Lab 2: Sorting and Linked Lists

Week of October 1, 2018

Objective

To insertion sort a linked list.

Exercises

The file `Lab2.java` contains a nearly-complete program to insertion sort a linked list, except it needs TWO methods added to the `LinkedList` class (details below).

Note that the file contains THREE classes. It contains an ordinary `LinkedList` class, an ordinary linked-list `Node` class, and the `Lab2` class containing the `main()` method and the code that tests the sorting method.

Add the two required methods (and any helper methods needed for the two methods) to `Lab2.java` without changing any of the existing code.

The two methods you will write:

1. A method with header `public boolean isSorted()`:
   This method returns `true` if the linked list it is called on (i.e., implicit parameter "`this`") is in ascending order; otherwise, it returns false.

   Challenge: Recursively check if the linked list is sorted.

2. A method with header `public void insertionSort()`.
   This method uses insertion sort to put the list it is called on (i.e., implicit parameter "`this`") into ascending order.

   You probably shouldn't directly translate every line of insertion sort working on arrays into code working on a linked list. But you should capture the essence of insertion sort:

   - One at a time, you should place unprocessed nodes into their correct position in the sorted part of the list.
   - The sorted part should start out consisting of just one node.

   Challenge: Recursively insertion sort the linked list.

Note: You can change the constants defined at the beginning of the `Lab2` class, which control how many tests the code does and what size of lists the tests use.