Project 1 – Unit 6

In this unit, you have to provide following operations, for persisting the LinkedHashMap into a Database:

- Create
- Update
- Delete

The methods used for above operations should be called in following cases:

1. When a new Automobile is added to the LinkedHashMap.
2. When an Automobile is deleted from the LinkedHashMap.
3. When an Automobile is updated in the LinkedHashMap.

You will need to design and implement:

1. A database schema (be sure to apply rules of normalization, discussed in the class).
2. Create a set of classes, for managing Create, Update and Delete operations.
3. You should write a driver class (with main method) and test out the functionality, in a console program on the server side.

Do's and Dont's

1. Implement Database functionality in the server.
2. Do not implement any functionality from the client side.
3. Do not save Option Choices.
4. Client interaction for Create, Update and Delete operations are not to be implemented or tested.
5. All functionality should be tested in a driver from the server side. In later versions (not meant for this course), the interactions with client can be setup.

How?

1. You will be given detailed examples in class, on how to setup a database and connectivity details.

Grading your Submission

1. Program Specification/Correctness (25 points)
    a. Compilation/Execution
        - No errors, program always works correctly and meets the specification(s).
    b. Code Reusability
        - Code could be reused as a whole or each routine could be reused.
    c. Design
        - Reusable and Extensible.
    d. Interfaces and Abstract Classes
        - To be applied for both units 5 and 6.
    e. Database
        - Normalized Schema.
        - Database classes are setup in a way, such that SQL is read from a text file (improves modifiability).
    f. Testing:
        - Code is adequately tested and test runs are shown for units 5 and 6.

2. Readability (5 points)
    a. No errors, code is clean, understandable and well-organized.
    b. Code has been packaged and authored, based on Java Coding Standards.

3. Documentation (5 points)
    a. The documentation is well written and clearly explains the functionality implemented.
    b. Detailed class diagram is provided.

4. Code Efficiency (10 points)
    a. No errors, code uses the best approach in every case. The code is extremely efficient, readable and understandable.

Reclaiming lost points

If you lost points in Units 1 through 4 you should:

1. Modify your project and fix issues that were reported to you (i.r. for which points were taken off).
2. Create a change log that reflects following:
    a. What were you asked to change?
    b. How many points were taken off?
    c. What changes were made in what files?
    d. Show test cases for changes made (if applicable).
    e. In your opinion, how many points should be added back?

3. TA's will review this change log, grade and return the lost points, if they feel that issue(s) have been corrected.

Lessons Learned

Please document the following:

- Things or concepts you learnt, which could be applied in 2nd Mini. – 50 points.
- Well-organized list of Design lessons learned from Project 1 and submit to cislabs04@gmail.com.

You will earn one point for each item (content quality is important in this context).