Assignment: Unique IPs

In this assignment you will continue to build on the **LogEntry** and **LogAnalyzer** classes, which you learned about in the last lesson. You will continue to use the method **parseEntry** from the **WebLogParser** class, and you should not modify this class. You will write several methods to solve problems about web logs.

Specifically, you should do the following:

- In the **LogAnalyzer** class, write the method **countUniqueIPs** that has no parameters. This method should return an integer representing the number of unique IP addresses. It should also assume that the instance variable **records** already has its ArrayList of Strings read in from a file, and should access **records** in computing this value. For help, refer to the lectures in this lesson on the unique IP algorithm and code.
- In the **Tester** class (or you can write a new class for testing) write the void method **testUniqueIP** that has no parameters. This method should create a LogAnalyzer, read from the file **short-test_log**, and then test the method **countUniqueIPs**.
- In the **LogAnalyzer** class, write the void method **printAllHigherThanNum** that has one integer parameter **num**. This method should examine all the web log entries in records and print those LogEntrys that have a status code greater than **num**. Be sure to add code in the **Tester** class to test out this method with the file **short-test_log**.
- In the **LogAnalyzer** class, write the method **uniqueIPVisitsOnDay** that has one String parameter named **someday** in the format “MMM DD” where MMM is the first three characters of the month name with the first letter capitalized and the others in lowercase, and DD is the day in two digits (examples are “Dec 05” and “Apr 22”). This method accesses the web logs in **records** and returns an ArrayList of Strings of unique IP addresses that had access on the given day. (Note that the dates in LogEntrys are stored as a Date object, but using **toString** will allow you to access the characters in the Date. For example, consider that **d** is a Date. **String str = d.toString();** allows you to now use a String representation of the date.) Be sure to test your program with code in the **Tester** class. Using the file **weblog-short_log** you should see that the call to **uniqueIPVisitsOnDay(“Sep 14”)** returns an ArrayList of 2 items and **uniqueIPVisitsOnDay(“Sep 30”)** returns an ArrayList of 3 items.
- In the **LogAnalyzer** class, write the method **countUniqueIPsInRange** that has two integer parameters named **low** and **high**. This method returns the number of unique IP addresses in **records** that have a status code in the range from **low** to **high**, inclusive. Be sure to test your program on several ranges. For example, using the file **short-test_log**, the call **countUniqueIPsInRange(200,299)** returns 4, as there are four unique IP addresses that have a status code from 200 to 299. The call **countUniqueIPsInRange(300,399)** returns 2. In this case, note that there are three entries in the file that have a status code in the 300 range, but two of them have the same IP address.

Link to FAQ page for this course: http://www.dukelearntoprogram.com/course3/faq.php

Programming Exercise - Finding Unique I…

✓ Complete