DATA STRUCTURES
Using C
DATA STRUCTURES
Using C

Rohit Khurana
Founder and CEO
ITLES, Delhi

VIKAS PUBLISHING HOUSE PVT LTD
Data Structures Using C
ISBN: 978-93259-7565-1

Vikas® is the registered trademark of Vikas Publishing House Pvt Ltd

Copyright © Author, 2014

All rights reserved. No part of this publication which is material protected by this copyright notice may be reproduced or transmitted or utilized or stored in any form or by any means now known or hereinafter invented, electronic, digital or mechanical, including photocopying, scanning, recording or by any information storage or retrieval system, without prior written permission from the publisher.

Information contained in this book has been published by VIKAS® Publishing House Pvt Ltd and has been obtained by its Authors from sources believed to be reliable and are correct to the best of their knowledge. However, the Publisher and its Authors shall in no event be liable for any errors, omissions or damages arising out of use of this information and specifically disclaim any implied warranties or merchantability or fitness for any particular use. Disputes if any are subject to Delhi Jurisdiction only.

Printed in India.
Dedication

I would like to dedicate this book to my uncle,
Shri G K Jaju
for his understanding, constant support and trust in me.

He has always inspired me to aim higher and achieve bigger.
He has been a great friend, philosopher and guide.

ROHIT KHURANA

Founder and CEO
ITLESL, Delhi
Data is the essence of every application that we design. The performance of any application largely depends on the way the data is organized and structured. The organization of data also plays an important role in the design and implementation of efficient algorithms and program development. There exist many ways of organizing data and each specific organization of data can be represented by means of a logical or mathematical model, referred to as data structure. A variety of data structures are available depending on the type of application. The choice of any data structure depends on its simplicity and effectiveness in processing of data.

Keeping pace with the ever increasing importance of data structures, most universities have integrated this subject in their curriculum. The book in your hands, *Data Structures Using C*, covers all the necessary material in the subject as per the syllabi of most universities. The text has been carefully organized to make it particularly easy for the students to understand the introductory as well as advanced concepts of data structures. It lays emphasis on theory and design of data structures and their implementation using ‗C‘ programming language. An attempt has been made to make the book self-contained so that the students can learn the subject by them.

**Key Features**
- Lucid, well-explained concepts with clear examples
- Complete coverage of the subject as per the syllabi of most universities
- Relevant illustrations for better understanding and retention
- Emphasis on the practical application of theory
- Chapter-end summary and questions for quick review
- Numerous programming examples for better clarity of concept
- A comprehensive index at the end of the book for quick access to the topics

**Text Layout**
The text is organized into 12 chapters.
- *Chapter 1* provides an overview of the programming language C. It discusses various elements of C such as data types, operators, expressions, etc.
• Chapter 2 introduces various data structures and discusses how algorithms can be analyzed.
• Chapter 3 deals with one-, two-, and multi-dimensional arrays with programming examples and discusses various application areas where they can be used. It also discusses string handling.
• Chapter 4 covers the concept of linked list, operations that can be performed on them and various application areas where they can be used.
• Chapter 5 covers the concept of stacks, operations that can be performed on them and various application areas where they can be used.
• Chapter 6 focuses on the concept of queues, operations that can be performed on them and various application areas where they can be used.
• Chapter 7 explains the concept of trees, operations that can be performed on them, their types and applications.
• Chapter 8 covers the concept of graphs, operations that can be performed on them and their various applications.
• Chapter 9 presents various sorting and searching algorithms along with their implementation in C and complexity.
• Chapter 10 gives an overview of hashing, various hashing functions and collision resolution techniques.
• Chapter 11 is devoted to set data structure. It describes different representations of sets, disjoint set ADT and various applications of sets.
• Chapter 12 introduces basic terminology related to files and various file organization techniques.

ACKNOWLEDGEMENTS

In all my efforts towards making this book a reality, my special thanks goes to my technical and editorial team, without whom this work would not have achieved its desired level of excellence. I sincerely extend my thanks to my research and development team for devoting their time and relentless effort in bringing out this high-quality book. I convey my gratitude to my publisher Vikas Publishing House Pvt. Ltd for sharing this dream and giving all the support in realizing it.

In our attempt towards further improvement, I welcome you all to send your feedback to itlesl@rediffmail.com. I will highly appreciate all your constructive comments.

I hope you will enjoy reading the book and hope it proves to be a good resource for all.

Rohit Khurana
Founder and CEO
ITLESL, Delhi
Contents

Preface vii

1. Overview of C Language 1
   1.1 Introduction 1
   1.2 Data Types 2
   1.3 Control Statements 4
   1.4 Functions 16
   1.5 Pointers 18
   1.6 Memory use in C 26
   1.7 Debugging Pointers 28
   1.8 Structures 30

2. Introduction to Data Structures and Algorithms 39
   2.1 Introduction 39
   2.2 Elementary Data Organization 40
   2.3 Data Type 40
   2.4 Data Structure 42
   2.5 Algorithms 47
   2.6 Mathematical Notations and Functions 55
   2.7 Algorithm Analysis 58
   2.8 Pseudocode 62
   2.9 Programs 64

3. Arrays and Strings 69
   3.1 Introduction 69
   3.2 Arrays 69
   3.3 Single-Dimensional Arrays 70
   3.4 Multi-Dimensional Arrays 82
   3.5 Use of Arrays in Function 89
   3.6 Pointers with Arrays 92
   3.7 Character String Handling 95
3.8 Polynomial Representation and Manipulation 104
3.9 Sparse Matrices 109

4. Linked List 124
4.1 Introduction 124
4.2 Array Based Implementation of List 124
4.3 Linked List Implementation 125
4.4 Singly Linked Lists 125
4.5 Circular Linked Lists 153
4.6 Doubly Linked Lists 165
4.7 Multilinked Lists 179
4.8 Header Linked Lists 180
4.9 Cursor-Based Linked List 185
4.10 Generalized List 185
4.11 Applications of Linked Lists 186
4.12 Garbage Collection and Compaction 206

5. Stacks 211
5.1 Introduction 211
5.2 Stack: Definition and Concept 211
5.3 Operations on Stacks 212
5.4 Implementation of Stacks 212
5.5 Multi-Stacks 223
5.6 Applications of Stacks 230
5.7 Recursion 255

6. Queues 274
6.1 Introduction 274
6.2 Queues: Definition and Concept 274
6.3 Operations on Queues 275
6.4 Representation of Queues 275
6.5 Circular Queue 287
6.6 Priority Queue 290
6.7 Deque 297
6.8 Application of Queues 306

7. Trees 312
7.1 Introduction 312
7.2 General Trees 312
7.3 Binary Trees 313
7.4 Trees and Forests 325
7.5 Binary Search Tree 330
7.6 Height Balanced Tree (AVL Tree) 350
7.7 Expression Tree 355
7.8 Threaded Binary Tree 356
7.9 Heap Tree 361
7.10 B-Trees 372
7.11 Splay Trees 379
7.12 Red-Black Tree 380
7.13 Applications of Trees 382

8. Graphs 394
  8.1 Introduction 394
  8.2 Graph: Definition and Terminologies 394
  8.3 Representations of Graph 398
  8.4 Operations on Graphs 403
  8.5 Applications of Graphs 416

9. Sorting and Searching 444
  9.1 Introduction 444
  9.2 Use of Various Data Structures for Searching and Sorting 445
  9.3 Internal Sorting 445
  9.4 External Sorting 485
  9.5 Comparison of Sorting Methods 489
  9.6 Skip List 493
  9.7 Searching 493
  9.8 Comparison of Searching Methods 501

10. Hashing 507
  10.1 Introduction 507
  10.2 Basic Idea of Hashing 507
  10.3 Hash Functions 508
  10.4 Collision Resolution Techniques 510
  10.5 Rehashing 521

11. Sets 525
  11.1 Introduction 525
  11.2 Sets: Definition and Terminologies 525
  11.3 Representation of Sets 527
  11.4 Disjoint Set ADT 529
  11.5 Applications of Sets 535
12. **File Structures** 541
   12.1 Introduction 541
   12.2 File Concept 542
   12.3 Organization of Records in Files 546
   12.4 File Systems 548
   12.5 File Organization 553
   12.6 Indexing 558

Index 569
1.1 INTRODUCTION

In the early 1970s, the high-level programming language ‘C’ was developed by Dennis Ritchie at the Bell Laboratories in USA for general purpose. The language was named ‘C’ because it was the successor to a language named ‘B’, which was developed by Ken Thompson in 1969–70. Dennis Ritchie extended the features of B and turned into C by adding some more features in it. C was developed as a high-level language so that it could be used to rewrite the UNIX operating system which was earlier written in assembly language. C was initially used to develop system software such as operating systems, compilers, interpreters, assemblers, databases, text editors, utilities, etc. One of the most interesting features of C language is that its compiler is written in C itself.

In 1978, Brian Kernighan and Dennis Ritchie wrote a book “The C Programming Language”. After this book, the de facto standard for C programming language was known as K & R standard for many years. However, there were many changes made unofficially to the C language but that were not present in the K & R standard. Due
to this reason, a group of compiler vendors and software developers approached
the American Standards Institute (ANSI) in 1983 to build a standard for the C
language, and by the end of 1989, the committee approved the ANSI standard for
C programming language.

This chapter will give a brief review of the C programming language and discusses
features of C in data structures. For more details on the language, readers can
consult a textbook on C. We have adhered to the ANSI standard definition of C. Our
discussion is based on the assumption that the reader is familiar with C language or
any other similar high-level language.

1.2 DATA TYPES

A data type determines the value that a variable can take and the operations that
can be performed on that variable. C provides several primitive (built-in) data types.
It provides the facility of defining new data types according to the requirements of
the programmer. In this section, we will discuss the primitivedata types provided
by C and composite data types such as arrays and pointers that are derived from
the primitive data types. We will also discuss structures and union which areuser-
defined data types.

1.2.1 Primitive Data Types

The basic (fundamental) data types provided by C are \texttt{int}, \texttt{float}, \texttt{char} and \texttt{double}.
The data types \texttt{char} and \texttt{int} are collectively known as \textit{integral data types}. The \texttt{char}
data type occupies 1 byte of memory (that is, it holds only one character at a time).
It can be preceded by \texttt{signed} and \texttt{unsigned} modifiers. The \texttt{int} data type is used
to store integers such as 41, 12, 521, –321, etc. Like \texttt{char} data type, \texttt{int} can also be
qualified with \texttt{signed} and \texttt{unsigned} modifiers. In addition to these modifiers, it
can be also be qualified with \texttt{short} and \texttt{long} modifiers. The default modifier of \texttt{int}
data type is \texttt{signed}. The \texttt{float} and \texttt{double} data types are used to store numbers
with decimal point. The \texttt{float} specifies single-precision floating-point numbers and
\texttt{double} specifies double-precision floating-point numbers. The various primitive
data types with their size and range are listed in Table 1.1.

<table>
<thead>
<tr>
<th>Type</th>
<th>Size (in bytes)</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{int}</td>
<td>2</td>
<td>–32,768 to 32,767</td>
</tr>
<tr>
<td>\texttt{signedint}</td>
<td>2</td>
<td>–32,768 to 32,767</td>
</tr>
<tr>
<td>\texttt{unsignedint}</td>
<td>2</td>
<td>0 to 65,535</td>
</tr>
<tr>
<td>\texttt{shortint}</td>
<td>2</td>
<td>–32,768 to 32,767</td>
</tr>
<tr>
<td>\texttt{signedshortint}</td>
<td>2</td>
<td>–32,768 to 32,767</td>
</tr>
<tr>
<td>\texttt{unsignedshortint}</td>
<td>2</td>
<td>0 to 65,535</td>
</tr>
<tr>
<td>\texttt{longint}</td>
<td>4</td>
<td>–2,147,483,648 to 2,147,483,647</td>
</tr>
</tbody>
</table>

(Contd...)