COSC 311 WINTER2016 HOMEWORK 1/7/2016

Due: 1/12/2016.

(1)
Write Java code to implement a sorted (ascending) linked list of nodes. Each node in the linked list will have:

a data field that is an `int`

a reference `next`*, that points to the next node in the list.*

Implement the following methods that act on a list:

```
void insert (Node n);     // inserts node n to the list.
                          // Duplicates are allowed.

Node find (int x);        // return reference to first
                          // encountered node with
                          // data value == x;
                          //return null if x is not present

void delete (Node n);     // remove node n from list.

void delete (int x);      // remove node with data == x

String toString();        // print list in sorted order.
```

Implement additional methods to list and to node as are desirable.

(2) You must use good object-oriented design. For this assignment, that means three classes:

`Node`
`LinkedList`
`Driver`

Do not use internal classes.

(3) Do not use any of the Java Collections framework (no `ArrayList`, etc)

(4) You may use generics if you wish (this will be required in the next homework).

(5) You may do error handling with exceptions, if you wish (this will be required in the next homework). Otherwise, assume the input is perfect.

(6) You must provide UML diagram (use `http://yed.yworks.com/support/manual/uml.html`) or other drawing tool.

(7) You must use javadoc to document (see https://en.wikipedia.org/wiki/Javadoc for a brief overview of javadoc comments.

Each class must have `@author, @version 1.0,` and a short comment describing the class.

Each method must be described with a one to two line comment
Each method parameter must be described with `@param`
Each method must have `@return`

(8) You must give the documentation pages produced using the Javadoc comments.

(9) Demonstrate your program working by giving a screen shot showing its functioning on the following (pseudo-codish) input:

insert(3)

insert(5)

insert(1)

toString()

Node n = find(3)

delete(n)

toString()

delete(1)

toString()

(10) Turn in:
Hard copy of code.
UML diagram
Screen shot of running program

(11) Grading based on:
Satisfying specs
Readability
Design
Elegance