**SEMESTER I**
**Name of the Programme: MCA**
**Course Code: CSA-500**
**Title of Course: Data Structures & Algorithms**
**Number of Credits: 2 (2L-0T-0P)**
**Effective from AY: 2022-23**

| | | |
|---|---|---|
| **Prerequisites for the course** | Programming using any Programming Language | |
| **Objectives** | The aim of the course is to emphasize the importance of data structures in implementing efficient algorithms. It provides an exposure to various algorithm design techniques and an introduction to algorithm analysis. | |
| **Content** | **Revision of Programming & Data Structures**<br>Problem solving, Data Types: Primitive and User Defined<br>Selection Constructs, Repetition Constructs, Recursion<br>Pointers<br>Algorithm Representation: - Pseudocode and flowcharts<br>Three level Approach<br>Abstract Data Types (ADTs)<br>Basic Linear Data Structures (LinkedList, Stack, Queue) | 5 hours |
| | **Algorithm Analysis**<br>Analysis of Algorithms<br>Algorithm Complexity: Space and Time<br>Cases of Complexity: Best, Worst and Average<br>Growth of Functions: Asymptotic Notation | 3 hours |
| | **Advanced Linear Data Structures**<br>Variants of Linked List and its applications (e.g. Polynomial addition, Sparse matrices)<br>Applications of stacks (e.g. Infix-to-Postfix conversion, Evaluating Postfix Expressions, Bracket Matching)<br>Variants of Queue and Applications | 4 hours |
| | **Nonlinear Data Structures:**<br>Trees: Binary Search Trees, AVL Trees, B-trees & variants.<br>Tree Traversal Algorithms<br>Heaps and its applications (e.g. implementation of Priority Queue)<br>Graph: Adjacency Matrix and Adjacency List Representations<br>Graph Traversal Algorithms: Breadth First Search and Depth First Search | 10 hours |
| | **Divide & Conquer Strategy**<br>Algorithms based on Divide and Conquer Strategy:<br>Sorting Algorithms (QuickSort, MergeSort)<br>Binary Search | 3 hours |
| | **Greedy Algorithms**<br>Huffman Coding Algorithm<br>Minimum Cost Spanning Tree (Prim's, Kruskal's)<br>Single Source Shortest Path (Dijkstra's) | 2 hours |
| | **Dynamic Programming**<br>Coin Change Problem<br>Longest Common Subsequence<br>All-pair shortest Path (floyd-warshall) | 3 hours |
| **Pedagogy** | • Lectures/Tutorials/Assignments/Quizzes<br>• Each data structure should be explained along with implementation of its ADT, its applications and complexity | |
| **References/ Readings** | 1. Horowitz, Ellis, Sartaj Sahni, and Susan Anderson-Freed. "Fundamentals of data structures in C" WH Freeman & Co., Latest Edition.<br>2. Thomas H. Cormen, Charles E. Leiserson, et al "Introduction to Algorithms", | |

| | |
|---|---|
| | Latest Edition<br>3. Allen, Weiss Mark. Data structures and algorithm analysis in C. Pearson Education India, Latest Edition.<br>4. Dasgupta, Papadimitriou, and Vazirani,Algorithms, by McGraw-Hill.<br>5. Jeri R. Hanly and Eliot B. Koffman “Problem Solving and Program Design in C” Pearson Education, VII Edition, 2012<br>6. R.G.Dromey "How to Solve it by Computer ", PHI , Latest Edition |
| **Course Outcomes** | Upon successful completion of the course, a student will be able to<br>● Implement common data structures such as lists, stacks, queues, graphs, and binary trees for solving programming problems.<br>● Identify and use appropriate data structures in the context of a solution to a given problem.<br>● Be able to analyze the complexity of a given algorithm |

(Back to top)