Sample EXAM

PROBLEM #1

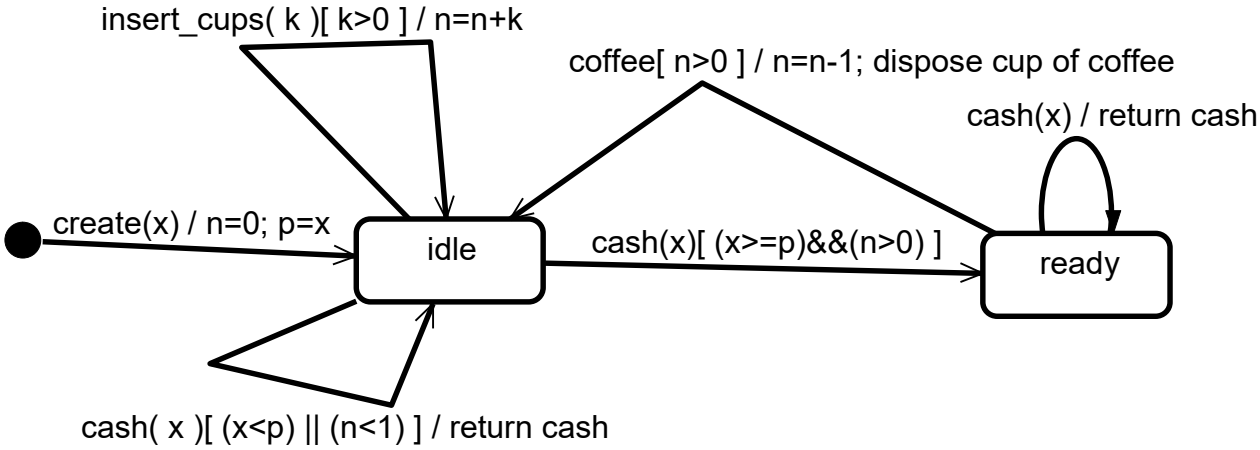
An EFSM (Extended Finite State Machine) of a component is shown below. The component supports the following operations: *create(float x), cash(float x), insert_cups(int k), coffee()*

Design the system using the **State design pattern**. You should use the **de-centralized** version of this pattern.

In your solution:

a. Provide a class diagram for the component. For each class list all operations with parameters and specify them using **pseudo-code**. In addition, for each class provide its attributes and data structures. Make the necessary assumptions for your design. Notice that the components in your design should be de-coupled as much as possible. In addition, components should have high cohesion.

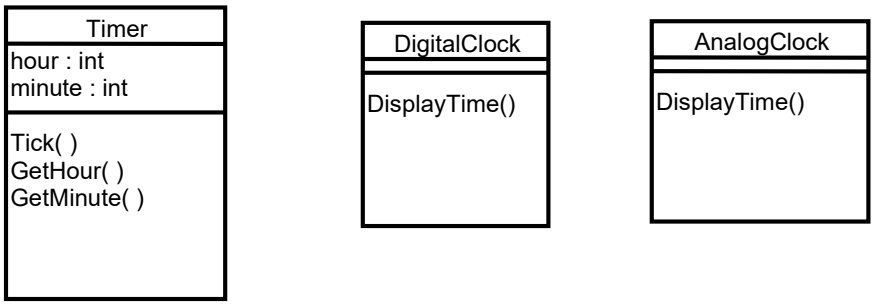
b. Provide a **sequence diagram** for the following operation sequence:
create(2.5), *insert_cups(10)*, *cash(3)*, *coffee()*



PROBLEM #2

In the system there exists a class *Timer* whose object stores and maintains the time of a day. This class supports the following operations: *Tick()*, *GetHour()*, and *GetMinute()*. The *Tick()* operation is called by an internal timer every 1 second. *Tick()* operation updates the *Timer*'s internal state (time data structure). Operations *GetHour()* and *GetMinute()* provide the interface for retrieving individual time units such as an hour and a minute.

In addition, there exist clock components in the system (e.g., *DigitalClock*, *AnalogClock*, etc.) that are responsible for displaying the time of the *Timer* component with a precision to a minute. Design a software subsystem using the **Observer** design pattern in which interested clock components can be updated about the current time of the *Timer* component.

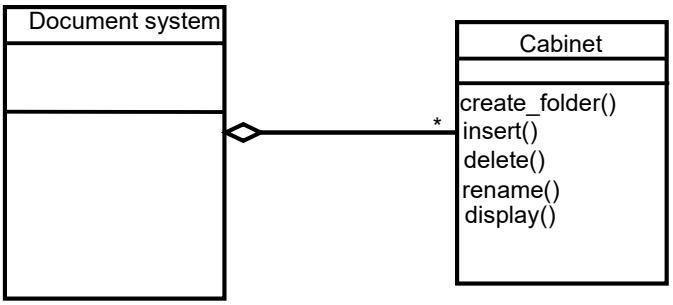


1. Provide a class diagram for the system. The class diagram should include classes *Timer*, *DigitalClock* and *AnalogClock* (if necessary introduce new classes and operations). In your design it should be easy to introduce new types of clock components (e.g., *AlarmClock*) that are interested in observing the time of the *Timer* component. For each class list operations and briefly describe their (operations) functionality. Notice that the components in your design should be de-coupled as much as possible. In addition, components should have high cohesion.

2. Provide a sequence diagram showing how the system notifies a registered digital clock and an analog clock about time change.

Note: Assume that the *Timer* and *Clocks* are in the same time zone and use the *24-hour time format* (e.g., 16:24).

PROBLEM #3
A document system consists of a set of cabinets as shown below:



Each cabinet contains folders and documents. A folder is a group of documents. In addition, a folder may contain another folders. Currently, the document system supports two types of documents: specification documents and design documents. In addition the system supports the following operations:

create_folder() – a folder is created
insert() – a document or a folder is inserted
delete() - a document or folder is deleted
rename() - a document or folder is renamed
display() - a document is displayed

Develop a class diagram for the document system using the **Whole-Part design pattern**.

Provide a class diagram for the document system. Identify operations and major attributes for each class. Identify necessary changes to the design when a new type of document is incorporated into the document system. Notice that required changes should be **minimal**.

Note: You do not have to specify the operations in classes.