Operating Systems (8 cr), separate exam 20.1.2017 *Toisella puolella suomeksi*

Write in each answer sheet the course name, date, your name, signature and student id.
This exam is a separate (final) exam. Answer all questions. For each question, it is sufficient to give a 1-2 page answer.

1. [6 p] **Critical section problem**
   a. [2 p] What is the critical section problem? Who waits for whom and when?
   b. [2 p] Give a pseudocode level example on two code segment critical section problem. Give a scenario with erroneus end result, when critical section problem is not solved.
   c. [2 p] Give a semaphore solution to the critical section problem in previous part (b). Explain why your solution works also in the scenario you gave in part (b).

2. [6 p] **Deadlocks.**
   a. [2 p] Give a pseudocode level example where processes P, Q and R are deadlocked. Explain.
   b. [2 p] How can you detect that some processes have deadlocked. How can you recover from such detected deadlock?
   c. [2 p] You can prevent deadlocks from ever occurring by reserving the resources always in some given order. Why does this method work? Show, how this method would prevent the deadlock in your part (a) answer. What problem is there with this deadlock solution?

3. [6 p] **Memory management and virtual memory**
   a. [3 p] Which problem is solved with Buddy System, and how does is work?
   b. [3 p] Which problem does the Clock algorithm solve, and how does it work in main principles?

4. [6 p] **Real Time Scheduling**
   a. [3 p] How does the scheduling for real-time systems differ from that for normal systems?
   Why are FIFO (FCFS) or Round-Robin usually not good for real-time systems scheduling?
   b. [3 p] A real-time system needs to handle concurrently (i) two (voice) phone calls and (ii) one video stream. Each phone call runs every 5 msec and consumes 1 msec of CPU time per burst. The video stream has 25 frames/sec, with each frame requiring 15 msec of CPU time. These three processes are schedulable with RMS (rate monotonic scheduling). Explain why the processes are schedulable with RMS and what is the resulting schedule.

5. [6 p] **Disk management**
   a. [2 p] What problem does disk management algorithm SCAN (elevator) solve?
   In main principles, how does SCAN work?
   b. [2 p] Why is C-SCAN (Circular SCAN) better than SCAN?
   In main principles, how does C-SCAN work?
   c. [2 p] Which problem relating to C-SCAN is solved with Linux Deadline Scheduler?
   In main principles, how does Linux Deadline Scheduler work?

6. [6 p] **File managment**
   a. [4 p] What is an indexed file (IF) and how does it differ from indexed sequential file (ISF)? When should IF be used? In which case is IF slower than ISF? How many indexes are there with IF and how large are they?
   b. [2 p] Why would it be useful to implement the IF index with a B-tree? What advantage does a B-tree implementation has as compared to an ordinary (multi-level) index?