Data Structures

Quiz - 3

Question:

A binary tree is *balanced* if, at each node, the difference between the height of the left and right children is less than 1.

(a) (2 points) Draw balanced binary trees of heights 2, 3, and 4.

(b) (1 point) Draw an imbalanced binary tree of any height.

(c) (6 points) Write an efficient recursive method

```
public boolean isBalanced()
```

for `LinkedBinaryTree` that checks if the tree is balanced. **Hint:** use height as an auxiliary variable.

(d) (1 point) What is the worst-case asymptotic computational complexity of `isBalanced()` if the tree has n positions?

Solution:

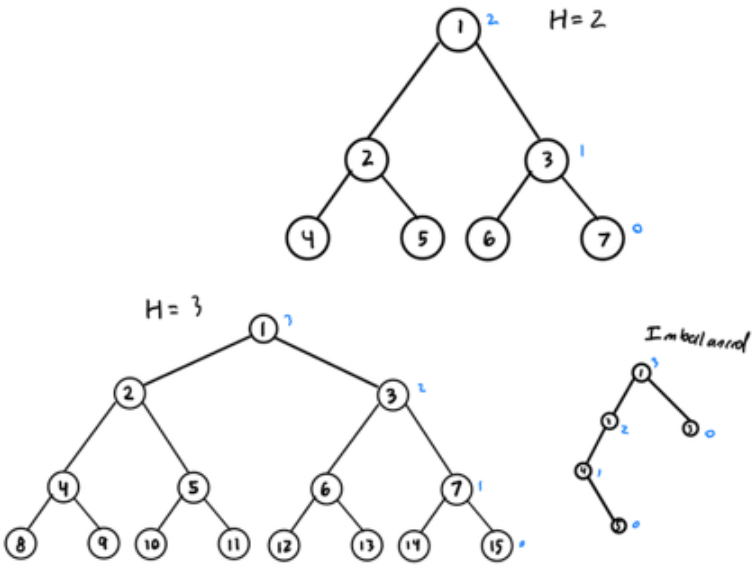

Figure 1: Part a and b

```java
public boolean isBalanced() {
    return helper((LinkedBinaryTree<E>.Node) root()) != -2;
}

private int helper(Node node) { // O(N)
    if (node == null) return -1;

    int left = helper((LinkedBinaryTree<E>.Node) node.getLeft());
    int right = helper((LinkedBinaryTree<E>.Node) node.getRight());

    if (left == -2) return -2;
    if (right == -2) return -2;
    if (Math.abs(left - right) > 1) return -2;

    return Math.max(left, right) + 1;
}
```