Brief Contents

Preface to the Second Edition v

Preface to the First Edition viii

1. Introduction to C 1

2. Introduction to Data Structures and Algorithms 43

3. Arrays 66

4. Strings 115

5. Structures and Unions 138

6. Linked Lists 162

7. Stacks 219

8. Queues 253

9. Trees 279

10. Efficient Binary Trees 298

11. Multi-way Search Trees 344

12. Heaps 361

13. Graphs 383

14. Searching and Sorting 424

15. Hashing and Collision 464

16. Files and Their Organization 489

Appendix A: Memory Allocation in C Programs 505

Appendix B: Garbage Collection 512

Appendix C: Backtracking 514

Appendix D: Josephus Problem 516

Appendix E: File Handling in C 518

Appendix F: Address Calculation Sort 520

Appendix G: Answers 522

Index 528

Detailed Contents

Preface to the Second Edition *v*
Preface to the First Edition *viii*

1. Introduction to C **1**

1.1 Introduction *1*
1.2 Identifiers and Keywords *2*
1.3 Basic Data Types *2*
1.4 Variables and Constants *3*
1.5 Writing the First C Program *5*
1.6 Input and Output Functions *6*
1.7 Operators and Expressions *9*
1.8 Type Conversion and Typecasting *16*
1.9 Control Statements *17*
1.9.1 Decision Control Statements *17*
1.9.2 Iterative Statements *22*
1.9.3 Break and Continue Statements *27*
1.10 Functions *28*
1.10.1 Why are Functions Needed? *29*
1.10.2 Using Functions *29*
1.10.3 Passing Parameters to Functions *31*
1.11 Pointers *34*
1.11.1 Declaring Pointer Variables *35*
1.11.2 Pointer Expressions and Pointer Arithmetic *36*
1.11.3 Null Pointers *36*
1.11.4 Generic Pointers *36*
1.11.5 Pointer to Pointers *37*
1.11.6 Drawback of Pointers *38*

2. Introduction to Data Structures and Algorithms **43**

2.1 Basic Terminology *43*
2.1.1 Elementary Data Structure Organization *45*
2.2 Classification of Data Structures *45*
2.3 Operations on Data Structures *49*
2.4 Abstract Data Type *50*
2.5 Algorithms *50*
2.6 Different Approaches to Designing an Algorithm *51*
2.7 Control Structures Used in Algorithms *52*
2.8 Time and Space Complexity *54*
2.8.1 Worst-case, Average-case, Best-case, and Amortized Time Complexity *54*
2.8.2 Time–Space Trade-off *55*
2.8.3 Expressing Time and Space Complexity *55*
2.8.4 Algorithm Efficiency *55*
2.9 Big O Notation *57*
2.10 Omega Notation (Ω) *60*
2.11 Theta Notation (Θ) *61*
2.12 Other Useful Notations *62*

3. Arrays **66**

3.1 Introduction *66*
3.2 Declaration of Arrays *67*
3.3 Accessing the Elements of an Array *68*
3.3.1 Calculating the Address of Array Elements *68*
3.3.2 Calculating the Length of an Array *69*
3.4 Storing Values in Arrays *69*
3.5 Operations on Arrays *71*
3.5.1 Traversing an Array *71*

3.5.2 Inserting an Element in an Array *76*
3.5.3 Deleting an Element from an Array *79*
3.5.4 Merging Two Arrays *82*
3.6 Passing Arrays to Functions *86*
3.6.1 Passing Individual Elements *86*
3.6.2 Passing the Entire Array *87*
3.7 Pointers and Arrays *90*
3.8 Arrays of Pointers *92*
3.9 Two-dimensional Arrays *93*
3.9.1 Declaring Two-dimensional Arrays *93*
3.9.2 Initializing Two-dimensional Arrays *95*
3.9.3 Accessing the Elements of Two-dimensional Arrays *96*
3.10 Operations on Two-Dimensional Arrays *99*
3.11 Passing Two-dimensional Arrays to Functions *103*
3.12 Pointers and Two-dimensional Arrays *105*
3.13 Multi-dimensional Arrays *107*
3.14 Pointers and Three-dimensional Arrays *109*
3.15 Sparse Matrices *110*
3.16 Applications of Arrays *111*

4. Strings 115

4.1 Introduction *115*
4.1.1 Reading Strings *117*
4.1.2 Writing Strings *118*
4.2 Operations on Strings *118*
4.3 Arrays of Strings *129*
4.4 Pointers and Strings *132*

5. Structures and Unions 138

5.1 Introduction *138*
5.1.1 Structure Declaration *138*
5.1.2 `Typedef` Declarations *139*
5.1.3 Initialization of Structures *140*
5.1.4 Accessing the Members of a Structure *141*
5.1.5 Copying and Comparing Structures *142*
5.2 Nested Structures *144*
5.3 Arrays of Structures *146*
5.4 Structures and Functions *148*
5.4.1 Passing Individual Members *149*
5.4.2 Passing the Entire Structure *149*
5.4.3 Passing Structures through Pointers *152*
5.5 Self-referential Structures *155*
5.6 Unions *155*
5.6.1 Declaring a Union *156*
5.6.2 Accessing a Member of a Union *156*
5.6.3 Initializing Unions *156*
5.7 Arrays of Union Variables *157*
5.8 Unions Inside Structures *158*

6. Linked Lists 162

6.1 Introduction *162*
6.1.1 Basic Terminologies *162*
6.1.2 Linked Lists versus Arrays *164*
6.1.3 Memory Allocation and De-allocation for a Linked List *165*
6.2 Singly Linked Lists *167*
6.2.1 Traversing a Linked List *167*
6.2.2 Searching for a Value in a Linked List *167*
6.2.3 Inserting a New Node in a Linked List *168*
6.2.4 Deleting a Node from a Linked List *172*
6.3 Circular Linked Lists *180*
6.3.1 Inserting a New Node in a Circular Linked List *181*
6.3.2 Deleting a Node from a Circular Linked List *182*
6.4 Doubly Linked Lists *188*
6.4.1 Inserting a New Node in a Doubly Linked List *188*

6.4.2 Deleting a Node from a Doubly Linked List *191*
6.5 Circular Doubly Linked Lists *199*
6.5.1 Inserting a New Node in a Circular Doubly Linked List *200*
6.5.2 Deleting a Node from a Circular Doubly Linked List *201*
6.6 Header Linked Lists *207*
6.7 Multi-linked Lists *210*
6.8 Applications of Linked Lists *211*
6.8.1 Polynomial Representation *211*

7. Stacks 219

7.1 Introduction to Stacks *219*
7.2 Array Representation of Stacks *220*
7.3 Operations on a Stack *221*
7.3.1 Push Operation *221*
7.3.2 Pop Operation *221*
7.3.3 Peek Operation *222*
7.4 Linked Representation of Stacks *224*
7.5 Operations on a Linked Stack *224*
7.5.1 Push Operation *224*
7.5.2 Pop Operation *225*
7.6 Multiple Stacks *227*
7.7 Applications of Stacks *230*
7.7.1 Reversing a List *230*
7.7.2 Implementing Parentheses Checker *231*
7.7.3 Evaluation of Arithmetic Expressions *232*
7.7.4 Recursion *243*

8. Queues 253

8.1 Introduction to Queues *253*
8.2 Array Representation of Queues *254*
8.3 Linked Representation of Queues *256*
8.4 Types of Queues *260*
8.4.1 Circular Queues *260*
8.4.2 Deques *264*
8.4.3 Priority Queues *268*
8.4.4 Multiple Queues *272*
8.5 Applications of Queues *275*

9. Trees 279

9.1 Introduction *279*
9.1.1 Basic Terminology *279*
9.2 Types of Trees *280*
9.2.1 General Trees *280*
9.2.2 Forests *280*
9.2.3 Binary Trees *281*
9.2.4 Binary Search Trees *285*
9.2.5 Expression Trees *285*
9.2.6 Tournament Trees *286*
9.3 Creating a Binary Tree from a General Tree *286*
9.4 Traversing a Binary Tree *287*
9.4.1 Pre-order Traversal *287*
9.4.2 In-order Traversal *288*
9.4.3 Post-order Traversal *289*
9.4.4 Level-order Traversal *289*
9.4.5 Constructing a Binary Tree from Traversal Results *290*
9.5 Huffman's Tree *290*
9.6 Applications of Trees *294*

10. Efficient Binary Trees 298

10.1 Binary Search Trees *298*
10.2 Operations on Binary Search Trees *300*
10.2.1 Searching for a Node in a Binary Search Tree *300*
10.2.2 Inserting a New Node in a Binary Search Tree *301*
10.2.3 Deleting a Node from a Binary Search Tree *301*
10.2.4 Determining the Height of a Binary Search Tree *303*
10.2.5 Determining the Number of Nodes *303*
10.2.6 Finding the Mirror Image of a Binary Search Tree *305*
10.2.8 Finding the Smallest Node in a Binary Search Tree *305*
10.2.9 Finding the Largest Node in a Binary Search Tree *306*
10.3 Threaded Binary Trees *311*
10.3.1 Traversing a Threaded Binary Tree *314*

10.4 AVL Trees *316*
10.4.1 Operations on AVL Trees *317*
Searching for a Node in an AVL Tree *317*
10.5 Red-Black Trees *327*
10.5.1 Properties of Red-Black Trees *328*
10.5.2 Operations on Red-Black Trees *330*
10.5.3 Applications of Red-Black Trees *337*
10.6 Splay Trees *337*
10.6.1 Operations on Splay Trees *338*
10.6.2 Advantages and Disadvantages of Splay Trees *340*

11. Multi-way Search Trees 344

11.1 Introduction to M-Way Search Trees *344*
11.2 B Trees *345*
11.2.1 Searching for an Element in a B Tree *346*
11.2.2 Inserting a New Element in a B Tree *346*
11.2.3 Deleting an Element from a B Tree *347*
11.2.4 Applications of B Trees *350*
11.3 B+ Trees *351*
11.3.1 Inserting a New Element in a B+ Tree *352*
11.3.2 Deleting an Element from a B+ Tree *352*
11.4 2-3 Trees *353*
11.4.1 Searching for an Element in a 2-3 Tree *354*
11.4.2 Inserting a New Element in a 2-3 Tree *354*
11.4.3 Deleting an Element from a 2-3 Tree *356*
11.5 Trie *358*

12. Heaps 361

12.1 Binary Heaps *361*
12.1.1 Inserting a New Element in a Binary Heap *362*
12.1.2 Deleting an Element from a Binary Heap *364*
12.1.3 Applications of Binary Heaps *364*
12.2 Binomial Heaps *365*
12.2.1 Linked Representation of Binomial Heaps *366*
12.2.2 Operations on Binomial Heaps *366*
12.3 Fibonacci Heaps *373*
12.3.1 Structure of Fibonacci Heaps *373*
12.3.2 Operations on Fibonacci Heaps *374*
12.4 Comparison of Binary, Binomial, and Fibonacci Heaps *379*
12.5 Applications of Heaps *379*

13. Graphs 383

13.1 Introduction *383*
13.2 Graph Terminology *384*
13.3 Directed Graphs *385*
13.3.1 Terminology of a Directed Graph *385*
13.3.2 Transitive Closure of a Directed Graph *386*
13.4 Bi-connected Components *387*
13.5 Representation of Graphs *388*
13.5.1 Adjacency Matrix Representation *388*
13.5.2 Adjacency List Representation *390*
13.5.3 Adjacency Multi-List Representation *391*
13.6 Graph Traversal Algorithms *393*
13.6.1 Breadth-First Search Algorithm *394*
13.6.2 Depth-first Search Algorithm *397*
13.7 Topological Sorting *400*
13.8 Shortest Path Algorithms *405*
13.8.1 Minimum Spanning Trees *405*
13.8.2 Prim's Algorithm *407*
13.8.3 Kruskal's Algorithm *409*
13.8.4 Dijkstra's Algorithm *413*
13.8.5 Warshall's Algorithm *414*

13.8.6 Modified Warshall's Algorithm *417*
13.9 Applications of Graphs *419*

14. Searching and Sorting 424

14.1 Introduction to Searching *424*
14.2 Linear Search *424*
14.3 Binary Search *426*
14.4 Interpolation Search *428*
14.5 Jump Search *430*
14.6 Introduction to Sorting *433*
14.6.1 Sorting on Multiple Keys *433*
14.6.2 Practical Considerations for Internal Sorting *434*
14.7 Bubble Sort *434*
14.8 Insertion Sort *438*
14.9 Selection Sort *440*
14.10 Merge Sort *443*
14.11 Quick Sort *446*
14.12 Radix Sort *450*
14.13 Heap Sort *454*
14.14 Shell Sort *456*
14.15 Tree Sort *458*
14.16 Comparison of Sorting Algorithms *460*
14.17 External Sorting *460*

15. Hashing and Collision 464

15.1 Introduction *464*
15.2 Hash Tables *465*
15.3 Hash Functions *466*
15.4 Different Hash Functions *467*
15.4.1 Division Method *467*
15.4.2 Multiplication Method *467*
15.4.3 Mid-Square Method *468*
15.4.4 Folding Method *468*
15.5 Collisions *469*
15.5.1 Collision Resolution by Open Addressing *469*
15.5.2 Collision Resolution By Chaining *481*
15.6 Pros and Cons of Hashing *485*
15.7 Applications of Hashing *485*
Real World Applications of Hashing *486*

16. Files and Their Organization 489

16.1 Introduction *489*
16.2 Data Hierarchy *489*
16.3 File Attributes *490*
16.4 Text and Binary Files *491*
16.5 Basic File Operations *492*
16.6 File Organization *493*
16.6.1 Sequential Organization *493*
16.6.2 Relative File Organization *494*
16.6.3 Indexed Sequential File Organization *495*
16.7 Indexing *496*
16.7.1 Ordered Indices *496*
16.7.2 Dense and Sparse Indices *497*
16.7.3 Cylinder Surface Indexing *497*
16.7.4 Multi-level Indices *498*
16.7.5 Inverted Indices *499*
16.7.6 B-Tree Indices *500*
16.7.7 Hashed Indices *501*

Appendix A: Memory Allocation in C Programs 505
Appendix B: Garbage Collection 512
Appendix C: Backtracking 514
Appendix D: Josephus Problem 516
Appendix E: File Handling in C 518
Appendix F: Address Calculation Sort 520
Appendix G: Answers 522

Index 528