Problem 4: Construct a BST from a preorder traversal

```
class TreeNode:
    def __init__(self, value):
        self.val = value
        self.left = None
        self.right = None



def construct_bst(preorder):
    if not preorder:
        return None

    root = TreeNode(preorder[0])
    stack = [root]

    for value in preorder[1:]:
        node = TreeNode(value)

        if value < stack[-1].val:
            stack[-1].left = node
        else:
            while stack and value > stack[-1].val:
                last = stack.pop()

            last.right = node

        stack.append(node)

    return root
```

```
def inorder_traversal(root):
    if root is None:
        return []

    result = []
    stack = []

    while stack or root:
        if root:
            stack.append(root)
            root = root.left
        else:
            node = stack.pop()
            result.append(node.val)
            root = node.right

    return result

preorder = [8, 5, 1, 7, 10, 12]
bst = construct_bst(preorder)

inorder = inorder_traversal(bst)
print("Inorder traversal:", inorder)
```