- **BCA305: Data Structure using C (Discipline Specific Core Course) Credit: 06**
  **Total Marks: 100 Marks (Theory: 75 Marks, Internal Assessment: 25 Marks)**
  **Workload: 4 Lectures (Per Week), 4 Practical (Per Week)**

**Course Objective**

This course aims at developing the ability to use basic data structures like array, stacks, queues, lists, trees and hash tables to solve problems.

**Course Learning Outcomes**

At the end of the course, students will be able to:

1. Implement and empirically analyses linear and non-linear data structures like Arrays, Stacks, Queues, Lists, Trees, Heaps and Hash tables as abstract data structures.

2. Write a program, choosing a data structure, best suited for the application at hand.

3. Re-write a given program that uses one data structure, using a more appropriate/efficient data structure.

4. Write programs using recursion for simple problems. Explain the advantages and disadvantages of recursion.

**Detailed Syllabus**

UNIT-I: 10 mark
Arrays: single and multi-dimensional arrays, analysis of insert, delete and search operations in arrays (both linear search and binary search), implementing sparse matrices, and applications of arrays to sorting: selection sort, insertion sort, bubble sort, comparison of sorting techniques via empirical studies. Introduction to Vectors.

UNIT-II: 15 mark
Linked Lists: Singly- linked, doubly-linked and circular lists, analysis of insert, delete and search operations in all the three types, implementing sparse matrices. Introduction to Sequences.

UNIT-III: 10 mark
Queues: Array and linked representation of queue, de-queue, comparison of the operations on queues in the two representations. Applications of queues.

UNIT-IV: 15 mark
Stacks: Array and linked representation of stacks, comparison of the operations on stacks in the two representations, implementing multiple stacks in an array; applications of stacks: prefix, infix and postfix expressions, utility and conversion of these expressions from one to another; applications of stacks to recursion: developing recursive solutions to simple problems, advantages and limitations of recursion.

UNIT-V: 15 mark
Trees and Heaps: Introduction to tree as a data structure; binary trees, binary search trees, analysis of insert, delete, search operations, recursive and iterative traversals on binary search trees. Height-balanced trees (AVL), B trees, analysis of insert, delete, search operations on AVL and B trees. Introduction to heap as a data structure, Analysis of insert, extract-min/max and delete-min/max operations, applications to priority queues.

UNIT-VI: 10 mark
Hash Tables: Introduction to hashing, hash tables and hashing functions -insertion, resolving collision by open addressing, deletion, searching and their analysis, properties of a good hash function.

## Practical

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 polynomial.

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.

24. WAP to implement heap operations.

Text book:

1. Drozdek, A., (2012), Data Structures and algorithm in C++. 3rd edition. Cengage Learning.

2. Goodrich, M., Tamassia, R., & Mount, D., (2011). Data Structures and Algorithms Analysis in C++. 2nd edition. Wiley.

3. G.S. Baluja ,Data Structure through C ( A practical Approach) , Dhanpat Rai & Co.(p) LTD, New Delhi

References:

Foruzan, B.A. (2012) Computer Science: A Structured Approach Using C++, Cengage Learning

Lafore, R. (2008). Object Oriented Programming in C++. 4th edition. SAMS Publishing.

Sahni, S. (2011). Data Structures, Algorithms and applications in C++. 2ndEdition, Universities Press
Tenenbaum, A. M., Augenstein, M. J., & Langsam Y., (2009), Data Structures Using C and C++. 2nd edition. PHI