Task 1: Balanced Binary Tree Check. Write a function to check if a given binary tree is balanced. A balanced tree is one where the height of two subtrees of any node never differs by more than one.

1. **Implementation:**
   - **Node Class:** Represents a node in the binary tree.
   - **BinaryTree Class:** Contains methods to insert nodes and to check if the tree is balanced.

CODE:

```
Package Com.day13
class Node {
    int data;
    Node left, right;
    Node(int value) {
        data = value;
        left = right = null;
    }
}
class BinaryTree {
    Node root;
    private int checkHeight(Node node) {
        if (node == null) {
            return 0;
        }
        int leftHeight = checkHeight(node.left);
        if (leftHeight == -1) {
            return -1;
        }
        int rightHeight = checkHeight(node.right);
        if (rightHeight == -1) {
            return -1;      }
        if (Math.abs(leftHeight - rightHeight) > 1) {
```

```
            return -1;
        }
        return Math.max(leftHeight, rightHeight) + 1;
    }
    public boolean isBalanced() {
        return checkHeight(root) != -1;
    }
    public void insert(int data) {
        root = insertRec(root, data);
    }
    private Node insertRec(Node root, int data) {
        if (root == null) {
            root = new Node(data);
            return root;        }
        if (data < root.data) {
            root.left = insertRec(root.left, data);
        } else if (data > root.data) {
            root.right = insertRec(root.right, data);
        }
        return root;    }
    public static void main(String[] args) {
        BinaryTree tree = new BinaryTree();
        // Insert nodes into the tree
        tree.insert(10);
        tree.insert(5);
        tree.insert(15);
        tree.insert(3);
        tree.insert(7);
        tree.insert(12);
```

```
        tree.insert(18);
        if (tree.isBalanced()) {
            System.out.println("The tree is balanced.");
        } else {
            System.out.println("The tree is not balanced.");
        }
        tree.insert(1);
        if (tree.isBalanced()) {
            System.out.println("The tree is balanced.");
        } else {
            System.out.println("The tree is not balanced.");
        }   }   }
```

Explanation:

1. **Node Class:** Represents each node in the binary tree with an integer value and pointers to left and right children.
2. **BinaryTree Class:** Contains the root node, methods to insert nodes, and the isBalanced method to check if the tree is balanced.
   - **checkHeight(Node node):** Recursively calculates the height of the tree and checks if the tree is balanced at each node.
   - **isBalanced():** Calls checkHeight starting from the root. If checkHeight returns -1, the tree is not balanced.
   - **insert(int data):** Helper method to insert nodes into the tree for testing purposes.
3. **BalancedBinaryTreeCheck Class:** Contains the main method to create a binary tree, insert nodes, and check if the tree is balanced.