ILP COURSEWORK 1 MARKS AND FEEDBACK

This report is for student s1915409

Submission time of marked submission: 2021-10-15-12-14-48

The raw overall mark is (23/25)

The awarded mark includes a deduction for any days late
(5% per calendar day).

Days late: (0)
The awarded mark is (23/25)

As described in the coursework instructions, marks are given for
three aspects: correctness, documentation and readability. See below
for details on each aspect.

1. Correctness (15 marks)

We asked that your application should correctly implement the classes
and methods described in Section 1.4 of the instructions.

Correctness mark: (15/15)

Correctness checking first involved us running Maven and your provided
pom.xml file to compile your code together with the JUnit 4 test class
we gave you for checking your code is working as intended. Part of
the correctness mark is based on whether this build was automatic or
whether manual intervention of some kind (e.g. editing your pom.xml
file) was needed for the build to be successful.

Build comments (blank if no issues):
BEGIN

END

The results of running the 21 JUnit tests are as follows.

Number of tests passed: 21
Number of tests failed: 0

Number of tests with errors: 0

Report on Tests:
BEGIN
All tests passed.

END

2. Documentation (5 marks)

We asked that the methods you implement should have JavaDoc comments which provide brief but clear descriptions of the purpose of the method and its return value and the role of each parameter passed to the method.

Documentation mark: (5/5)

Comments:
BEGIN
You have provided good JavaDoc documentation giving helpful information about the methods which you have implemented. Well done.
END

3. Code readability (5 marks)

Your Java code should be well-structured and clear with idiomatic use of the Java language. You should consider the readability of your code, thinking that it will be passed on to the developers of the drone delivery service to extend and maintain as their needs change.

Aspects of code readability that markers focussed on included:
+ how well you structured your code judged by how well you divided up functionality into methods,
+ use of descriptive names and final modifiers for constants,
+ use of HttpClient to access web content as recommended in class,
+ use of the Google Gson parser to parse JSON documents,
+ use of an efficient map-like data-structure for storing the mapping from item names to their prices.

Readability mark: (3/5)

Readability comments:
BEGIN

You have a largely unstructured solution. There were opportunities to define methods here to structure your application, but you have not chosen to do this. This makes your code more difficult to read and more difficult to maintain.

You have declared some final values for constants in your application; this is good practice.

You declared a static instance of HttpClient to access the web server. This is good practice for a heavyweight object such as HttpClient.

You used the Gson parser to parse JSON documents obtained from the web server. This is a good approach.

You have not used a HashMap, Hashtable or TreeMap to store the menu which maps items to prices; this was a missed opportunity which required you to write code to search for a price instead of just looking it up from a map-like data-structure.
END

END OF REPORT