Challenge 2: Binary Tree Traversal

Consider a binary tree:

```cpp
class Tree {
    Tree left, right, parent;
    bool mark;
}
```

We are given a binary tree with the following properties:

- It is well formed, in the sense that following a child pointer (left or right) and then following a parent pointer brings us to the original node. Moreover, the parent pointer of the root is null.
- It has at least one node, and each node has 0 or 2 children.

We do not know the initial value of the mark fields.

Our goal is to set all mark fields to true. The algorithm below (Morris 1979) works in time linear in the number of nodes, as usual, but uses only a constant amount of extra space.

```cpp
void markTree(Tree root) {
    Tree x, y;
    x = root;
    do {
        x.mark = true;
        if (x.left == null && x.right == null) {
            y = x.parent;
        } else {
            y = x.left;
            x.left = x.right;
            x.right = x.parent;
            x.parent = y;
        }
        x = y;
    } while (x != null);
}
```

**Tasks.** Prove that:

1. upon termination of the algorithm, all mark fields are set
2. the tree shape does not change
3. the code does not crash, and
4. the code terminates.

As a bonus, prove that the nodes are visited in depth-first order.