DSA Lab

Mr. ALEEM AHMAD

A logo of a university

Description automatically generated

Bahria University

**Lab # 9**

**Binary Tree Implementation**

LAB Journal

Asim Ali (01-131232-015)

**Lab 9: Binary Tree Implementation**

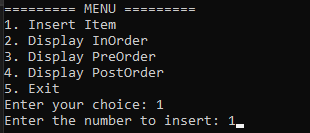
**TASK:**

Binary Tree Implementation.

**Lab Task GitHub Link:**

[Link](https://github.com/iasimkhan2005/DSA.git)

**OUTPUT:**



A screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

**CODE:**

#include<iostream>

using namespace std;

class Tree {

private:

struct Node {

int num;

Node\* left, \* right;

};

Node\* root;

public:

Tree() {

root = NULL;

}

bool isEmpty() {

return (root == NULL);

}

void insertItem(int n) {

insertHelper(root, n);

}

void insertHelper(Node\*& ptr, int n) {

if (ptr == NULL) {

ptr = new Node;

ptr->num = n;

ptr->right = ptr->left = NULL;

}

else if (n > ptr->num) {

insertHelper(ptr->right, n);

}

else if (n < ptr->num) {

insertHelper(ptr->left, n);

}

else {

cout << "Can't add duplicate in tree\n";

}

}

//------------------------------------

void inOrder() {

if (!isEmpty()) {

inOrderHelper(root);

}

else {

cout << "Tree Empty\n";

}

}

void inOrderHelper(Node\* ptr) {

if (ptr == NULL)

{

return;

}

inOrderHelper(ptr->left);

cout << ptr->num << " ";

inOrderHelper(ptr->right);

}

//-----------------------------------

void preOrder() {

if (!isEmpty())

{

preOrderHelper(root);

}

else {

cout << "Tree Empty\n";

}

}

void preOrderHelper(Node\* ptr) {

if (ptr == NULL) {

return;

}

cout << ptr->num << " ";

preOrderHelper(ptr->left);

preOrderHelper(ptr->right);

}

//------------------------------------

void postOrder() {

if (!isEmpty())

{

postOrderHelper(root);

}

else {

cout << "Tree Empty\n";

}

}

void postOrderHelper(Node\* ptr) {

if (ptr == NULL) {

return;

}

postOrderHelper(ptr->left);

postOrderHelper(ptr->right);

cout << ptr->num << " ";

}

};

int main() {

Tree tree;

int choice, value;

while (true) {

system("cls");

cout << "========= MENU =========" << endl;

cout << "1. Insert Item" << endl;;

cout << "2. Display InOrder" << endl;;

cout << "3. Display PreOrder"<< endl;

cout << "4. Display PostOrder"<<endl;

cout << "5. Exit" << endl;

cout << "Enter your choice: ";

if (!(cin >> choice)) {

cout << "Invalid input! Please enter a valid choice.\n";

cin.clear();

cin.ignore();

continue;

}

switch (choice) {

case 1:

cout << "Enter the number to insert: ";

if (!(cin >> value)) {

cout << "Invalid input! Please enter a valid number." << endl;

cin.clear();

cin.ignore();

continue;

}

tree.insertItem(value);

cout << value << " inserted successfully."<< endl;

break;

case 2:

cout << "Display in InOrder: ";

tree.inOrder();

cout << endl;

break;

case 3:

cout << "Display in PreOrder: ";

tree.preOrder();

cout << endl;

break;

case 4:

cout << "Display in PostOrder: ";

tree.postOrder();

cout << endl;

break;

case 5:

cout << "Exiting program...\n";

exit(1);

default:

cout << "Invalid choice. Please select a valid option.\n";

}

system("pause");

}

return 0;

}