#include "stdafx.h"

#include<iostream>

#include <list>

#include <queue>

using namespace std;

class Node {

public:

int data;

Node \*left;

Node \*right;

Node() { data = -1; left = NULL; right = NULL; };

void addNode(int key, Node \*leaf);

void levelOrder(Node \*temp);

void Node::asdads(Node \*temp);

void setKey(int key) { data = key; };

void setLeft(Node\* Left) { left = Left; };

void setRight(Node\* Right) { right = Right; };

int getKey() { return data; }

Node \*getLeft() { return left; }

Node \*getRight() { return right; }

};

void Node::addNode(int key, Node \*leaf)

{

if (leaf->getKey() == -1)

{

leaf->setKey(key);

}

else

{

if (key <= leaf->getKey()) {

if (leaf->getLeft() != NULL)

addNode(key, leaf->getLeft());

else {

Node \*n = new Node();

n->setKey(key);

leaf->setLeft(n);

}

}

else {

if (leaf->getRight() != NULL)

addNode(key, leaf->getRight());

else {

Node \*n = new Node();

n->setKey(key);

leaf->setRight(n);

}

}

}

}

void Node::levelOrder(Node \*temp)

{

if (temp == NULL)

return;

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

int k = 0;

if (k%2==0) {

levelOrder(temp->left);

k = k + 1;

}

if (k%2==1) {

levelOrder(temp->right);

k = k + 1;

}

}

void Node::asdads(Node \*temp)

{

queue<Node\*> q;

if (temp) {

q.push(temp);

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

}

else {

cout << "NULL ";

}

while (!q.empty()) {

const Node \* const temp\_node = q.front();

q.pop();

if (temp\_node->left) {

q.push(temp\_node->left);

cout << temp\_node->left->data << " ";

}

if (temp\_node->right) {

q.push(temp\_node->right);

cout << temp\_node->right->data << " ";

}

} }

int main()

{

int a;

Node \*tree = new Node();

tree->addNode(5, tree);

tree->addNode(4, tree);

tree->addNode(36, tree);

tree->addNode(3, tree);

tree->addNode(34, tree);

tree->addNode(8, tree);

tree->addNode(1, tree);

cout << "The binary tree we have is " << endl;

tree->asdads(tree);

cin >> a;

return 0;

}