Name of the Course: Data Structures

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Data Structures course introduces students to the foundational building blocks of programming and software development. It covers the systematic organization, management, and storage of data for efficient access and modification using various data structures. **Relevance:** Data structures are essential for solving computational problems efficiently, making this subject central to the study of computer science. Understanding how data is stored, manipulated, and accessed directly impacts the performance of applications. **Usefulness:** Knowledge of data structures allows students to choose the most appropriate structure for any problem, enabling optimal algorithm design and effective memory utilization. This leads to better program performance and scalability. **Application:** Data structures find wide applications in database indexing, compiler design, network routing, artificial intelligence, and graphics. Real-life scenarios such as job scheduling, expression evaluation, searching, and pathfinding extensively use stacks, queues, trees, and graphs. **Interest:** Students often find this subject intellectually stimulating as it challenges them to think logically and solve problems efficiently. Implementing structures like AVL Trees or graph traversals can be deeply engaging and rewarding. **Connection with Other Courses:** This course is closely linked with algorithms, operating systems, databases, and programming languages. Many advanced subjects assume a working knowledge of data structures, making it a prerequisite for deeper computer science learning. **Demand in the Industry:** Almost every tech role—from software development to system design—requires a strong understanding of data structures. Employers frequently test these concepts during technical interviews and coding assessments.

		Job Prospects: Proficiency in data structures opens doors to careers in software engineering, data science, system architecture, web development, and cybersecurity. It provides a solid base for roles that involve designing efficient and scalable solutions.
2	**Vertical:**	Major
3	**Type:**	Theory
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory)
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives (CO):** **CO 1.** To introduce students to the concept of Abstract Data Types (ADTs) and various data structures for efficient data representation and manipulation. **CO 2.** To develop an understanding of linked structures, including singly and doubly linked lists, and their applications. **CO 3.** To study stack and queue data structures, their implementation, and real-life applications like expression conversion and job scheduling. **CO 4.** To explore non-linear data structures such as trees and graphs, their operations, and their applications in problem-solving. **CO 5.** To understand the principles of priority queues, heaps, and hashing, and their role in efficient data access and management.	
8	**Course Outcomes (OC):** After successful completion of this course, students would be able to - **OC 1.** Define and implement various data structures using Abstract Data Types (ADTs) and understand their classifications and use cases. **OC 2.** Apply operations on linked lists, including traversal, insertion, deletion, and use them in practical applications like polynomial manipulations. **OC 3.** Implement stack and queue operations with array and linked representations, and apply them in real-world scenarios like delimiter checking and scheduling. **OC 4.** Design and traverse tree structures including binary trees, BSTs, AVL trees, and understand their applications in encoding and searching. **OC 5.** Implement graph structures, perform traversals using BFS and DFS, and solve shortest path and connectivity problems. **OC 6.** Use heaps and hashing techniques effectively for priority management, efficient searching, and collision handling in various applications.	
9	**Modules** **Module 1 (15 hours):**	
	Abstract Data Type: Different Data Types, different types of data structures & their classifications, Introduction to ADT, Creating user-specific ADT **Linked Structures**: ADT for linked list, Advantages & Disadvantages, Singly Linked List-Traversing, Searching, Prepending and Removing Nodes, applications of	

	linked list like polynomial equation, ADT of doubly linked list, Advantages & Disadvantages, Insertion and deletion of nodes at various positions **Stacks**: Stack ADT for Stack, Advantages & Disadvantages, Applications of stack like balanced delimiter, prefix to postfix notation **Queues**: Queue ADT, Advantages & Disadvantages, linked representations. Circular Queue operations, Dequeues, applications of queue like job scheduling queues	
	Module 2 (15 hours):	
	Trees: ADT for Tree Structure. Advantages & disadvantages, Binary Tree-Properties, Implementation and Traversals, Binary Search Tree, Balanced BST, Threaded Binary Trees, AVL Trees, Applications of Tree like Huffman Coding, **Priority Queues & Heaps:** Priority Queue, Priority Queue ADT, Advantages and Disadvantages, Applications, Heaps, types of heaps, Heapifying the element, **Graph:** Introduction, Graph ADT, Advantages and Disadvantages, Graph Representation using adjacency matrix and adjacency list, Graph operations like insertion and deletion of nodes, Graph Traversals using BFS & DFS, Applications of Graphs like shortest path algorithms, **Hashing:** Hash Table ADT, Advantages & Disadvantages, Concept of hashing, hash table, hash functions, collision, collision avoidance techniques, Applications of hashing	
10	**Text Books** 1. Introduction to Algorithm, Thomas H Cormen, PHI 2. Data Structures And Algorithms Made Easy, Narasimha Karumanchi, 2021	
11	**Reference Books** 1. Fundamentals of Computer Algorithms, Sartaj Sahni and Sanguthevar Rajasekaran Ellis Horowitz, Universities Press, 2018 2. Data Structures and Algorithms in Python, Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, Wiley, 2016	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**