Algorithmics I

Laboratory Exercise - Extending and using a Trie class

The purpose of the exercise is to extend the `Trie` class given in the lectures to incorporate a method that generates a list of all of the words represented in the trie. This extended class will then be used in a program that accepts as input a text file, and produces a list of all of the words occurring in it.

Tries will figure in the assessed exercise for this course, so some practice in their use will be valuable.

Requirements: the setup files are available under Moodle (these include files `Trie.java` and `Node.java`, as presented in lectures).

Task 1: add a new method extract, specified as follows, to the Trie class:

```
/** method traverses the trie and both extracts and returns a list
* of all of the distinct words represented in the trie
*/
public LinkedList<String> extract();
```

Hint: one possible solution uses a recursive helper method that takes as its parameters (i) a reference to a trie node, (ii) the string represented by the parent of that node, and (iii) a reference to the LinkedList being constructed.

Task 2: the class `Main.java` obtained from the set-up files contains a main program designed to read an input text file, named as the first program parameter, and to output a list of all of the distinct words contained in that file together with a word count. (The string Scanner object used in this program has its delimiter altered to enable it to recognise words.)

Test the code you have written by running this program using the text file `text1.txt` and `text2.txt` as input. (The number of words should be `779` and `3212` respectively.)

Task 3: make a suitable amendment to the `Trie` class (not the main program) to ensure that the list of words is produced in dictionary order.
