4. Trees :: Binary Trees :: Traversals :: Depth First :: Inorder

An **inorder** traversal is one in which we traverse the left subtree of the root node first, then we visit the root node, and then we traverse the right subtree of the root node. For the example tree, the nodes would be visited in this order:

Note that our definition of inorder traversal is recursive in nature:

1. Perform an inorder traversal of the subtree rooted at the left child of the root node (if it exists).
2. Visit the root node.
3. Perform an inorder traversal of the subtree rooted at the right child of the root node (if it exists).

Here is the pseudocode:

```
-- Performs an inorder traversal of the subtree rooted at pRoot. pVisitor is an object which
-- implements a method named visit() which will be called as each Node is visited.
Method inorderTraversal(In: Node pRoot, In: pVisitor) Returns Nothing
   -- Perform an inorder traversal of the subtree rooted at the left child pRoot if it exists.
   If pRoot has a left child Then inorderTraversal(pRoot.leftChild, pVisitor)
   -- Visit the root node passing the data stored in the root node.
   pVisitor.visit(pRoot.data)
   -- Perform an inorder traversal of the subtree rooted at the right child pRoot if it exists.
   If pRoot has a right child Then inorderTraversal(pRoot.rightChild, pVisitor)
End Method inorderTraversal
```

4. Trees :: Binary Trees :: Traversals :: Depth First :: Inorder