COMP 250 Assignment #4 – Written Responses

Alex Hale
ID: 260672475

2)

a)

Step 1: place node $floor(\frac{n}{2})$ at root.

Step 2: divide remaining nodes into two groups: the group of nodes that were above the root will go under the right child, and the group of nodes that were below the root will go under the left child

Step 3: Repeat the process with the sub-groups.

- Height = $ceiling(\log_2(n+1))$

b) Find, Insert, and Remove all have a worst-case running time of $O(\log(n))$

c)

Tree with maximal height is an unbalanced tree.

Option 1: node n inserted at root, node n inserted as its left child, … , node 1 inserted. No node has a right child.

Option 2: node 1 inserted at root, node 2 inserted as its right child, … , node n inserted. No node has a left child.

- Height = n

d) Find, Insert, and Remove all have a worst-case running time of $O(n)$

3)

a) 7 9 9 7 8 6 2 1 2 3

b) 9 9 7 8 6 2 1 2 3 7

c) 7 3 9 2 6 8 9 1 2 7

d) 7 3 2 1 2 6 9 8 7 9 - this is the same result as a pre-order traversal

4)

Algorithm boolean isomorphic(treeNode r, treeNode s)

Input: two treeNodes r and s to be checked

Output: Returns true if r and s are isomorphic, false otherwise

```
if ((r == null && s != null) || (r != null && s == null)) return false;
if (r == null && s == null) return true;
if (r.getValue() != s.getValue()) return false;
x <- isomorphic(r.getLeftChild(), s.getLeftChild()) && isomorphic(r.getRightChild(), r.getRightChild());
y <- isomorphic(r.getLeftChild(), s.getRightChild()) && isomorphic(r.getRightChild(), r.getLeftChild());
return (x || y);
```