Assignment 4
(**Due date: Thursday, 11/5/2009, in class**)

Your name:	Date:

Provide **brief** answers to the following Chapter Exercises questions**:**

6.3 (**no submission—check solutions from the course website**) Explain why Solaris, Windows XP, and Linux implement multiple locking mechanisms. Describe the circumstances under which they use spinlocks, mutexes, semaphores, and condition variables. In each case, explain why the mechanism is needed.

6.8 Race conditions are possible in many computer systems. Consider a banking system with the following two functions: *deposit(amount)* and *withdraw(amount).* These two functions are passed the amount that is to be deposited or withdrawn from a bank account. Assume a shared bank account exists between a husband and wife and concurrently the husband calls the *withdraw()* function and the wife calls *deposit().* Describe how a race condition is possible and what might be done to prevent the race condition from occurring.

6.11 What is the meaning of the term busy waiting? What other kinds of waiting are there in an operating system? Can busy waiting be avoided altogether? Explain your answer.

6.14 Explain why interrupts are not appropriate for implementing synchronization primitives in multiprocessor systems.

6.16 (**no submission**) Describe how the algorithm presented in Figure 6.8 (on page 233 of the textbook, or slide 6.17 of lecture slides) satisfy **3 requirements**: (1) mutual exclusion, (2) progress, and (3) bounded waiting. Describe how the Swap() instruction can be used to provide mutual exclusion that satisfies the bounded-waiting requirement. (This question will appear in **our final exam**).

6.17 Servers can be designed to limit the number of open connections. For example, a server may wish to have only *N* socket connections at any point in time. As soon as *N* connections are made, the server will not accept another incoming connection until an existing connection is released.
Explain how semaphores can be used by a server to limit the number of concurrent connections.

6.18 Show that, if the wait() and signal() semaphore operations are not executed atomically, then mutual exclusion may be violated.

6.23 Write a bounded-buffer monitor in which the buffers (portions) are embedded within the monitor itself.

6.29 A file is to be shared among different processes, each of which has a unique number. The file can be accessed simultaneously by several processes, subject to the following constraint: The sum of all unique numbers associated with all the processes currently accessing the file must be less than ***n***. Write a monitor to coordinate access to the file.

7.5 (**no submission—check solutions from the course website**) Consider a computer system that runs 5,000 jobs per month and has no deadlock-prevention or deadlock-avoidance scheme. Deadlocks occur about twice per month, and the operator must terminate and rerun about 10 jobs per deadlock. Each job is worth about $2 (in CPU time), and the jobs terminated tend to be about half-done when they are aborted.

A systems programmer has estimated that a deadlock-avoidance algorithm (like the banker's algorithm) could be installed in the system with an increase in the average execution time per job of about 10 percent. Since the machine currently has 30 percent idele time, all 5,000 jobs per month could still be run, although turnaround time would increase by about 20 percent on average.

a) What are the arguments for installing the deadlock-avoidance algorithm?
b) What are the arguments against installing the deadlock-avoidance algorithm?

7.7 (**no submission—check solutions from the course website**) Consider the following resource-allocation policy. Requests for and releases of resources are allowed at any time. If a request for resources cannot be satisfied because the resources are not available, then we check any processes that are blocked waiting for resources. If a blocked process has the desired resources, then these resources are taken away from it and are given to the requesting process. The vector of resources for which the blocked process is waiting is increased to include the resources that were taken away.

For example, consider a system with three resource types and the vector *Available* initialized to (4, 2, 2). If process P_0 asks for (2, 2, 1), it gets them. If P_1 asks for (1, 0, 1), it gets them. Then, if P_0 asks for (0, 0, 1), it is blocked (resource not available). If P_2 now asks for (2, 0, 0), it gets the available one (1, 0, 0) and one that was allocated to P_0 (since P_0 is blocked). P_0's *Allocation* vector goes down to (1, 2, 1), and its *Need* vector goes up to (1, 0, 1).

a) Can deadlock occur? If you answer "yes", give an example. If you answer "no", specify which necessary condition cannot occur.
b) Can indefinite blocking occur? Explain your answer.

7.8 (**no submission—check solutions from the course website**) Suppose that you have coded the deadlock-avoidance safety algorithm and now have been asked to implement the deadlock-detection algorithm. Can you do so by simply using the safety algorithm code and redefining $Max_i = Waiting_i + Allocation_i$, where $Waiting_i$ is a vector specifying the resources for which process I is waiting and $Allocation_i$ is as defined in Section 7.5? Explain your answer.

7.9 (**no submission—check solutions from the course website**) Is it possible to have a deadlock involving only a single process? Explain your answer.
No. This follows directly from the hold-and-wait condition.

7.11 Consider the deadlock situation that could occur in the dining-philosophers problem when the philosophers obtain the chopsticks one at a time. Discuss how the four necessary conditions for deadlock indeed hold in this setting. Discuss how deadlocks could be avoided by eliminating any one of the four conditions.

7.14 In a real computer system, neither the resources available nor the demands of processes for resources are consistent over long periods (months). Resources break or are replaced, new processes come and go; new resources are bought and added to the system. If deadlock is controlled by the banker's algorithm, which of the following changes can be made safely (without introducing the possibility of deadlock), and under what circumstances?

a. Increase Available (new resources added)
b. Decrease Available (resource permanently removed from system)
c. Increase Max for one process (the process needs more resources than allowed, it may want more)
d. Decrease Max for one process (the process decides it does not need that many resources)
e. Increase the number of processes
f. Decrease the number of processes

7.15 Consider a system consisting of four resources of the same type that are shared by three processes, each of which needs at most two resources. Show that the system is deadlock-free.

7.20 Consider the following snapshot of a system:

	Allocation	Max	Available
	ABCD	ABCD	ABCD
P0	0 0 1 2	0 0 1 2	1 5 2 0
P1	1 0 0 0	1 7 5 0	
P2	1 3 5 4	2 3 5 6	
P3	0 6 3 2	0 6 5 2	
P4	0 0 1 4	0 6 5 6	

Answer the following questions using the *banker's algorithm*:

a. What is the content of the matrix Need?

b. Is the system in a safe state?

c. If a request from process P1 arrives for (0,4,2,0), can the request be granted immediately?

7.22 A single-lane bridge connects the two Vermont villages of North Tunbridge and South Tunbridge. Farmers in the two villages use this bridge to deliver their produce to the neighboring town. The bridge can become deadlocked if both a northbound and a southbound farmer get on the bridge at the same time (Vermont farmers are stubborn and are unable to back up). Using semaphores, design an algorithm that prevents deadlock. Initially, do not be concerned about starvation (the situation in which northbound farmers prevent southbound farmers from using the bridge, and vice versa).

==========**Important Notes**==========

- Solutions must be **typewritten**. You can use lists, bullets for the write-up (short phrases are OK; complete sentences *not* necessary).
- Put all your solutions **in the same order** as the above questions.
- Use this question paper as **cover page** and **staple them together**.
- *Electronic* submissions will be accepted ***only under*** an excusable circumstances (the above rules still apply)—in this case, put your solutions including this question paper as cover page into ***ONE SINGLE*** Word or PDF file and send it to me via email. I'll *grade your work based on this file* and send back only your grade (without corrections of your errors).
- The full score for this homework is **100** points. **You will lose 5-10 points for missing ANY ONE of the followings in your submission:**
  - No name
  - No cover page
  - Your work submitted *not* in proper order
  - Not a single Word or PDF file (if submitted via email)

==========