From Product Backlog product to Sprint Backlog, a story of filters

Download the Agile.zip file and integrate the sources into a new Java project with an UTF-8 encoding.

This archive is composed by:

- TypeUserStory : enumeration of the different types of user stories handled in a BackLog.
- UserStory : describes a user story and contains the three sorting (or filtering) criteria: its business value, its complexity, and its type.
- BackLog : abstract class, mother of classes BackLogProduct et SprintBackLog.
- BackLogProduct : contains all the user-stories still to be done in a project.
- SprintBackLog : contains all the user-stories of a sprint.
- ProductOwner : represents the work of the product owner who is in charge of creating the Product Backlog and proposing various sprint backlogs.
- ProductOwnerTest : contains the two non-regression tests, ensuring the correct integration of the chain of responsibilities pattern and his two instances, one for the method `productToSprintBacklogStrategieOne` and the other for the method `productToSprintBacklogStrategieTwo`.

The goal of these two methods is to chain different filters to have an automatic sprint backlog proposal.

1. As you did in practical work on the different cash dispenser strategies, integrate the Chain of Responsibility design pattern to have 6 filters encapsulated by 6 different classes. **Export your code to a zip file named responsibilities.**
2. **First refactoring**: realize that we can only have three classes of filters instantiated using a business value, or using a complexity, or using the type of user story that we want to delete. Simplify the data of the two chains of responsibilities **Export your code to a zip file named firstRefactoring**.
3. **Second refactoring**: you have three classes with the same loop code and only the condition of deleting a user story that changes. Use an abstract method userStoryToRemove and redefine it in the three filter subclasses, in order to factorize the identical code. Export your code to a zip file named **secondRefactoring**.

Don't forget to upload your 3 zip files in the homework rendering.