```java
package binarytree;

/* Java program to determine if binary tree is
height balanced or not */



class BinaryTree19 {
    Node root;

    /* Returns true if binary tree with root as root is height-balanced */
    boolean isBalanced(Node node)
    {
        int lh; /* for height of left subtree */

        int rh; /* for height of right subtree */

        /* If tree is empty then return true */
        if (node == null)
            return true;

        /* Get the height of left and right sub trees */
        lh = height(node.left);
        rh = height(node.right);

        if (Math.abs(lh - rh) <= 1
                && isBalanced(node.left)
                && isBalanced(node.right))
            return true;

        /* If we reach here then tree is not height-balanced */
        return false;
    }

    /* UTILITY FUNCTIONS TO TEST isBalanced() FUNCTION */
    /* The function Compute the "height" of a tree. Height is the
        number of nodes along the longest path from the root node
        down to the farthest leaf node.*/
    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 + Math.max(height(node.left), height(node.right));
    }

    public static void main(String args[])
    {
        BinaryTree19 tree = new BinaryTree19();
        tree.root = new Node(1);
        tree.root.left = new Node(2);
        tree.root.right = new Node(3);
        tree.root.left.left = new Node(4);
        tree.root.left.right = new Node(5);
        tree.root.left.left.left = new Node(8);

        if (tree.isBalanced(tree.root))
            System.out.println("Tree is balanced");
        else
            System.out.println("Tree is not balanced");
    }
}
```

```
// This code has been contributed by Mayank Jaiswal(mayank_24)
```