2019 Java Coding Competition

The Problem

As a new developer at SF Analysis, a project has been provided to assist in the implementation of an application using natural disaster data provided by https://ourworldindata.org/natural-disasters. The natural disaster files can be found in src/main/resources:

- natural-disasters-by-type
- significant-earthquake
- significant-volcanic-eruption

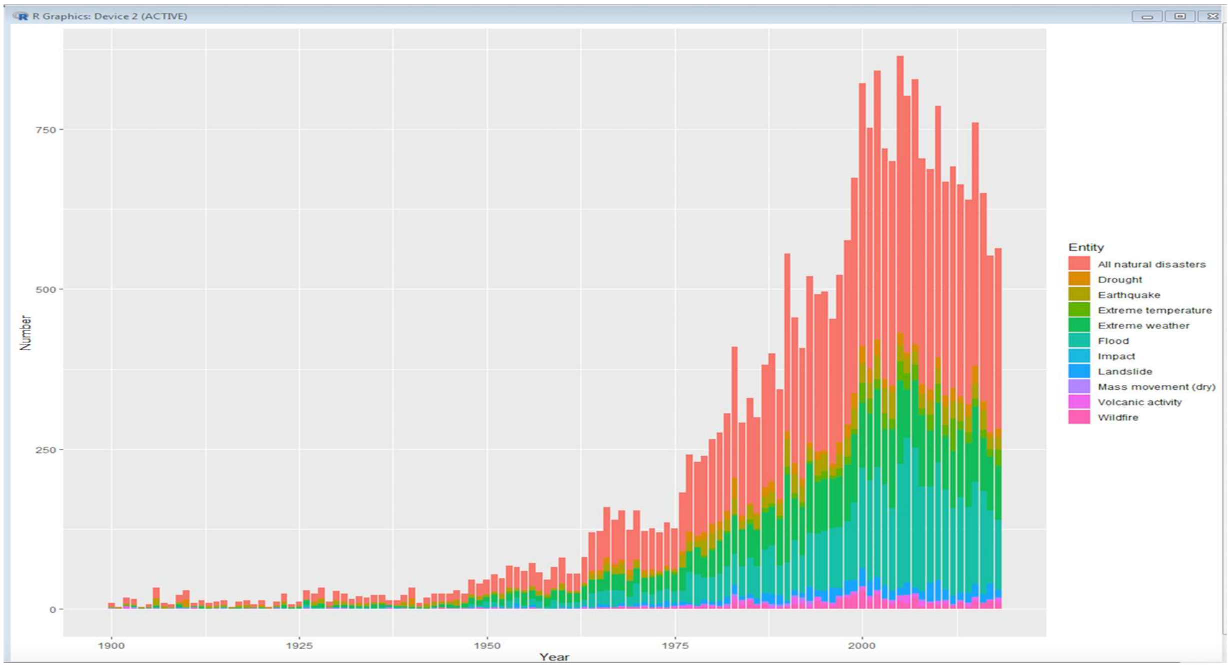
There are some basic requirements (JUnits) that must be completed before you are able to showcase your creativity in using the data. The problem is separated in multiple pieces and allows you to build confidence as you pass the simple problem of reading the data and completing data queries, etc. JUnit tests are the highest authority associated to the basic requirements and can be found at *src/test/java*.

Bonus "Nice to have" features:

- Bonus Credit – Do not complete any bonus features unless you have completed all the required functionality/JUnits pass.
  - To help visualize the data, we recommend using R to draw charts (https://www.r-project.org/). If you're familiar with Python, you can also download Anaconda for charting the data instead of R (https://www.anaconda.com/distribution/).
  - Create additional functionality and demonstrate through a Graphical User Interface (GUI), command line interface, JUnits, or web UI.
  - If you can think of any other useful features to add, we appreciate ingenuity and will gladly accept any useful enhancements. Be sure to document any bonus features in the feedback text file.

If you decide to create a GUI or print info from the data provided – you must give credit to 'Our World in Data' - https://ourworldindata.org/natural-disasters. *Our World in Data* is created as a public good and all visualizations are Creative Commons licensed. Please give appropriate credit when doing bonus work.

An example of a chart using R is shown below, but you are welcome to use any chart to represent the data from https://ourworldindata.org/natural-disasters.



2019 Java Coding Competition

********* **Do not change anything in the JUnit tests!** **********

First Actions:

- Import the problem statement into your IDE.
- We have provided Maven dependency for JUnit 4. If you are not set up with the recommended IDE (Spring Tool Suite), you may need to add JUnit 4.
- If you identify any additional libraries you would like to use, please add them to the pom.xml file or copy the .jar files into the resources folder
- Run your JUnit tests, code, and repeat.

When you are done:

- Update the feedback.txt file and include the following information:
  - Your team – name of each individual participating.
  - How many JUnits you were able to execute successfully.
  - Document and describe the additional "nice to have" features included, to help the judges properly grade your submission and explain how to properly execute new enhancements.

- Push your changes to one single branch for you and your teammate. Open a single pull request against the main State Farm Coding Competition repository before 11:59PM CST on October 5, 2019.
  - If you make any commits after midnight without prior approval from codingcompetition@statefarm.com, your submission will be disqualified.
  - If you so choose, you may open a pull request at any time during the competition and continue to update it as long as you do not make any commits after midnight.

Rules

- Contestants cannot seek help from individuals outside their team.
- Teams are expected to have the necessary tools and JARs preloaded on their machines **prior** to the competition.
- If you believe this document and the JUnit tests conflict, the JUnit tests are the highest authority.

How you will be Graded

- 100% core requirements met, including:
  - Number of JUnits that pass using correct functionality in the program
  - Maintaining Object Oriented Programming principles
  - Code documentation
  - Code must compile and execute
- Do not complete any Bonus unless you have all the JUnit tests completed
  - Bonus credit awarded for any extra features added (up to 10%)

In the event of a tie, we will further judge your solution based on: code cleanliness, maintainability, and adherence to object-orientated principles.