Name:
No:
Signature:

Q1	Q2	Total
/50	/50	

08.04.2015

Computer Operating Systems Midterm Exam

(90 minutes)

Please note:

* No questions are allowed.
* Write your answers in the spaces provided below the questions. Answers written elsewhere will NOT receive any points.
* 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 answer the following questions.

a) *(5 pts)* Can a process make a transition from the "*ready*" state to the "*suspended*" state? Why or why not? Explain your answer.

b) *(10 pts)* Explain how binary semaphores can be used for (i) "*mutual exclusion*" and (ii) "*synchronization*". Also give an example usage for each.

c) *(5 pts)* What does it mean for a process to be "*I/O bound*"? What does it mean for it to be "*CPU bound*"?

d) *(5 pts)* Compare and contrast "*monolithic kernel*" design and "*modular kernel*" design (i.e. give definitions, advantages and disadvantages for both, explain when and why one should be preferred over the other, etc)

e) *(10 pts)* Consider the following line of code:

```
while (TRUE) fork();
```

What is the problem if you execute this as part of a program on a computer that runs a UNIX based operating system? Propose a solution to remedy this problem. Explain your answer.

Name:
No:
Signature:

f) *(10 pts)* Define "*busy waiting*"? In terms of using processor time, is "*busy waiting*" always less efficient than a "*blocking wait*"? Explain your answer.

g) *(5 pts)* Describe how "*system calls*" work.

Q2. Answer the questions based on the following problem description.

"Three kinds of threads/processes share access to a singly-linked list: "*searchers*", "*inserters*" and "*deleters*". "*Searchers*" merely examine the list; hence they can execute concurrently with each other. "*Inserters*" add new items to the end of the list; insertions must be mutually exclusive to prevent two inserters from inserting new items at about the same time. However, one insert can proceed in parallel with any number of searches. Finally, "*deleters*" remove items from anywhere in the list. At most one "*deleter*" can access the list at a time, and deletion must also be mutually exclusive with searches and insertions."

a) *(15 points)* Do you prefer to model the "*searchers*", "*inserters*" and "*deleters*" as processes or threads? Explain.

b) *(35 points)* Write the pseudocode for "*searchers*", "*inserters*" and "*deleters*". Use the model you chose in the first part of this question, i.e. either threads or processes.