#include <bits/stdc++.h>

using namespace std;

class Node {

public:

int data;

Node\* left;

Node\* right;

Node(int d)

{

int data = d;

left = right = NULL;

}

};

// Function to calculate the height of a tree

int height(Node\* node)

{

// base case tree is empty

if (node == NULL)

return 0;

// If tree is not empty then

// height = 1 + max of left height

// and right heights

return 1 + max(height(node->left), height(node->right));

}

// Returns true if binary tree

// with root as root is height-balanced

bool isBalanced(Node\* root)

{

// for height of left subtree

int lh;

// for height of right subtree

int rh;

// If tree is empty then return true

if (root == NULL)

return 1;

// Get the height of left and right sub trees

lh = height(root->left);

rh = height(root->right);

if (abs(lh - rh) <= 1 && isBalanced(root->left)

&& isBalanced(root->right))

return 1;

// If we reach here then tree is not height-balanced

return 0;

}

// Driver code

int main()

{

Node\* root = new Node(1);

root->left = new Node(2);

root->right = new Node(3);

root->left->left = new Node(4);

root->left->right = new Node(5);

root->left->left->left = new Node(8);

if (isBalanced(root))

cout << "Tree is balanced";

else

cout << "Tree is not balanced";

return 0;

}