DATA STRUCTURES

Upon successful completion of this course, students will be able to:

1. Recognize and explain the concept of different data structures and Abstract Data Types (ADT) in the context of programming and problem-solving.
2. Apply basic ideas of complexity analysis, including Big-Oh, Small-Oh, Big-Omega, Small Omega, and Big-Theta notations, to analyze algorithms and data structures.
3. Implement and evaluate different representations and applications of arrays, including address translation techniques.
4. Compare and contrast linear and binary search algorithms, identifying their advantages and disadvantages in various scenarios.
5. Illustrate the representations of linked lists, distinguish between different types such as singly linked lists, doubly linked lists, and circular linked lists, and perform operations unique to each type.
6. Define stacks, implement stack using arrays and linked lists, and demonstrate the use of Reverse Polish Notation for stack operations.
7. Define queues, implement queues using arrays and linked lists, and distinguish between different types of queues like circular queue and priority queue.
8. Explain binary trees, analyze quantitative properties such as height and depth, classify different types of binary trees, implement array, and linked representations, and perform tree traversals like in-order, pre-order, and post-order.
9. Describe binary search trees, identify their properties, demonstrate operations such as insertion, deletion, and searching, and define and discuss properties of AVL trees in maintaining balance.
10. Implement and compare different sorting algorithms like Bubble Sort, Selection Sort, Insertion Sort, Shell Sort, Merge Sort, Quick Sort, and Heap Sort, and evaluate their efficiency and performance in different scenarios.