**Name: Usman Ul Haq**

**Roll# BSAIM-035**

**AI LAB TASKs**

**LAB TASK 5:**

**DFS with Stack & Node**

class Node:

def \_init\_(self, value):

self.value = value

self.left = None

self.right = None

class Graph:

def \_init\_(self):

self.adj\_list = {}

def add\_edge(self, node, neighbor):

if node not in self.adj\_list:

self.adj\_list[node] = []

self.adj\_list[node].append(neighbor)

def dfs\_stack(self, start):

stack = [start]

visited = set()

while stack:

node = stack.pop()

if node not in visited:

print(node, end=" ")

visited.add(node)

if node in self.adj\_list:

for neighbor in reversed(self.adj\_list[node]):

if neighbor not in visited:

stack.append(neighbor)

graph = Graph()

graph.add\_edge('A', 'B')

graph.add\_edge('A', 'C')

graph.add\_edge('B', 'D')

graph.add\_edge('B', 'E')

graph.add\_edge('C', 'F')

graph.add\_edge('C', 'G')

print("DFS Traversal using Stack:")

graph.dfs\_stack('A')

**Output**

**DFS Traversal using Stack:**

**A B D E C F G**

**2. Research about "Inorder, Preorder, Postorder" and implement in DFS**

class TreeNode:

def \_\_init\_\_(self, value):

self.value = value

self.left = None

self.right = None

def inorder\_traversal(root):

if root:

inorder\_traversal(root.left)

print(root.value, end=" ")

inorder\_traversal(root.right)

def preorder\_traversal(root):

if root:

print(root.value, end=" ")

preorder\_traversal(root.left)

preorder\_traversal(root.right)

def postorder\_traversal(root):

if root:

postorder\_traversal(root.left)

postorder\_traversal(root.right)

print(root.value, end=" ")

root = TreeNode('A')

root.left = TreeNode('B')

root.right = TreeNode('C')

root.left.left = TreeNode('D')

root.left.right = TreeNode('E')

root.right.left = TreeNode('F')

root.right.right = TreeNode('G')

print("\nInorder Traversal:")

inorder\_traversal(root)

print("\nPreorder Traversal:")

preorder\_traversal(root)

print("\nPostorder Traversal:")

postorder\_traversal(root)

**Output**

**Inorder Traversal:**

**D B E A F C G**

**Preorder Traversal:**

**A B D E C F G**

**Postorder Traversal:**

**D E B F G C A**