



Unit - I

Chapter 1 : Introduction to Algorithms
1-1 to 1-14

1.1	Introduction to Algorithms.....	1-1
1.1.1	Definition and Characteristics	1-1
1.1.2	Algorithm Design Tools	1-3
1.1.2(A)	Flowcharting.....	1-3
1.1.2(B)	Pseudo-Language	1-3
1.1.3	Relation between Data Structure and Algorithm	1-5
1.2	Program Development.....	1-5
1.3	Algorithm Analysis (Using Step Count).....	1-6
1.3.1	Measuring the Running Time of a Program (Time Complexity)	1-8
1.3.2	Measurement of Growth Rate (Asymptotic Growth Rate)	1-8
1.3.2(A)	Asymptotic Consideration.....	1-8
1.3.2(B)	Constant Factor in Complexity Measure	1-9
1.3.3	Notation O : (Pronounced as Big-Oh), ($O(n^2)$ is Pronounced as Big-Oh of n^2)	1-9
1.3.4	Best Case, Worst Case and the Average Case Behaviour.....	1-11
1.3.5	Different Algorithm Asymptotic Notation	1-12
1.3.5(A)	The Notation O (Big-Oh).....	1-12
1.3.5(B)	The Notation Ω (Omega).....	1-12
1.3.5(C)	The Notation θ (Theta).....	1-13
1.3.6	Graphical Interpretation of Different Notations	1-13
1.4	Programming Constructs-linear, Quadratic, Cubic and Logarithmic.....	1-13
1.4.1	Constant-Time Algorithms [$O(1)$].....	1-13
1.4.2	Linear-Time Algorithm [$O(n)$]	1-14

1.4.3	Quadratic-Time Algorithm [$O(n^2)$].....	1-14
1.4.4	Cubic-Time Algorithm [$O(n^3)$].....	1-14
1.4.5	Logarithmic-Time Algorithm [$\log(n)$].....	1-14

Chapter 2 : Data Structures and Algorithmic Strategies
2-1 to 2-16

2.1	Data	2-1
2.1.1	Data Types.....	2-1
2.1.2	Abstract Data Types (ADT)	2-1
2.1.3	Data Object	2-3
2.2	Data Structures	2-3
2.2.1	Types/Classification of Data Structures	2-4
2.2.1(A)	Primitive and Non-Primitive	2-4
2.2.1(B)	Linear and Non-Linear	2-4
2.2.1(C)	Static and Dynamic	2-5
2.2.1(D)	Persistent and Ephemeral	2-6
2.2.1(E)	Relationship among Data Object, Data Type, Data Structure and Data Representation	2-6
2.3	Algorithmic Strategy	2-7
2.4	Divide and Conquer.....	2-7
2.4.1	Binary Search.....	2-7
2.4.2	Examples of Divide and Conquer Strategy	2-8
2.4.2(A)	'Divide and Conquer' Strategy for Tower of Hanoi	2-8
2.4.2(B)	Merge Sort	2-8
2.4.2(C)	Quick Sort	2-9
2.4.2(D)	Finding of Maxima	2-9
2.5	The Greedy Method.....	2-9
2.5.1	Job Sequencing with Deadlines.....	2-11
2.5.2	Graph Colouring Problem	2-11
2.6	Generating Functions	2-12
2.7	Multiplication Technique	2-16



Unit - II

Chapter 3 : Linear Data Structures**using Sequential Organization 3-1 to 3-54**

<p>3.1 Sequential Organization3-1</p> <p>3.2 Introduction to Arrays3-2</p> <p>3.3 Representation and Analysis3-2</p> <p>3.4 One-Dimensional Arrays3-3</p> <p>3.5 Operations with Arrays3-4</p> <p>3.5.1 Deletion3-4</p> <p>3.5.2 Insertion3-5</p> <p>3.5.3 Search3-6</p> <p>3.5.4 Merging of Sorted Arrays3-7</p> <p>3.5.5 Reversing an Array3-9</p> <p>3.6 Two-Dimensional Arrays3-9</p> <p>3.6.1 Initializing Two-Dimensional Arrays3-10</p> <p>3.6.2 Address Calculation3-10</p> <p>3.7 Multi-Dimensional Arrays3-12</p> <p>3.8 Application of Arrays3-12</p> <p>3.8.1 Addition of Two 2-D Matrices3-12</p> <p>3.8.2 Transpose of Square Matrix3-14</p> <p>3.8.3 Finding whether a given Square Matrix is Symmetrical3-15</p> <p>3.8.4 Multiplication of Two Matrices $A_{m \times n}$ and $B_{n \times p}$3-16</p> <p>3.8.5 Saddle Point3-17</p> <p>3.9 Character String in C-Language3-18</p> <p>3.9.1 String Comparison3-20</p> <p>3.9.2 Reversing a String3-20</p> <p>3.9.3 String Searching3-21</p> <p>3.9.4 String Concatenation3-22</p> <p>3.10 Ordered List3-24</p> <p>3.11 Representation of a Polynomial of Degree n in an Array3-25</p>	<p>3.11.1 Pseudo C - Algorithm for Addition of Two Polynomials3-25</p> <p>3.11.2 C-Function for Addition of Two Polynomials Represented in an Array3-26</p> <p>3.11.3 Representing a Polynomial $A(x, y)$3-26</p> <p>3.11.4 Representing a Polynomial $A(x, y, z)$3-28</p> <p>3.12 Polynomial as an Ordered List (using Structure)3-29</p> <p>3.12.1 Function for Initialisation of a Polynomial3-30</p> <p>3.12.2 Inserting a Term3-30</p> <p>3.12.3 Reading a Polynomial3-31</p> <p>3.12.4 Printing a Polynomial3-31</p> <p>3.12.5 Addition of Two Polynomials3-31</p> <p>3.12.6 Multiplication of Two Polynomials3-32</p> <p>3.12.7 Evaluation of a Polynomials3-33</p> <p>3.13 Sparse Matrix as an Ordered List3-36</p> <p>3.14 Operations on a Sparse Matrix3-37</p> <p>3.14.1 From 2D-Representation to Sparse Representation3-37</p> <p>3.14.2 From Sparse Representation (List of Triplets) to 2D-representation (Conventional)3-37</p> <p>3.14.3 Transpose of a Sparse Matrix (Simple/slow)3-40</p> <p>3.14.4 Pseudo-C Algorithm for Simple Transpose3-41</p> <p>3.14.5 Timing Complexity of the Function Transpose()3-43</p> <p>3.15 Fast Transpose3-43</p> <p>3.15.1 Pseudo-C Algorithm for Fast Transpose3-45</p> <p>3.15.2 Calculation of Timing Complexity of "Fast-Transpose()"3-45</p> <p>3.15.3 Addition of Two Sparse Matrices3-46</p> <p>3.15.4 Multiplication of Two Sparse Matrices3-48</p> <p>3.15.5 Applications of Sparse Matrices3-51</p> <p>3.16 Representing a Polynomial of Two Variables using Sparse Matrix3-52</p> <p>3.17 Case Study - Use of Sparse Matrix in Social Networks and Maps3-53</p> <p>3.17.1 Use of Polynomials to Model Economic Growth Pattern3-53</p> <p>3.17.2 Polynomial to Describe the Behaviour of Covid-19 Virus3-53</p> <p>● Model Question Paper - I (In Sem.) Q-1 to Q-2</p>
---	--



Unit - III

Chapter 4 : Searching

4-1 to 4-13

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.....	4-3
4.3.1	Comparison between Linear and Binary Search	4-8
4.4	Fibonacci Search.....	4-8
4.4.1	Binary Search versus 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	Index Sequential Search.....	4-11
4.5.1	Indexing.....	4-11
4.5.2	Sequential File.....	4-12
4.5.3	Indexed Sequential File	4-12
4.6	Sentinel Search.....	4-13
4.7	Case Study - Use of Fibonacci Search	4-13

Chapter 5 : Sorting

5-1 to 5-58

5.1	Sorting.....	5-1
5.1.1	Sort Stability	5-1
5.1.2	Sort Efficiency	5-2
5.1.3	Passes	5-3
5.1.4	Importance of Sorting and Searching	5-3
5.2	Insertion Sort.....	5-3
5.2.1	Sorting an Array of Strings using Insertion Sort.....	5-6
5.2.2	Sorting an Array of Records on the given Key using Insertion Sort.....	5-7
5.3	Bubble Sort	5-10
5.4	Selection Sort.....	5-15
5.5	Quick Sort	5-17
5.5.1	Picking a Pivot.....	5-18

5.5.2	Partitioning	5-18
5.5.3	Running Time of Quick Sort	5-31
5.5.3(A)	Worst-Case Analysis for Quick Sort to Sort List of Numbers in Ascending Order	5-32
5.5.3(B)	Best-Case Analysis	5-32
5.5.3(C)	Average-Case Analysis	5-32
5.5.4	Role of Pivot in Efficiency of Quick Sort	5-35
5.6	Two-Way Merge Sort.....	5-35
5.6.1	Merging	5-37
5.6.2	Analysis of Merge Sort	5-44
5.6.3	Non-Recursive Merge Sort	5-45
5.7	Counting Sort	5-45
5.8	Radix Sort	5-47
5.8.1	Algorithm for Radix Sort	5-48
5.8.2	C-Function for Radix Sort.....	5-48
5.8.3	Analysis of Radix Sort	5-50
5.9	Bucket Sort	5-51
5.10	Shell Sort	5-54
5.10.1	C-Function for Shell Sort	5-54
5.11	External versus Internal Sorting.....	5-56
5.12	Comparison of Sorting Algorithms	5-57
5.13	Best-case, Worst-case and Average-case Analysis of Sorting Algorithm.....	5-58
5.14	Case Study - Timesort is a Hybrid Stable	5-58

Unit - IV

Chapter 6 : Linked List

6-1 to 6-58

6.1	Representation and Implementation of Singly Linked Lists	6-1
6.1.1	Comparison between Array and Linked Lists.....	6-1
6.1.2	Dynamic Memory Management.....	6-2
6.1.3	Memory Management Function during Runtime	6-2
6.1.4	Representation	6-3



6.1.5	Implementation.....	6-3	6.3.1	Applications of Circular Linked List.....	6-27
6.1.6	Types of Linked List	6-4	6.4	Doubly Linked List.....	6-27
6.1.6(A)	Singly Linked List	6-4	6.4.1	Creation of a Doubly Linked List.....	6-27
6.1.6(B)	Doubly Linked List	6-5	6.4.2	Deletion of a Node.....	6-31
6.1.6(C)	A Circular Linked List	6-5	6.5	Doubly Linked Circular List.....	6-32
6.2	Basic Linked List Operations	6-5	6.6	Applications of Linked Lists	6-34
6.2.1	Creating a Linked List.....	6-6	6.6.1	Polynomials as Linked Lists	6-34
6.2.2	Traversing a Linked List	6-7	6.6.2	Addition of Two Polynomials	6-35
6.2.3	Counting Number of Nodes in a Linked List through Count Function.....	6-7	6.7	Concept of Skip List	6-41
6.2.4	Printing a List through Print Function.....	6-8	6.8	Generalized Linked List	6-41
6.2.5	Inserting an Item	6-8	6.8.1	Introduction	6-41
6.2.5(A)	Inserting an Item at the End of a Linked List.....	6-9	6.8.2	Representation of Generalized Lists.....	6-42
6.2.5(B)	Inserting a Data 'x' at a given Location 'LOC' in a Linked List, Referenced by 'head'	6-10	6.9	Representation of Polynomial using Generalized List	6-47
6.2.5(C)	Inserting an Element in a Priority Linked List.....	6-12	6.10	Linked List in Array.....	6-54
6.2.6	Deleting an Item	6-13	6.11	Memory Allocation.....	6-54
6.2.6(A)	Deletion of the Last Node of a Linked List	6-14	6.11.1	First-Fit Allocation	6-54
6.2.6(B)	Deletion of a Node at Location 'LOC' from a Linked List.....	6-14	6.11.1(A)	Algorithm for First-Fit.....	6-55
6.2.6(C)	Delete a Linked List, Referenced by the Pointer Head	6-15	6.11.2	Best-Fit Allocation	6-56
6.2.7	Concatenation of Two Linked Lists	6-16	6.11.2(A)	Algorithm for Best-Fit.....	6-57
6.2.8	Inversion of Linked List.....	6-16	6.11.3	Worst Fit	6-57
6.2.9	Searching a Data 'x' in a Linked List, Referenced by the Pointer Head	6-18	6.11.4	Next Fit.....	6-57
6.2.10	Searching an Element x in a Sorted Linked List	6-19	6.12	Garbage Collection.....	6-57
6.2.11	New Linear Linked List by Selecting Alternate Element.....	6-19	Unit - V		
6.2.12	Handling of Records through Linked List	6-20	<hr/>		
6.2.13	Merging of Sorted Linked Lists	6-20	Chapter 7 : Stacks 7-1 to 7-60		
6.2.14	Splitting a Linked List at the Middle and Merge with Second Half as First Half.....	6-21	7.1	Introduction	7-1
6.2.15	Removing Duplicate Elements from a Linked List.....	6-22	7.2	Operations on Stacks	7-2
6.3	Circular Linked List.....	6-23	7.3	Array Representation.....	7-2
			7.3.1	'C' Functions for Primitive Operations on a Stack.....	7-2
			7.3.2	Program Showing Stack Operations.....	7-3
			7.3.3	Operations on Stack Considering Overflow and Underflow : [Array Implementation]	7-5



<p>7.4 Linked Representation of a Stack.....7-6</p> <p>7.4.1 Functions for Stack Operations7-7</p> <p>7.5 Application of Stack.....7-8</p> <p>7.5.1 Expression Representation.....7-9</p> <p>7.5.2 Evaluation of a Postfix Expression using a Stack7-9</p> <p>7.5.3 Evaluation of a Prefix (Polish) Expression7-12</p> <p>7.5.3(A) Conversion of an Expression from Infix to Postfix.....7-15</p> <p>7.5.4 Conversion of an Expression from Infix to Prefix7-26</p> <p>7.5.5 Conversion of Expression from Postfix into Infix7-29</p> <p>7.5.6 Conversion of Expression from Postfix into Prefix7-31</p> <p>7.5.7 Conversion of Expression from Prefix to Infix7-33</p> <p>7.5.8 Conversion of Expression from Prefix into Postfix7-35</p> <p>7.5.9 Fully Parenthesizing an Infix Expression7-37</p> <p>7.5.10 Conversion of a Fully Parenthesized Infix Expression into Postfix7-40</p> <p>7.5.11 Conversion of a Fully Parenthesized Infix Expression into Prefix Form.....7-43</p> <p>7.5.12 Well Formedness of Parenthesis.....7-47</p> <p>7.5.12(A) Algorithm to Read in a Parenthesized Infix Expression and Check Well-formedness of Parenthesis.....7-47</p> <p>7.6 Representation of Two Stacks in an Array.....7-48</p> <p>7.7 Representation of Multiple Stacks in an Array7-49</p> <p>7.8 Recursion7-50</p> <p>7.8.1 Introduction7-50</p> <p>7.9 Converting a Recursive Function to an Equivalent C-Function.....7-51</p> <p>7.9.1 Finding Factorial of an Integer Number7-51</p> <p>7.10 Examples of Recursion.....7-52</p> <p>7.10.1 Finding Sum of the Elements Stored in an Array7-53</p> <p>7.10.2 Finding Length of a String7-53</p> <p>7.10.3 Reversing a String.....7-53</p>	<p>7.10.4 Searching a Number in an Array7-54</p> <p>7.10.5 Finding Largest Element in an Array.....7-54</p> <p>7.10.6 Binary Search.....7-54</p> <p>7.10.7 Tower of Hanoi Problem.....7-54</p> <p>7.11 Backtracking.....7-57</p> <p>7.12 Variants of Recursion7-57</p> <p>7.13 Backtracking Algorithmic Strategy.....7-58</p> <p>7.13.1 Use of Stack in Backtracking.....7-58</p> <p>7.13.2 Removal of Recursion7-58</p> <p>7.14 Case Study : The n-queens Problem (8 Queen as a Case)7-58</p> <p>7.15 Case Study : Android-Multiple Tasks / Multiple Activities and Back Stack7-60</p>
--	---

Unit - VI

Chapter 8 : Queues

8-1 to 8-35

<p>8.1 Array Implementation of Queues.....8-1</p> <p>8.1.1 Definition8-1</p> <p>8.1.2 Application of Queues8-1</p> <p>8.1.3 Array Representation and Implementation of Queues8-1</p> <p>8.2 Operations on Queue8-3</p> <p>8.2.1 Operations on Queue Implemented using Array8-3</p> <p>8.2.2 Queue as an ADT.....8-6</p> <p>8.2.3 Operations on Queue Implemented using Linked Structure8-8</p> <p>8.3 Circular Queues8-13</p> <p>8.3.1 Queue using a Circular Array8-13</p> <p>8.3.1(A) Implementation of a Circular Movement inside a Linear Array.....8-14</p>
--



8.3.2	Queue using a Circular Linked List.....	8-17	8.7.1	Josephus Problem.....	8-32
8.4	Dequeues.....	8-22	8.7.2	Job Scheduling.....	8-33
8.4.1	Implementation of Dequeue using a Circular Array....	8-23	8.7.3	Queue Simulation.....	8-34
8.5	Priority Queue	8-27	8.8	Case Study - Priority Queue in Bandwidth Management	8-34
8.5.1	Implementation of Priority Queues	8-27			
8.6	Representation of Multiple Queues in an Array.....	8-31	•	Model Question Paper - I (End Sem.)	Q-1 to Q-2
8.7	Applications of Queue	8-32			