Institut for Matematik og Datalogi
Syddansk Universitet

Assignment 2 — Introduction to Computer Science 2014

This is your second assignment in DM534. The assignment is due at 8:15 on Tuesday, October 21. You may write this either in Danish or English. It must be made in LaTeX. Write your full name, your section number, and your "instruktor"s name (Magnus Gausdal Find or Christian Kudahl) clearly on the first page of your assignment (on the top, if it's not a cover page). You should turn it in as a PDF file via Blackboard through your DM534 course. The assignment hand-in is in the menu for the course and is called "SDU Assignment". Choose the correct one for your section number, D1 or D2. Keep the receipt it gives you proving that you turned your assignment in on time. Blackboard will not allow you to turn in an assignment late.

Cheating on this assignment is viewed as cheating on an exam. You are allowed to talk about course material with your fellow students, but working together on this assignment is cheating. If you have questions about the assignment, come to Joan Boyar, Rolf Fagerberg or your "instruktor" for DM534.

Please note that you must have this assignment approved in order to pass DM534. If it is not turned in on time, or if you do not get it approved, it will count as one of your two retries in the course, and you must have it approved on your single allowed retry for this assignment. Recall that there are at most two retries allowed in all for assignments (the study start project does not count in this number).

Assignment 2

Do the following problems and write your solutions in LaTeX. (Do not include the statements of the problems or other information not asked for in the problems.

1. This problem is about critical regions and mutual exclusion. Consider a system where there are n processes running, all of which may possibly at one or more points need to execute code from the same critical region. Call the processes $P_0, P_1, ..., P_{n-1}$. Assume that they

can all update a number, S, either adding something to or subtracting something from S. Suppose the operations for updating S are Zero(S), which sets S to zero, Add(x, S), which adds x to S, and Subtract(x,S), which subtracts x from S. The following example demonstrated that there is a problem if no attempt has been made to ensure mutual exclusion:

- At time 0: Process P_0 executes Zero(S) and finishes.
- At time 1: Process P_1 starts executing Add(15,S), reads the value of S, but is interrupted is interrupted before adding.
- At time 2: Process P_2 starts executing Subtract(3,S) and finishes.
- At time 3: Process P_1 continues the interrupted execution of Add(15,S) and finishes.

At the end, the result will be 15 because process P_1 will add 15 to the value zero it reads, without noticing that process P_2 tried to subtract. The actions of process P_2 have no affect on the value of S. Note that the result should have been 12.

(a) Consider the following method for allowing mutual exclusion. Suppose there is a shared variable p which indicates which processor can update S, trying to arrange that the processes take turns updating S. Assume that p is initially set to zero. Process P_i may or may not want to update S. Each time it does want to update S, it executes the following code:

while $(p \neq i)$ **do skip**
Execute update to S
$p \leftarrow p + 1 \pmod{n}$

The **skip** operation does nothing, so that process P_i waits until $p = i$ before doing its update to S. The operation $\pmod{n}$ computes the remainder after division by n. Thus, $n \pmod{n} = 0$, and the counting starts over at zero after process P_{n-1} get its turn. Note that only one process at a time can update S, but there will be a problem in some cases. Explain in general what the problem is, and give an example where the problem would occur.

(b) Describe a solution using semaphores which does work. Write correct code for process P_i. State all of your assumptions.

(c) Can you get a problem if processes can "die" (due to some error) in the middle of executing the critical section? Why or why not?

2. On page 197, do problem 29.

3. Include your LaTeX code for this assignment at the end.