//============================================================================

// Name : DSA\_2.cpp

// Author : Aditya Darade

// Version :

// Copyright : Your copyright notice

// Description : Hello World in C++, Ansi-style

//============================================================================

#include <iostream>

#include<bits/stdc++.h>

using namespace std;

class node{

friend class BT;

friend int getH(node\*);

int data;

node\* left , \*right;

public:

node(int d){

data = d;

left = right = NULL;

}

};

int getH(node\* root){

if(root == NULL) return 0;

int l = getH(root->left);

int r = getH(root->right);

return (max(l,r) + 1);

}

class BT{

node\* root;

int leaves , internal;

public:

BT(){

root = NULL;

leaves = internal = 0;

}

node\* Root(){

return this->root;

}

void maketree(){

int d;

cout << "Enter data for root :- ";

cin >> d;

if(d == -1){

return;

}

this->root = new node(d);

queue<node\*> q;

q.push(root);

while(!(q.empty())){

bool c1 = true , c2 = true;

node\* temp = q.front();

q.pop();

int left, right;

cout << "Enter data in left of " << temp->data << " :- ";

cin >> left;

if(left != -1){

c1 = false;

temp->left = new node(left);

q.push(temp->left);

}

cout << "Enter data in right of " << temp->data << ":- ";

cin >> right;

if(right != -1){

c2 = false;

temp->right = new node(right);

q.push(temp->right);

}

if( c1 && c2 ) leaves++;

else{

internal++;

}

}

}

void getnodesnum(){

cout<<"Internal Nodes in this tree are :- "<<internal<<endl;

cout<<"Leaf Nodes in this tree are :- "<<leaves<<endl;

}

void height(){

cout<<"Height of tree is :- " << getH(this->root) << endl;

}

void invert(){

if(root == NULL){

cout<<"Empty Tree\n";

return;

}

queue<node\*> q;

q.push(this->root);

while(!q.empty()){

node\* temp = q.front();

q.pop();

if(temp->left != NULL){

q.push(temp->left);

}

if(temp->right != NULL){

q.push(temp->right);

}

node\* t = temp->left;

temp->left = temp->right;

temp->right = t;

}

cout<<"Inverted\n";

}

void preorder(){

if(root == NULL){

cout<<"Empty Tree\n";

return;

}

stack<node\*> st;

cout<<"Preorder is : - ";

st.push(root);

while(!st.empty()){

node\* temp = st.top();

st.pop();

cout<<temp->data<<" ";

if(temp->right != NULL){

st.push(temp->right);

}

if(temp->left != NULL){

st.push(temp->left);

}

}

cout<<endl;

}

void inorder(){

if(root == NULL){

cout<<"Empty Tree\n";

return;

}

stack<node\*> st;

//st.push(root);

node\* temp = root;

cout<<"Inorder is :- ";

while(!st.empty() || temp != NULL){

while(temp != NULL){

st.push(temp);

temp = temp->left;

}

temp = st.top();

st.pop();

cout<<temp->data<<" ";

temp = temp->right;

}cout<<endl;

}

void postorder(){

if(root == NULL){

cout<<"Empty Tree\n";

return;

}

node\* temp = NULL;

stack<node\*> st;

stack<int> ans;

st.push(root);

while(!st.empty()){

temp = st.top();

st.pop();

if(temp->left) st.push(temp->left);

if(temp->right) st.push(temp->right);

ans.push(temp->data);

}

cout<<"Post Order is :- ";

while(!ans.empty()){

cout<<ans.top()<<" ";

ans.pop();

}

}

void operator=(BT &t1){

if(this->root == NULL){

cout<<"'=' operator overloaded for copying tree\n";

cout<<"Empty Tree\n";

return;

}

if(t1.root == NULL) return;

node\* temp = new node(t1.root->data);

this->root = temp;

node\* copy = this->root;

queue<node\*> q , q2;

q.push(t1.root);

q2.push(this->root);

while(!q.empty()){

temp = q.front();

copy = q2.front();

q2.pop();

q.pop();

if(temp->left != NULL){

q.push(temp->left);

copy->left = new node(temp->left->data);

q2.push(copy->left);

}

if(temp->right != NULL){

q.push(temp->right);

copy->right = new node(temp->right->data);

q2.push(copy->right);

}

cout<<"'=' operator overloaded for copying tree\nCopying Complete\n";

}

}

void deletebt(node\* node){

if(node == NULL) return ;

deletebt(node->left);

deletebt(node->right);

cout<<"Deleting node :- "<<node->data<<endl;

delete node;

}

};

int main() {

BT t1 , t2;

cout<<"-----------------Contructing Tree by levels ---------------- \n";

t1.maketree();

cout<<"Contruction Complete\n\n-----------------MENU-------------------\n";

cout<<"1)Inorder\n2)Postorder\n3)preorder\n4)Get Height of Tree\n5)Number of leaf & internal Nodes\n6)Invert the Tree\n7)Copy Tree to another Tree\n8)Delete entire tree\n\n";

int n;

while(1){

cout<<"Enter your option :- ";

cin>>n;

cout<<"===================================================================\n";

if(n==1){

t1.inorder();

}else if(n==2){

t1.postorder();

}else if(n==3){

t1.preorder();

}else if(n==4){

t1.height();

}else if(n==5){

t1.getnodesnum();

}else if(n==6){

t1.invert();

}else if(n==7){

t2 = t1;

}else if(n==8){

t1.deletebt(t1.Root());

}else{

break;

}

cout<<"\n===================================================================\n";

}

return 0;

}