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.

Ans:

```
package Day13;
class TreeNode {
   int val;
   TreeNode left;
   TreeNode right;
   TreeNode(int x) {
      val = x;
   }
}
public class Task1 {
   public boolean isBalanced(TreeNode root) {
      return checkHeight(root) != -1;
   }
   private int checkHeight(TreeNode node) {
      if (node == null) {
         return 0;
      }
      int leftHeight = checkHeight(node.left);
      if (leftHeight == -1) {
         return -1; // Not balanced
      }
      int rightHeight = checkHeight(node.right);
      if (rightHeight == -1) {
         return -1; // Not balanced
      }
      if (Math.abs(leftHeight - rightHeight) > 1) {
         return -1; // Not balanced
      }
      return Math.max(leftHeight, rightHeight) + 1;
   }
   public static void main(String[] args) {

      TreeNode root = new TreeNode(1);
```

```
        root.left = new TreeNode(2);
        root.right = new TreeNode(3);
        root.left.left = new TreeNode(4);
        root.left.right = new TreeNode(5);
        Task1 treeChecker = new Task1();
        System.out.println(treeChecker.isBalanced(root));  // Output: true
        TreeNode unbalancedRoot = new TreeNode(1);
        unbalancedRoot.left = new TreeNode(2);
        unbalancedRoot.left.left = new TreeNode(3);
        System.out.println(treeChecker.isBalanced(unbalancedRoot));
    }
}
```

OUTPUT:

```
true
false
```