ILP CW2 Specifications (Programming Task)

10.10.2023

As you read through the ILP Course Specification you have gained an idea what the ILP requirements, specifications and demands are all about. As CW1 was mainly focused on getting you started and setup, CW2 now is the real implementation of the PizzaDronz service. You will need and reuse everything you did in CW1, just add additional items to the solution.

As the ILP Spec was slightly modified in some details, you should retrieve the latest version of it.

In general, CW2 is about creating a runnable application, which can be started like:

```
java -jar PizzaDronz-1.0-SNAPSHOT.jar 2023-11-15 https://ilp-rest.azurewebsites.net
```

In the application you are supposed:

- Read orders for the specified day (and only the day) and restaurants plus other relevant data from the REST-Server whose URL is passed in (see the spec)
    - *Orders retrieved for a day will only contain data for that day as the REST-Server only delivers filtered data if queried by date.*
- Validate orders
- Calculate the flightpaths for all valid orders in the exact sequence you received them
- Write the 3 result files **in a folder resultfiles** (create if not exists)

If you call the program several times for the same day you are supposed to overwrite previous results. If called for different days, you will just add more files to the folder.

During marking we will apply an auto-marker which will check your results. This auto-marker will build and start your application and check the result files for structure, content and logic (so it is a kind of black-box testing). As everything the application does will appear finally in the resulting files, this is a very straightforward way to see if all rules were obeyed and the application works according to specification. This approach is done in the industry very often as well, as the internal workings are usually of secondary concern, yet the results are what is checked.

In addition, we will check (to some extent) your code for structure, efficiency, elegance, commenting, resiliance and algorithms to make sure that you produce good quality software.

Thus, please make sure you have comments (JavaDoc and inline where appropriate), use proper class structures and your code is not just a big blob of lines, but really structured. In addition, good error catching, and recovery are recommended as we will try to destabilize your code with wrong / invalid data as well as false URLs, etc. Terminating the app gracefully and producing no results (if data doesn't allow it) is fine – just do not slip exceptions through.

Remove unused parts of application as well, please.

A result of this analysis is available in HTML format upon request.
Currently we cannot publish the results online (due to data protection) and sending them individually to 260+ students is impossible. We are investigating our options here, yet for now that is the situation.

So, should you have questions regarding your mark, we can provide these files ***on demand after marking and availability of marks****.*

This CW2 programming task has a maximum mark of 50 / 100 points (in relation to the entire ILP course).

For your submission, please zip the entire solution directory to ilp.zip and submit accordingly.