SEMESTER- II
Name of the course: **Data Structures**
Course code: **UGCMSMAC03**
Total Class Hours: **120** Credit: **4+2 (Theory & Lab)**

Course Objectives:

1. Knowledge of fundamental data structures like: array, linked list, stack, queue, tree , graph, etc.
2. Capability to apply the knowledge for choosing a data structure to suitably model any data used in computer applications.

SYLLABUS

A Theory (60 Hours) 4 Credits

1. **Introduction** (2 L)

   Data and Information, Data Structure, Classification of Data Structures, Primitive Data Types, Abstract Data Types
2. **Arrays** (5 L)

   One Dimensional Array, Memory Representation of One Dimensional Array, Traversing, Insertion, Deletion, Searching, Merging of Arrays, Advantages and Limitations of Arrays, Multidimensional Arrays, Memory Representation of Two Dimensional Arrays, Row major, Column major representation, General Multi-Dimensional Arrays, Sparse Matrices (Array and Linked Representation)
3. **Linked Lists** (10 L)

   Singly, Doubly, Circular, Circular doubly lists, Insertion, Deletion, Traversal and Manipulation of all types of lists, Representation of Stack using Lists;
4. **Stacks** (5 L)

   Implementing single / multiple stacks in an Array; Prefix, Infix and Postfix expressions, Utility and conversion of these expressions from one to another; Applications of stack; Limitations of Array representation of stack
5. **Queues** (5 L)

   Array and Linked representation of linear Queue and circular Queue, enqueue and dequeue

operations, Deque, Priority Queues

6. **Recursion** (3 L)

   Developing Recursive Definition of Simple Problems and their implementation; Advantages and Limitations of Recursion; Understanding what goes behind Recursion (Internal Stack Implementation)

7. **Trees** (20 L)

   Introduction to Tree as a data structure; Binary Trees (Insertion, Deletion , Recursive and Iterative Traversals on Binary Search Trees); Threaded Binary Trees (Insertion, Deletion, Traversals); Heap, Memory Representation of Heap, Operation on Heap, Height-Balanced Trees (Various operations on AVL Trees).

8. **Searching and Sorting** (5 L)

   Linear Search, Binary Search, Comparison of Linear and Binary Search, Selection Sort, Insertion Sort, Insertion Sort, Shell Sort, Comparison of Sorting Techniques.

9. **Hashing** (5 L)

   Introduction to Hashing, Hash function, Address calculation techniques, Common hashing functions, Collision resolution, Chaining, Linear probing, Quadratic probing, Double hashing, Bucket hashing, Deletion and rehashing, Perfect Hashing Function

B Practical (60 Hours) 2 Credits

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates to sort a list of elements. Give user the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation. Use Templates.

8. Perform Queues operations using Circular Array implementation. Use Templates.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using linked list and add two polynomials.
11. WAP to calculate factorial and to compute the factors of a given no. (i)using recursion, (ii)using iteration
12. WAP to display Fibonacci series (i)using recursion, (ii) using iteration
13. WAP to calculate GCD of 2 number (i) with recursion (ii) without recursion
14. WAP to create a Binary Search Tree and include following operations in tree:
    a) Insertion (Recursive and Iterative Implementation)
    b) Deletion by copying
    c) Deletion by Merging
    d) Search a no. in BST
    e) Display its preorder, postorder and inorder traversals Recursively
    f) Display its preorder, postorder and inorder traversals Iteratively
    g) Display its level-by-level traversals
    h) Count the non-leaf nodes and leaf nodes
    i) Display height of tree
    j) Create a mirror image of tree
    k) Check whether two BSTs are equal or not
15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.
16. WAP to reverse the order of the elements in the stack using additional stack.
17. WAP to reverse the order of the elements in the stack using additional Queue.
18. WAP to implement Diagonal Matrix using one-dimensional array.
19. WAP to implement Lower Triangular Matrix using one-dimensional array.
20. WAP to implement Upper Triangular Matrix using one-dimensional array.
21. WAP to implement Symmetric Matrix using one-dimensional array.
22. WAP to create a Threaded Binary Tree as per inorder traversal, and implement operations like finding the successor / predecessor of an element, insert an element, inorder traversal.
23. WAP to implement various operations on AVL Tree.

Course Outcomes:

CO No.	Course Outcomes	Cognitive Level	PO Addressed	PSO Addressed
CO1	Ability to define fundamental data structures and with the manner in which these data structures can best be implemented.	R(1)	PO1	PSO1 PSO2
CO2	Ability to understand the complexity of basic operations like insert, delete, search on these data structures.	U(2)	PO2 PO3	PSO1 PSO2
CO3	Ability to analyse and know the applications of algorithms for sorting, pattern matching etc	An(4)	PO4	PSO3 PSO4
CO4	Ability to choose a data structure to suitably model any data used in computer applications.	E(5)	PO4	PSO4
CO5	Ability to assess efficiency trade-offs among different data structure implementations.	E(5)	PO5	PSO4 PSO5
CO6	Design programs using various data structures including hash tables, Binary and general search trees, heaps, graphs etc.	C(6)	PO6	PSO6

R= remembering, U = understanding, Ap = applying, An = analysing, E = evaluating, and C = creating

Reference Books

1. Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, "Data Structures and Algorithms", Pearson Education, 2009.
2. Mark Allen Weiss, "Data Structures and Algorithm Analysis in C", Pearson Education, 2nd edition.
3. Ellis Horowitz and Sartaj Sahni, “Fundamentals of Data Structures in C”, Universities Press
4. Seymour Lipschutz, " Data Structures ", Schaums Series, McGraw Hill Education.
5. Aaron M. Tenenbaum, Moshe J. Augenstein, Yedidyah Langsam, "Data Structures Using C and C++, 2nd edition, Pearson.
6. Michael T. Goodrich , Roberto Tamassia , David M. Mount “Data Structures and Algorithms in C++”, 2nd edition, JohnWiley & Sons, Inc.