CSCI3200 Spring 2018 LAB3

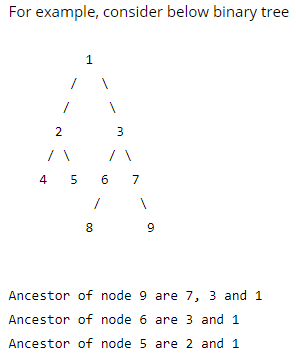
Objective: the objective of this lab is to help the students have a better understanding of the reference based implementation of a Binary Search tree. The LAB will go back to a common node structure with each node only has two references refers to the left child and right child (no direct reference to its parent). This will help the students better adjust their learning to popular interview questions they may have to deal with in their future interviews. The LAB provides a simpler implementation for the linked binary tree discussed in class but slightly different (the size variable is removed from the binary tree class, the parent reference variable for each node has also been removed).

1. Create a project and put/copy all java files in the *LAB3* folder and paste into the *src* folder of your created project. Remark: the size variable and parent reference have already been removed. You are recommended to read the provided code first, to figure out what is already available.
2. Please based on this change, make necessary changes to the following methods
3. public int size() : return the number of nodes
4. public BinaryNode<E> getParent(BinaryNode<E> p) throws IllegalArgumentException : return the parent node for a node referred by p
5. private int numChildren(BinaryNode<E> p) : return the number of children for a node referred by p
6. public Boolean search(E e)：check if element e is on the tree or not
7. public void insert(E e): insert element e into the binary search tree, do nothing if found duplicate element
8. public void remove(E e): remove the node contains element e from the tree
9. public BinaryNode<E> findMax(BinaryNode<E> p){//find and return the node contains the maximum element on a binary search tree rooted at p
10. Add and implement the following methods within the class:
11. public boolean isIdentical(BinaryNode <E> p)***:*** check if two binary search trees rooted at this.root and p are identical or not. i.e. if they have identical structure & their contents are also same. Return true for yes and false, otherwise.

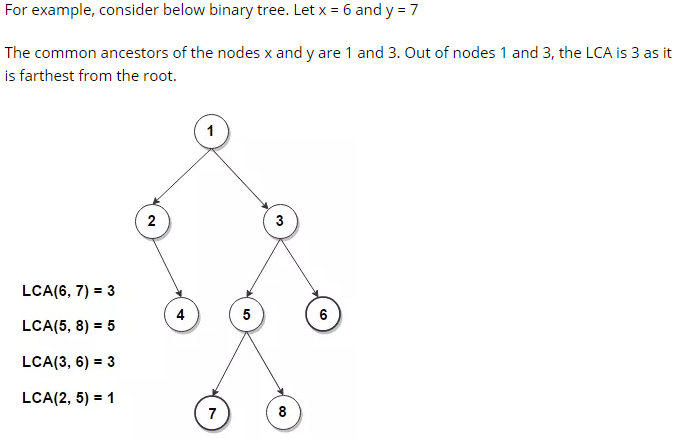
http://www.techiedelight.com/check-if-two-binary-trees-are-identical-not-iterative-recursive/

1. ***public void printAncestors(***BinaryNode ***<E> p):*** print all ancestors of a given node referred by p on the tree.

http://www.techiedelight.com/find-ancestors-of-given-node-binary-tree/



1. public ArrayList<E> inorderTraversal(BinaryNode<E> p){//find and return the inorder travesal for elements stored on the tree rooted at p
2. ***(Bonus) public Node<E> findLCA(Node<E> p1, Node<E> p2):*** find lowest common ancestor (LCA) of p1 and p2 in it.



<http://www.techiedelight.com/find-lowest-common-ancestor-lca-two-nodes-binary-tree/>

Remark:

* If you want to add extra methods to support the implementation of the required methods, please put them also at the end of the file for clarity.
* The method definition can be different (such as return type or parameter list) as long as the objectives are the same.