Written Assignment #1

Due: May 5, 2006 (in lecture)

1. Consider the following code. Explain a scenario under which threads in the program will deadlock (i.e., make no progress).

```
public class Example
{
    private Object lock1, lock2, lock3;
        ...
    public void operationOne()
    {
        synchronized (lock1)
        {
            synchronized (lock2)
            {
                synchronized (lock3)
                {
                }
            }
        }
    }

    public void operationTwo()
    {
        synchronized (lock3)
        {
            synchronized (lock2)
            {
                synchronized (lock1)
                {
                }
            }
        }
    }
}
```

2. Java (as well as other systems) supports reentrant locks. This means that a thread that attempts to acquire a lock it already holds will not block on the second synchronization point. As such, the following code does not cause a thread invoking `doubleOperation` to block forever.

   (a) Rewrite the code such that the same synchronization guarantees are preserved, but in a style that does not rely on reentrant locks. Your solution may introduce code, but must not introduce duplicate code (i.e., do not violate "good" programming style simply to address this problem).

```
public class Example
{
    private int i;
    public synchronized void singleOperation()
    {
        ... some long complex operation ...
    }

    public synchronized void doubleOperation()
    {
        singleOperation();
        singleOperation();
    }
}
```

(b) Consider what must be done internally to support reentrant locks. Compare reentrant locks with non-reentrant locks.

3. If presented with a threading package, can you determine programmatically (i.e., write a program to do so) if the package supports preemptive or non-preemptive threads? Discuss how, or why not. Be sure to state any assumptions implicit in your argument.

4. In many of our discussions, it has been stated that one should attempt to implement stateless servers.

   (a) Name two advantages of stateless servers over stateful servers.

   (b) In lab #2, you were asked to implement a simple distributed lock server. Is it possible to make this server stateless?