**Invert a Binary Tree**

bool invertHelper(TreeNode<int> \*node, TreeNode<int> \*leaf, stack<TreeNode<int> \*> &stk) {

    stk.push(node);

    if (!node->left && !node->right) {

        if (node->data == leaf->data)

            return true;

        else {

            stk.pop();

            return false;

        }

    }

    if (node->left) {

        if (invertHelper(node->left, leaf, stk))

            return true;

    }

    if (node->right) {

        if (invertHelper(node->right, leaf, stk))

            return true;

    }

    stk.pop();

    return false;

}

TreeNode<int> \*invertBinaryTree(TreeNode<int> \*root, TreeNode<int> \*leaf) {

    if (!root)

        return NULL;

    stack<TreeNode<int> \*> stk;

    bool found = invertHelper(root, leaf, stk);

    TreeNode<int> \*newRoot = stk.top();

    stk.pop();

    TreeNode<int> \*parent = newRoot;

    while (!stk.empty()) {

        auto curr = stk.top();

        stk.pop();

        if (curr->left == parent) {

            curr->left = NULL;

            parent->left = curr;

        } else {

            curr->right = curr->left;

            curr->left = NULL;

            parent->left = curr;

        }

        parent = curr;

    }

    return newRoot;

}