NAME:

Sufyan Ajmal

# Code Documentation

## 1) DFS with Stack

## This program performs DFS traversal on a graph using an explicit stack instead of recursion.

## The graph is stored as a dictionary where each node has a list of its neighbors.

## A visited set is used to keep track of already visited nodes, so the algorithm does not repeat or fall into loops.

## The algorithm starts from the given node, pushes it into the stack, and then explores as deep as possible before backtracking.

## Neighbors are added in reverse order so that the left-most node is visited first (same as recursive DFS).

## Example flow (Start = A):

## Visit A → push its neighbors → B visited → then D and E → finally C.



## 2) Research about "In order, Preorder, Post order

**1. Preorder (Root → Left → Right)**

* First visit the **root node**
* Then the **left subtree**
* Finally, the **right subtree**  
   **Use:** Converting an expression tree into **prefix form**.

**2. In order (Left → Root → Right)**

* First visit the **left subtree**
* Then the **root**
* Finally, the **right subtree**  
   **Use:** In a **Binary Search Tree (BST)**, in order traversal gives the nodes in **sorted order**.

**3. Post order (Left → Right → Root)**

* First visit the **left subtree**
* Then the **right subtree**
* Finally, the **root**  
  👉 **Use:** Useful for **deleting a tree** or generating **postfix expressions**.

