Quiq

Programming Exercise

As part of your interview process, we ask that you complete this programming exercise. The purpose of this exercise is to allow us to see how you solve problems and write code as well as to foster discussion during your potential in-person/on-site interview. We ask that you spend between 2-4 hours working on this problem. You may use any programming language you like. You may use third-party libraries so long as they do not directly solve the problem at hand. When complete, please email your source code or a link to a private GitHub repository to brad.pascoe@goquiq.com. Thank you for your time and effort.

The Problem

JSON is a popular serialization format supported by virtually all modern programming languages. It is typically schema-less and can store arbitrary data. Your task is to write a command-line utility that takes two files containing arbitrary JSON documents and outputs a score between 0.0 and 1.0 indicating the degree of similarity between the JSON. 0.0 means no similarity and 1.0 means complete equivalence of the data. There is no single 'correct' way to score the similarity of two unequal documents, but the scores you assign will ideally be intuitive and useful to a user of the application. We've provided some example JSON documents for you to test with that are attached to the same email as this exercise.

Your Solution

- Should take two files as command line arguments. Positional or named arguments are fine.
- Should output 1.0 for any pair of JSON documents that contain the same data
- Should output a value < 1.0 for any pair of documents that are not equal
- Will ideally use intermediate scores to convey a degree of similarity between the documents
- We suggest focusing on detecting equality before assessing similarity
- You should not be writing any code that directly parses the JSON. Use a builtin language feature or a third-party library like one of the following:
    - Java: https://github.com/google/gson
    - .NET: http://www.newtonsoft.com/json
- Feel free to output additional information in addition to the similarity score that might help the user
- Please include a README file that describes any assumptions you made and also highlights anything about your solution you'd like us to know