Lab 08: Call Log Analyzer

Assigned: Thursday, April 4, 2019
Due: Wednesday, April 10 at 11:30pm

1 Objective

In this assignment you will practice reading data from files.

2 Background

In 2006 and again in 2013, many Americans were surprised to find out about the National Security Administration's MAINWAY database, which contains metadata about every telephone call in the United States: who placed the call, who received the call, the length of the call, and other details such as the date, time, and perhaps locations of the callers. The database is used for intelligence and law enforcement investigations.

In this lab we are going to explore how someone who had access to such a database might analyze it to discover interesting information. We will use a simple metadata format in which each line of a file contains information about a single telephone call. First it lists the telephone number from which the call was made, then the telephone number that received the call, and finally the length of the call in minutes. Here is an example line:

```
(717)555-1234 (570)555-6789 12
```

This line says that (717)555-1234 called (570)555-6789, and that they talked for approximately 12 minutes. (All lengths are rounded to the nearest integer.)

While telephone numbers sound like numbers, they should be represented as `String`s in Java, because you would not do arithmetic with phone numbers. We will always write them with an area code surrounded by parenthesis, a hyphen in the middle, and no spaces.

3 Setup

Create a new Java Project in Eclipse named `Lab08`. Download the `callslong.txt` and `callsshort.txt` files from D2L and copy them into the project directory. Then create a new class named `CallLogAnalyzer` .

4 Assignment

Make the following changes to the project:

1. (20 points) Add a method named `totalMinutesUsed` to the `CallLogAnalyzer` class. This method should receive two parameters:

- A `Scanner` that is connected to a data file.
- A phone number in the format specified above.

It should return the total number of minutes of all calls that involved that number (either as the caller or the recipient).

To test this method, you should have your `main` method read in a file name and telephone number from the user, create a `Scanner` connected to the specified file, and call the method with that `Scanner` and telephone number.

If you test it with the `callslong.txt` file and the telephone number `"(850)789-1207"`, you should get the answer `193`.

If you test with the `callsshort.txt` file and the telephone number `"(850)789-1207"`, you should get the answer `0`.

If you test with the `callsshort.txt` file and the telephone number `"(521)262-5923"`, you should get the answer `67`.

2. (20 points) Add a method named `countCalls` to the `CallLogAnalyzer` class. This method should receive three parameters:

   - A `Scanner` that is connected to a data file.
   - A phone number in the format specified above.
   - Another phone number in the format specified above.

   It should return the number of different calls that were made between those two numbers. (A call should be included if it was from the first to the second, or from the second to the first.)

   If you test it with the `callslong.txt` file and the telephone numbers `"(417)865-5822"` and `"(827)922-1650"`, you should get the answer `2`.

   If you test it with the `callsshort.txt` file and the telephone numbers `"(417)865-5822"` and `"(827)922-1650"`, you should get the answer `0`.

   If you test it with the `callsshort.txt` file and the telephone numbers `"(250)126-1494"` and `"(888)682-8161"`, you should get the answer `7`.

3. (20 points) Add a method named `hadContact` to the `CallLogAnalyzer` class. This method should receive three parameters:

   - A `Scanner` that is connected to a data file.
   - A phone number in the format specified above.
   - Another phone number in the format specified above.

   It should return `true` if there was at least one call between those two numbers, or `false` if there was not. This should sound very similar to the previous method that you wrote, and it is. The difference is that while we always needed to process the entire file to answer the previous question, for this question you may find sufficient evidence to be certain of the answer without having processed the entire file.

For full credit, you must correctly code an early exit (without break statements or multiple returns), and I will deduct from the points that AutoLab gives you if I do not see one.

When you test this, you should get `true` on any example for which the prior method returned a positive number and `false` on any example for which the prior method returned `0`.

4. (20 points) Add a method named `findLongestCall` to the `CallLogAnalyzer` method. This method should receive two parameters:

   - A `Scanner` that is connected to a data file.
   - A phone number in the format specified above.

   It should return the telephone number with which that number had its longest call. (It could have been the caller or the receiver of that call.) If multiple calls are tied for longest, it should return the first one that it finds in the file. If there were no calls involving the number, it should return `"(none)"`.

   If you test it with the `callslong.txt` file and the telephone number `"(850)789-1207"`, you should get the answer `"(804)227-5808"`.

   If you test with the `callsshort.txt` file and the telephone number `"(850)789-1207"`, you should get the answer `"(none)"`.

   If you test with the `callsshort.txt` file and the telephone number `"(521)262-5923"`, you should get the answer `"(715)764-4806"`.

5. (20 points) AutoLab will now automatically check for Javadoc comments and good style. (It can't find all types of bad style, though, so I will still be checking some things.) Ensure that you are making it happy and that you think you will make me happy as well.

5 Submitting Your Work

Make sure that you have submitted your final version to the `Lab 08: Call Log Analyzer` AutoLab page, and that (unless you are satisfied with a lower score) you have 100/100 autograded points.

You do not need to submit your work through D2L – I will look at whatever your last AutoLab submission is.