Exercise sheet 7 – Process synchronisation 2

Goals:

- Understand and solve advanced synchronisation issues

Exercise 7.1: Reader-Writer problem 1 (theoretical, pseudo C code)

Given is the following reader-writer pseudo C code. There exists one instance of the `writer` and 3 instances of the `reader`.

```c
1   //initialise semaphores
2
3
4
5   //writer and readers
6
7   writer() {                          27   reader() {
8     while(1) {                        28     while(1) {
9                                       29
10      data = collect_data();          30
11                                      31
12                                      32
13                                      33
14                                      34
15                                      35
16      write_data("data_file", data);  36       data = read_data("data_file");
17                                      37
18                                      38
19                                      39
20                                      40
21                                      41
22                                      42
23                                      43
24                                      44
25    }                                 45       work_with_data(data);
26  }                                   46     }
                                        47   }
```

(a) What is the problem here if nothing is synchronised?

(b) Solve the race condition with appropriate semaphores and a `num_readers` variable that counts the readers that are currently reading from the file.

(c) Who is preferred here, the readers or the writer?

Exercise 7.2: Reader-Writer problem 2 (theoretical, pseudo C code)

There are three processes `P1`, `P2`, `P3` which are periodically accessing the same file. There is an order for accessing the file: At first `P1` accesses the file. When it has finished, `P2` and `P3` should access the file at the same time. When both processes are finished with this, a new cycle starts and again `P1` accesses the file, and so on ... . Write pseudo C code which uses semaphores for accessing the file.

Exercise 7.3: Producer-Consumer problem (theoretical, pseudo C code)

Consider a hot day in summer: You have a butler, a mini refrigerator and an endless supply of beer. The problem is that the beer is warm and there is only space for one bottle in the refrigerator. If you take the beer from the refrigerator, the butler puts in a new bottle. The beer is cold enough when it was in the refrigerator for at least one hour. Only one person can access the refrigerator at the same time. Describe this procedure with pseudo C code and solve the synchronisation with semaphores! Tip: You need only 2 semaphores to solve this.

Exercise 7.4: Mutual exclusion/synchronisation problem (theoretical, pseudo C code)

There are three processes P1, P2, and P3 which are periodicly accessing the same file. There is an order for accessing the file: P1, then P2, and at last P3. When P3 has finished, a new cycle starts and again P1 accesses the file, and so on ... . Write pseudo C code which uses semaphores for accessing the file.