Mathematics and Computer Science
Technical University of Denmark
Building 324
DK–2800 Lyngby
Denmark

HHL 2024–10-21

02158 CONCURRENT PROGRAMMING FALL 2024

Exercise Class 3

Thursday October 24

Monitor Construction

Unless stated otherwise, you should assume (multiple) condition queues with *signal-and-continue* (SC) semantics and without spurious wakeups (ie. the standard semantics used in [Andrews]). Furthermore the predicate *empty*(c) and the function *length*(c) on a condition queue c can be used.

1. Do Exercise Mon.1 in [Aux].

2. Do Mon.4(a) (ie. write a monitor with two operations *sleep*() and *wakeup*() that implements the synchronization mechanism of Andrews Ex. 4.6).

[With the standard monitor semantics this is readily solved with a few lines of code. Right?. Compare with the semaphore solution given in the solutions for the Week 6 exercises.]

3. State some similarities and differences between semaphores and condition queues.

4. What happens if a `synchronized` Java method contains `while (!b) Thread.sleep(100);`?

5. Now, in Mon.1, the two operations $SYNC_A$ and $SYNC_B$ should be changed to functions that each takes an integer argument and returns the sum of the two arguments provided by process P_1 and process P_2 when they have met.

The signature of the two operations should thus be:

function $SYNC_{A/B}(x$: *integer*$)$ **returns** *integer*;

6. Do Mon.4(b)