Some of these exercises are taken from Magee and Kramer’s book ‘Concurrency: State Models and Java Programs’, which contains further exercises for you to attempt. Some of the other exercises were prepared by Mark Jacobson, an ex-student of the School.

**Important**

Clearly, several of the answers to the exercises below can be found by utilising the available tools (LTSA, Java). However, you should do these exercises by hand first and then check using the LTSA tool.

Answers to exercises are to be found on the course website, BUT you should try the exercises and only use the answers to confirm your solutions, or if you are stuck.

**Try out your own examples:** As we progress with the course, please do not constrain yourself to the given exercises and examples alone — try out your own modelling, analysis and implementation examples, e.g.

- Does your third year project involve concurrency? If so, try building abstract models in FSP,
- Have you got your Threads tangled in the past? If you have encountered problems then try reconstructing the example and analyse it,
- Other course units,
- Applications of your own.

**Topic 2.1: Basic FSP Processes**

1. Make sure you have drawn the three ‘DAY’ LTSs, representing the actions of someone getting up and going to work:
   
   (a) **DAY1**: get up (action up), then have tea (action tea), then go to work (action work), then stop
   
   (b) **DAY2**: do DAY1 repeatedly
   
   (c) **DAY3**: do DAY2, but choose between tea and coffee

2. Write the FSP process definitions for the above. You can check these using the LTSA tool.

3. Extend DAY3 to include the effects of an alarm with a snooze button, so prior to the up action, an alarm action is performed. However instead of then doing up you may do a snooze action and go back to the start.
For the following, write an FSP specification and draw the corresponding LTS diagram. Check manually that your FSP and LTS correspond. Again, you can use LTSA tool to confirm this (LTSA may also be used to animate the specifications)

4. A variable stores values in the range $0..N$ and supports the actions read and write. Model the variable as a process, VARIABLE, using FSP.

For $N = 2$, check that it can perform the actions given by the trace:

$$\text{write}2\rightarrow\text{read}2\rightarrow\text{read}2\rightarrow\text{write}1\rightarrow\text{write}0\rightarrow\text{read}0$$

5. A bistable digital circuit receives a sequence of trigger inputs and alternately outputs 0 and 1. Model the process BISTABLE using FSP, and check that it produces the required output; i.e. it should perform the actions given by the trace:

$$\text{trigger} \rightarrow 1 \rightarrow \text{trigger} \rightarrow 0 \rightarrow \text{trigger} \rightarrow 1 \rightarrow \text{trigger} \rightarrow 0 \cdots$$

(Hint: The alphabet of BISTABLE is $\{[0],[1],\text{trigger}\}$.)

6. A sensor measures the water level of a tank. The level (initially 5) is measured in units $0..9$. The sensor outputs a low signal if the level is less than 2, a high signal if the level is greater than 8 and otherwise it outputs normal. Model the sensor as an FSP process, SENSOR.

7. A drinks dispensing machine charges 15p for a can of Sugarola. The machine accepts coins with denominations 5p, 10p and 20p and gives change. Model the machine as an FSP process, DRINKS.

8. A miniature portable FM radio has three controls. An on/off switch turns the device on and off. Tuning is controlled by two buttons scan and reset which operate as follows. When the radio is turned on or reset is pressed, the radio is tuned to the top frequency of the FM band (108 MHz). When scan is pressed, the radio scans towards the bottom of the band (88MHz). It stops scanning when it locks on to a station or it reaches the bottom (end). If the radio is currently tuned to a station and scan is pressed then it starts to scan from the frequency of that station towards the bottom. Similarly, when reset is pressed the receiver tunes to the top. Using the alphabet $\{\text{on}, \text{off}, \text{scan}, \text{reset}, \text{lock}, \text{end}\}$, model the FM radio as an FSP process, RADIO.