Lab 11: Saving Objects to Disk

In this lab we will improve upon last week's `WordChecker` by letting it save its tree to disk, so it doesn't need to be recalculated each time. This class will be called `WordChecker2`.

The first thing your program will do is to ask the user for a file. If this file is a `.lex` file, it will load all the words in that file into a brand new TreeSet, save that TreeSet to disk, and exit. The filename it uses should be the same as the input file, with the `.lex` extension replaced by `.set`. Be sure to print out a couple of lines of text to the user explaining what it did, like this:

```
Welcome to the Word Checker 2.0!
Please enter a .lex or .set file:  english.lex
Loading file "english.lex", which contains 62970 words.
Saving new file "english.set"...done.
Goodbye!
```

On the other hand, if the file is a `.set` file, the program will load it up, print the total number of words, and then enter a query loop identical to that in last week's lab:

```
Welcome to the Word Checker 2.0!
Please enter a .lex or .set file:  english.set
Loading file "english.set", which contains 62970 words.
Please enter a word, or hit enter to quit:
> moose
"moose" is a valid word.
> honorificabilitudinitatibus
"honorificabilitudinitatibus" is NOT a valid word.
>
Goodbye!
```

Fortunately, the `TreeSet` class already implements the `Serializable` interface (with a `serialVersionUID`), so you can load and save it easily.

To save a file, you will first need to create a `FileOutputStream` object. Fortunately, this has an easy-to-use constructor, that takes the file name as its lone argument. Once you have the `FileOutputStream` created, you can make an `ObjectOutputStream`. The constructor for this object takes the `FileOutputStream` as its argument.

Once you have the `ObjectOutputStream`, writing is easy. Just call that object's `writeObject()` method, passing it the `Serializable` object. Be sure to `close()` the `ObjectOutputStream` when you are finished.

Reading a file is similar. Create an `ObjectInputStream` object from a `FileInputStream` object, and then use its `readObject()` method followed by its `close()` method. `readObject()` takes no arguments, and returns an `Object`, which you'll need to cast to a `TreeSet`.