VIT®
BHOPAL
www.vitbhopal.ac.in

TERM END EXAMINATIONS (TEE) – August 2024

Programme	: B.Tech.-BAI & Int. M.Tech.	Semester	:	Fall Semester 2024-2025
Course Name	: Data Structures and Analysis of Algorithms	Course Code	:	CSD3009
Date/Session	: 28 Aug 2024/Session-II	Slot	:	B22+B23+D24+E21+E22
Time	: 3 Hrs.	Max. Marks	:	100

Answer ALL the Questions

Q. No.		Question Description	Marks
		PART A – (60 Marks)	
1	(a)	Explain the fundamental role of data structures in organizing and managing data in a way that aligns with the goals of the algorithm and discuss how the design and implementation of data structures reflect computational thinking and problem-solving approaches.	12
		OR	
	(b)	Define complexity analysis and its importance in evaluating the efficiency of an algorithm and discuss the significance of best, worst, and average case complexities, providing examples of algorithms where each case is particularly relevant.	12
2	(a)	Provide pseudocode examples to demonstrate how key operations like insertion, deletion are implemented in both Singly Linked Lists and Circular Linked Lists. Highlight the differences and explain why Circular Linked Lists might be more suitable for certain applications.	12
		OR	
	(b)	Describe the Tower of Hanoi problem, including its rules and the objective.Explain the recursive algorithm to solve the problem, detailing recursive case and discuss how to simulate the recursive process iteratively, and provide an example with a small number of disks..	12
3	(a)	Discuss the concept of time complexity in the context of linear data structures(such as arrays, linked lists, and stacks). Explain the significance of best case, worst case, and average case time complexities with respect to common operations like search, insertion, and deletion. Provide detailed examples to illustrate how these complexities vary depending on the scenario.	12
		OR	
	(b)	Define the structure of Array-based Queues, Explain how enqueue and dequeue methods are handled in queues, including edge cases like queue overflow in Array-based Queues. Discuss the time complexity of these operations for implementations. Highlight the memory usage, especially how dynamic memory allocation in Queues.	12
4	(a)	Define BFS and explain how it explores the graph level by level. Discuss the differences in	12

how BFS operates on undirected graphs. Provide pseudocode for implementing BFS, detailing how the algorithm uses a queue to keep track of nodes to visit. Use a detailed example with a graph, showing step-by-step how BFS traverses the graph.

OR

(b) Describe Depth First Search (DFS) in detail. Explain the DFS algorithm, use recursive implementations with pseucode. Discuss how DFS manages graph traversal, especially in the presence of cycles. Additionally, analyze the time and space complexity of DFS and compare it with Breadth First Search (BFS). 12

5 (a) Explain the concept of a Minimum Spanning Tree (MST) and its importance in graph theory. Develop Kruskal's algorithm for finding the MST using a graph with at least 7 nodes. Provide a step-by-step explanation of the algorithm, along with pseudocode. Apply the algorithm to your custom graph, demonstrating each step in detail. Finally, analyze the time complexity of Kruskal's algorithm 12

OR

(b) Describe the purpose of Dijkstra's Algorithm and the specific problem it solves. Provide a detailed explanation of the algorithm's steps, including the initialization of distances, the selection process for the next node, and the method for updating path lengths. Explain how Dijkstra's Algorithm handles varying edge weights, and apply the algorithm to a custom example graph with at least 7 nodes and weighted edges. Demonstrate how the shortest path from a chosen source node to all other nodes is calculated, and analyze the time complexity of the algorithm based on this example. 1

PART B – (40 Marks)

6 Define a Binary Tree and its basic structure. Describe the different types of Binary Trees (Full, Complete, Perfect) with diagrams and examples.Provide examples of real-world applications where each type of Binary Tree is used.

7 Derive the time complexity of the Merge Sort algorithm using the recurrence relation $T(n) = T(n/2) + T(n/2) + n$. Explain each step of your derivation in detail, and discuss how this relates to the efficiency of Merge Sort. Include a discussion on how Merge Sort compares to other sorting algorithms in terms of time complexity.

8 Define a Binary Search Tree and explain its properties that make it unique (left child < parent < right child).Discuss how these properties influence the efficiency of operations searching, analysis time complexity BSTs and provide detailed examples and pseudocode for implementing these search operations, highlighting how the BST property is maintained. 8

9 Explain the Insertion Sort algorithm in detail. Provide a step-by-step explanation of its working process, including a pseudocode example. Discuss its time and space complexity, and compare it to other sorting algorithms. Additionally,describe real-world scenarios where Insertion Sort is particularly effective, and analyze situations where its complexity may present challenges.

10 Explain the differences between the Divide and Conquer approach and Greedy approache. Provide examples of each and discuss their applications.

⇔⇔⇔