

**DEPARTMENT OF INFORMATION TECHNOLOGY & COMPUTER  
ENGINEERING**



**LESSON PLAN**

**SUBJECT: DATA STRUCTURE LABORATORY**

**BRANCH: 3RD SEM, IT, CE&IOT**

**FACULTY NAME: DR. ANITARANI BRAHMA**

## **Jharsuguda Engineering School, Jharsuguda**

### **LEARNING OUTCOMES:**

After completion of the course, the students will be able to:

- ☐ Explain basic terminologies and operations on data structures.
- ☐ Perform asymptotic and worst-case analysis of algorithms.
- ☐ Implement linear data structures.
- ☐ Apply trees and graphs to solve problems.
- ☐ Implement sorting and searching algorithms.

|                                                |
|------------------------------------------------|
| <b>SUBJECT NAME: DATA STRUCTURE Laboratory</b> |
| <b>SUBJECT CODE: PR3</b>                       |
| <b>FACULTY NAME: DR ANITARANI BRAHMA</b>       |
| <b>HOUR: 60 HR (WEEKLY LOAD 4HR)</b>           |

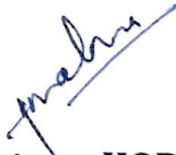
| Lab<br>Sl<br>no. | Module                                                                   | Topic/Program covered                                                                       | Session activities                                                                                                                                      |
|------------------|--------------------------------------------------------------------------|---------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1                | Module I:<br>Introduction to<br>Data Structures (2<br>Hours)             | Introduction to Data<br>Structures: Analyzing and<br>comparing time<br>complexity.          | Lab: Write programs to analyze<br>and compare the time<br>complexity of basic operations<br>(e.g., searching, insertion) on<br>arrays and linked lists. |
| 2                | Module II: Linear<br>Data Structures:<br>Stacks and Queues<br>(12 Hours) | Stack operations (push,<br>pop, peek) using arrays.                                         | Lab: Implement stack operations<br>(push, pop, peek) using arrays.                                                                                      |
| 3                |                                                                          | Stack operations (push,<br>pop, peek) using linked<br>lists.                                | Lab: Implement stack operations<br>(push, pop, peek) using linked<br>lists.                                                                             |
| 4                |                                                                          | Applications of stacks:<br>Infix-to-postfix<br>conversion.                                  | Lab: Develop a program for infix-<br>to-postfix conversion using a<br>stack.                                                                            |
| 5                |                                                                          | Applications of stacks:<br>Postfix evaluation.                                              | Lab: Develop a program for<br>postfix evaluation using a stack.                                                                                         |
| 6                |                                                                          | Queue operations<br>(enqueue, dequeue) using<br>arrays.                                     | Lab: Implement queue<br>operations (enqueue, dequeue)<br>using arrays.                                                                                  |
| 7                |                                                                          | Queue operations<br>(enqueue, dequeue) using<br>linked lists, Circular<br>Queues, Dequeues. | Lab: Implement queue<br>operations (enqueue, dequeue)<br>using linked lists. Write<br>programs for circular queues                                      |

|    |                                                                     |                                                                                 |                                                                                                                                                             |
|----|---------------------------------------------------------------------|---------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    |                                                                     |                                                                                 | and dequeues.                                                                                                                                               |
| 8  | Module III: Linked Lists (10 Hours)                                 | Singly linked list operations: Insertion.                                       | Lab: Implement functions for insertion operations (at beginning, end, specific position) in a singly linked list.                                           |
| 9  |                                                                     | Singly linked list operations: Deletion.                                        | Lab: Implement functions for deletion operations (from beginning, end, specific position, by value) in a singly linked list.                                |
| 10 |                                                                     | Singly linked list operations: Traversal.                                       | Lab: Implement functions for traversing and displaying elements of a singly linked list.                                                                    |
| 11 |                                                                     | Circular and Doubly linked lists: Creation and manipulation.                    | Lab: Write programs to create and manipulate circular linked lists (insertion, deletion, traversal).                                                        |
| 12 |                                                                     | Doubly linked lists: Creation and manipulation, Stack/Queue using linked lists. | Lab: Write programs to create and manipulate doubly linked lists (insertion, deletion, traversal). Implement stack and queue operations using linked lists. |
| 13 | Module IV: Non-Linear Data Structures (Trees and Graphs) (16 Hours) | Binary tree operations: Insertion.                                              | Lab: Implement a function to insert nodes into a binary tree.                                                                                               |
| 14 |                                                                     | Binary tree operations: Deletion.                                               | Lab: Implement a function to delete nodes from a binary tree.                                                                                               |
| 15 |                                                                     | Binary tree operations: Traversal (Inorder, Preorder, Postorder).               | Lab: Implement recursive and iterative functions for inorder, preorder, and postorder traversals of a binary tree.                                          |

|    |                              |                                                        |                                                                                                                                |
|----|------------------------------|--------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| 16 |                              | Binary Search Tree (BST) operations.                   | Lab: Develop programs for creating and performing operations (insertion, searching, deletion) on a Binary Search Tree.         |
| 17 |                              | AVL Tree operations (conceptual/basic implementation). | Lab: (Conceptual) Trace AVL tree balancing steps. (Basic implementation) Implement a simple AVL tree insertion with rotations. |
| 18 |                              | Graph representations: Adjacency List.                 | Lab: Implement graph representation using an adjacency list for both directed and undirected graphs.                           |
| 19 |                              | Graph representations: Adjacency Matrix.               | Lab: Implement graph representation using an adjacency matrix for both directed and undirected graphs.                         |
| 20 |                              | Basic graph traversals: BFS.                           | Lab: Implement Breadth-First Search (BFS) traversal for a given graph.                                                         |
| 21 |                              | Basic graph traversals: DFS.                           | Lab: Implement Depth-First Search (DFS) traversal for a given graph.                                                           |
| 22 | Module V: Sorting (10 Hours) | Sorting algorithms: Bubble Sort and Selection Sort.    | Lab: Implement Bubble Sort and Selection Sort algorithms and analyze their performance.                                        |
| 23 |                              | Sorting algorithms: Insertion Sort.                    | Lab: Implement Insertion Sort algorithm and analyze its performance.                                                           |
| 24 |                              | Sorting algorithms: Merge Sort.                        | Lab: Implement Merge Sort algorithm and analyze its performance.                                                               |

|    |                                 |                                                                                          |                                                                                                                                                  |
|----|---------------------------------|------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| 25 |                                 | Sorting algorithms: Quick Sort.                                                          | Lab: Implement Quick Sort algorithm and analyze its performance.                                                                                 |
| 26 | Module VI: Searching (10 Hours) | Searching using Binary Search Trees (BST).                                               | Lab: Write programs for searching elements efficiently using Binary Search Trees.                                                                |
| 27 |                                 | Searching using Hash Tables: Implementation.                                             | Lab: Implement a hash table with basic hash functions and collision resolution techniques (e.g., linear probing).                                |
| 28 |                                 | Searching using Hash Tables: Operations.                                                 | Lab: Implement insertion, deletion, and searching operations in a hash table.                                                                    |
| 29 |                                 | Symbol table operations using balanced search trees (e.g., AVL/Red-Black Tree concepts). | Lab: (Conceptual) Discuss and trace symbol table operations using balanced search trees. (Basic implementation) Implement a simple symbol table. |
| 30 |                                 | Project/Case Study: Applying Data Structures.                                            | Lab: Work on a mini-project or case study that requires the application of various data structures learned throughout the course.                |

  
Signature of Faculty

  
Signature, HOD i/c