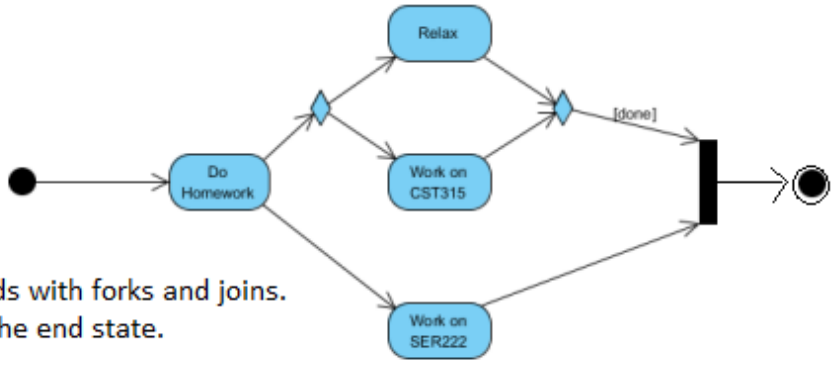
CST315 Final Exam Prep.

1. **Behavior Modeling**: The following is an example of an activity diagram with incorrect syntax. Identify **one** of the issues in this diagram: describe it, and suggest how it might be fixed. [2 points]



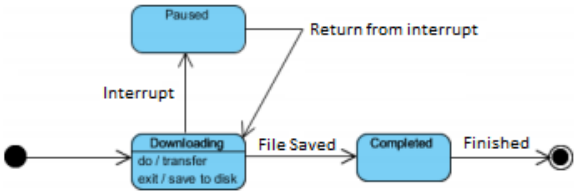
Replace diamonds with forks and joins.
Terminate with the end state.

2. **Behavior Modeling**: Activity diagrams and Statecharts are both ways of describing the behavior of a system. How do they differ? (Do not explain in terms of syntax - explain in terms of intent.) [2 points]

Activity diagram describes the flow of logic in the form of activities. Activity diagram is similar to flow-chart.

State-Chart describes sequence of states an object goes through. the cause the transition from one state to other and the action that result from a state change.

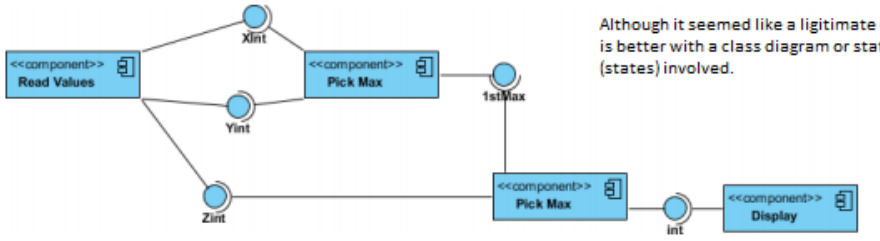
3. **Behavior Modeling**: The following statechart shows the state of a FileDownload object. Give a label for each of the transitions - use proper syntax. [3 points]



4. **Structure Modeling:** In general, if we are trying to build a loosely coupled system, would we be better off using generalization or realization relationships? Explain why. [2 points]

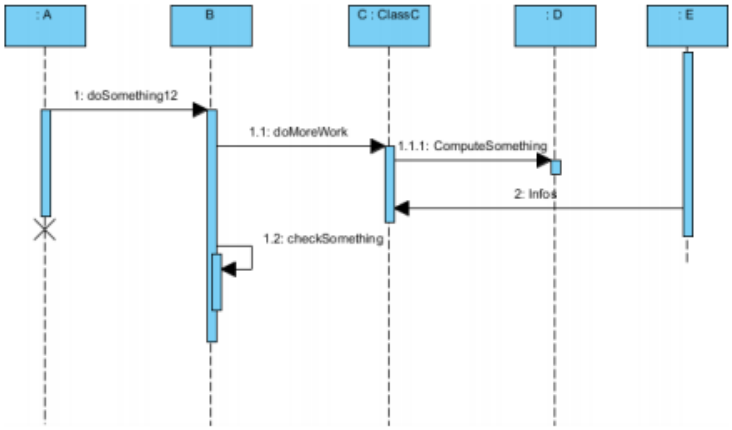
We would be better off using realization because it interfaces the superclass and it doesn't matter how the interface is implemented, hence abstraction is maintained, and the code base is loosly coupled.

5. **Structure Modeling**: The following is an example of a component diagram with correct syntax. Does this diagram seem like legitimate use of a component diagram? Explain. [2 points]



Although it seemed like a ligitimate use of component diagram, it is better with a class diagram or state chart because of the fields (states) involved.

6. **Interaction Modeling:** The following is an example of a malformed sequence diagram. Identify **one** of the issues in this diagram: describe it, and suggest how it might be fixed. [2 points]



(1) Give a start state at :A sequence.
(2) Give an element creation point.
(3) extend the lifeline at object:E to the same length as other objects:classes

7. **Interaction Modeling:** Sequence and Communication diagrams can portray almost the same information - is there any reason to use both diagram types? Or should you pick one and stay with it? Explain. [2 points] (This is an opinion question - provide a cogent argument for full credit.)

A use case diagram provides documentation of functional requirements needed for the system.

An activity diagram is used to portray the sequential flow of business process activities, the steps of a use case, or the logic of an object's behavior (i.e., its methods).

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time.

A collaboration diagram, also called a communication diagram or interaction diagram, illustrates the relationships and interactions among software objects.

A class diagram shows how the different entities (people, things, and data) relate to each other; in other words, it shows the static structures of the system.

State chart diagrams are used to describe the behavior of a system by describing all of the possible states of an object as systems events occur.

As we can see, while each of the different diagrams portrays a different aspect of the system and the objects within it; when taken together, a clearer and more integrated picture of what those objects are and how they interact with other objects and processes within the system can be attained.

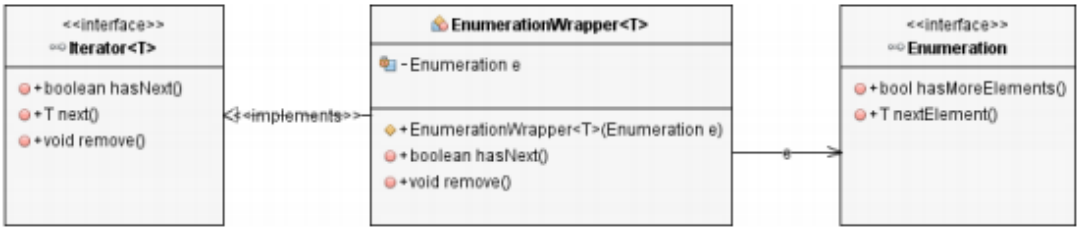
8. **Design Patterns:** What is a significant advantage of having a "vocabulary" of design patterns?

Design Patterns give you a shared vocabulary with other developers. Once you've got the vocabulary you can more easily communicate with other developers.

9. **Design Patterns:** Consider the task of a writing a class to manage logging information during a program's execution, and saving it into a single text file. What design pattern could you apply to this problem? Explain your choice.

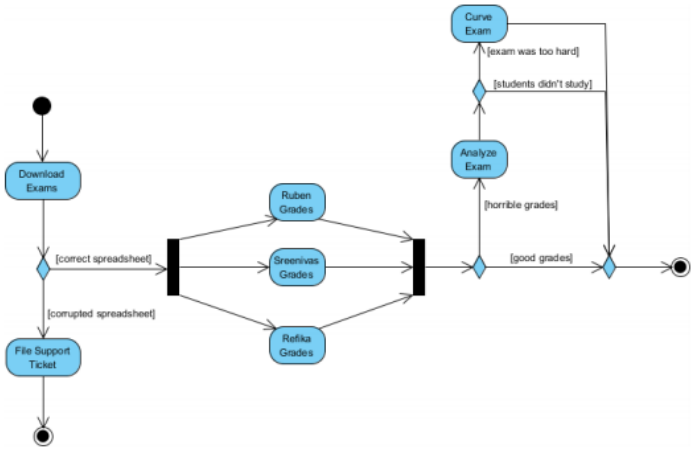
I would use the Singleton design pattern because the Singleton pattern is a design pattern that restricts the instantiation of a class to one object. This is useful when exactly one object is needed to coordinate actions across the system, which is this scenario.

10. **Design Patterns:** Review the following UML class diagram. What design pattern appears to be used here? Explain. [2 point]



the adapter pattern is a software design pattern that allows the interface of an existing class to be used from another interface. It is often used to make existing classes work with others without modifying their source code.

11. **(Scenario) Behavior Modeling:** The follow is an example of a valid activity diagram:



Describe this process in plain English. Your description should implicitly contain the meaning of each element in the diagram.

Interpretation:
Initially the exams are to be downloaded.

If the correction spreadsheet is corrupted, then a support ticket is filed and the process is done.

Else, make the correction spreadsheet available to Ruben, Sreenivas, and Refika to collect the grades and merge into one file for analysis.

If the grades are horrible, move on to analyze the exam. If the edxam is too hard, then move on to grade the dxam using a curve system. Otherwise, if the students didn't study (i.e the exam wasn't too hard,) keep the original grades and finalize the process.

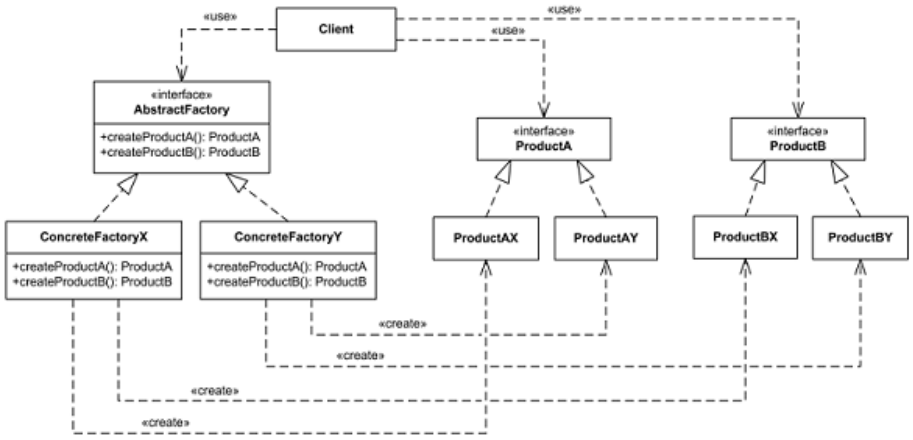
Else, the grades are finalized and the process is complete.

12. **(Scenario) Interaction Modeling**

FYI: if this was the real exam, you would probably be looking at a question asking for you to read or fix a class diagram

13. **(Scenario) Design Patterns:** Consider the following system description: "YouTube is a video sharing website where users can view videos, create channels, and upload videos to channels. Viewers who create accounts are able to subscribe to channels. If they subscribe, they will be notified of new videos on the channel. Viewers also have the ability to unsubscribe from a channel if they no longer want to be notified of new videos."

Describe with text how the subscription subsystem could be structured with a design pattern. Include a rough bullet point list of the classes/interfaces that will be used, together with their attributes and operations (appropriately indented). Be sure to explictly name the pattern that you selected. (Instructor's note: This description would be more detailed on an actual exam - here, I am assuming everyone is familiar with YouTube.)



I would design this system with abstract factory pattern.