Exercise 1 (2 points) Warm-up

Draw a Kripke model that represents the behavior of a simple microwave oven. The oven has a door which is either open or closed, and the oven can only be cooking if the door is closed. If the door is opened during operation, cooking stops automatically. The cooking mode can be set at anytime between thawing, heating or grilling.

(a) Define an appropriate set of propositional variables to the problem.
(b) Draw the Kripke model
(c) Can you enhance readability of the model by using a Finite State Machine? How?

Exercise 2 (4 points)
Consider the modal axiom (D) (□p → ◊p)

(a) Show that (D) is valid in all frames where the transition relation is serial: ∀x∃y ((x,y) ∈ R)
(b) (*) Show that the validity of (D) implies that the transition relation is serial.

Exercise 3 (2 points) Modelling the world wide web in modal logic

The world wide web consists of a large number of web pages, which are connected via hyperlinks. Assume that you have a homepage, which links to a page with your hobbies and to a page with your work projects. The page with your hobbies links back to the homepage, as well as to an outside page belonging to a club of which you are a member. Your work page links to the home page of your lab.

The following set of modal formulas could describe this situation.

(i) (homepage → (◊hobbypage ∧ ◊workpage))
(ii) (hobbypage → (◊homepage ∧ ◊clubpage))
(iii) (workpage → (◊workpage ∧ ◊labpage))

Questions you might be interested in asking about this structure include:

- Are there no “dead links”, i.e., does every page link to some other page?
- Is it possible to come back from any page to the homepage?

(a) Draw a picture of a Kripke model where the formulas (i) – (iii) are valid.
(b) Give formalizations of the two properties, and prove or disprove them on your model.

Exercise 4 (6 points) Modelling tools

a) Install Eclipse and Papyrus
b) Develop a model for a service multiplexer. The multiplexer mediates service between a client and two servers. When the multiplexer receives a request from a client, it forwards the request to both servers. Since both servers provide exactly the same service, they will always answer, and they will always answer with the same result. However, one can do so before the other one (for example, depending on the current server load). The servers always provide their answer to the multiplexer, and the multiplexer forwards the first answer it receives to the client, ignoring the second one. Keep in mind that:
   a. Only one client may have an active request at any time (HINT: model a unique client).
   b. The client should never receive duplicate answers, or answers to requests that were already finalized.

   c) Modify the model in b) so that now the multiplexer tells the second server to cancel work once it receives the first answer, to avoid useless work.
   d) Express the following properties in modal logic
      a. If a request was sent, it is necessary that the client receives an answer
      b. If a request was sent, it is necessary that the client receives a unique answer.
      c. If a request is sent, a second one is not possible before the answer is received.
   e) Extend the model in c) to many servers.