

Operating Systems Comprehensive Examination
Fall Quarter, 2005

ID _____

NOTE: Please write your ID on every page of the exam.
Answer each question on its own page; you may use the back if you need extra space.
Answers on any other pages of the exam will NOT be graded.

Question	Score	Points
1		10
2		10
3		10
4		10
5		10
Total		50

1. (10 pts) **Monitors.** For the purposes of this question, assume a system with thread preemption.

- a) The Linux operating system implements the OS inside a big monitor (euphemistically called the "Big Kernel Lock" or BKL). Using the approach and terminology from the Mesa paper, describe what problem(s) this poses and how it can be avoided. Finally, discuss what new problem is presented by the Mesa solution to this problem. You don't need to know anything special about Linux to answer this question.

- b) Considering the following Mesa monitor code:

```
PROCEDURE Test ( )
  BEGIN
    SIGNAL ( c ) ;
  END
```

In a system with two different thread priorities, how might this construction be inefficient (i.e. waste CPU time)? However, in this particular case (SIGNAL is last statement in the monitor) the overhead could be easily eliminated. How? If there are three different thread priorities how might a medium-priority thread prevent a high-priority thread from running indefinitely without ever entering the monitor?

3. (10pts) **Synchronization.** A relational database consists of a set of tables. A transaction will access a set of these tables. A transaction that accesses a table T obtains an exclusive lock on that table. This lock must be held at the times the transaction accesses the table. For example, if a transaction X accesses tables T_1 , T_2 , and T_3 , then it will obtain an exclusive lock on each table T_1 , T_2 , and T_3 , and the lock for table T_i must be held at least at the times that the transaction accesses table T_i .

The two-phase locking protocol uses the following rule: once a transaction releases the lock on any table, it cannot acquire the lock for any other table. For example, once transaction X unlocks T_1 , X can't acquire the lock for any of the tables. Thus, a transaction using two-phase locking will first acquire locks (the first phase) and then release locks (the second phase).

The reason for this rule (it ensures serializability) is unimportant for this problem. You can assume that a transaction X knows at the beginning of its execution which tables it will access.

- a) Show, by example, that there is a two-phase locking protocol that can deadlock. Do not consider self-deadlocks, such as a transaction attempting to acquire a lock on a table that it has already locks.

- b) Do you expect that a common technique for a database to avoid deadlock is to use hierarchical locking? Explain your reasoning.

- c) Do you expect that a common technique for a database to avoid deadlock is to use the Banker's algorithm? Explain your reasoning.

4. (10pts) Structure of High Performance Services

- a)** Describe some of the relative advantages of using multi-threaded, multi-process, and event-driven network services. Consider issues of portability to multiple platforms, performance, and simplicity of the overall code. Please briefly justify your position on the relative merits of each approach.
- b)** What are some of the challenges and techniques for making a remote procedure call transparent? What role do idempotent operations play in this space? What about marshaling/unmarshaling arguments?

5. (10pts) **Mutual Exclusion.** Consider the following mutual exclusion algorithm. There are a set of threads each executing the same code. The variable i is a local variable, and names the thread. The variable x is shared by all the threads. The statement `delay(T)` causes the thread to delay execution for at least T milliseconds.

```
while (true) {
    do {
        while (x != 0) ;
        x = i;
        delay(T);
    } while ( x != i );
    critical section
    x = 0;
}
```

- a) Show, by giving an example, that if T is too small, then this algorithm will not implement mutual exclusion.

- b) How large must T be for this algorithm to ensure that mutual exclusion is provided? Explain.