Aim-converting into avl

#include <iostream>

using namespace std;

struct TreeNode {

int key;

TreeNode\* left;

TreeNode\* right;

int height;

TreeNode(int k) : key(k), left(nullptr), right(nullptr), height(1) {}

};

class AVLTree {

public:

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

if (!root)

return new TreeNode(key);

if (key < root->key)

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

else if (key > root->key)

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

else

return root;

root->height = 1 + max(getHeight(root->left), getHeight(root->right));

int balance = getBalance(root);

if (balance > 1 && key < root->left->key)

return rightRotate(root);

if (balance < -1 && key > root->right->key)

return leftRotate(root);

if (balance > 1 && key > root->left->key) {

root->left = leftRotate(root->left);

return rightRotate(root);

}

if (balance < -1 && key < root->right->key) {

root->right = rightRotate(root->right);

return leftRotate(root);

}

return root;

}

void inOrder(TreeNode\* root) {

if (root) {

inOrder(root->left);

cout << root->key << " ";

inOrder(root->right);

}

}

private:

int getHeight(TreeNode\* node) {

return node ? node->height : 0;

}

int getBalance(TreeNode\* node) {

return node ? getHeight(node->left) - getHeight(node->right) : 0;

}

TreeNode\* leftRotate(TreeNode\* x) {

TreeNode\* y = x->right;

TreeNode\* T2 = y->left;

y->left = x;

x->right = T2;

x->height = 1 + max(getHeight(x->left), getHeight(x->right));

y->height = 1 + max(getHeight(y->left), getHeight(y->right));

return y;

}

TreeNode\* rightRotate(TreeNode\* y) {

TreeNode\* x = y->left;

TreeNode\* T3 = x->right;

x->right = y;

y->left = T3;

y->height = 1 + max(getHeight(y->left), getHeight(y->right));

x->height = 1 + max(getHeight(x->left), getHeight(x->right));

return x;

}

};

int main() {

AVLTree tree;

TreeNode\* root = nullptr;

root = tree.insert(root, 10);

root = tree.insert(root, 20);

root = tree.insert(root, 30);

root = tree.insert(root, 40);

root = tree.insert(root, 50);

root = tree.insert(root, 25);

cout << "In-order traversal of the AVL tree: ";

tree.inOrder(root);

return 0;

}