Exercise 1.  Recommended Reading (4 Marks)
Read the article “Can Programming be Liberated, Period?” by David Harel, IEEE Computer Society 41:1, 2008. [http://www.wisdom.weizmann.ac.il/~harel/papers/LiberatingProgramming.pdf](http://www.wisdom.weizmann.ac.il/~harel/papers/LiberatingProgramming.pdf) and answer the following questions. Use your own words and try to answer each question with no more than 200 words.

a. What characterizes reactive systems? What is the difference between inter-object behavior and intra-object behavior?

b. What is the analogy of the “good citizen” and executing a program? Can you think an example of unrealizable “pieces of behavior”?

Exercise 2.  Modal Sequence Diagrams. (6 Marks)
Figure 1 shows the Model Sequence Diagram PrepareCoffee for a coffee machine and its corresponding Büchi automaton, which accepts the same language as the MSD. The intuition behind this MSD is follows: When the user presses the button of the coffee machine (assume there is only one button), the controller must order the brewer unit of the coffee machine to boil the water. When the brewer unit then reports that the water has boiled, the controller must order the brewer unit prepare the coffee.

Assume that the coffee machine object system consists of the objects u, c, and bu (a somewhat incomplete coffee machine, actually). The lifelines in the MSD represent these objects. The alphabet, i.e., the set of possible message events in the object system is:

\[ \Sigma = \{(u, pressButton, c), (c, boilWater, bu), (bu, waterBoiled, c), (c, prepareCoffee, bu), (c, stopPrepareCoffee, bu), (bu, noWater, c), (bu, noCoffee, c)\} \]
We abbreviate the message events as follows:
\[ \Sigma = \{ pB, bW, wB, pC, sPC, nW, nC \} \]

Figure 1: MSD \textit{PrepareCoffee} and the corresponding Büchi automaton

Note that message events \( sPC, nW, nC \) cannot be unified with any message in the MSD \textit{PrepareCoffee}. Hence, the alphabet of the MSD \textit{PrepareCoffee} is \( \Sigma_{pc} = \{ pB, bW, wB, pC \} \). If in the Büchi automaton a transition arrow is labeled with a comma-separated list of message events, then this means that there is one transition for each event in the list. A transition arrow can also be labeled with a set of message events, e.g. \( \Sigma \setminus \Sigma_{pc} \) means that there is a transition for all the events in the set \( \Sigma \) without the elements of the set \( \Sigma_{pc} \), which is \( \Sigma \setminus \Sigma_{pc} = \{ sPC, nW, nC \} \). \( \Sigma \setminus pB \) means the set \( \Sigma \) without the element \( pB \).

Figure 2 shows the MSD \textit{PrepareCoffee2}, an extended version of the above MSD. The extension here describes that if the user presses the button again after the controller ordered the brewing unit to prepare the coffee, then the controller must order the brewing unit to stop preparing coffee.

Figure 2: MSD \textit{PrepareCoffee2}

\textbf{a.} State the set of message events \( \Sigma_{pc2} \) that can be unified with messages in the MSD \textit{PrepareCoffee2}.

\textbf{b.} Sketch a Büchi automaton that accepts the same language as the MSD \textit{PrepareCoffee2} with an iterative interpretation of the MSD.
c. Sketch a Büchi automaton that accepts the same language as the MSD PrepareCoffee2 with an invariant interpretation of the MSD.

*Hint: For drawing (Büchi) automata in LibreOffice/OpenOffice, there is a template on the lecture website.*