

Implement a function to calculate the height of a binary tree
Implement a function to calculate the count of leaf nodes in a binary tree

CODE :

```
#include<iostream>
using namespace std;
class node
{
public:
    int data;
    node* left, *right;
    node(int k)
    {
        data = k;
        left = right = NULL;
    }
};

class Trees
{
public:
    node *create_node(int value)
    {
        node *newnode = new node(value);
        newnode->left = NULL;
        newnode->right = NULL;
        return newnode;
    }
    node* insert(node* root, int value)
    {
        if (root == NULL)
        {
            return create_node(value);
        }
        else
        {
            if (value < root->data)
            {
                root->left = insert(root->left, value);
            }
            else
            {
                root->right = insert(root->right, value);
            }
            return root;
        }
    }
    void inorder_traversal(node*root_temp)
    {
        if (root_temp != NULL)
        {
            inorder_traversal(root_temp->left);
            cout << " " << root_temp->data;
            inorder_traversal(root_temp->right);
        }
    }
};
```

```

    }
}

int heightOf_tree(node*root_temp)
{
    int height = 0;
    if (root_temp == NULL)
    {
        return 0;
    }
    else if (root_temp != NULL)
    {
        int left_subtree = heightOf_tree(root_temp->left);
        int right_subtree = heightOf_tree(root_temp->right);
        if (left_subtree > right_subtree)
        {
            height = left_subtree+1;
        }
        else
        {
            height = right_subtree+1;
        }
    }
    return height;
}

int number_of_leaf_nodes(node* root)
{
    if (root == NULL)
    {
        cout << "Root is null " << endl;
        return 0;
    }
    else if (root->left == NULL && root->right == NULL)
    {
        return 1;
    }
    else
    {
        int L_leaf_node = number_of_leaf_nodes(root->left);
        int R_Leaf_node = number_of_leaf_nodes(root->right);
        return L_leaf_node + R_Leaf_node;
    }
}

};
int main()
{

```

```

cout << "Muhammad Zeeshan\nf2022266312\nBSCS\nV-1\nDSA LAB" << endl

```

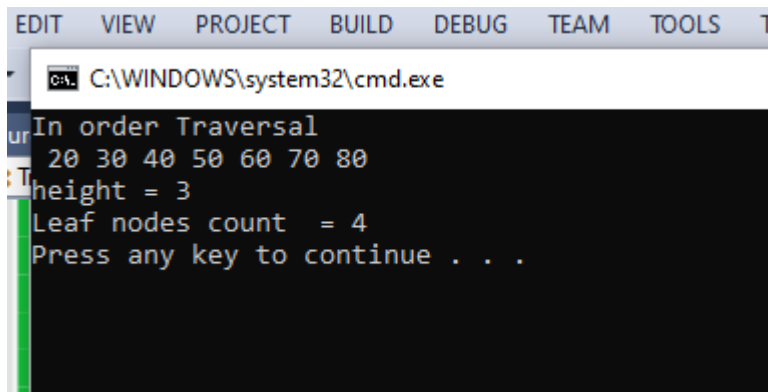
```

Trees obj1;
node* root = NULL;
root = obj1.insert(root, 50);
root = obj1.insert(root, 30);
root = obj1.insert(root, 20);
root = obj1.insert(root, 40);
root = obj1.insert(root, 70);
root = obj1.insert(root, 60);
root = obj1.insert(root, 80);

```

```
    cout << "In order Traversal";  
    cout << endl;  
    obj1.inorder_traversal(root);  
    cout << endl;  
  
    cout << "height = " << obj1.heightOf_tree(root) << endl;  
    cout << "Leaf nodes count = " << obj1.number_of_leaf_nodes(root) << endl;  
    return 0;  
}
```

Output :



```
EDIT  VIEW  PROJECT  BUILD  DEBUG  TEAM  TOOLS  T  
C:\WINDOWS\system32\cmd.exe  
In order Traversal  
20 30 40 50 60 70 80  
height = 3  
Leaf nodes count = 4  
Press any key to continue . . .
```