Inorder, Preorder, Postorder Traversals in DFS

Tree traversal is the process of visiting each node in a tree in a specific order. Depth First Search (DFS) traversals are one of the most common ways to explore trees. There are three main types of DFS traversals:  
1. Inorder  
2. Preorder  
3. Postorder

# 1. Inorder Traversal (Left → Root → Right)

Steps:  
- First visit the Left subtree  
- Then visit the Root  
- Finally, visit the Right subtree  
  
Inorder traversal is often used with Binary Search Trees because it gives the nodes in sorted order.

Example Tree:  
 A  
 / \  
 B C  
 / \  
 D E  
  
Inorder = D → B → E → A → C

# 2. Preorder Traversal (Root → Left → Right)

Steps:  
- First visit the Root  
- Then visit the Left subtree  
- Finally, visit the Right subtree  
  
Preorder traversal is useful for creating a copy of a tree or for expression trees.

Example Tree:  
 A  
 / \  
 B C  
 / \  
 D E  
  
Preorder = A → B → D → E → C

# 3. Postorder Traversal (Left → Right → Root)

Steps:  
- First visit the Left subtree  
- Then visit the Right subtree  
- Finally, visit the Root  
  
Postorder traversal is useful for deleting or freeing nodes in a tree because children are processed before the parent.

Example Tree:  
 A  
 / \  
 B C  
 / \  
 D E  
  
Postorder = D → E → B → C → A

# Python Implementation

class Node:  
 def \_\_init\_\_(self, value):  
 self.value = value  
 self.left = None  
 self.right = None  
  
# Inorder Traversal (Left → Root → Right)  
def inorder(root):  
 if root:  
 inorder(root.left)  
 print(root.value, end=" ")  
 inorder(root.right)  
  
# Preorder Traversal (Root → Left → Right)  
def preorder(root):  
 if root:  
 print(root.value, end=" ")  
 preorder(root.left)  
 preorder(root.right)  
  
# Postorder Traversal (Left → Right → Root)  
def postorder(root):  
 if root:  
 postorder(root.left)  
 postorder(root.right)  
 print(root.value, end=" ")  
  
# Example Tree  
root = Node("A")  
root.left = Node("B")  
root.right = Node("C")  
root.left.left = Node("D")  
root.left.right = Node("E")  
  
print("Inorder: ")  
inorder(root) # Output: D B E A C  
print("\nPreorder: ")  
preorder(root) # Output: A B D E C  
print("\nPostorder: ")  
postorder(root) # Output: D E B C A