CADT IDT វិទ្យាស្ថានបច្ចេកវិទ្យាឌីជីថល Institute of Digital Technology

Course T1Y2: Advanced Algorithms

Lecturer: Bou Channa

Student's name: Chea Ilong

ID: 100022

Group: 1 SE Gen10

Assignment 7

✈ Exercise

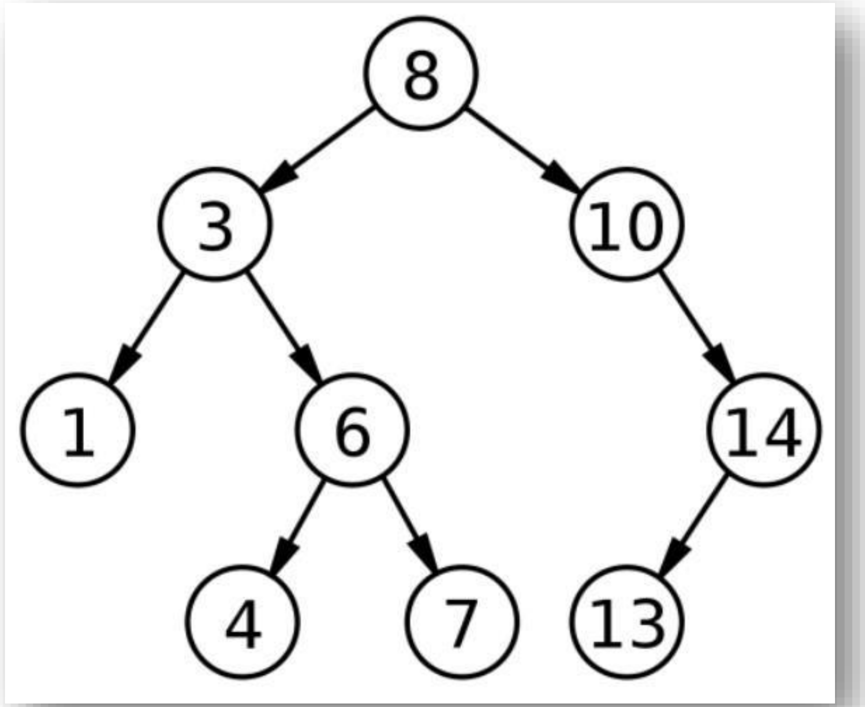
Given a binary search tree (BST) below.

What are the output of the following tree traversal?

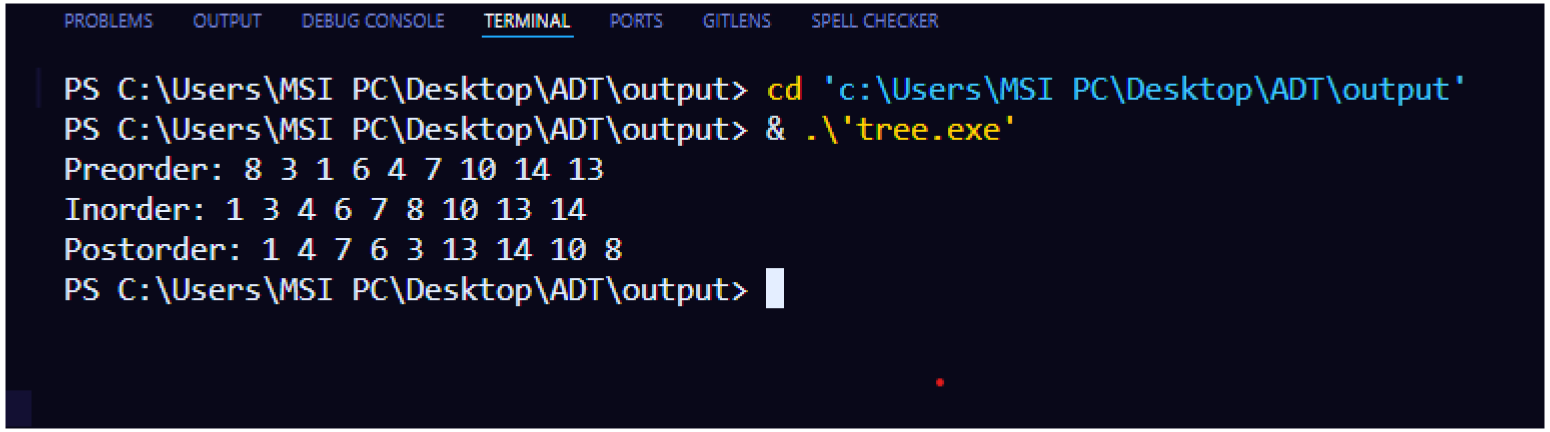
a. Pre-order traversal

b. In-order traversal

c. Post-order traversal

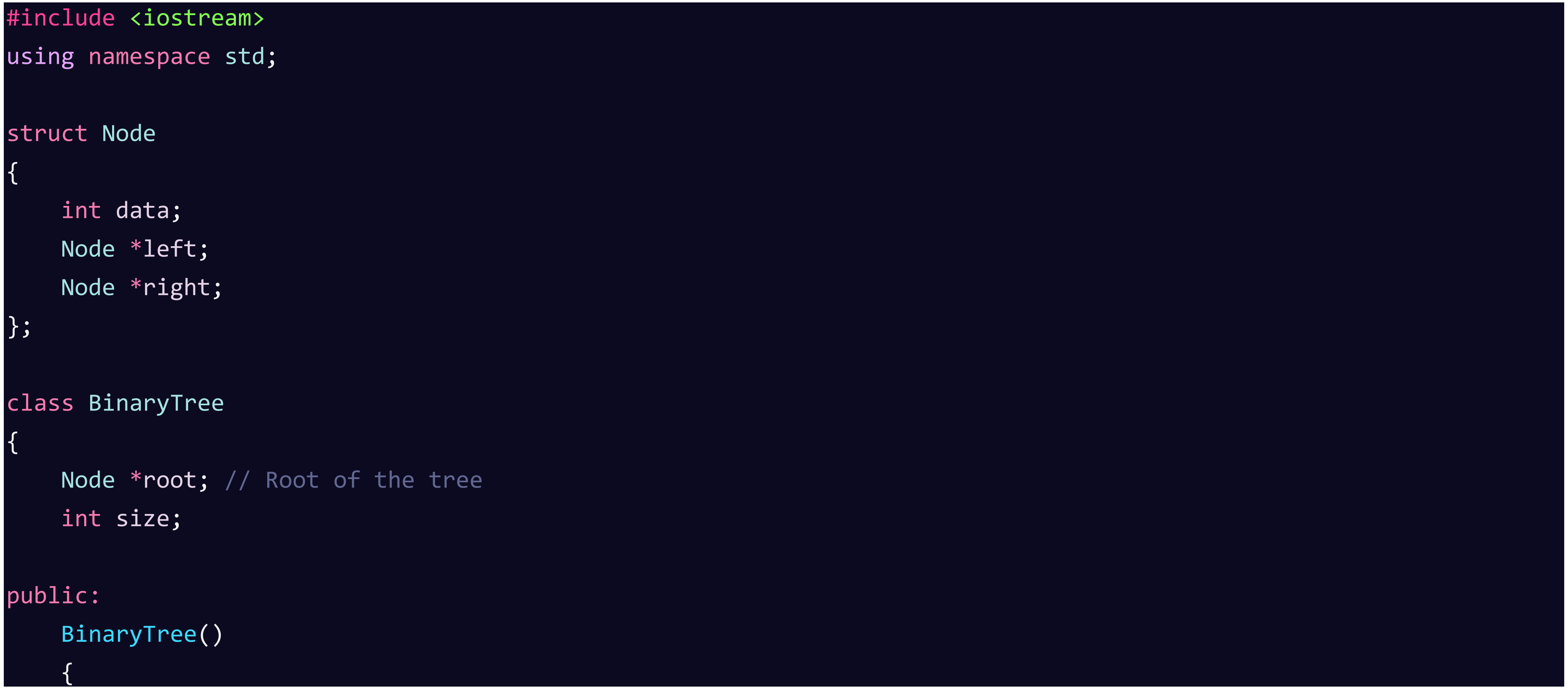


Result:



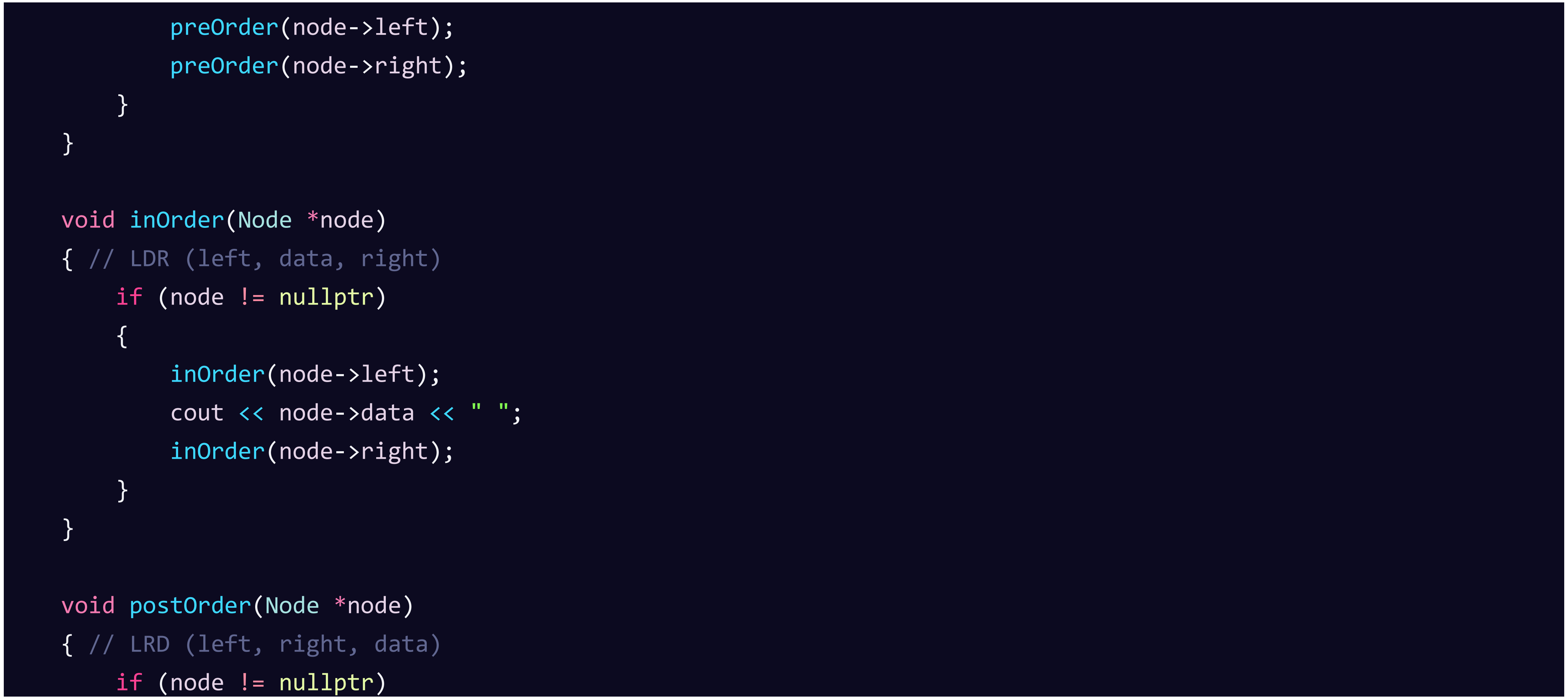
```
PS C:\Users\MSI PC\Desktop\ADT\output> cd 'c:\Users\MSI PC\Desktop\ADT\output'
PS C:\Users\MSI PC\Desktop\ADT\output> & .\'tree.exe'
Preorder: 8 3 1 6 4 7 10 14 13
Inorder: 1 3 4 6 7 8 10 13 14
Postorder: 1 4 7 6 3 13 14 10 8
PS C:\Users\MSI PC\Desktop\ADT\output>
```

Source code:



```cpp
#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()
```

```cpp
    {
        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;
}
```