

SEMESTER-II

COURSE 3: DATA STRUCTURES USING C

Theory

Credits: 3

3 hrs/week

Course Objectives:

1. Understand the fundamentals of computer programming, Apply structured problem-solving approaches using algorithms, flowcharts, and C programming constructs.
2. Develop efficient logic using decision-making, loop, and jump control statements.
3. Utilize derived data types like arrays and strings for modular program design.
4. Design and implement modular solutions using functions, recursive logic, pointer operations, and dynamic memory management.
5. Handle complex data structures including structures, unions, and text file operations.

Course Outcomes:

At the End of the Course, The Students will be able to:

1. Understand basic computing concepts, programming paradigms and write structured C programs.
2. Apply control flow statements to solve logical and repetitive tasks in C.
3. Implement arrays and string operations to manage and manipulate data efficiently.
4. Design modular code using functions, recursion, and appropriate parameter passing.
5. Utilize pointers and memory operations for effective data handling. Demonstrate competence in dynamic memory allocation and text file processing.

Unit 1. Basic Concepts:

Algorithm: Definition and characteristics, Complexity analysis: Space Complexity, Time Complexity, Asymptotic Notations.

Introduction to Data structures: Definition, Types of Data structures, Abstract Data Types (ADT), Introduction to Linked Lists, Representation of linked lists in Memory, Comparison between Linked List and Array.

Unit 2. Linked Lists:

Types of Linked Lists - Singly Linked list, Doubly Linked list, Circularly Singly Linked list, Circularly Doubly Linked list; Implementation of Single Linked List ADT: Creating a List, Traversing a linked list, Searching in linked list, Insertion and deletion into linked list (At first Node, Specified Position, Last node).

Unit 3. Stacks and Queues:

Introduction to stack ADT, Implementation of stacks using array and Linked List, Application of stacks - Polish Notations - Converting Infix to Post Fix Notation - Evaluation of Post Fix Notation.

Queues: Introduction to Queue ADT, Implementation of Queues using array and Linked List, Application of Queues Types of Queues- Circular Queues, De-queues, Priority Queue, Heaps.

Unit 4. Searching and Sorting:

Linear or Sequential Search, Binary Search, Hashing and collision resolution.

Sorting: Selection Sort, Bubble Sort, Insertion Sort, Quick Sort and Merge Sort

Unit 5. Trees and Graphs:

Tree Terminology, Binary Tree Representation, Traversal techniques, Expression Tree, Binary Search Tree- Definition, Operations on a Binary Search Tree: Creation, Search, Insertion & deletion.

Graphs: Introduction to Graphs, Terminology, Representation (Adjacency Matrix, Adjacency List), Traversal of Graphs (DFS, BFS), Applications of Graphs, Concept of Shortest Path Problems, Concept of Minimum Cost Spanning Tree

Textbooks:

1. Data Structures Using C, Balagurusamy E. Tata MCGraw Hill
2. Data Structures using C, Reema Thareja, Third Edition, Oxford University Press

Reference Books:

1. Data Structures, Lipschutz, Schaum's Outline Series, Tata Mcgraw-hill
2. Data Structures Using C, Ch. Vijay Kumar, Pen Press International

Activities:

Outcome: Explain algorithm characteristics, time and space complexity, and asymptotic notations with clarity

Activity: Create a comparative chart of algorithms with different notations related to time and space complexities.

Evaluation Method: Rubric-based assessment of the chart for correctness, clarity, and depth of explanation on a 10-point scale.

Outcome: Implement and analyze different types of linked lists, including insertion, deletion, and traversal operations

Activity: Code a menu-driven program in C to implement single linked lists with all basic operations.

Evaluation Method: Practical lab assessment with test cases and Viva-style questioning to explain pointer manipulation.

SEMESTER-II

COURSE 3: DATA STRUCTURES USING C

Practical

Credits: 1

2 hrs/week

List of Experiments

1. Write a program to read 'N' numbers of elements into an array and also perform the following operation on an array
 - a. Add an element at the beginning of an array
 - b. Insert an element at given index of array
 - c. Update an element using a values and index
 - d. Delete an existing element
2. Write a program to implement Single Linked List with insertion, deletion and traversal operations
3. Write a program to implement Doubly Linked List with insertion, deletion and traversal operations
4. Write a program to implement the Stack operations using Arrays and Linked Lists.
5. Write a program to convert a given infix expression to a postfix expression using stacks.
6. Write a program to implement the Queue operations using Arrays and Linked Lists.
7. Write a program to implement the Circular Queue operations using Arrays.
8. Write a program for Binary Search Tree Traversals
9. Write a program to search an item in a given list using the following Searching Algorithms
 - a. Linear Search
 - b. Binary Search.
10. Write a program for implementation of the following Sorting Algorithms
 - a. Bubble Sort
 - b. Insertion Sort
 - c. Quick Sort
 - d. Merge Sort