CS4240, Principles of SW Design: Some Sample Exam 1 Questions

Sample Question #1: Class modeling problem.
We are going to model the requirements for the permission strategy that allows users and groups of users to access files and directories in a new operating system. Create a class model for this system that includes what is described at the end of these instructions. (You might want to sketch it out in advance quickly to get the boxes and lines arranged.)

Important: Focus on identifying classes, creating the obvious relationships between them, and the attributes and any operations that can be easily inferred from this description. There's no need to include operations. For associations, indicate multiplicity whenever possible (but not navigability) and document the meaning of each association in your diagram. You are not required to include any of the more advanced UML notations beyond classes and their relationships, multiplicity, operations, and attributes. (But some of these might be useful in this model.)

- For each user, we store a user-id and a password. Users belong to one or more groups. Each group has a unique id.
- Certain users may be super-users, who will be able to access files or directories that they have not been given explicit permission to access.
- Files and directories each have a name. We also store a world-permission and a group-permission field. These field values can be "read" or "write" or "no access". The system stores the size (in bytes) for each file.
- Like in UNIX or Windows, directories can be nested, i.e. subdirectories are possible.
- Every file or directory that is created "belongs to" a user and has that user's group setting. (Just like UNIX.)
- Our system does not support any kind of link (as in UNIX) or shortcut (as in Windows).
- Like modern versions of UNIX, our system will support Access Control Lists (ACLs). A file or directory can have an ACL that provides a list of groups and/or users who have permissions beyond those stored for each file or directory.

For example, a file might be protected as "no access" for both group and world permission. But an ACL would contain entries on the list that grant "exemptions" to these settings. E.g. we could grant "write" privilege to user "Bob" and "read" privilege to the group "faculty". Access by users other than Bob and all the faculty would be governed by the file's permission fields.

Sample Question #2: In this Java code, which of the two method calls (if any) are allowed according to the Law of Demeter? <Snip>

Sample Question #3: What are some reasons or situations in OO design where we should prefer aggregation over inheritance and why?

Sample Question #4: One OO design principle states "program to an interface not a particular implementation". Answer the following questions about this principle… <Snip>

Sample Question #5: Here's a situation (blah blah blah). Which of the design patterns we've studied might be applied here and why?