CSE 332 – Operating Systems
A (Very) Extended List of Questions
Chapters 6-8

Chapter 6

1. Describe the critical section problem. Do not forget to mention shared resources. List and discuss 2 out of 3 requirements that a solution to the critical section problem must satisfy.

2. Assume that two threads run the following two functions, where wallet variable is shared. Show how context switches can produce conflict on the shared variable.

```
void buy(float amount) {

        float currentWallet = wallet;

        currentWallet = currentWallet-amount;

        wallet = currentWallet;


}
```

```
void sell(float amount) {

        float currentWallet = wallet;

        currentWallet = currentWallet+amount;

        wallet = currentWallet;

}
```

3. Assume that two threads run the following two functions, where noMilk and noNote variables are shared and initialized to true, and buyMilk updates the shared variable noMilk. Show how context switches can produce conflict on the shared variable.

```
if (noMilk) {

        if (noNote) {

                leaveNote()

                buyMilk()

                removeNote()

        }

}
```

```
if (noMilk) {

        if (noNote) {

                leaveNote()

                buyMilk()

                removeNote()

        }

}
```

4. Assume that two threads run the following two functions, where lock is a shared variable that is initialized with false, noMilk is a shared variable that is initialized to true, and buyMilk updates the shared variable noMilk. Show how context switches can produce conflict on the shared variable.

```
while (lock) {}

lock = true;

if (noMilk)


if (noMilk)

        buyMilk();

lock = false;
```

```
while (lock) {}

lock = true;

if (noMilk)


if (noMilk)

        buyMilk();

lock = false;
```

5. What is an atomic instruction in the context of process synchronization?

6. Fix the synchronization problem in questions 2 and 3 using the testAndSet approach.

7. Fix the synchronization problem in questions 2 and 3 using the swap approach.

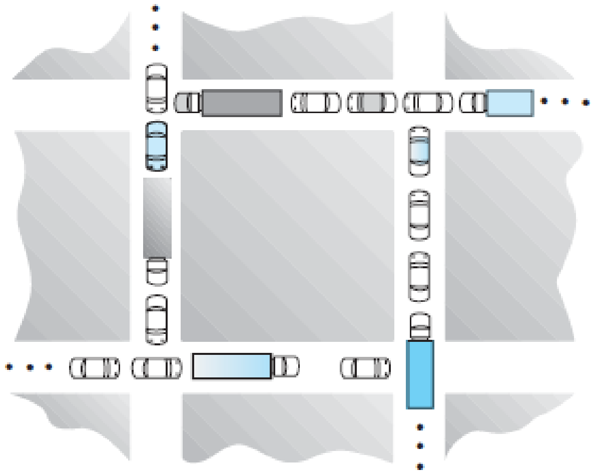
8. Why do we dislike busy waiting? How can starvation occur with preemptive priority scheduling and busy waiting?

9. Write a Semaphore class for a binary mutex including the wait and signal methods. You can describe adding and removing a process to and from a waiting list, sleeping a process, and waking up a process as comments in your code. Solve the synchronization problem of Problem 2 using your Semaphore class.

Chapter 7

10. List and discuss 3 out of 4 necessary conditions for deadlocks to occur.

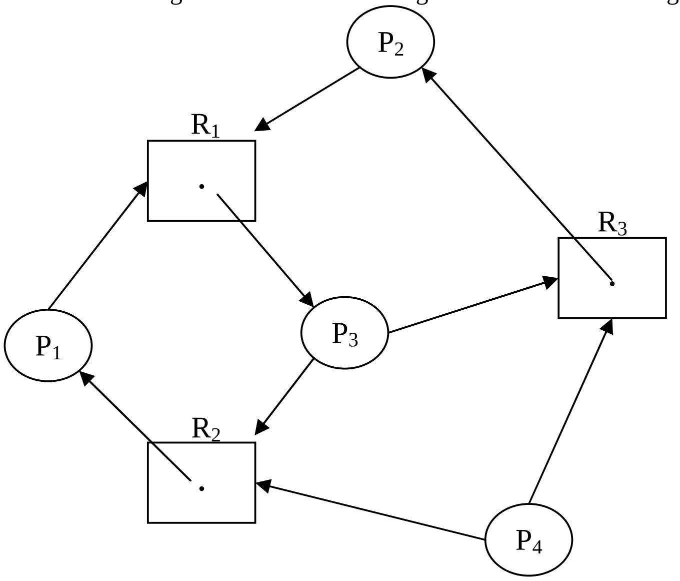
11. Consider the following traffic deadlock.



Show on this example, the four necessary conditions of deadlock exist.

12. Explain how we can avoid deadlocks by attacking the necessary conditions of deadlock.

13. For the following resource allocation diagram derive the wait-for graph and determine if there is a deadlock or not.



14. Given Process, Resource, Request and Allocation sets:
Proc = {P1, P2, P3}
Res = {R1, R2, R3, R4}
Req = {P1 → R1, P2 → R3, P2 → R2}
Alloc = {R1 → P2, R2 → P1, R3 → P3}

a. Draw the resource allocation graph.
b. Draw the wait-for graph.
c. Detect deadlocks, if any.

Chapter 8

15. What are the base and limit registers in memory management? What happens if a process attempts an illegal memory access?

16. Discuss compile time, load time and execution time memory binding.

17. What are logical and physical address spaces? How is the conversion from logical to physical address performed?

18. What is dynamic loading? What is its primary advantage?

19. What is dynamic linking? What are its advantages?

20. Discuss the strategies of contiguous memory allocation.

21. What is external memory fragmentation? How can we deal with it?

22. What is paging? Why do we use paging? Show on a sketch how the conversion from logical to physical address is done with paging.

23. Discuss the internal and external fragmentation with the paging memory management strategy.

24. What is the Translation-Look-Aside (TLB) buffer. Why is it useful?

25. Show on a sketch how the Translation-Look-Aside (TLB) buffer is used to translate a logical address to a physical address.

26. What is swapping? How is a process swapped with (a) compile time binding, and (b) execution time binding?