Aim – lca of two nodes

#include <iostream>

using namespace std;

struct TreeNode {

int key;

TreeNode\* left;

TreeNode\* right;

TreeNode(int k) : key(k), left(nullptr), right(nullptr) {}

};

class BinaryTree {

public:

TreeNode\* insert(TreeNode\* root, int key) {

if (!root)

return new TreeNode(key);

if (key < root->key)

root->left = insert(root->left, key);

else

root->right = insert(root->right, key);

return root;

}

TreeNode\* findLCA(TreeNode\* root, int n1, int n2) {

if (!root)

return nullptr;

if (root->key > n1 && root->key > n2)

return findLCA(root->left, n1, n2);

if (root->key < n1 && root->key < n2)

return findLCA(root->right, n1, n2);

return root;

}

};

int main() {

BinaryTree tree;

TreeNode\* root = nullptr;

root = tree.insert(root, 20);

root = tree.insert(root, 10);

root = tree.insert(root, 30);

root = tree.insert(root, 5);

root = tree.insert(root, 15);

root = tree.insert(root, 25);

root = tree.insert(root, 35);

int n1 = 5, n2 = 15;

TreeNode\* lca = tree.findLCA(root, n1, n2);

if (lca)

cout << "Lowest Common Ancestor of " << n1 << " and " << n2 << " is: " << lca->key << endl;

else

cout << "No Common Ancestor Found!" << endl;

return 0;

}