Practical # 13

Objective:

Discuss tree data structure. ***Develop*** *a C++ program that traverse the binary tree.*

Theory:

In this Lab, we discuss the Tree data structure. A tree is a very popular non-linear data structure used in a wide range of applications. Tree is a non-linear data structure which organizes data in hierarchical structure and this is a recursive definition. A tree data structure can also be defined a collection of data (Node) which is organized in hierarchical structure recursively. A tree in which every **node can have a maximum of two children** is called **Binary Tree**. A binary tree data structure is represented using two methods:

- ✓ **Array Representation**
- ✓ **Linked List Representation**

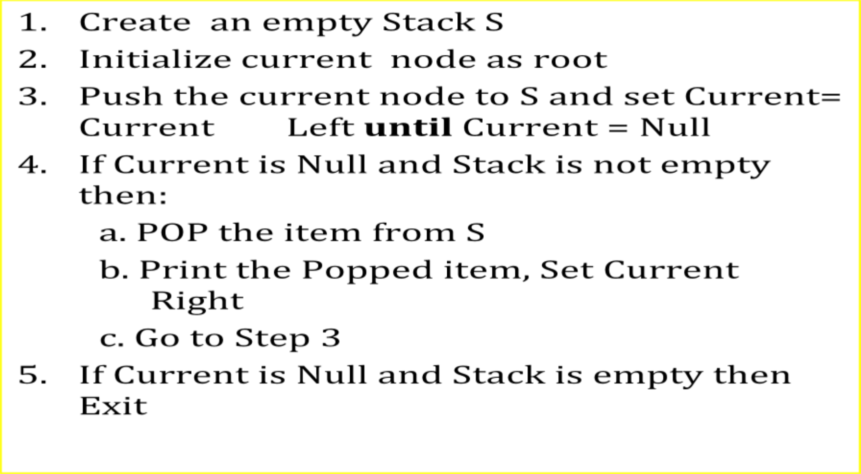
Displaying (or) visiting order of nodes in a binary tree is called as Binary Tree Traversal. There are three types of binary tree traversals.

1. **In - Order Traversal (leftChild - root - rightChild)**
2. **Pre - Order Traversal (root - leftChild - rightChild)**
3. **Post - Order Traversal (leftChild - rightChild - root)**

1. In - Order Traversal (leftChild - root - rightChild)

Following operations are carried out to traverse a binary tree using In-Order way:

a) Traverse the left most sub-tree starting at the left external node
b) Visit the root
c) Traverse the right sub-tree starting at the left external node

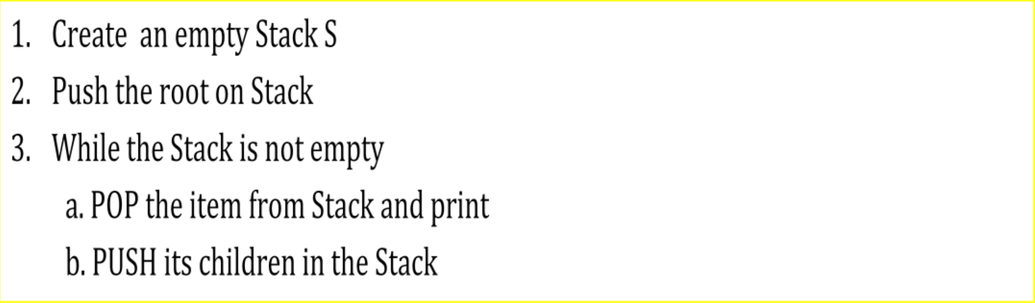


```
1.  Create  an empty Stack S
2.  Initialize current  node as root
3.  Push the current node to S and set Current=
    Current        Left until Current = Null
4.  If Current is Null and Stack is not empty
    then:
      a. POP the item from S
      b. Print the Popped item, Set Current
           Right
      c. Go to Step 3
5.  If Current is Null and Stack is empty then
    Exit
```

2. Pre - Order Traversal (root - leftChild - rightChild)

Following operations are carried out to traverse a binary tree using Pre-Order way:

a) Visit the root
b) Traverse the left most sub-tree starting at the left external node
c) Traverse the right sub-tree starting at the left external node

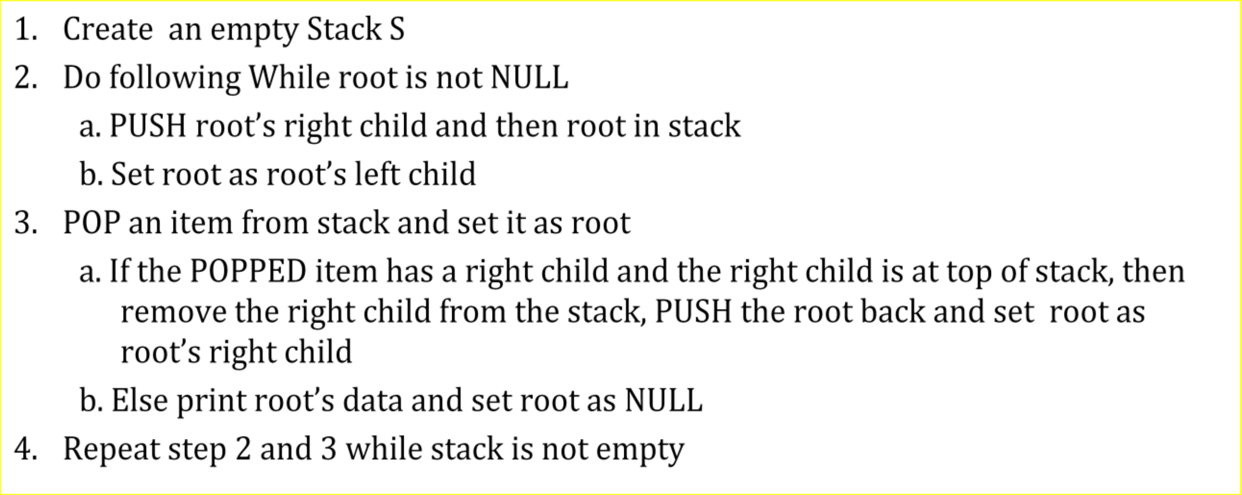


```
1. Create an empty Stack S
2. Push the root on Stack
3. While the Stack is not empty
    a. POP the item from Stack and print
    b. PUSH its children in the Stack
```

3. Post - Order Traversal (leftChild - rightChild - root)

Following operations are carried out to traverse a binary tree using Post-Order way:

a) Traverse the left most sub-tree starting at the left external node
b) Traverse the right sub-tree starting at the left external node
c) Visit the root

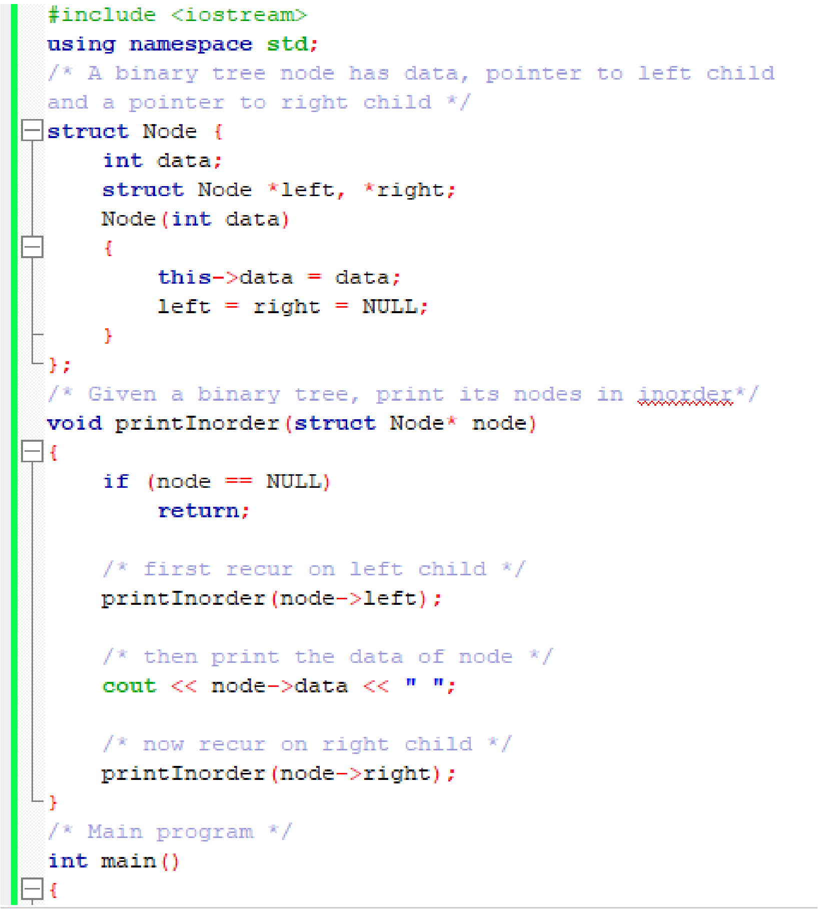


```
1. Create an empty Stack S
2. Do following While root is not NULL
    a. PUSH root's right child and then root in stack
    b. Set root as root's left child
3. POP an item from stack and set it as root
    a. If the POPPED item has a right child and the right child is at top of stack, then
       remove the right child from the stack, PUSH the root back and set root as
       root's right child
    b. Else print root's data and set root as NULL
4. Repeat step 2 and 3 while stack is not empty
```

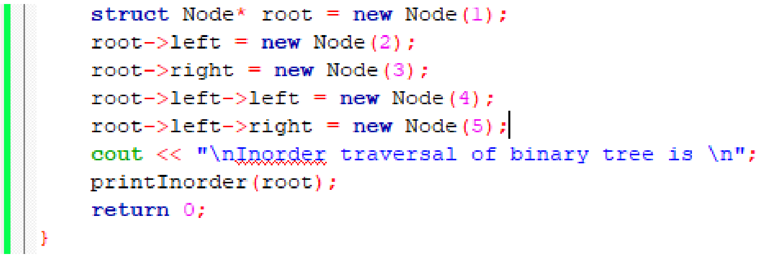
Lab Objectives:

- To be able to write C++ program for performing operations on Tree data structure.

C++ program: Write C++ program to traverse binary tree using In-order traversal method.

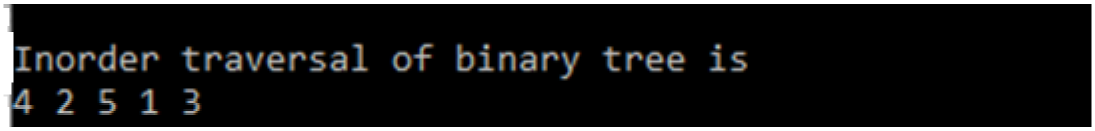


```cpp
#include <iostream>
using namespace std;
/* A binary tree node has data, pointer to left child
and a pointer to right child */
struct Node {
    int data;
    struct Node *left, *right;
    Node(int data)
    {
        this->data = data;
        left = right = NULL;
    }
};
/* Given a binary tree, print its nodes in inorder*/
void printInorder(struct Node* node)
{
    if (node == NULL)
        return;

    /* first recur on left child */
    printInorder(node->left);

    /* then print the data of node */
    cout << node->data << " ";

    /* now recur on right child */
    printInorder(node->right);
}
/* Main program */
int main()
{
```



```cpp
    struct Node* root = new Node(1);
    root->left = new Node(2);
    root->right = new Node(3);
    root->left->left = new Node(4);
    root->left->right = new Node(5);
    cout << "\nInorder traversal of binary tree is \n";
    printInorder(root);
    return 0;
}
```

OUTPUT



```

Inorder traversal of binary tree is
4 2 5 1 3
```

Review Questions/ Exercise:

1. Write a C++ program traverse the tree using pre-order traversal method.

   ____

   ____

   ____

2. Write a C++ program traverse the tree using post-order traversal method.

   ____

   ____

   ____

3. Design a C++ program that uses functions to perform the following:
   a) Create a binary search tree of integers
   b) Search for an integer key in the above binary search tree non recursively.
   c) Search for an integer key in the above binary search tree recursively.

   ____

   ____

   ____

Name: ___________________________

Roll #: ___________________________

Date: ___________________________

Subject Teacher

Remarks: