COMP 282 – Homework 02

The purpose of this project is to introduce you to building and utilizing trees. It will consist of multiple parts, each corresponding to a lecture.

There are to be absolutely no packages used in your submissions. The use of the default package is, generally, not advised for enterprise-level work; this, however, is not that.

The following files **should** be present in your submission:

- Tree.java
- BinaryTree.java

You may include any interfaces described here, but they will not be examined. A successful project need only include the list of files mentioned above – with appropriate implementations, of course.

PART 1 - THE TREE CLASS

In order to build more complex tree-based structures, you need to start with the basics. Namely, you will need to build a Tree class to store some arbitrary set of elements. We will extend this class throughout the rest of the project.

The basic interface definition is as follows:

```
public interface ITree<T> {
  public T getItem();
  public ITree<T> find(T item);
  public ITree<T> insert(T item);
}
```

This should be included in your project as ITree.java. You must provide an implementation for this interface in form of a Tree class – to be defined in Tree.java:

```
public class Tree<T> implements ITree<T> {
  // ...
  public Tree(T item) {
    // ...
  }
  // ...
}
```

This is the only file required from this part of the project.

PART 2 - BINARy SEARch TREES

Now it's time to take the general implementation of a tree, and extend from it a binary search tree. To do this, we will need to be able to only accept items that are ordinal:

```
public class BinaryTree<T extends Comparable<T>> extends Tree<Comparable<T>> {
  // ...
}
```

You will want to override the find and insert methods of your Tree class in order to ensure you are using this new tree as efficiently as possible. The BinaryTree.java file will be the only required file from this part of the project.

PART 3 - TRAVERSAL

In this part, we will supply four methods for traversing our tree structures. The goal is to implement the following interface in your BinaryTree class:

```
import java.util.*;

public interface ITraversable<T> {
  public ArrayList<T> nlr(); // Pre-order
  public ArrayList<T> lnr(); // In-order
  public ArrayList<T> lrn(); // Post-order
  public ArrayList<T> bfs(); // Breadth-first
}
```

Each method should return an ArrayList of node values, based on the appropriate traversal.

PART 4 - MEASUREMENT

In order to implement some more sophisticated trees, we will need an easy way of determining their heights. In order for to do this, we will implement another interface:

```
public interface IMeasurable {
  public int size();
  public int height();
}
```

These should be fairly self descriptive: the size method will return the total number of elements in the tree, while the height method returns its height. **Remember: the height of a tree is the longest path from the root to any of its leaves.**

Note: It is OK to implement this homework using other programing languages like C++ and Python. In addition, you can implement the above tree/binary tree in your own way. For example, you can implement only binary tree without implementing the tree. Or, you can define binary tree with integer keys (values).

INSTRUCTIONS for SUBMISSION

You do not need to submit anything as this moment as this is only a part of the homework 02. More information will be provided later.