++ for You++: An Introduction to Programming and Computer Science*(1998), which became one of the leading high school textbooks for AP Computer Science courses, and of the earlier editions of the *Java Methods* books. Maria is also the co-author of *Be Prepared for the AP Computer Science Exam in Java* and *Mathematics for the Digital Age and Programming in Python* (Skylight Publishing, 2010).

Gary Litvin has worked in many areas of software development including artificial intelligence, pattern recognition, computer graphics, and neural networks. As founder of Skylight Software, Inc., he developed SKYLIGHTS/GX, one of the first GUI prototyping and development tools for C and C++ programmers. Gary led in the development of several state-of-the-art software products including interactive touch screen development tools, OCR and handwritten character recognition systems, and credit card fraud detection software. He is the co-author of *C++ for You++*, the *Java Methods* series, *Be Prepared for the AP Computer Science Exam in Java*, and *Mathematics for the Digital Age and Programming in Python*.

Part One

Programs:
Syntax and Style

Part Two

Classes and Data Structures