| Name | Signature | Student Id Nr | *Points* |
|---|---|---|---|
|  |  |  |  |

## Operating Systems, miniexam 3, 29.2.2016 (6p)

Write your answer on this exam paper in the space given. Please notice, that the exam paper is 2-sided.

a) [3 p] Monitor. Process P is implemented with 4 threads: A, B1, B2, and B3. Threads Bi will run their initialization code and then wait in (code position) Binit, until thread A has completed its own initialization and reached Ainit. Thread A will then wait in Ainit until all threads Bi have completed their calculations in Bdone. After this all threads will run into completion.
The synchronization problem between these processes is solved with monitor Sync.
The pseudocodes for processes A and Bi are

```
A                   B1, B2, B3
...                 ....
Sync.Ainit();       Sync.Binit();
...                 ...
                    Sync.Bdone();
                    ...
```

Give the pseudocode solutions for monitor Sync with methods Ainit(), Binit(), and Bdone(). By default, your monitor uses Hoare's *signal-and-wait* signaling semantics. If you want to use some other signaling semantics, mention it.

TURN

b) [1 p] There are many processes in a distributed system, and they all may want to do an update to three data bases in different servers in a critical section. How can you solve such critical section problem in distributed system?

c) [1 p] Operating systems threads P, Q, R and S can execute concurrently. They all use 7 critical regions (CS1-CS7). Usually, you only need one critical region at a time. However, sometimes with CS2 you also need regions CS1 or CS3. Also, sometimes with CS3 or CS6 you also need regions CS2 or CS5.
How can you prevent deadlocks in this case? Can you prove the correctness of your solution?
(The solution must allow that a process usually needs and reserves only one critical region for its use.)

d) [1 p] What data and what data structures are needed for the deadlock detection algorithm (DDA)?