DAT 103
Datamaskiner og operativsystemer (Computers and Operating Systems)

Supplementary exercises (Set 5)

Problem 1

Explain why spinlocks are not appropriate for single-processor systems yet are often used in multiprocessor systems.

Problem 2

Describe how the `signal()` operation associated with monitors differs from the corresponding operation defined for semaphores.

Problem 3

Describe how deadlock is possible with the dining-philosophers problem.

Problem 4

Let `l1` and `l2` be locks, and `x` a shared `int` variable initialised to `0`. Consider two processes P_1 and P_2 that concurrently run the following pieces of code:

```
1 // Run by P1 //
2 acquire(l1);
3 x++;
4 release(l1);
```

```
5 // Run by P2 //
6 acquire(l2);
7 x--;
8 release(l2);
```

List all possible values that `x` could have when both processes have finished.

Problem 5

Now, assume P_1 and P_2 concurrently run the two methods in Listing 1 to access their critical sections instead. Complete Listing 1 by filling in the boolean condition of the while-loops in Lines 8 and 18 such that it is a solution to the critical section problem, i.e., it guarantees both mutual exclusion and progress.

```
1  bool lock1 = false; bool lock2 = false;
2  int i;
3
4  //  Run by P1 //
5  while (true) {
6    lock1 = true;
7    i = 1;
8    while (???  // Fill in your code //) {skip} ;
9    ;; critical section
10   lock1 = false ;
11   ;; non critical section
12 }
13
14 //  Run by P2 //
15 while (true) {
16   lock2 = true;
17   i = 2;
18   while (???  // Fill in your code //) {skip} ;
19   ;; critical section
20   lock2 = false ;
21   ;; non critical section
22 }
```

Listing 1: