ZAIN ALI
BSAI-3A(AI-LAB)
SU92-BSAIM-F24-046

➢ Explanation of Two Tasks

Task 1: DFS using a Stack

This is an iterative Depth-First Search algorithm implemented using a stack. It explores as far as possible along each branch before backtracking.

Steps:
1. Push the start node onto the stack.
2. While the stack is not empty:
   - Pop the top node.
   - If it's not visited:
     - Print it.
     - Mark it visited.
     - Push its unvisited neighbors (in reversed order for consistent traversal).

Because it always pops the last pushed node, the traversal is depth-first.

Example Output on your graph: A B D E F C

Task 2: Preorder / Inorder / Postorder Traversal of a Tree

This task performs three standard traversals on a binary tree stored as a dictionary with each node having [left, right] children.

Preorder (root-left-right): Visit the node first, then left subtree, then right subtree.
Output: A B D E C F

Inorder (left-root-right): Visit the left subtree first, then the node, then right subtree.
Output: D B E A C F

Postorder (left-right-root): Visit the left subtree first, then right subtree, then the node last.
Output: D E B F C A

These functions all recursively visit the children in the specified order.

Key Differences Between the Two Tasks

• DFS Stack works on a general graph and uses an explicit stack to avoid recursion.
• Tree Traversals work on a binary tree and use recursion directly (implicit call stack).
• DFS Stack requires a visited set to avoid cycles, while Tree Traversals do not because a tree has no cycles.