CS 319 Spring'22 - Design Patterns Assignment

In this assignment, you are going to implement a simple task management app. The app will allow users to create tasks and lists. A task has a description, status, and target date (i.e., deadline). Task status can be one of the following: *created, in progress, completed*. Initially, tasks will have *created* status. Users can move a task from *created* to *in progress* or *completed* statuses. Tasks that have *in progress* status can only move to *completed* status. The status of *completed* tasks cannot be changed.

Users can add a couple of options to their tasks, which are as follows :

- *TrackElapsedTime*: this option will allow users to track the elapsed time (in days) since the creation date of the task. Tasks with this option should present the elapsed time alongside their description.
- *TrackStatusHistory*: the app should track the statuses of tasks with this option enabled and present the history alongside their description.

Users can also create lists that consist of tasks and other lists. Lists will also have a description, and they can store the tasks sorted by their alphabetical order, added order, or target dates (all in ascending order). The sorting only applies to tasks (lists always appear after the tasks in a list). Consider the following sample scenario, where we have an initial list named "My Todos", and its tasks are sorted by their added order:

```
My Todos [Add Order] {
-Fix lights 2022-05-22 [In Progress]
-Attend Seminar 2022-05-10 [Created]
CS 319 [Target Date Order] {
-Prepare iteration 1 reports 2022-04-10 [Completed]
-Submit design patterns HW 2022-04-26 [In Progress]
-Address TA/Instructor feedback 2022-05-02 [Created] [Elapsed time: 0 day(s)]
Implementation [Target Date Order] {
-Define classes 2022-04-20 [Completed] [Status History: Created->In Progress->Completed]
-Design backend APIs 2022-04-30 [In Progress]
-Implement front-end components 2022-05-01 [In Progress] [Elapsed time: 0 day(s)] [Status
History: Created->In Progress]
}
}
Grocery [Add Order] {
Fruits [Alphabetical Order] {
-Apples 2022-04-27 [Created]
-Bananas 2022-04-25 [Completed]
-Oranges 2022-04-22 [Completed]
}
Dairy [Add Order] {
-Milk 2022-04-29 [Completed]
-Yoghurt 2022-04-23 [Created]
}
}
}
```

The contents of a list are presented inside curly brackets. For example, the “My Todos” list has two tasks (“Fix lights” and “Attend Seminar”) and two lists (“CS 319” and “Grocery”). The “Grocery” list has two lists inside called “Fruits” and “Dairy.” The “CS 319” list has three tasks and one list (“Implementation”). The “Implementation” list has three tasks. The order information of lists should be provided alongside their descriptions. The tasks inside a list are presented by a hyphen (-). For each task, its description, target date, and status should be presented. If an option is enabled for a task, additional information regarding that option should also be presented. For example, the “Implement front-end components” task has both options enabled, so the elapsed time and history of its statuses are provided. The “Address TA/Instructor feedback” task has only the *TrackElapsedTime* option enabled.

For this assignment, you need to provide a report and the implementation of classes in Java. The report must consist of a maximum of 2 pages. It should include a UML class diagram and indicate which patterns you used, where, why, and how you used them. You also need to implement the sample scenario above by instantiating class objects manually in the **main** method. When running your app, it should print the given output from the scenario. You can use built-in Java libraries (e.g., java.util) for implementing your solutions, but not external libraries.

Submission Notes

- Deadline: **26 April 09:00 (No late submissions allowed).**
- Create a single zip file for your solution named STUDENTID.zip (e.g., 2200XXXX.zip) and email it to elgun@bilkent.edu.tr with the subject line of **CS319-HW**.
- The zip file should contain the following files:
    - Report.pdf (should contain the UML diagram and explanations)
    - Solution.java (should contain the classes and the implementation of the scenario)

We will follow the Bilkent University Code of Academic Integrity for this assignment, as explained in the Student Disciplinary Rules and Regulation. Violations of the rules will not be tolerated. By submitting the assignment, you accept the honor code that you will solve the assignment on your own without any collaboration. Violation of this policy will result in appropriate disciplinary action.