# Homework 4
## Creating a Domain Model
Check Canvas Assignment Posting for Due Date

In Homework 2, we asked you to improve the design of EvalSorts through refactoring. Refactoring amounts to a bottom-up design approach: you have an implementation (bottom) and you try to rearrange and re-express that implementation so that it is easier to change (a better design) and then modify the functionality.

In this Homework, we want you to create a new design for EvalSorts, that meets the requirements of Homework 2, but that does not make any attempt to preserve any of the existing implementation -- instead we want you to focus on creating a "clean-slate" design that supports the desired functionality, but instead, is obtained by applying the analysis techniques we have discussed (and, hopefully, fully exploits the design principles we have discussed.)

**Deliverables**

1. **List of brief use cases** for all the use cases in the application. ["run a single experiment from the command line (no user input) using data specified in a file" and other use-cases evident to you]. For each of the use cases, you should supply a one or two sentence description.

Example 'brief' style use case description (from a design for an ATM):

> User logs in: User swipes card and enters his pin and successfully logs in. He is provided with menu for possible actions.

2. **A use case drawing** for the full application.

3. **A fully-detailed "happy path" scenario for the use case:** "run a single experiment from the command line (no user input) using data specified in a file." See the Template (slide 5) in the Jan 26 lecture, the example from jan. 29 lecture, and slide 3 in Jan 31 lecture.

4. A set of candidate **domain concepts based on relevant noun phrases**.

Sources for candidate noun phrases:

- Use cases defined in part 1,3 (including the detailed scenario).
- Requirements given in the Homework 2 question
- Solution provided for Homework 2
- Slides explaining Homework 2 (feb 5)

Remember, you are only looking for relevant nouns. You are NOT reverse-engineering the design of the posted code -- you are, hopefully, producing a much better, original to you, design.

5. Analyze and choose domain elements from your list of candidate domain concepts in part 4. **Draw a domain model with important attributes and associations.** Please include annotations (number and type) on the associations.

6. A Glossary of terms. For each of the domain elements in your model, supply a short textual definition that specifies what the element represents, its major responsibilities, and how it participates in the system (does it create other objects, direct their activity, respond to queries, etc.)

Format:

Domain element name:
Definition:
Responsibilities:
Interaction with others: (Relations - aggregation, creation, serves)