

EXPERT DATA STRUCTURES With C

SECOND EDITION

```
int temp, current, j;  
for (current = 0; current < size; current++)  
for (j = current + 1; j < size; j++)  
if (array[current] > array[j])  
{  
temp = array[current];  
array[current] = array[j];  
array[j] = temp;  
}
```

R B Patel

CONTENTS

I INTRODUCTION.....

1 - 70

1.1	DEVELOPMENT PHASE OF C	2
1.2	FEATURES OF C	2
1.3	CONSTANTS	3
1.3.1	Escape Sequences	5
1.3.2	Enumeration Constants	7
1.4	TYPE CONVERSION	8
1.5	ARITHMETIC OPERATORS WITH ASSIGNMENTS	10
1.6	PRECEDENCE AND ASSOCIATIVITY	11
1.7	CONDITIONAL EXPRESSIONS	12
1.8	HOW THE OVERFLOW ARISE	13
1.9	STRUCTURES	14
1.10	UNION	18
1.11	POINTERS	20
1.11.1	Arrays Of Pointers	25
1.11.2	Pointer To Structures	29
1.12	DYNAMIC MEMORY ALLOCATION	34
1.13	FUNCTION	38
1.13.1	Function Invocation	39
1.13.2	Function Termination	40
1.13.3	Parameter Passing Techniques	42
1.13.3.1	Pass By Value	42
1.13.3.2	Pass By Reference	43
1.13.4	Prototypes	44
1.14	FLOWCHART	46
1.15	ALGORITHMS	50
1.15.1	Analysis Of Algorithms	51
1.15.1.1	Rate Of Growth	51
1.15.1.2	Time Analysis	52
1.15.1.3	Complexity (Big O Analysis)	54
1.16	DATA STRUCTURE	64
1.16.1	Linear Data Structures	64
1.16.1.1	Arrays	64
1.16.1.2	Linked List	65
1.16.1.3	Stack And Queues	65
1.16.2	Nonlinear Data Structures	66
1.16.2.1	Tree	66
1.16.2.2	Graph	67

1.17 DATA STRUCTURE OPERATIONS	67
EXERCISES	68

4

71-111

2 RECURSION

2.1 USING RECURSION TO SOLVE PROBLEMS	75
2.1.1 An Every Day Examples	75
2.1.2 A Mathematical Examples	75
2.1.3 Data Structures Examples	77
2.1.4 Recursion As A Programming Technique	78
2.2 TYPES OF RECURSION	86
2.3 EXPONENTIAL POWER	86
2.4 FIBONACCI NUMBERS (REVISITED)	91
2.5 GREATEST COMMON DIVISOR	93
2.6 THE TOWERS OF HANOI	94
2.7 DIVIDE AND CONQUER	97
2.8 FUNCTION OVERHEAD	103
2.9 EFFICIENT RECURSION	104
EXERCISES	108

3 STRING MANIPULATION AND PATTERN MATCHING

112 - 156

3.1 STRUCTURE OF STRING STORAGE	114
3.1.1 Fixed Length	114
3.1.2 Variable Length Fixed Maximum	119
3.1.3 Linked List Storage	124
3.2 STRING OPERATIONS	125
3.2.1 Indexing	127
3.2.2 Concatenation	132
3.2.3 Length	133
3.3 APPLICATIONS OF STRING OR TEXT PROCESSING	134
3.3.1 Text Editing	135
3.3.1.1 Insertion	135
3.3.1.2 Deletion	137
3.3.1.3 Pattern Matching	141
3.3.2 Text Formatting	147
3.4 MARKOV ALGORITHM	148
3.4.1 Basic Concept	148
3.4.2 Identity Property	149
3.4.3 Associative Property	149
3.4.4 Production	149
3.4.5 Termination	150
3.4.6 Algorithm	151
EXERCISES	155

5

4 ARRAYS.....

157 - 242

4.1	INTRODUCTION	157
4.2	ONE DIMENSIONAL ARRAYS	158
4.3	ONE DIMENSIONAL ARRAYS IN MEMORY	160
4.4	TRAVERSING ONE DIMENSIONAL ARRAY	161
4.5	INSERTION IN ONE DIMENSIONAL ARRAY	164
4.6	DELETION FROM ONE DIMENSIONAL ARRAY	169
4.7	SEARCHING	177
4.8	MULTIDIMENSIONAL ARRAYS	182
4.8.1	Two Dimensional Arrays	183
4.8.2	Representation Of Two Dimensional Arrays In Memory	186
4.8.3	Traversing Two Dimensional Arrays	187
4.9	ALGEBRA OF MATRICES	191
4.9.1	Addition Of Matrices	191
4.9.2	Subtraction Of Matrices	196
4.9.3	Multiplication Of Two Matrices	200
4.9.4	Transpose Of A Matrix	205
4.9.5	Orthogonal Of A Matrix	209
4.9.6	Singular Matrix	214
4.9.7	Inverse Of A Matrix	217
4.9.8	Symmetric Matrix	223
4.9.9	Rank Of A Matrix	226
4.10	GENERAL MULTIDIMENSIONAL ARRAYS	232
4.11	SPARSE MATRIX	235
	EXERCISES	238

5 LINKED LIST.....

243 - 396

5.1	LINKED LIST	243
5.2	BUILDING A LINKED LIST	245
5.3	TRAVERSING A LINKED LIST	253
5.4	INSERTION IN A LINKED LIST	256
5.4.1	Insertion As A First Node	257
5.4.2	Insertion As A Last Node	262
5.4.3	Insertion Of A Node At Specific Location	267
5.5	DELETION IN A LINKED LIST	278
5.5.1	Deletion Of First Node	280
5.5.2	Deletion Of Last Node	285
5.5.3	Deletion Of A Desired Node	291
5.6	SEARCHING A LINKED LIST	302
5.7	SORTING A LINKED LIST	307
5.8	DOUBLY LINKED LIST	317
5.9	TRAVERSING A DOUBLY LINKED	322
5.10	INSERTION IN A DOUBLY LINKED LIST	326

5.10.1	Inserting First Node	327
5.10.2	Inserting A Node At The End	333
5.10.3	Inserting A Node At Desired Place	340
5.11	DELETION IN A DOUBLY LINKED LIST	342
5.11.1	Deletion Of Left Most Node	343
5.11.2	Deletion Of Right Most Node	348
5.11.3	Deletion Of A Desired Node	355
5.12	MERGING TWO LISTS	362
5.13	HEADER LINKED LIST	369
5.13.1	Grounded Header Linked List	369
5.13.2	Circular Header Linked List	378
5.14	APPLICATION OF HEADER LINKED LIST	385
5.14.1	Polynomials	385
5.14.2	Addition Of Polynomials	388
	EXERCISES	391

6 STACKS AND QUEUES.....

397 - 449

6.1	STACK	397
6.2	REPRESENTATION OF STACK	398
6.3	IMPLEMENTATION OF STACK	399
6.3.1	Array's Implementation	400
6.3.1.1	Push Operation	400
6.3.1.2	Pop Operation	401
6.3.1.3	Peep Operation	408
6.3.1.4	Update Operation	412
6.4	POLISH NOTATION	416
6.4.1	Uses Of Stack	418
6.4.1.1	Evaluation Of The Postfix Expression	424
6.5	QUEUES	424
6.5.1	Implementation Of Queues	425
6.6	CIRCULAR QUEUES	433
6.7	DOUBLE ENDED QUEUE	441
6.8	PRIORITY QUEUES	444
	EXERCISES	446

7 TREES.....

450 - 590

7.1	INTRODUCTION	450
7.2	BINARY TREE	452
7.3	COMPLETE BINARY TREE	462
7.4	EXTENDED BINARY TREE OR 2-TREE	467
7.5	REPRESENTATION OF BINARY TREE	467

7.5.1	Sequential Representation	467
7.5.2	Linked List Representation	470
7.6	TRAVERSING BINARY TREE	471
7.6.1	Preorder	471
7.6.2	Inorder	472
7.6.3	Postorder	474
7.7	MINIMUM WEIGHTED PATH LENGTH ALGORITHM	479
7.8	HUFFMAN ENCODING	481
7.8.1	Decoding Huffman Encoded Data	483
7.8.2	Transmission And Storage Of Huffman Encoded Data	483
7.9	BINARY SEARCH TREE	485
7.10	BALANCED BINARY TREE	499
7.10.1	Height Balanced binary tree (AVL Tree)	502
7.10.2	Tree Rotation: A General Algorithm	507
7.10.3	Insertion In AVL Tree	513
7.10.4	Deletion Of A Node From An AVL Tree	514
7.10.5	The Height Of An AVL Tree	517
7.11	PAGED BINARY TREE	531
7.12	M-WAY SEARCH TREE	532
7.12.1	B-Tree (Perfectly Height Balanced M-Way Search Tree)	534
7.13	RED-BLACK TREE	568
7.14	THREADED BINARY TREES	582
	EXERCISES	583

8 GRAPHS.....

591 - 665

8.1	BASIC CONCEPTS AND DEFINITIONS	591
8.2	PATH	597
8.3	REPRESENTATION OF GRAPH	597
8.3.1	Sequential Representation	598
8.4	SHORTEST PATH ALGORITHMS	604
8.4.1	Warshall's Algorithm	604
8.4.2	Warshall's Modified Algorithm	610
8.4.3	Dijkstra's Technique	619
8.4.4	Floyd's Technique	626
8.5	ADJACENCY LIST	633
8.6	GRAPH TRAVERSAL	634
8.6.1	Depth First Search	635
8.6.2	Breadth First Search	640
8.7	TOPOLOGICAL SORTING	648
8.8	MINIMUM SPANNING TREE	650
8.8.1	Algorithm For Computing A Minimum Spanning Tree	651
8.9	KRUSKAL'S ALGORITHM	653
8.10	DIJKSTRA'S ALGORITHM (REVISITED)	657

9 SORTING AND SEARCHING.....

9.1 LINEAR SEARCH 666
 9.2 BINARY SEARCH 671
 9.3 BUBBLE SORT 676
 9.4 INSERTION SORT 679
 9.5 QUICK SORT 685
 9.6 SELECTION SORT 690
 9.7 SHELL SORT 694
 9.8 MERGE SORT 698
 9.9 HEAP SORT 711
 9.10 RADIX SORT 721
 9.11 RADIX EXCHANGE SORT 734
 9.12 SKIP LISTS 735
 9.13 HASHING 740
 9.13.1 Collisions 747
 9.13.2 Linear Probing 747
 9.13.3 Deletion 749
 9.13.4 Clustering 750
 9.13.5 Rehashing 750
 9.13.6 Bucket And Chaining 752
 9.13.7 Selecting Good Hash Function 753
 9.14 EXTERNAL SORTING 759
 EXERCISES 761

10 FILE STRUCTURES.....

10.1 WHY FILE STRUCTURES? 767
 10.2 GOALS OF FILE STRUCTURES 769
 10.3 FILE SYSTEMS 769
 10.3.1. Basic Concepts Of Files And File Systems 770
 10.3.1.1 File System Services 770
 10.3.1.2 Disk Space Allocation 771
 10.3.2. The MS-DOS Fat File System 772
 10.3.2.1 File Allocation Table 773
 10.3.2.2 The Tree-Structured Directory System 774
 10.3.3. The Unix File System 776
 10.3.3.1 The i-Node File System (Flat File System) 777
 10.3.3.2 The Directory File System 778
 10.4 FILE STRUCTURE ANALYSIS CONCEPTS 779
 10.4.1 User's View Of A File: The Logical File 779

10.4.2 Keys	781
10.4.3 System Architecture	783
10.4.4 Physical Files	785
10.4.5 Primary vs. Secondary Structure	786
10.4.6 File Performance	787
10.4.7 File Structure Implementation	789
10.4.8 File Expansion	789
10.5 SECONDARY STORAGE DEVICES	791
10.5.1 Disk Storage	792
10.5.1.1 Buffer Management	794
10.5.1.2 Hard Disk Drives	795
10.5.1.3 Reducing Seeking	796
10.5.1.4 Disk Capacity	797
10.5.1.5 Blocked Disks	799
10.5.1.6 Disk Access	803
10.5.1.7 Data Transfer Rate	804
10.5.2 Tape Storage	808
10.5.2.1 Tape Utilization	812
10.5.2.2 Effective Data Transfer Rate	813
10.5.2.3 File Structure On Tape	814
10.6 FILE ORGANIZATION	815
10.6.1 Indexes	817
10.6.2 Sequential Files	818
10.6.3 Inverted Files	819
10.6.4 Index-Sequential Files	819
10.6.5 Multi-Lists	820
10.6.6 Cellular Multi-Lists	821
10.6.7 Ring Structures	821
10.6.8 Self-Organizing Files	828
10.6.9 Sorted File And Binary Searching	829
10.6.10 Entry-Sequenced Files And An Index	829
10.7 EXTERNAL SORTING (REVISITED)	830
10.7.1 Basic Underlying Method Of Sorting	831
10.7.2 Creating Large Runs	834
10.7.3 Fast M-Way Merging	837
10.7.4 Pipeline Merging: Minimizing Elapsed Time	839
10.8 HASHING (REVISITED)	852
10.8.1 Linear Hashing	854
10.8.1.1 An Algorithm For Insertion	857
10.8.2 Dynamic Hashing	861
10.8.2.1 Virtual Addressing	861
10.8.2.2 Virtual Address Splitting	862
10.8.2.3 Organizing Virtual Addresses	862
10.8.2.4 Mapping Records To Buckets	863
10.8.2.5 Design Of Nodes In The Address Trie	864
10.8.2.6 Multiple Address Tries	866

10.8.2.7 Linear Splitting	867
10.9 B-TREE (REVISITED)	869
10.9.1 Splitting In A Search Tree	873
10.9.2 Retrieval Performance	874
10.9.3 Insertion	876
10.9.4 Performance Of Insertion	882
10.9.5 Deletion	883
10.10 B+-TREE	888
10.10.1 Insertion Of A Record	890
10.10.2 Performance Of Insertion	891
10.10.3 B+-Tree As A Secondary Index	891
EXERCISES	895
<i>INDEX</i>	903 - 909