



**Unit - I**

---

**Chapter 1 : Introduction to Data Structures**
**1-1 to 1-27**

1.1	Data .....	1-1
1.2	Data Types .....	1-1
1.2.1	Data Types in 'C' .....	1-1
1.2.2	User-Defined Type .....	1-2
1.2.3	Derived Data Types .....	1-2
1.2.3(A)	Structure .....	1-2
1.2.3(B)	Union .....	1-3
1.2.4	Abstract Data Types (ADT) .....	1-3
1.3	Data Object .....	1-4
1.4	Data Structures .....	1-4
1.4.1	Classification / Types of Data Structures .....	1-4
1.4.1(A)	Primitive and Non-Primitive .....	1-4
1.4.1(B)	Linear and Non-Linear .....	1-5
1.4.1(C)	Static and Dynamic .....	1-6
1.4.1(D)	Dynamic Memory Management .....	1-6
1.4.1(E)	Memory Management Function During Runtime .....	1-6
1.4.1(F)	Persistent and Ephemeral Data Structure .....	1-7
1.5	Introduction to Algorithms .....	1-8
1.5.1	Definition and Characteristics .....	1-8
1.5.2	Algorithm Design Tools .....	1-10
1.5.2(A)	Flowcharting .....	1-10
1.5.2(B)	Pseudo-Language .....	1-10
1.5.3	Relation between Data Structure and Algorithm .....	1-12
1.6	Algorithm Analysis .....	1-12
1.6.1	Measuring the Running Time of a Program (Time Complexity) .....	1-14

1.6.2	Measurement of Growth Rate (Asymptotic Growth Rate) .....	1-15
1.6.2(A)	Asymptotic Consideration .....	1-15
1.6.2(B)	Constant Factor in Complexity Measure .....	1-15
1.6.3	Notation O : (Pronounced as Big-oh), (O(n <sup>2</sup> ) is Pronounced as Big-oh of n <sup>2</sup> ) .....	1-15
1.6.4	Best Case, Worst Case and the Average Case Behaviour .....	1-17
1.6.5	Different Algorithm Asymptotic Notation .....	1-23
1.6.5(A)	Notation O .....	1-23
1.6.5(B)	$\Omega$ Notation .....	1-23
1.6.5(C)	Notation $\theta$ .....	1-23
1.7	Program Development .....	1-24
1.8	University Questions and Answers .....	1-26

---

**Chapter 2 : Sequential Organization**
**2-1 to 2-22**

2.1	Sequential Organization .....	2-1
2.2	Introduction to Arrays .....	2-2
2.3	Representation and Analysis .....	2-2
2.4	One-Dimensional Arrays .....	2-2
2.5	Operations with Arrays .....	2-3
2.5.1	Deletion .....	2-3
2.5.2	Insertion .....	2-4
2.5.3	Search .....	2-5
2.5.4	Merging of Sorted Arrays .....	2-6
2.5.5	Reversing an Array .....	2-7
2.6	Two-Dimensional Arrays .....	2-8
2.6.1	Initializing Two-Dimensional Arrays .....	2-9
2.6.2	Address Calculation .....	2-9
2.7	Multi-Dimensional Arrays .....	2-11



2.8	Application of Arrays .....	2-11	3.2.5(A)	Inserting an Item at the End of a Linked List.....	3-7
2.8.1	Addition of Two 2-D Matrices.....	2-11	3.2.5(B)	Inserting a Data 'x' at a given Location 'LOC' in a Linked List, Referenced by 'head' .....	3-8
2.8.2	Transpose of Square Matrix .....	2-13	3.2.5(C)	Inserting an Element in a Priority Linked List.....	3-9
2.8.3	Finding whether a given Square Matrix is Symmetrical .....	2-14	3.2.6	Deleting an Item .....	3-10
2.8.4	Multiplication of Two Matrices $A_m \times n$ and $B_n \times p$ .....	2-14	3.2.6(A)	Deletion of the Last Node of a Linked List .....	3-11
2.8.5	Saddle Point.....	2-16	3.2.6(B)	Deletion of a Node at Location 'LOC' from a Linked List.....	3-11
2.9	Character String in C-Language .....	2-16	3.2.6(C)	Delete a Linked List, Referenced by the Pointer Head .....	3-12
2.9.1	String Comparison.....	2-17	3.2.7	Concatenation of Two Linked Lists.....	3-12
2.9.2	Reversing a String.....	2-18	3.2.8	Inversion of Linked List.....	3-13
2.9.3	String Searching.....	2-19	3.2.9	Searching a Data 'x' in a Linked List, Referenced by the Pointer Head.....	3-14
2.9.4	String Concatenation.....	2-20	3.2.10	Searching an Element x in a Sorted Linked List .....	3-15
2.10	University Questions and Answers .....	2-21	3.2.11	New Linear Linked List by Selecting Alternate Element.....	3-16
<b>Chapter 3 : Linked Organization</b>		<b>3-1 to 3-30</b>	3.2.12	Handling of Records through Linked List .....	3-16
3.1	Representation and Implementation of Singly Linked Lists.....	3-1	3.2.13	Merging of Sorted Linked Lists .....	3-16
3.1.1	Comparison between Array (Sequential) and Linked Lists .....	3-1	3.2.14	Splitting a Linked List at the Middle and Merge with Second Half as First Half.....	3-17
3.1.2	Representation .....	3-1	3.2.15	Removing Duplicate Elements from a Linked List.....	3-18
3.1.3	Implementation.....	3-2	3.3	Circular Linked List.....	3-19
3.1.4	Types of Linked List .....	3-3	3.3.1	Applications of Circular Linked List.....	3-23
3.1.4(A)	Singly Linked List .....	3-3	3.4	Doubly Linked List .....	3-23
3.1.4(B)	Doubly Linked List.....	3-3	3.4.1	Creation of a Doubly Linked List.....	3-23
3.1.4(C)	A Circular Linked List.....	3-3	3.4.2	Deletion of a Node.....	3-25
3.2	Basic Linked List Operations .....	3-3	3.5	Doubly Linked Circular List.....	3-27
3.2.1	Creating a Linked List.....	3-4	3.6	University Questions and Answers .....	3-29
3.2.2	Traversing a Linked List .....	3-5			
3.2.3	Counting Number of Nodes in a Linked List through Count Function .....	3-5			
3.2.4	Printing a List through Print Function.....	3-6			
3.2.5	Inserting an Item .....	3-6			



## Unit - II

### **Chapter 4 : Searching and Sorting** **4-1 to 4-57**

4.1	Searching .....	4-1
4.2	Sequential / Linear Search .....	4-1
4.2.1	Sequential Search on a Sorted Array .....	4-2
4.3	Binary Search.....	4-2
4.3.1	'C' Function for Binary Search (Non-Recursive) .....	4-4
4.3.2	Recursive Function for Binary Search .....	4-5
4.4	Fibonacci Search.....	4-8
4.4.1	Binary Search Vs Fibonacci search.....	4-9
4.4.2	Selection of Searching Algorithm.....	4-9
4.4.3	Timing Complexity of Fibonacci Search.....	4-10
4.5	Sorting.....	4-11
4.5.1	Sort Stability .....	4-11
4.5.2	Sort Efficiency .....	4-12
4.5.3	Passes .....	4-12
4.5.4	Importance of Sorting and Searching .....	4-12
4.6	Insertion Sort.....	4-13
4.6.1	Sorting an Array of Strings using Insertion Sort .....	4-17
4.6.2	Sorting an Array of Records on the given Key using Insertion Sort .....	4-17
4.7	Bubble Sort .....	4-20
4.8	Selection Sort.....	4-23
4.9	Quick Sort .....	4-26
4.9.1	Picking a Pivot.....	4-26
4.9.2	Partitioning .....	4-26
4.9.3	Running Time of Quick Sort .....	4-40
4.9.3(A)	Worst-Case Analysis for Quick Sort to Sort List of Numbers in Ascending Order .....	4-40
4.9.3(B)	Best-Case Analysis .....	4-40
4.9.3(C)	Average-Case Analysis .....	4-40
4.9.4	Role of Pivot in Efficiency of Quick Sort .....	4-43

4.10	Two-Way Merge Sort.....	4-43
4.10.1	Merging .....	4-44
4.10.2	Analysis of Merge Sort .....	4-49
4.10.3	Non-Recursive Merge Sort .....	4-49
4.11	Shell Sort .....	4-50
4.11.1	C Function for Shell Sort .....	4-50
4.12	Comparison of Sorting Algorithms .....	4-51
4.13	Best-case, Worst-case and Average-case Analysis of Sorting Algorithm .....	4-52
4.14	External Vs Internal Sorting.....	4-54
4.15	University Questions and Answers .....	4-55

## Unit III

### **Chapter 5 : Stack** **5-1 to 5-35**

5.1	Introduction .....	5-1
5.2	Operations on Stacks .....	5-1
5.3	Array Representation.....	5-2
5.3.1	'C' Functions for Primitive Operations on a Stack .....	5-2
5.3.2	Program Showing Stack Operations.....	5-3
5.4	Linked Representation of a Stack .....	5-5
5.4.1	Functions for Stack Operations .....	5-6
5.4.2	Class Declaration for Stack using a Linked List.....	5-6
5.5	Implicit and Explicit Stack.....	5-8
5.6	Removal of Recursion using Stacks .....	5-8
5.7	Application of Stack.....	5-9
5.7.1	Expression Representation.....	5-9
5.7.2	Evaluation of a Postfix Expression using a Stack .....	5-10
5.7.3	Evaluation of a Prefix Expression .....	5-14
5.7.4	Conversion of an Expression from Infix to Postfix.....	5-15
5.7.5	Conversion of an Expression from Infix to Prefix .....	5-19
5.7.6	Conversion of Expression from Postfix into Infix.....	5-24
5.7.7	Conversion of Expression from Postfix into Prefix .....	5-25



5.7.8	Conversion of Expression from Prefix to Infix .....	5-29
5.7.9	Conversion of Expression from Prefix into Postfix .....	5-30
5.7.10	Conversion of a Fully Parenthesised Infix Expression into Prefix Form .....	5-31
5.8	University Questions and Answers .....	5-35

---

**Chapter 6 : Queue** **6-1 to 6-35**


---

6.1	Array Implementation of Queues .....	6-1
6.1.1	Definition .....	6-1
6.1.2	Applications of Queues.....	6-1
6.1.3	Array Representation and Implementation of Queues .....	6-1
6.2	Operations on Queue .....	6-2
6.2.1	Operations on Queue Implemented using Array .....	6-2
6.2.2	Queue as an ADT.....	6-5
6.2.3	Operations on Queue Implemented using Linked Structure.....	6-8
6.3	Circular Queues .....	6-12
6.3.1	Queue using a Circular Array .....	6-12
6.3.1(A)	Implementation of a Circular Movement inside a Linear Array .....	16-3
6.3.2	Queue using a Circular Linked List.....	6-18
6.4	Dequeues.....	6-22
6.4.1	Implementation of Dequeue using a Circular Array....	6-23
6.4.2	Dequeue using Linked Structure.....	6-28
6.5	Priority Queue .....	6-29
6.5.1	Implementation of Priority Queues .....	6-29
6.5.2	Priority Queue using a Linked List .....	6-33
6.6	University Questions and Answers .....	6-34

**Unit - IV**

---

**Chapter 7 : Trees** **7-1 to 7-47**


---

7.1	Basic Terminology.....	7-1
7.1.1	Linear Data Structures.....	7-1
7.1.2	Non-linear Data Structures .....	7-1
7.1.3	Introduction to Tree Terminology.....	7-1
7.1.4	Basic Terms .....	7-2
7.2	Binary Tree.....	7-2
7.3	Representation of a Binary Tree using an Array .....	7-2
7.4	Linked Representation of a Binary Tree.....	7-4
7.4.1	Program for Creation of a Sample Binary Tree.....	7-5
7.4.2	C++ Function for Creation of a Binary Tree .....	7-5
7.5	A General Tree.....	7-6
7.5.1	Node Declaration for a Tree .....	7-6
7.6	Types of Binary Tree .....	7-8
7.6.1	Full Binary Tree .....	7-8
7.6.2	Complete Binary Tree.....	7-8
7.6.3	Skewed Binary Tree .....	7-8
7.6.4	Strictly Binary Tree .....	7-9
7.6.5	Extended Binary Tree (2-Tree) .....	7-9
7.7	Binary Tree Traversal .....	7-9
7.7.1	Preorder Traversal (Recursive).....	7-9
7.7.2	Inorder Traversal (Recursive) .....	7-10
7.7.3	Postorder Traversal (Recursive).....	7-11
7.7.4	Non-Recursive Preorder Traversal .....	7-12
7.7.4(A)	'C++' Function for Non-Recursive Preorder of Tree along with the ADT Stack .....	7-12
7.7.5	Non-Recursive Inorder Traversal.....	7-12
7.7.5(A)	'C++' Function for Non-Recursive Inorder Traversal of a Binary Tree .....	7-13
7.7.6	Non-Recursive Postorder Traversal.....	7-13
7.7.6(A)	'C++' Function for Non-Recursive Postorder Traversal .....	7-15
7.7.7	Tree Traversal Examples .....	7-16
7.8	Basic Tree Operations.....	7-18
7.8.1	'C++' Function for Counting of Nodes in a Tree.....	7-18
7.8.2	'C++' Function for Counting of Leaf Nodes in a Tree (Recursive).....	7-19
7.8.3	'C++' Function for Counting of Nodes of Degree 1 (Recursive) .....	7-19



<p>7.8.4 'C++' Function for Counting of ..... Nodes of Degree 2 (Recursive) .....7-20</p> <p>7.8.5 'C++' Function to Create an Exact ..... Copy of a Tree (Recursive).....7-20</p> <p>7.8.6 'C++' Function for Checking Equivalence of Two Binary Trees .....7-20</p> <p>7.8.7 'C++' Function for Finding Height of a Tree (Recursive).....7-20</p> <p>7.8.8 'C++' Function for Swapping of Left and Right Children of Every Node (Mirror) .....7-21</p> <p>7.8.9 Function to List the DATA Fields of the Node of a Binary Tree T by Level, within Levels Nodes are Listed Left to Right.....7-21</p> <p>7.8.10 Non-Recursive Algorithm for Height of a Binary Tree.....7-22</p> <p>7.9 Creation of a Binary Tree from Traversal Sequence .....7-23</p> <p>7.9.1 Creation of Binary Tree from Preorder and Inorder Traversals .....7-23</p> <p>7.9.2 Creation of Tree from Postorder and Inorder Traversal.....7-23</p> <p>7.9.3 Examples on Tree Creation from Traversal Sequence .....7-24</p> <p>7.10 Binary Tree as an ADT .....7-27</p> <p>7.11 Binary Search Tree (BST) .....7-29</p> <p>7.11.1 Definition .....7-29</p> <p>7.11.2 Operations on a Binary Search Tree .....7-30</p> <p>7.11.2(A) Initialize Operation.....7-30</p> <p>7.11.2(B) Find Operation.....7-30</p> <p>7.11.2(C) Makeempty Operation .....7-30</p> <p>7.11.2(D) Insert Operation.....7-31</p> <p>7.11.2(E) Example on Creation of a BST .....7-31</p> <p>7.11.2(F) Delete Operation .....7-33</p> <p>7.11.2(G) Create .....7-35</p> <p>7.11.2(H) FindMin .....7-35</p>	<p>7.11.3 Program for Various Operations on BST (BST as an ADT) .....7-35</p> <p>7.12 Recursion .....7-40</p> <p>7.13 Application of Trees.....7-40</p> <p>7.13.1 Expression Trees .....7-40</p> <p>7.13.2 Program on Expression Tree from Postfix Expression.....7-41</p> <p>7.13.3 Conversion of an Expression into Binary Tree .....7-43</p> <p>7.13.4 Construction of an Expression Tree from Infix Expression.....7-44</p> <p>7.14 University Questions and Answers .....7-46</p>
--	--

**Unit - V**

---

**Chapter 8 : Graphs**
**8-1 to 8-57**

<p>8.1 Terminology and Representation.....8-1</p> <p>8.1.1 Definition .....8-1</p> <p>8.1.2 Undirected Graph .....8-1</p> <p>8.1.3 Directed Graph .....8-1</p> <p>8.1.4 A Complete Graph.....8-2</p> <p>8.1.5 Weighted Graph .....8-2</p> <p>8.1.6 Adjacent Nodes .....8-2</p> <p>8.1.7 Path.....8-2</p> <p>8.1.8 Cycle .....8-2</p> <p>8.1.9 Connected Graph .....8-2</p> <p>8.1.10 Subgraph.....8-3</p> <p>8.1.11 Component.....8-3</p> <p>8.1.12 Degree of a Vertex .....8-3</p> <p>8.1.13 Self Edges or Self Loops .....8-3</p> <p>8.1.14 Multigraph .....8-3</p> <p>8.1.15 Tree.....8-3</p> <p>8.1.16 Spanning Trees .....8-4</p> <p>8.1.17 Minimal Spanning Tree.....8-4</p>
--



8.1.18	Applications of Spanning Tree.....	8-4	8.4.4(A)	'C++' Function for Computing Transitive Closure of a Graph using Warshall's Algorithm .....	8-27
8.2	Representation of Graphs.....	8-4	8.5	Minimum Cost Spanning Tree .....	8-27
8.2.1	Adjacency Matrix .....	8-5	8.5.1	Prim's Algorithm .....	8-28
8.2.2	Adjacency List.....	8-6	8.5.1(A)	'C++' Function for Finding the Minimum Cost Spanning Tree .....	8-29
8.2.3	Examples on Graph Representation.....	8-7	8.5.1(B)	Program for Prim's Algorithm .....	8-30
8.2.4	Programs on Graph Representation .....	8-9	8.5.1(C)	Timing Complexity of Prim's Algorithm .....	8-31
8.3	Traversal of Graphs.....	8-12	8.5.1(D)	Examples on Prim's Algorithm.....	8-32
8.3.1	Depth First Search (DFS) .....	8-12	8.5.2	Kruskal's Algorithm .....	8-35
8.3.1(A)	Algorithm for DFS.....	8-12	8.5.2(A)	'C++' Function for find().....	8-37
8.3.1(B)	Program for DFS using Adjacency Matrix.....	8-13	8.5.2(B)	'C++' Function for Finding Minimum Cost Spanning Tree of a Graph using Kruskal's Algorithm .....	8-37
8.3.1(C)	Program for DFS using Adjacency List.....	8-14	8.5.2(C)	Program for Kruskal's Algorithm.....	8-37
8.3.1(D)	Non-recursive DFS Traversal .....	8-15	8.5.2(D)	Timing Complexity of Kruskal's Algorithm.....	8-39
8.3.1(E)	Examples on DFS .....	8-16	8.5.2(E)	Comparison of Time Complexity of Prim's and Kruskal's Algorithm.....	8-39
8.3.1(F)	DFS Spanning Tree.....	8-17	8.5.2(F)	Examples on Kruskal's Algorithm .....	8-39
8.3.1(G)	'C++' Function for Checking whether the given Graph is Connected.....	8-18	8.6	Shortest Path Algorithm .....	8-45
8.3.1(H)	'C++' Function for Finding Components of a Graph .....	8-18	8.6.1	Dijkstra Algorithm .....	8-45
8.3.2	Breadth First Search (BFS) .....	8-19	8.6.1(A)	Timing Complexity.....	8-45
8.3.2(A)	Algorithm for BFS.....	8-19	8.6.2	Program on Dijkstra's Algorithm .....	8-45
8.3.2(B)	Examples on BFS .....	8-20	8.6.3	Examples on Dijkstra's Algorithm .....	8-47
8.3.2(C)	Program for BFS using Adjacency Matrix .....	8-21	8.7	Topological Sorting.....	8-51
8.3.2(D)	Program for BFS using Adjacency List .....	8-22	8.7.1	Program for Topological Sorting .....	8-52
8.3.2(E)	BFS Spanning Tree.....	8-23	8.8	University Questions and Answers .....	8-55
8.4	Connected Components.....	8-26	<b>Chapter 9 : Symbol Table</b>		<b>9-1 to 9-68</b>
8.4.1	'C++' Function for Printing Connected Components of a Graph .....	8-26	9.1	Notion of Symbol Tables .....	9-1
8.4.2	Transitive Closure and Connectedness .....	8-26	9.2	Representation of Symbol Table .....	9-1
8.4.3	Examples on Transitive Closure .....	8-27	9.2.1	Static Tree Tables .....	9-1
8.4.4	Warshall's Algorithm for Computing Transitive Closure of a Graph G .....	8-27	9.2.2	Dynamic Tree Tables .....	9-2



9.3	Optimal Binary Search Tree (OBST) .....	9-2	9.5.3(A) Basic Operations for Max Heap : .....	9-38
9.3.1	OBST with Failure Nodes .....	9-3	9.5.3(B) Heap Creation – A Better Approach .....	9-41
9.3.2	Equation of the Root for OBST .....	9-6	9.5.3(C) Heap as an ADT .....	9-42
9.3.3	C-Algorithm for Finding Minimum Cost Binary Search Tree .....	9-8	9.5.4 Application of Heaps .....	9-45
9.3.4	Program for Constructing an OBST .....	9-8	9.5.4(A) Sorting in Ascending Order (Algorithm) .....	9-45
9.4	AVL Trees .....	9-11	9.5.4(B) Sorting in Ascending Order (Program) .....	9-46
9.4.1	Height Balanced Tree .....	9-11	9.5.4(C) Timing Complexity of Heap Sort .....	9-46
9.4.2	Balance Factor .....	9-11	9.5.4(D) Priority Queues using a Heap Data Structure .....	9-68
9.4.3	Structure of a Node in AVL Tree .....	9-12	9.6 University Questions and Answers .....	9-68
9.4.4	'C++' Function for Finding the Balance Factor of a Node .....	9-12		
9.4.5	Insertion of a Node into an AVL Tree .....	9-13		
9.4.5(A)	Rotate Left .....	9-13		
9.4.5(B)	Rotate Right .....	9-14		
9.4.5(C)	Single Rotation and Double Rotation .....	9-15		
9.4.5(D)	'C++' Function for Insertion of an Element into an AVL Tree .....	9-26		
9.4.5(E)	'C++' Function to Find Height of AVL Tree .....	9-26		
9.4.5(F)	'C++' Function to Rotate Right .....	9-26		
9.4.5(G)	'C++' Function to Rotate Left .....	9-27		
9.4.5(H)	'C++' Function for RR .....	9-27		
9.4.5(I)	'C++' Function for LL .....	9-27		
9.4.5(J)	'C++' Function for LR .....	9-27		
9.4.5(K)	'C++' Function for RL .....	9-27		
9.4.5(L)	C Function for Deletion .....	9-27		
9.4.6	Program for AVL Tree .....	9-28		
9.5	Heap Sort .....	9-37		
9.5.1	Heaps .....	9-37		
9.5.2	Types of Heaps .....	9-38		
9.5.3	Basic Heap Operations .....	9-38		

**Unit - VI**

**Chapter 10 : Hashing****10-1 to 10-24**

10.1	Hash Tables .....	10-1
10.1.1	What is Hashing ? .....	10-1
10.1.2	Hash Table Data Structure .....	10-2
10.1.2(A)	Open Hashing Data Structure .....	10-2
10.1.2(B)	Closed Hashing Data Structure .....	10-2
10.2	Hashing Functions .....	10-2
10.2.1	Characteristics of a Good Hash Function .....	10-3
10.2.2	Division-Method .....	10-3
10.2.3	Midsquare Methods .....	10-3
10.2.4	Folding Method .....	10-3
10.2.5	Digit Analysis .....	10-3
10.2.6	Length Dependent Method .....	10-4
10.2.7	Algebraic Coding .....	10-4
10.2.8	Multiplicative Hashing .....	10-4
10.3	Collision Resolution Strategies (Synonym Resolution) .....	10-4
10.3.1	Separate Chaining .....	10-4
10.3.2	Open Addressing .....	10-5



10.3.2(A) Linear Probing.....	10-5	11.7 File Organization.....	11-16
10.3.2(A)(1) Linear Probing with Chaining (Without Replacement).....	10-8	11.7.1 Primitive Operations on a File.....	11-16
10.3.2(A)(2) Linear Probing with Chaining (with Replacement).....	10-12	11.7.1(A) Creation.....	11-16
10.3.2(B) Quadratic Probing.....	10-21	11.7.1(B) Reading.....	11-17
10.3.3 Double Hashing.....	10-21	11.7.1(C) Insertion.....	11-17
10.3.4 Rehashing.....	10-21	11.7.1(D) Deletion.....	11-17
10.4 Extendible Hashing.....	10-22	11.7.1(E) Updation.....	11-17
10.5 University Questions and Answers.....	10-23	11.7.2 Sequential Files.....	11-17
<hr/>		11.7.2(A) Operations on a Sequential File.....	11-18
<b>Chapter 11 : File Organization</b>		11.7.2(A)(1) Creation.....	11-18
<b>11-1 to 11-39</b>		11.7.2(A)(2) Reading (Printing).....	11-19
<hr/>		11.7.2(A)(3) Insertion.....	11-19
11.1 Introduction to Files ( in 'C' ).....	11-1	11.7.2(A)(4) Deletion.....	11-20
11.1.1 Files I/O in 'C'.....	11-1	11.7.2(A)(5) Searching.....	11-20
11.1.2 Compare Text File and Binary File.....	11-3	11.7.2(A)(6) Updation (Modifications).....	11-21
11.2 Basic File Operations (in 'C').....	11-3	11.7.2(A)(7) Packing.....	11-21
11.3 Sequential and Random Access Files (in 'C').....	11-5	11.8 Indexing and Hashing.....	11-24
11.3.1 Binary Files.....	11-8	11.8.1 Indexing.....	11-24
11.4 Introduction to Files (in C++).....	11-10	11.8.1(A) Advantages of Indexing over Sequential File.....	11-25
11.4.1 Creating a File.....	11-10	11.8.2 Hashing (Direct File Organization).....	11-25
11.4.2 Selecting File Open Mode.....	11-10	11.8.3 Operations on a Direct File.....	11-26
11.4.3 Reading from a File.....	11-10	11.9 Types of Indexes.....	11-29
11.4.4 Writing to a File.....	11-10	11.9.1 Primary Indexes (Indexed Sequential File).....	11-29
11.4.5 Reading and Writing of Objects.....	11-11	11.9.1(A) Operations on an Indexed Sequential File.....	11-30
11.4.6 Random Access Files.....	11-11	11.9.2 Clustering Indexes.....	11-33
11.5 Physical Storage Media File Organization.....	11-12	11.9.3 Secondary Indexes (Simple Index File).....	11-33
11.5.1 Magnetic Disk.....	11-12	11.9.3(A) Operations on a Simple Index File.....	11-34
11.5.1(A) Disk Access.....	11-13	11.10 Indexing (or Sequential) and Hashing Comparison.....	11-37
11.5.1(B) Data Organisation and Formatting.....	11-13	11.11 Difference between Sequential, Index Sequential and Direct Access File.....	11-38
11.5.2 CD-ROM.....	11-14	11.12 University Questions and Answers.....	11-38
11.6 Organization of Records into Blocks.....	11-15		