Delta College

CST 283
Programming Assignment 1

Winter 2024
Instructor: T. Klingler

Objective

This program provides an opportunity to practice Java files, String objects, and basic array searching.

Overview & Instructions

Write a program that will allow scientific queries to be performed on (actual) earthquake data. The (very large) file quakes.zip is provided and includes all earthquakes worldwide with magnitudes 4.0 (on Richter Scale) from 2010 through Summer 2018. The file format is:

`{date-time}  {latitude}  {longitude}  {magnitude}  {location}`

with data delimited by a pipe (**|**) character. A seismologist is requesting that an *API* (application program interface) be created to allow basic queries to be performed on the dataset with minimum keystrokes.

Initially, read the entire contents of the data file to an array-based data structure. You can choose parallel arrays or an array of objects. You will then be be performing searches on the array(s). Please avoid use of specialized Java "container" classes such as `ArrayLists`, `HashMaps`, etc.

Your API will allow the user to key in a string into the program interface

`R,minLat,maxLat,minLon,maxLon`	List all quakes in dataset by *region* with north/south boundary between **`minLat`** and **`maxLat`**, and east/west boundary between **`minLon`** and **`maxLon`**.
`D,minDate,maxDate`	List all quakes in dataset by *date* with the calendar date of the quake from **`minDate`** and **`maxDate.`** Include quakes from the minimum date through and including the maximum date.
`M,minMag`	List all quakes in dataset with *magnitude* **`minMag`** or greater.

Queries must be validated prior to processing. First, validate that the format matches one of the patterns above. This implies that each must begin with a character **`R`**, **`D`**, or **`M`** only. Next, be sure that the number of comma-delimited tokens match the expected pattern. Finally, be sure the data contained in each token are valid. Earthquake magnitudes must be 4.0 or greater. Latitudes must be –90.0 to 90.0 degrees. The minimum latitude would be the southern boundary. Longitudes can only be –180.0 to 180.0. The minimum longitude would be the western boundary. To validate the calendar dates, you have permission to use algorithms demonstrated in class. Along with two valid dates, be sure the first date is less than or equal to the second date.

For any queries that do not match this format, provide an error message and allow the user to start again. Some valid sample queries could be:

```
R,40.0,45.0,-90.0,-80.0
M,6.2
D,2017-01-01,2017-12-31
```

For your interface,use simple console input (via the Scanner class). Similarly, output can be directed to the Java console to restrict your earthquake data system to a *command-line* application. A simple query will likely generate a long list of earthquake events. Pull all matching quakes from you data (stored in the array-based data structure) and list all that satisfy the query. Once all information has been displayed, be sure to return to the menu to give the user the the option to perform another query or quit.

Formatting of the "raw" quake data is expected. For example, if the following data line from the file matches the query:

```
2015-05-02T16:23:07.580Z|42.2357|-85.4285|4.2|5km S of Galesburg; Michigan
```

reformat it for your output to look like:

```
02 MAY 15 1623Z, (42.24,-85.43), Mag: 4.2, 5km S of Galesburg; Michigan
```

Be sure to round the geographical coordinates to two decimal places. Also, reformat the event date-time from the coded format to a "friendlier" format. For example:

Change: `2015-03-07T00:58:34.380Z` to: `07 MAR 15 0059Z`

where each month will be abbreviated to three characters and the universal (**Z**) time is rounded to the nearest minute with all colons removed. Note also that the day of the month will always be two digits and the time will always be four digits (including any leading zeros). Make sure the minute is correctly rounded up or down to the nearest minute.

Design your solution with appropriate modularity in mind. Either a procedural approach or an object oriented are acceptable.

Deliverables

Demonstrate the development steps of your program with at least two version commits to the assignment Git repository.

Deliver the following to the eLearning system **Assignment Dropbox** as your final product:

- **Upload** your **source code** (.java) file(s); preferably zipped in more than one file.
