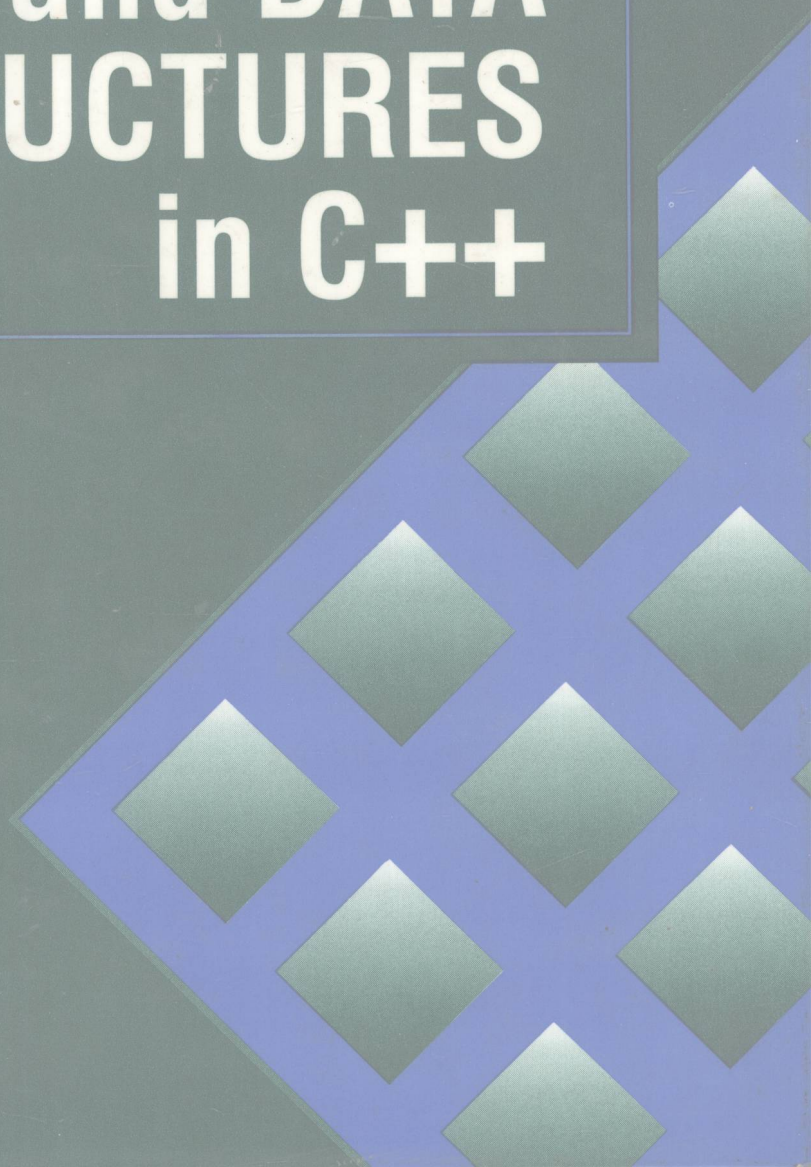


Alan Parker

ALGORITHMS and DATA STRUCTURES in C++



p312
238

9463027

ALGORITHMS and DATA STRUCTURES in C++

Alan Parker
The University of Alabama in Huntsville



E9463027



CRC Press

Boca Raton Ann Arbor London Tokyo

Library of Congress Cataloging-in-Publication Data

Parker, Alan, 1959–

Algorithms and data structures in C++ / Alan Parker.

p. cm.—(CRC series in computer engineering)

Includes bibliographical references and index.

ISBN 0-8493-7171-6

1. C++ (Computer program language). 2. Computer algorithms. 3. Data structures (Computer science). I. Title. II. Series.

QA76.C153P37 1993

005.13'—dc20

DNLN/DLC

for Library of Congress

93-20537

CIP

This book contains information obtained from authentic and highly regarded sources. Reprinted material is quoted with permission, and sources are indicated. A wide variety of references are listed. Reasonable efforts have been made to publish reliable data and information, but the author and the publisher cannot assume responsibility for the validity of all materials or for the consequences of their use.

Neither this book nor any part may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, microfilming, and recording, or by any information storage or retrieval system, without prior permission in writing from the publisher.

CRC Press, Inc.'s consent does not extend to copying for general distribution, for promotion, for creating new works, or for resale. Specific permission must be obtained in writing from CRC Press for such copying.

Direct all inquiries to CRC Press, Inc., 2000 Corporate Blvd., N.W., Boca Raton, Florida 33431.

© 1993 by CRC Press, Inc.

No claim to original U.S. Government works

International Standard Book Number 0-8493-7171-6

Library of Congress Card Number 93-20537

Printed in the United States of America 1 2 3 4 5 6 7 8 9 0

Printed on acid-free paper

ALGORITHMS and DATA STRUCTURES in C++

CRC Press
Computer Engineering Series

Series Editor
Udo W. Pooch
Texas A&M University

Published books:

Telecommunications and Networking

Udo W. Pooch, Texas A&M University
Denis P. Machuel, Telecommunications Consultant
John T. McCahn, Networking Consultant

**Spicey Circuits: Elements of Computer-Aided
Circuit Analysis**

Rahul Chattergy, University of Hawaii

**Microprocessor-Based Parallel Architecture
for Reliable Digital Signal Processing Systems**

Alan D. George, Florida State University
Lois Wright Hawkes, Florida State University

Discrete Event Simulation: A Practical Approach

Udo W. Pooch, Texas A&M University
James A. Wall, Simulation Consultant

Algorithms and Data Structures in C++

Alan Parker, The University of Alabama in Huntsville

Forthcoming books:

Handbook of Software Engineering

Udo W. Pooch, Texas A&M University

PREFACE

This text is designed for an introductory quarter or semester course in algorithms and data structures for students in engineering and computer science. It will also serve as a reference text for programmers in C++. The book presents algorithms and data structures with heavy emphasis on C++. Every C++ program presented is a stand-alone program. Except as noted, all of the programs in the book have been compiled and executed on multiple platforms.

When used in a course, the students should have access to C++ reference manuals for their particular programming environment. The instructor of the course should strive to describe to the students every line of each program. The prerequisite knowledge for this course should be a minimal understanding of digital logic. A high-level programming language is desirable but not required for more advanced students.

The study of algorithms is a massive field and no single text can do justice to every intricacy or application. The philosophy in this text is to choose an appropriate subset which exercises the unique and more modern aspects of the C++ programming language while providing a stimulating introduction to realistic problems.

I close with special thanks to my friend and colleague, Jeffrey H. Kulick, for his contributions to this manuscript.

Alan Parker
Huntsville, AL
1993

to

Valerie Anne Parker

PREFACE.....	v
LIST of FIGURES.....	xi
LIST of PROGRAMS and OUTPUT.....	xiii

1 Data Representations 1

1.1 Integer Representations	1
1.1.1 Unsigned Notation	5
1.1.2 Signed-Magnitude Notation	6
1.1.3 2's Complement Notation	7
1.1.4 Sign Extension	11
1.1.5 C++ Program Example	14
1.2 Floating Point Representation	16
1.2.1 IEEE 754 Standard Floating Point Representations	16
1.2.2 Bit Operators in C++	20
1.2.3 Examples	25
1.2.4 Conversion from Decimal to Binary	25
1.3 Character Formats — ASCII	26
1.4 Putting it All Together	31
1.5 Problems.....	35

2 Algorithms 37

2.1 Order.....	37
2.1.1 Justification of Using Order as a Complexity Measure	40
2.2 Induction.....	42
2.3 Recursion.....	45
2.3.1 Factorial	45
2.3.2 Fibonacci Numbers	46
2.3.3 General Recurrence Relations	50
2.3.4 Tower of Hanoi	51
2.3.5 Boolean Function Implementation	61
2.4 Graphs and Trees	62
2.5 Parallel Algorithms.....	70
2.5.1 Speedup and Amdahls Law	70
2.5.2 Pipelining	71
2.5.3 Parallel Processing and Processor Topologies	74
2.6 The Hypercube Topology	77
2.6.1 Definitions	77
2.6.2 Message Passing	78
2.6.3 Efficient Hypercubes	80
2.6.4 Visualizing the Hypercube: A C++ Example	87
2.7 Problems.....	97

3	Data Structures and Searching	101
3.1	Pointers and Dynamic Memory Allocation.....	101
3.1.1	A Double Pointer Example	106
3.1.2	Dynamic Memory Allocation with New and Delete	110
3.1.3	Arrays	112
3.1.4	Overloading in C++	117
3.2	Arrays	119
3.3	Stacks	122
3.4	Linked Lists.....	126
3.4.1	Singly Linked Lists	126
3.4.2	Circular Lists	133
3.4.3	Doubly Linked Lists	133
3.5	Operations on Linked Lists	134
3.5.1	A Linked List Example	135
3.6	Linear Search.....	148
3.7	Binary Search	149
3.8	QuickSort	150
3.9	Binary Trees	164
3.9.1	Traversing the Tree	164
3.10	Hashing.....	164
3.11	Simulated Annealing	165
3.11.1	The Square Packing Problem	165
3.12	Problems.....	184
4	Algorithms for Computer Arithmetic	187
4.1	2's Complement Addition.....	187
4.1.1	Full and Half Adder	189
4.1.2	Ripple Carry Addition	191
4.1.3	Carry Lookahead Addition	197
4.2	A Simple Hardware Simulator in C++	213
4.3	2's Complement Multiplication.....	215
4.3.1	Shift-Add Addition	221
4.3.2	Booth Algorithm	223
4.3.3	Bit-Pair Recoding	228
4.4	Fixed Point Division.....	232
4.4.1	Restoring Division	233
4.4.2	Nonrestoring Division	234
4.4.3	Shifting over 1's and 0's	240
4.4.4	Newton's Method	241
4.5	Residue Number System	244
4.5.1	Representation in the Residue Number System	244

4.5.2	Data Conversion — Calculating the Value of a Number	246
4.5.3	C++ Implementation	247
4.6	Problems.....	253

Index	255
-------------	-----

LIST of FIGURES

FIGURE 1.1	Packing Attributes into One Character	23
FIGURE 1.2	Memory Implementation for Variable t.....	33
FIGURE 1.3	Mapping of each Union Entry	33
FIGURE 2.1	Tower of Hanoi Problem	51
FIGURE 2.2	Class Structure.....	55
FIGURE 2.3	PostScript Rendering	56
FIGURE 2.4	Recursive Model for Boolean Function Evaluation	62
FIGURE 2.5	A Simple Graph.....	63
FIGURE 2.6	Subgraph of Graph in Figure 2.5.....	64
FIGURE 2.7	A Directed Graph.....	65
FIGURE 2.8	Paths in a Directed Graph.....	66
FIGURE 2.9	Cyclic and Acyclic Graphs	67
FIGURE 2.10	Trees	68
FIGURE 2.11	Planar Graph.....	69
FIGURE 2.12	Transitive Closure of a Graph	69
FIGURE 2.13	A Four Stage Pipeline.....	71
FIGURE 2.14	Pipelining	72
FIGURE 2.15	Full Crossbar Topology	75
FIGURE 2.16	Rectangular Mesh.....	76
FIGURE 2.17	Hypercube Topology	76
FIGURE 2.18	Cube-Connected Cycles	77
FIGURE 2.19	Hypercube Architecture.....	79
FIGURE 2.20	Hypercube with Failed Nodes	82
FIGURE 2.21	A 64-Node Hypercube.....	88
FIGURE 2.22	An 8-Node Hypercube.....	89
FIGURE 3.1	Memory Layout for C++ Program	103
FIGURE 3.2	Program Organization in Memory	108
FIGURE 3.3	Memory Organization for Code List 3.8	113
FIGURE 3.4	Packing Bits in Memory.....	117
FIGURE 3.5	Push and Pop in a LIFO Stack	123
FIGURE 3.6	Linked List	129
FIGURE 3.7	Circular List.....	134

FIGURE 3.8	Doubly Linked List	134
FIGURE 3.9	A Particular Game Sequence.....	136
FIGURE 3.10	Generic Simulated Annealing Algorithm	170
FIGURE 3.11	A Given Instance of the Square Packing Problem	171
FIGURE 3.12	Hill Climbing Analogy	172
FIGURE 4.1	Full and Half Adder Modules.....	191
FIGURE 4.2	Output Delay Calculation for a Full Adder	192
FIGURE 4.3	2's Complement 4-Bit Adder.....	192
FIGURE 4.4	Output Delay Calculation for a Half Adder.....	193
FIGURE 4.5	Delay Calculation	199
FIGURE 4.6	2's Complement 4-Bit CLA Adder Module	200
FIGURE 4.7	4-Bit CLA Adder Module Timing	201
FIGURE 4.8	2's Complement 4-Bit Module Representation	201
FIGURE 4.9	2's Complement 4-Bit CLA Adder Module	202
FIGURE 4.10	16-Bit CLA Adder with Group Lookahead	203
FIGURE 4.11	D Flip-Flop Circuit for Simulation.....	215
FIGURE 4.12	Transition Timing	215
FIGURE 4.13	Timing Diagram for Simulation	216
FIGURE 4.14	Booth Algorithm (page 1 of 2)	224
FIGURE 4.15	Bit Pair Recoding Algorithm.....	229

LIST of PROGRAMS and OUTPUT

Code List 1.1	Integer Example	3
Code List 1.2	Program Output of Code List 1.1	5
Code List 1.3	Assembly Language Example	15
Code List 1.4	Assembly Language Code.....	15
Code List 1.5	C++ Source Program	17
Code List 1.6	Output of Program in Code List 1.5	20
Code List 1.7	Testing the Binary Operators in C++	21
Code List 1.8	Output of Program in Code List 1.7	22
Code List 1.9	Bit Operators	23
Code List 1.10	Output of Program in Code List 1.9	24
Code List 1.11	Simple File I/O	25
Code List 1.12	Decimal to Binary Conversion	28
Code List 1.13	Decimal to Conversion C++ Program	29
Code List 1.14	Output of Program in Code List 1.13	30
Code List 1.15	Data Representations	34
Code List 1.16	Output of Program in Code List 1.15	34
Code List 2.1	Factorial	46
Code List 2.2	Output of Program in Code List 2.1	46
Code List 2.3	Fibonacci Sequence Generation	47
Code List 2.4	Output of Program in Code List 2.3	47
Code List 2.5	Fibonacci Program — Non Recursive Solution	49
Code List 2.6	Program to Display Tower of Hanoi	54
Code List 2.7	File Created by Program in Code List 2.6.....	59
Code List 2.8	Message Passing in an Efficient Hypercube	84
Code List 2.9	Output of Program in Code List 2.8	86
Code List 2.10	C++ Code to Visualize the Hypercube.....	88
Code List 2.11	Output of Program in Code List 2.10	94
Code List 3.1	Integer Pointer Example.....	101
Code List 3.2	Pointer Example	105
Code List 3.3	Output of Program in Code List 3.2	105
Code List 3.4	Double Pointer Example	106
Code List 3.5	Output of Program in Code List 3.4	107
Code List 3.6	Dynamic Memory Allocation in C++	110
Code List 3.7	Output of Program in Code List 3.6	111
Code List 3.8	Array Example	114
Code List 3.9	Output of Code in Code List 3.8	115
Code List 3.10	Output of Code in Code List 3.8	116
Code List 3.11	Operator Overloading Example.....	117
Code List 3.12	Output of Program in Code List 3.11	119
Code List 3.13	Creating an Array Class in C++	120
Code List 3.14	Output from Code List 3.13	121
Code List 3.15	LIFO Stack Class	122
Code List 3.16	Output of Program in Code List 3.15	126

Code List 3.17	Linked List Source	128
Code List 3.18	Output from Code List 3.17	133
Code List 3.19	Source Code for Game Simulation.....	137
Code List 3.20	Output of Program in Code List 3.19	146
Code List 3.21	Linear Search Code for Strings	148
Code List 3.22	Output of Program in Code List 3.21	149
Code List 3.23	Binary Search for Integers.....	149
Code List 3.24	Output of Program in Code List 3.23	150
Code List 3.25	Binary Search for Strings	151
Code List 3.26	Output of Program in Code List 3.25	152
Code List 3.27	QuickSort C++ Program.....	153
Code List 3.28	Output of Program in Code List 3.27	157
Code List 3.29	QuickSort Comparison.....	160
Code List 3.30	QuickSort For Double Types	161
Code List 3.31	Output for Program in Code List 3.30.....	161
Code List 3.32	QuickSort Program for Integers	162
Code List 3.33	Output for Program in Code List 3.32.....	162
Code List 3.34	QuickSort Program.....	163
Code List 3.35	Output of Program in Code List 3.34	163
Code List 3.36	Simulated Annealing	169
Code List 3.37	Output of Program in Code List 3.36	183
Code List 4.1	2's Complement Addition	187
Code List 4.2	Output of Program in Code List 4.1	189
Code List 4.3	Ripple Carry Addition	193
Code List 4.4	Output of Program in Code List 4.3	196
Code List 4.5	CLA Addition.....	204
Code List 4.6	Output of Program in Code List 4.5	212
Code List 4.7	Boolean Logic Simulator	216
Code List 4.8	Output of Program in Code List 4.7	219
Code List 4.9	Shift Add Technique	221
Code List 4.10	Output of Code List 4.9.....	222
Code List 4.11	Booth Algorithm	225
Code List 4.12	Output of Program in Code List 4.11	227
Code List 4.13	Bit-Pair Recoding Program	230
Code List 4.14	Output of Program in Code List 4.13	232
Code List 4.15	Nonrestoring Division	236
Code List 4.16	Output of Program in Code List 4.15	240
Code List 4.17	Floating Point Division	241
Code List 4.18	Output of Program in Code List 4.17	243
Code List 4.19	Residue Number System	248
Code List 4.20	Output of Program in Code List 4.19	251
Code List 4.21	Euler Totient Function	252
Code List 4.22	Output of Program in Code List 4.21	252

1 Data Representations

This chapter introduces the various formats used by computers for the representation of integers, floating point numbers, and characters. Extensive examples of these representations within the C++ programming language are provided.

1.1 Integer Representations

The tremendous growth in computers is partly due to the fact that physical devices can be built inexpensively which distinguish and manipulate two states at very high speeds. Since computers are devices which primarily act on two states (0 and 1), binary, octal, and hex representations are commonly used for the representation of computer data. The representation for each of these bases is shown in Table 1.1.

TABLE 1.1 Number Systems

Binary	Octal	Hexadecimal	Decimal
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8

TABLE 1.1 Number Systems (continued)

Binary	Octal	Hexadecimal	Decimal
1001	11	9	9
1010	12	A	10
1011	13	B	11
1100	14	C	12
1101	15	D	13
1110	16	E	14
1111	17	F	15
10000	20	10	16

Operations in each of these bases is analogous to base 10. In base 10, for example, the decimal number 743.57 is calculated as

$$743.57 = 7 \times 10^2 + 4 \times 10^1 + 3 \times 10^0 + 5 \times 10^{-1} + 7 \times 10^{-2} \quad (1.1)$$

In a more precise form, if a number, X , has n digits in front of the decimal and m digits past the decimal

$$X = a_{n-1}a_{n-2}\dots a_1a_0.b_{m-1}b_{m-2}\dots b_1b_0 \quad (1.2)$$

Its base 10 value would be

$$X = \sum_{j=0}^{n-1} a_j 10^j + \sum_{k=0}^{m-1} b_{m-1-k} 10^{-k} \quad 0 \leq a_j, b_j \leq 9 \quad (1.3)$$

For hexadecimal,

$$X = \sum_{j=0}^{n-1} a_j 16^j + \sum_{k=0}^{m-1} b_{m-1-k} 16^{-k} \quad 0 \leq a_j, b_j \leq F \quad (1.4)$$

For octal,

$$X = \sum_{j=0}^{n-1} a_j 8^j + \sum_{k=0}^{m-1} b_{m-1-k} 8^{-k} \quad 0 \leq a_j, b_j \leq 7 \quad (1.5)$$

In general for base r

$$X = \sum_{j=0}^{n-1} a_j r^j + \sum_{k=0}^{m-1} b_{m-1-k} r^{-k} \qquad 0 \leq a_j, b_j \leq r-1 \tag{1.6}$$

When using a theoretical representation to model an entity one can introduce a tremendous amount of bias into the thought process associated with the implementation of the entity. As an example, consider Eq. 1.6 which gives the value of a number in base r . In looking at Eq. 1.6, if a system to perform the calculation of the value is built, the natural approach is to subdivide the task into two subtasks: a subtask to calculate the integer portion and a subtask to calculate the fractional portion; however, this bias is introduced by the theoretical model. Consider, for instance, an equally valid model for the value of a number in base r . The number X is represented as

$$X = a_{n-1} a_{n-2} \dots a_k . a_{k-1} \dots a_0 \tag{1.7}$$

where the decimal point appears after the k th element. X then has the value:

$$X = r^{-k} \left(\sum_{j=0}^{n-1} a_j r^j \right) \tag{1.8}$$

Based on this model a different implementation might be chosen. While theoretical models are nice, they can often lead one astray.

As a first C++ programming example let's compute the representation of some numbers in decimal, octal, and hexadecimal for the integer type. A program demonstrating integer representations in decimal, octal, and hex is shown in Code List 1.1.

Code List 1.1 Integer Example

C++ Source Program
<pre>#include <iostream.h> int a[]={45,245,567,1014,-45,-1,256}; void main() { int i; for(i=0;i<sizeof(a)/sizeof(int);i++) { cout << endl << endl << "In decimal " << dec << a[i];</pre>