Vidyavardhini's College of Engineering & Technology, Vasai

First Year Engineering (NEP)
Academic Year 2025-26

Experiment No.8
Implementation of Binary Search Tree and its traversal methods. The program enters a sequence of numbers and constructs a BST for it. Display Preorder, Inorder and Postorder sequences of the BST traversal
Date of Performance:
Date of Submission:

Experiment No. 8 : Implementation of Binary Search Tree and its traversal methods.

Aim: Program enter sequence of numbers and construct BST for it. Display Preorder, Inorder and Postorder sequences of the BST traversal

Objectives:

1. Understand the Binary Search Tree (BST) data structure
2. Implement core operations of a BST: Insertion of nodes, Searching for a value, Deletion of nodes (optional/advanced)
3. Explore and implement tree traversal methods: In-order traversal, Pre-order traversal, Post-order traversal – used for deleting/freeing nodes

Theory:

Tree traversal is the process of systematically visiting all the nodes in a tree data structure, typically starting from the root, and visiting each node exactly once. The nodes are visited in a specific order, which can vary based on the traversal method, such as Inorder, Preorder, or Postorder, depending on whether the root, left child, or right child is processed first. This helps in performing operations like searching, sorting, or printing the elements of the tree.

Inorder Traversal (Left, Root, Right):

Inorder traversal is a method of visiting all the nodes of a binary tree where the nodes are recursively visited in the following order: left subtree, root, right subtree. This traversal is commonly used in binary search trees (BST) as it visits the nodes in sorted order.

Basic Operations in Inorder Traversal:

- Visit Left Subtree: Recursively visit the left subtree of the current node.
- Visit Root: Process the current node (typically by printing its value).
- Visit Right Subtree: Recursively visit the right subtree of the current node.

Preorder Traversal (Root, Left, Right):

Preorder traversal is a method of visiting all the nodes of a binary tree where the nodes are recursively visited in the following order: root, left subtree, right subtree. This traversal is useful for creating a copy of the tree or for operations where the root needs to be processed before its children.

Basic Operations in Preorder Traversal:

- Visit Root: Process the current node (typically by printing its value).
- Visit Left Subtree: Recursively visit the left subtree of the current node.
- Visit Right Subtree: Recursively visit the right subtree of the current node.

Postorder Traversal (Left, Right, Root):

Postorder traversal is a method of visiting all the nodes of a binary tree where the nodes are recursively visited in the following order: left subtree, right subtree, root. This traversal is useful for operations where we need to process the children of a node before processing the node itself, such as in tree deletion or evaluating expressions.

Basic Operations in Postorder Traversal:

- Visit Left Subtree: Recursively visit the left subtree of the current node.
- Visit Right Subtree: Recursively visit the right subtree of the current node.
- Visit Root: Process the current node (typically by printing its value).

Algorithm:

Algorithm for Preorder Traversal

```
Step 1: Repeat Steps 2 to 4 while TREE != NULL
Step 2:Write TREE -> DATA
Step 3: PREORDER(TREE -> LEFT)
Step 4: PREORDER(TREE -> RIGHT)
[END OF LOOP]
Step 5: END
```

Algorithm for inorder traversal

```
Step 1: Repeat Steps 2 to 4 while TREE != NULL
Step 2: INORDER(TREE -> LEFT)
Step 3: Write TREE -> DATA
Step 4: INORDER(TREE -> RIGHT)
[END OF LOOP]
Step 5: END
```

Algorithm for postorder traversal

```
Step 1: Repeat Steps 2 to 4 while TREE != NULL
Step 2: POSTORDER(TREE -> LEFT)
Step 3: POSTORDER(TREE -> RIGHT)
Step 4: Write TREE -> DATA
[END OF LOOP]
Step 5: END
```

Code and Output:

Conclusion: Students will conclude their understanding of Implementation of Binary Search Tree and its traversal methods.