Chris Piech
CS 106A

Section #8
March 3, 2019

Section Handout #8: Data Structures and The Internet

Problems by Brandon Burr, Patrick Young, Nick Troccoli and Andrew Tierno

Your task for this week is to write server program for an application called **FlightPlanner** allow the user to plan a round-trip flight route.

You will need to apply your knowledge of file reading, data structures, string parsing, internet applications, and more to create this program. We have included specific details about the client and server programs below.

Flight Planner

A critical issue in building these programs is designing appropriate data structures to keep track of the information you'll need in order to produce flight plans. You'll need to both have a way of keeping track of information on available flights that you read in from the **`flights.txt`** file, as described below, as well as a means for keeping track of the flight routes that the user is choosing in constructing their flight plan. Consider how both **`ArrayList`**s and **`HashMap`**s might be useful to store the information you care about.

1. Flight

We need a sensible data structure to hold information about each of the flights! Specifically, we need to keep track of where a flight is coming from, where it is going to, and how long it will take to get from its source to its destination.

Implement the **`Flight`** class in the empty file **`Flight.java`** with whatever instance variable, constructors, and methods you think are appropriate. Keep in mind the requests that **`FlightPlannerServer`** will eventually have to handle. Remember, when designing a class, you should to think about promises. What kind of information are you promising to the people who use your class? What sort of behaviors would be most useful for the context in which the class is used?

2. FlightPlannerServer

There are two types of requests that the server needs to handle from the client in order for the client to do its job:

Command	Parameters	Response
`getAllCities`	N/A	Returns the list of all cities, as a string, that the user can visit.
`getDestinations`	**`city (String)`**	Returns a list of all cities that the user can travel to and the times to get to each destination *from the given*

		city in the format specified below.

The flight data come from a file named **`flights.txt`**, which has the following format:

- Each line consists of a source city, a destination city, and the travel time between them in hours separated by commas as in,

  ```
  San Francisco,Singapore,20.9
  ```

- The file may contain blank lines for readability (you should just ignore these).

An excerpt of the data file appears below.

```
San Francisco,Singapore,20.9
San Francisco,London,10.4

New York,London,6.50
New York,San Francisco,6.16
New York,Singapore,22.3
New York,New Delhi,13.9

London,New York,8.16
London,San Francisco,11.3
London,Nairobi,15.6

New Delhi,New York,15.5
New Delhi,Singapore,6.00

Nairobi,Singapore,14.3
Nairobi,London,15.5
Nairobi,Lima,27.2
```

Your server should:

- Read in the flight information from the file **`flights.txt`** and store it in an appropriate data structure when it begins running.
- Respond to **`getAllCities`** and **`getDestinations`** requests as described above.
- Specifically, if we called the command **`getDestinations`** with the parameter **`"New Delhi"`** we would expect it to return the String

  ```
  [New Delhi->New York:15.5, New Delhi->Singapore:6.00]
  ```

  and more generally

  ```
  [source->destination1:time1, source->destination2:time2,...]
  ```

Hint: when a server receives a request, it must respond with a String. If you need to respond with a *list* (e.g. an **`ArrayList`**), one method is to return the value of that list's **`toString`** method. For instance:

```
ArrayList<Flight> myList = ...
String stringToSend = myList.toString();
```

when you call this method on an **`ArrayList`** of a class (like your **`Flight`** class),

`myList.toString()` will call the **`toString`** function of each element in your list! How can we take advantage of this fact to produce the desired response?