

CD-ROM Included

# DATA STRUCTURES THROUGH

# C

# IN C++

S. K. SRIVASTAVA  
DEEPALI SRIVASTAVA



PUBLICATIONS

Contents

**CHAPTER 1 INTRODUCTION ..... 1-28**

1.1 Abstract Data Type ..... 1

1.2 String Operations ..... 2

    1.21 Length ..... 2

    1.22 Concatenation ..... 3

    1.23 Comparison ..... 4

    1.24 Copy ..... 5

    1.25 Substring ..... 5

1.3 Pattern matching algorithm ..... 6

    1.31 First approach ..... 7

    1.32 Second Approach ..... 8

1.4 Complexity of Algorithm ..... 10

    1.41 Best case ..... 10

    1.42 Worst case ..... 11

    1.43 Average case ..... 11

1.5 O Notation ..... 11

1.6 Design of Algorithm ..... 12

    1.61 Greedy Algorithm ..... 12

    1.62 Divide and conquer ..... 12

    1.63 Non recursive algorithm ..... 12

    1.64 Randomized algorithm ..... 12

    1.65 Modular programming approach ..... 12

1.7 Mathematical Notations And Methods ..... 13

    1.71 Factorial ..... 13

    1.72 Series ..... 13

    1.73 Logarithms ..... 14

    1.74 Exponents ..... 14

    1.75 Modular operation ..... 14

    1.76 Fibonacci series ..... 15

    1.77 Arithmetic series ..... 15

1.8 Matrices ..... 16

    1.81 Matrix Addition ..... 16

    1.82 Matrix Multiplication ..... 17

    1.83 Lower triangular matrix ..... 17

    1.84 Upper triangular matrix ..... 18

    1.85 Tridiagonal matrix ..... 18

    1.86 Transpose matrix ..... 18

1.9 Recursion ..... 19

    1.91 Tail Recursion ..... 22

1.10 Recursion through stack ..... 23

1.11 Removal of recursion through iteration ..... 23

1.12 Tower of hanoi ..... 24

Exercise ..... 28

<b>CHAPTER 2 ARRAYS, POINTERS AND STRUCTURES .....</b>		<b>29-88</b>
2.1	What is array .....	29
2.2	Declaration of array .....	29
2.3	Processing with array .....	30
2.4	String library functions .....	30
2.41	strlen ( ) .....	30
2.42	strcmp ( ) .....	31
2.43	strcpy ( ) .....	31
2.44	strcat ( ) .....	31
2.5	Initialization of array .....	31
2.6	Use of array in function .....	32
2.7	Multidimensional array .....	33
2.8	What is pointer .....	35
2.9	Uses of pointer .....	35
2.10	The & and * operator .....	35
2.11	Declaration of pointer .....	36
2.12	Pointer to pointer .....	39
2.13	Pointer arithmetic .....	40
2.14	Pointers and functions .....	43
2.141	Call by value .....	44
2.142	Call by reference .....	45
2.15	Pointer and array .....	49
2.16	Pointer with multidimensional array .....	55
2.17	Array of pointers .....	58
2.18	Pointer and string .....	60
2.19	Two dimensional array of characters .....	62
2.20	Array of pointer to string .....	63
2.21	Dynamic memory allocation .....	64
2.211	sizeof ( ) .....	65
2.212	malloc ( ) .....	65
2.213	calloc ( ) .....	66
2.214	free ( ) .....	67
2.215	realloc ( ) .....	67
2.22	What is structure .....	68
2.23	Another type of declaration .....	69
2.24	Initialize value to the structure .....	69
2.25	Another type of initialization .....	69
2.26	Array of structures .....	70
2.27	Passing structure to function .....	70
2.28	Passing array of structure to function .....	70
2.29	Structure within structure .....	71
2.30	typedef .....	72
2.31	union .....	73
2.32	Pointer to structure .....	74
Exercise	.....	79

**CHAPTER 3 LINKED LIST ..... 89-138**

3.1 What is list ..... 89

3.2 Array Implementation of list ..... 89

    3.21 Traversing an array list ..... 90

    3.22 Searching in an array list ..... 90

    3.23 Insertion into an array list ..... 90

    3.24 Deletion from an array list ..... 91

3.3 Linked List ..... 95

    3.31 Traversing a linked list ..... 96

    3.32 Searching into a linked list ..... 96

    3.33 Insertion into a linked list ..... 97

    3.34 Deletion from a linked list ..... 98

3.4 Reverse linked list ..... 104

    3.41 Creation of reverse( ) ..... 104

3.5 Circular linked list ..... 108

    3.51 Creation of circular linked list ..... 108

    3.52 Traversal in circular linked list ..... 108

    3.53 Insertion into a circular Linked list ..... 109

    3.54 Deletion from circular linked list ..... 109

3.6 Sorted linked list ..... 114

3.7 Double linked list ..... 118

    3.71 Traversing a doubly linked list ..... 119

    3.72 Insertion into a doubly linked list ..... 119

    3.73 Deletion from doubly linked list ..... 121

3.8 Polynomial arithmetic with linked list ..... 127

3.9 Creation of polynomial linked list ..... 128

3.10 Addition with polynomial linked list ..... 128

    3.101 Implementation ..... 130

Exercise ..... 136

**CHAPTER 4 STACK AND QUEUE ..... 139-188**

4.1 Stack ..... 139

4.2 Array Implementation of Stack ..... 141

    4.21 Push operation on stack ..... 141

    4.22 Pop operation on stack ..... 141

4.3 Linked List Implementation ..... 143

    4.31 Push operation on Stack ..... 144

    4.32 Pop operation on Stack ..... 144

4.4 Queue ..... 146

4.5 Array Implementation of Queue ..... 147

    4.51 Add operation in queue ..... 148

    4.52 Delete operation in queue ..... 148

4.6 Linked List implementation ..... 151

    4.61 Add operation in Queue ..... 151

    4.62 Delete operation in Queue ..... 152

4.7	Circular Queue .....	15
4.71	Add operation in Circular Queue .....	15
4.72	Delete operation in Circular Queue .....	15
4.8	Priority Queue .....	16
4.9	Linked list implementation of priority queue .....	16
4.91	Operation in priority queue .....	16
4.911	Add operation in Priority Queue .....	16
4.912	Delete operation in Priority Queue .....	16
4.10	Dequeue .....	16
4.11	Array Implementation of Dequeue .....	16
4.111	Add and delete operation in Dequeue .....	16
4.12	Applications of stack .....	17
4.121	Reversal of string .....	17
4.122	Checking validity of an expression containing nested parentheses .....	17
4.13	Polish Notation with arithmetic expression .....	17
4.14	Polish Notation .....	17
4.15	Converting infix expression into postfix expression .....	17
4.16	Evaluation of postfix expression .....	17
4.17	Sparse Matrix .....	18
4.171	3-tuple Method .....	18
	Exercise .....	18

## **CHAPTER 5 TREES** ..... 189-297

5.1	Binary Tree .....	189
5.2	Strictly Binary Tree .....	191
5.3	Complete Binary Tree .....	191
5.4	Extended Binary Tree .....	192
5.5	Algebraic Expression representation in tree .....	192
5.6	Representation of Binary Tree .....	194
5.61	Linked Representation .....	194
5.7	Traversing in Binary Tree .....	195
5.71	Preorder Traversal .....	197
5.72	Inorder Traversal .....	197
5.73	Postorder Traversal .....	198
5.8	Non recursive functions for traversals .....	201
5.81	Preorder Traversal .....	201
5.82	Inorder Traversal .....	201
5.83	Postorder Traversal .....	202
5.9	Level order traversal .....	204
5.10	Creation of binary tree from preorder and inorder traversals .....	204
5.11	Creation of tree from postorder and inorder traversals .....	207
5.12	Shortcut method of creating the tree from preorder and inorder traversal .....	208
5.13	Binary Search Tree .....	210
5.131	Search and Insertion Operations .....	211
5.1311	Creation of find( ) .....	212
5.1312	Insertion in Binary Search tree .....	212

5.132	Deletion operation .....	213
5.1322	Creation of function case a() .....	217
5.1322	Creation of function case b() .....	217
5.1323	Creation of function case c() .....	218
5.133	Traversal in Binary Search Tree .....	221
5.1331	Preorder Traversal .....	221
5.1332	Inorder Traversal .....	221
5.1333	Postorder Traversal .....	221
5.134	Recursive Function for finding a node in Binary search tree .....	227
5.14	Threads .....	227
5.141	Finding inorder successor of a node in in-threaded tree .....	230
5.142	Finding inorder predecessor of a node in in-threaded tree .....	231
5.143	Inorder Traversal in in-threaded binary tree .....	231
5.144	Preorder traversal of in-threaded binary tree .....	232
5.145	Insertion and deletion in threaded binary tree .....	233
5.1451	Insertion in a threaded binary search tree .....	233
5.1452	Deletion from a threaded binary search tree .....	234
5.15	AVL Tree(Balanced Tree) .....	242
5.151	Insertion in AVL tree .....	244
5.152	AVL Rotations .....	246
5.1521	Left to Left rotation .....	246
5.1522	Right to Right rotation .....	247
5.1523	Left to right rotation .....	248
5.1524	Right to left rotation .....	250
5.16	Huffman Tree .....	258
5.161	Huffman Algorithm .....	259
5.162	Use in application .....	262
5.17	Heap .....	263
5.171	Insertion in Heap .....	264
5.172	Deletion in heap .....	267
5.18	General Tree .....	273
5.181	Linked representation of General Tree .....	274
5.19	B Tree .....	275
5.191	Insertion in B tree .....	276
5.192	Deletion in B-tree .....	280
5.20	B+ tree .....	289
5.201	Insertion in B+ tree .....	290
5.202	Deletion in B+ tree .....	290
5.21	Digital Search Tree .....	290
5.22	Trie .....	291
5.221	Traversal in trie .....	292
5.222	Insertion in Trie .....	292
5.223	Deletion in Trie .....	293
5.224	Analysis .....	293
5.23	Optimum Search Tree .....	293
Exercise	.....	297

<b>CHAPTER 6. GRAPH</b> .....	<b>298-378</b>
6.1 Undirected Graph .....	298
6.2 Directed Graph .....	298
6.3 Representation of Graph .....	301
6.31 Adjacency Matrix .....	301
6.32 Adjacency List .....	304
6.4 Operations on Graph .....	306
6.41 Insertion in Adjacency Matrix .....	307
6.411 Node insertion .....	307
6.412 Edge insertion .....	307
6.42 Deletion in adjacency matrix .....	308
6.421 Node deletion .....	308
6.422 Edge deletion .....	308
6.43 Insertion in adjacency list .....	312
6.431 Node insertion .....	312
6.432 Edge insertion .....	313
6.44 Deletion in adjacency list .....	313
6.441 Node deletion .....	313
6.442 Edge deletion .....	314
6.5 Path Matrix .....	320
6.6 Computing Path matrix from powers of adjacency matrix .....	320
6.7 Warshall's Algorithm .....	325
6.8 Modified Warshall's Algorithm .....	329
6.9 Traversal In Graph .....	334
6.91 Breadth First Search .....	334
6.911 Breadth First Search through queue .....	336
6.92 Depth First Search .....	338
6.921 Depth First Search through stack .....	339
6.10 Shortest Path Algorithm ( Dijkstra ) .....	347
6.11 Spanning Tree .....	356
6.12 Minimum Spanning Tree .....	356
6.13 Prim's Algorithm .....	356
6.14 Kruskal's Algorithm .....	363
6.15 Topological Sorting .....	370
Exercise .....	376
<b>CHAPTER 7 SORTING</b> .....	<b>379-423</b>
7.1 What is sorting .....	379
7.2 Efficiency Parameters .....	381
7.3 Efficiency of sorting .....	381
7.4 Bubble Sort .....	382
7.41 Analysis .....	384
7.5 Selection Sort .....	385
7.51 Analysis .....	387
7.6 Insertion Sort .....	387

7.61	Analysis .....	389
7.7	Shell Sort .....	390
7.71	Analysis .....	392
7.8	Merging .....	393
7.9	Merge Sort .....	396
7.91	Analysis .....	400
7.10	Radix Sort .....	400
7.101	Analysis .....	405
7.11	Address Calculation Sort .....	405
7.111	Analysis .....	409
7.12	Quick Sort .....	409
7.121	Analysis .....	413
7.13	Binary Tree Sort .....	413
7.131	Analysis .....	415
7.14	Heap Sort .....	415
7.141	Analysis .....	421
7.15	Comparison .....	421
Exercise	.....	422

## **CHAPTER 8 SEARCHING, HASHING AND STORAGE MANAGEMENT ..... 424**

8.1	Sequential searching .....	424
8.11	Analysis .....	425
8.2	Binary Search .....	426
8.3	Hashing .....	429
8.4	Choosing a hash function .....	431
8.41	Truncation Method .....	431
8.42	Mid square Method .....	431
8.43	Folding Method .....	432
8.44	Modular Method .....	432
8.5	Hash function for floating point numbers .....	433
8.6	Hash function for strings .....	435
8.7	Collision Resolution ( Open Hashing ) .....	436
8.71	Separate chaining .....	436
8.8	Closed Hashing (Open Addressing) .....	437
8.81	Linear Probing .....	437
8.82	Quadratic Probing .....	438
8.83	Double Hashing .....	438
8.9	Rehashing .....	439
8.10	Extendible Hashing .....	441
8.11	Storage Management .....	442
8.12	Garbage collection .....	442
8.13	Dynamic memory management .....	443
8.14	Method to select free block .....	443
8.141	First Fit .....	444
8.142	Best Fit .....	444
8.143	Worst Fit .....	445
8.15	Freeing Memory .....	445



Cont

<u>8.16</u>	<u>Boundary Tag Method</u> .....	4
<u>8.17</u>	<u>Buddy Systems</u> .....	4
<u>8.171</u>	<u>Binary Buddy System</u> .....	4
<u>8.172</u>	<u>Fibonacci Buddy System</u> .....	45
<u>Exercise</u>	.....	45

Index