

Course T1Y2: Advanced Algorithms

Lecturer: Bou Channa

Student’s name: Chea Ilong

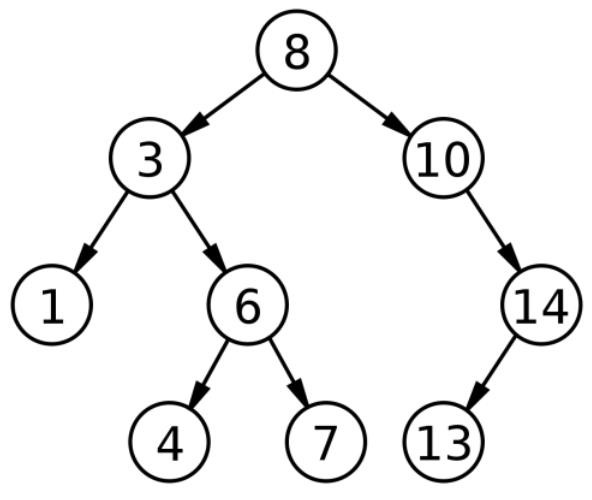
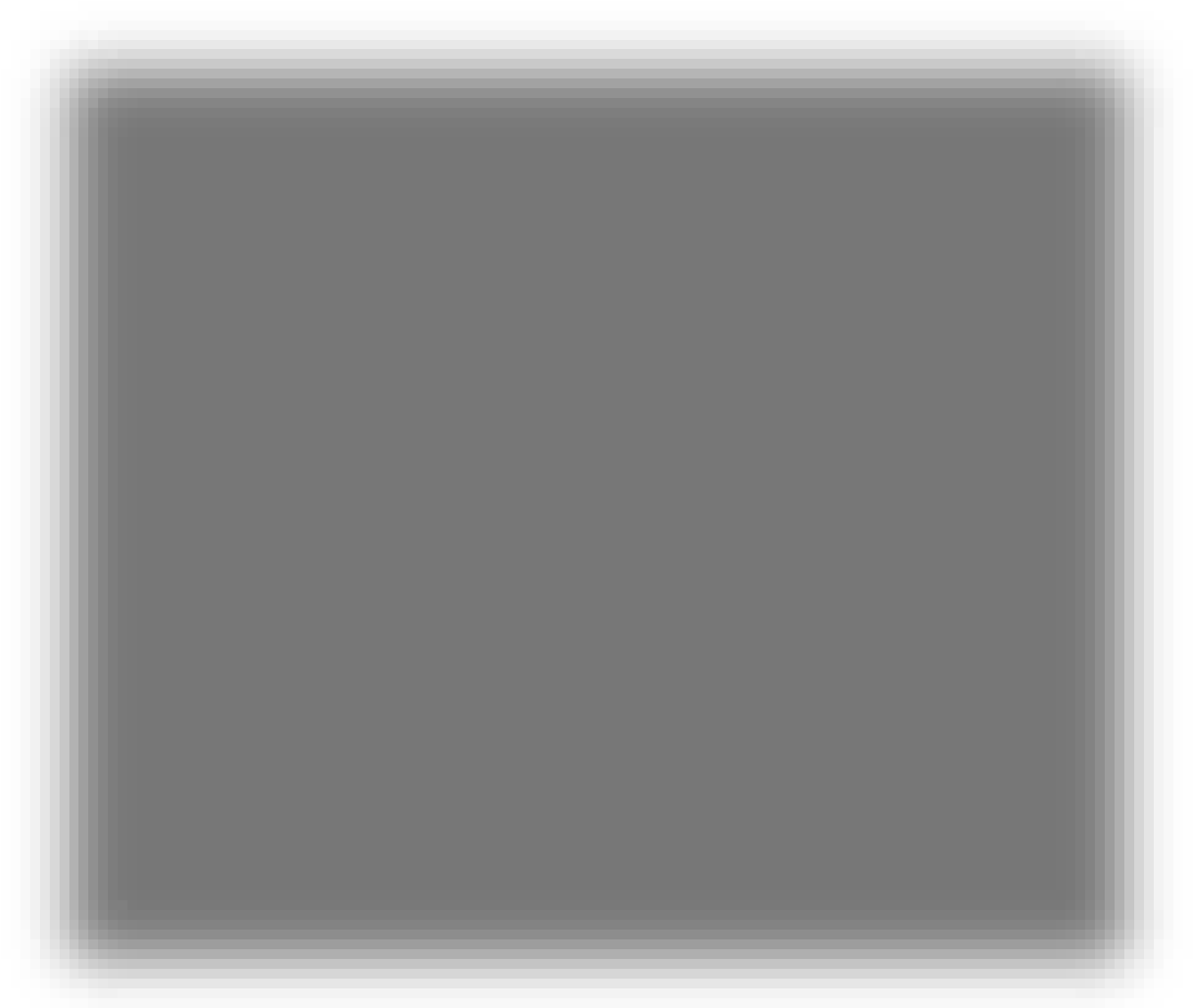
ID: 100022

Group: 1 SE Gen10

Assignment 7

❑ Exercise

Given a binary search tree (BST) below.



What are the output of the following tree traversal?

1. Pre-order traversal
2. In-order traversal
3. Post-order traversal

Result:

A screen shot of a computer

Description automatically generated

Source code:

#include <iostream>

using namespace std;

struct Node

{

    int data;

    Node \*left;

    Node \*right;

};

class BinaryTree

{

    Node \*root; // Root of the tree

    int size;

public:

    BinaryTree()

    {

        size = 0;

        root = nullptr;

    }

    Node \*insert(Node \*root, int newdata)

    {

        if (root == nullptr)

        {

            root = new Node;

            root->left = nullptr;

            root->right = nullptr;

            root->data = newdata;

            size++;

        }

        else if (newdata < root->data)

        {

            root->left = insert(root->left, newdata);

        }

        else if (newdata > root->data)

        {

            root->right = insert(root->right, newdata);

        }

        // Duplicates are ignored

        return root;

    }

    void insert(int newdata)

    {

        root = insert(root, newdata);

    }

    void preOrder(Node \*node)

    { // DLR (data, left, right)

        if (node != nullptr)

        {

            cout << node->data << " ";

            preOrder(node->left);

            preOrder(node->right);

        }

    }

    void inOrder(Node \*node)

    { // LDR (left, data, right)

        if (node != nullptr)

        {

            inOrder(node->left);

            cout << node->data << " ";

            inOrder(node->right);

        }

    }

    void postOrder(Node \*node)

    { // LRD (left, right, data)

        if (node != nullptr)

        {

            postOrder(node->left);

            postOrder(node->right);

            cout << node->data << " ";

        }

    }

    void preOrderTraversal()

    {

        preOrder(root);

    }

    void inOrderTraversal()

    {

        inOrder(root);

    }

    void postOrderTraversal()

    {

        postOrder(root);

    }

};

int main()

{

    BinaryTree tree;

    // Insert values

    tree.insert(8);

    tree.insert(3);

    tree.insert(1);

    tree.insert(6);

    tree.insert(4);

    tree.insert(7);

    tree.insert(10);

    tree.insert(14);

    tree.insert(13);

    // Display traversals

    cout << "Preorder: ";

    tree.preOrderTraversal();

    cout << endl;

    cout << "Inorder: ";

    tree.inOrderTraversal();

    cout << endl;

    cout << "Postorder: ";

    tree.postOrderTraversal();

    cout << endl;

    return 0;

}