Lab 5: Binary Tree Traversals and Iterators

- Download the following file
  - Lab5.zip

Binary tree traversals

During lectures, we have discussed binary tree traversals. Your first task is to implement recursive methods to do preorder, inorder and postorder traversals. This should be a simple task, based on pseudocodes seen in lecture, and take only a few minutes to code and test. More specifically complete the following methods inside LikedBinarySearchTree class:

- void preorderRecursive(Node)
- void inorderRecursive(Node)
- void postorderRecursive(Node)

Binary tree iterators

In a previous course, you probably have studied the concept of an iterator for a class; you might have studied in more detail list iterators.
In this lab you will apply the concept of an iterator for a binary tree. While a list iterator is a way to move through elements of a list from beginning to end, for a tree there are many ways to move from node to node.
For this part you will implement two types of iterators: a preorder and an order iterator.

You may need to review information about the Interface Iterator<E> provided in package java.util. Your TA will briefly review iterators during the lab. Your iterators will implement this interface, so you need to provide the the constructor plus methods hasNext() and next() for each iterator implemented.

The hardest method to implement for **class InorderIterator implements Iterator** is next() as it entails to figure out from the current node of your iterator, who is the next node to be visited in an inorder traversal of the tree. The same is true for **class PreorderIterator implements Iterator**.

The pseudocode of the algorithms to do these tasks have been discussed in your DGD and will also be reviewed by your TA in the lab.

Task details:

1. Review class LinkedBinaryTree and class TestTree.
2. Implement the traversal algorithms; test them using TestTree class.
3. Implement preorderNext and inorderNext iterators; based on pseudocodes. These are nontrivial algorithms.
4. Test them with TestTree class.