**Name:**
**No:**
**Signature:**

| Q1 | Q2 | Q3 | Q4 | Total |
|---|---|---|---|---|
| /30 | /30 | /20 | /20 | |

# Computer Operating Systems Midterm Exam

**Duration: 100 minutes** **No questions allowed.** **13.04.2016**

**Please note:**
* Put your answers in the spaces below the questions. You will NOT receive any points otherwise.
* You will NOT receive any points for answers which are not explained.
* Give all answers in English. You will NOT receive any points otherwise.
* You are not allowed to use any electronic equipment, books, notes, etc during the exam.

**Q1.** Briefly define the following.

(a) Busy waiting solutions

(b) Deadlock

(c) Process

(d) Critical section

(e) Process control block

(f) Deadlock avoidance technique

**Q2.**

(a) *(6 pts)* Define scheduling. What is “pre-emptive” and “non pre-emptive” scheduling. Explain.

(b) *(9 pts)* State whether each of the following scheduling methods are pre-emptive or non pre-emptive. Explain why.

(i) round robin

(ii) shortest job first

(iii) shortest time remaining first

(c) *(15 pts)* **Answer this question on the back of the second sheet.**

In a system with 5 processes assume that the process arrival times and the CPU times required by each process are as given below (times given as ut- unit of time) :

| Process | Arrival time | Required CPU time |
|---|---|---|
| P1 | 0 | 2 |
| P2 | 1 | 12 |
| P3 | 3 | 8 |
| P4 | 7 | 6 |
| P5 | 8 | 1 |

Multilevel queues with the maximum queue level as 3 is used for scheduling. At each level, the quantum (Q) assigned to each process at that level (v) is calculated based on the level as follows: $Q = 2^{v}$. Give the scheduling diagram with time as the x-axis. Show and explain all your steps.

**Name:**
**No:**
**Signature:**

**Q3.** Assume that an operating system does not support counting semaphores. We want to implement counting semaphores using binary semaphores and shared variables.
(a) Explain how this can be done.

(b) Write the pseudocodes for the P (wait) and V (signal) operations defined on counting semaphores implemented in this way.

**Q4.** Assume that two processes P1 and P2 run the following piece of code in parallel and there is no mutual exclusion. In this code x is a global variable shared by the processes and i is a local variable and thus is not shared.

```
x=0;
for (i=0; i<2; i++)
  x=x+2;
```

Give three different scenarios where x has a different final value when all processes finish execution. Explain each scenario.