Alan Parker

ALGORITHMS and DATA STRUCTURES in C++

ALGORITHMS and DATA STRUCTURES in C++

Alan Parker
The University of Alabama in Huntsville

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
DNLM/DLC
for Library of Congress 93-20537
CIP

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

LIST of FIGURES

LIST of PROGRAMS and OUTPUT

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

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

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

$$X = a_{n-1}a_{n-2}\ldots a_1 a_0 . b_{m-1}b_{m-2}\ldots b_1 b_0 \qquad (1.2)$$

Its base 10 value would be

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

For hexadecimal,

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

For octal,

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

In general for base r

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

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

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

where the decimal point appears after the kth 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

```
#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];
```