Lab10 - Lucky 21 states

Due: Fri June 5, 2015

This lab covers:

- ❏ Chapter 16 Sorting (well, sort of)
- ❏ Comparable and Comparator interfaces
- ❏ Collections.sort method

From my `Lab10` folder on the k: drive, please grab:

- `StateData.java` - this class holds all the stats for a state
- `lab10_state_data.csv` - CSV file of state stats

I compiled this data from two sources:

- ❖ www.infoplease.com/us/states/population-by-rank.html
- ❖ github.com/ubikuity/List-of-neighboring-states-for-each-US-state/blob/master/usa-states.csv

QUIZ!

To complete Lab10, you must answer these Quiz questions correctly… using only your powers of Java, of course.

1. What is the 21st state in the original data file?
2. What is the 21st state in alphabetical order?
3. What is the 21st most populous state?
4. What is the 21st state that entered The Union?

Ready. Set. GO!

STEP 1 - Original order

Read the data file and store it in a list. The data for each line (except the first header line) is used to create a new StateData object. Add these to a list. This is meatball surgery.

- ★ I put everything in `Lab10`
- ★ Hard-code the file name as a constant (`static final String`)
- ★ I added a static method:

```
ArrayList<StateData> readFile( String fileName) { … }
```

- ★ Use `StateData.parseLine()` to convert each line into an object

Print the list with a counter in front, so you can easily spot the 21st state in the original order. Notice that `StateData` has a nice `toString()` method.

```
Name,Code,StatehoodDate,Population
1 - Pennsylvania,PA,12/12/1787,12787209
2 - New Hampshire,NH,12/18/1787,1326813
3 - Georgia,GA,01/02/1788,10097343
… and so on…
```

Write it down - Which state is the 21st in the original order of the data file?

STEP 2 - Alphabetical order

Change `StateData` to make it `Comparable`. Make your method sort these objects alphabetically, by state name.

Back in your `main()`, use `Collections.sort()` to sort the list of states by name. The `sort()` method auto-magically uses your `Comparable` method to sort the objects in the list. Then, print the list and note the 21st one.

STEP 3 - Most populous to least

Create a `Comparator` class that sorts `StateData` objects by population. Put this class in its own file. I called my `PopulationComp`, but whatev.

Back to `main()`... the `Collections` class lets you sort using a `Comparator`… like this:

```
Collections.sort( List<T> list, Comparator<T> comp);
```

Since we want the most populous state first, then reverse your list order with `Collections.reverse()`. Print and jot down the 21st most populous state.

STEP 4 - Date of statehood order

Create another `Comparator` to sort states by when they joined The Union. Let's try something different and weird… make this guy a static class in Lab10. It has to be static to work in your static `main()` method. Try it!

Also, the `Calendar` objects that hold the statehood date are `Comparable`. In your `Comparator`, get these `Calendar` objects and call `compareTo()`.

Done!
thanks… yow, bill