Class Node:

Def \_\_init\_\_(self, data):

Self.left = None

Self.right = None

Self.data = data

# Insert Node

Def insert(self, data):

If self.data:

If data < self.data:

If self.left is None:

Self.left = Node(data)

Else:

Self.left.insert(data)

Else data > self.data:

If self.right is None:

Self.right = Node(data)

Else:

Self.right.insert(data)

Else:

Self.data = data

# Print the Tree

Def PrintTree(self):

If self.left:

Self.left.PrintTree()

Print( self.data),

If self.right:

Self.right.PrintTree()

# Inorder traversal

# Left -> Root -> Right

Def inorderTraversal(self, root):

Res = []

If root:

Res = self.inorderTraversal(root.left)

Res.append(root.data)

Res = res + self.inorderTraversal(root.right)

Return res

Root = Node(27)

Root.insert(14)

Root.insert(35)

Root.insert(10)

Root.insert(19)

Root.insert(31)

Root.insert(42)

Print(root.inorderTraversal(root))