Reedsy Test

Test : https://gist.github.com/pedrosanta/aa4ca7260cd7a3d658c739c194ec1743
Github for question 4 : https://github.com/AlexPavy/Reedsy-Text-Convert-Machine

Question 1

I'm currently a backend Java Software Engineer in France in a growing tech company in e-commerce.
Throughout my previous jobs, I've been the happiest when I had the most freedom in work as well as good communication with the team, so I'm looking ideally for a job in a startup.
As a software developer, I take special care on work efficiency as well as the structure and readability of the code.

In my spare time I like to follow tech news, artificial intelligence, philosophy, science, genetics and I like to do latin dances or play video games.

Question 2

Operational Transformation

Consistency maintenance and concurrency control in collaborative editing of plain text documents.

The actions are queued in order of timestamp and all previous actions are applied to each arriving action before it is applied to the document.

Example

If we have 2 operations
Op1= insert at position p1 and Op2= insert at position p2 <= p1 and Op2 is before Op1 (according to the timestamp) then we must modify Op1 with p1 = p1 + 1.

Sources

https://en.wikipedia.org/wiki/Operational_transformation

Question 3

Use the longest common subsequence algorithm, with memoization. Mark the longest common subsequence from both files with the same id, then exclude this sequence and move on to the next sequence. The id will be useful to visualize the differences, and mark the common subsequences.
The remaining sequences that are not common are either an addition if present in file2 but not in file1, a deletion if present in file1 but not in file2, or a modification if some text is present in both files.

Example

File1 = Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.
File2 = JavaScript runtime built on Chrome's V8 Java engine.

1) Longest common subsequence : “JavaScript runtime built on Chrome's V8”
File1 = Node.js® is a <same id=”1”>JavaScript runtime built on Chrome's V8</same> JavaScript engine.
File2 = <same id=”1”>JavaScript runtime built on Chrome's V8</same> Java engine.

2) Longest common subsequence : “engine”
File1 = Node.js® is a <same id=”1”>JavaScript runtime built on Chrome's V8</same> JavaScript <same id=”2”>engine.</same>
File2 = <same id=”1”>JavaScript runtime built on Chrome's V8</same> Java <same id=”2”>engine.</same>

3) No more common subsequence. Mark additions, deletions, and modifications in a simple linear operation, also with ids.
File1 = <delete id = “3”>Node.js® is a</add><same id=”1”> JavaScript runtime built on Chrome's V8 </same> <modify id=”4”>JavaScript</modify><same id=”2”> engine.</same>
File2 = <same id=”1”>JavaScript runtime built on Chrome's V8 </same><modify id=”4”>Java</modify><same id=”2”> engine.</same>

With tree structure of a json format : start the algorithm from the higher levels

Sources

https://en.wikipedia.org/wiki/Longest_common_subsequence_problem