My Problems:

1. Suppose that it is time for the operating system to dispatch a process and that there are processes in both
the Ready state and the Ready, suspend state, and that at least one process in the Ready, suspend state has
higher scheduling priority than any of the processes in Ready state. Two extreme policies are as follows:
(1) Always dispatch from a process in the Ready state, to minimize swapping, and (2) always give
preference to the highest-priority process, even though that may mean swapping when swapping is not
necessary. Compare these policies as regards to concerns of priority and performance. Can you suggest
some other policy?

**Answer:** Ready state means less swapping, also fairer in that there will be no starvation. Ready suspend
means that the process needs to swap (take memory away from) another process (in the ready Q?) and thus
is more time consuming. It also may lead to starvation. It does guarantee that the highest priority processes
are executing.

2. Suppose that the operating system wishes to implement the two system calls SUSPEND and RESUME.
The SUSPEND call can be used by one process to suspend the execution of another process. (A process
cannot SUSPEND itself.) The suspended process remains suspended until it is RESUMED by the process
that originally suspended it. While a process is suspended it cannot run. What are some dangers of these
system calls?

**Answer:** What if the process is in the middle of its critical section and is updating a database? This would
cause damage to the database (inconsistent state) as well as potentially leading to deadlock.

3. What does it mean to preempt a process?

**Answer:** To take a resource away from a process. One such resource is the CPU, and in fact preempt often
means to move a process from RUNNING state to READY state. The process involuntarily gives up the
CPU.

4. What is swapping and what is its purpose?

**Answer:** To maximize the number of processes in the system, we swap a process from the ready state to
the ready suspend state (i.e. give its memory to another process)

5. On the LINUX machines in Fulton 160 (any of the machines will do), type in: ps –a and then type ps -A.
Describe what you see.

**Answer:** Information about the processes in the machine, PID, STATUS, OWNER, etc.

6. Our discussion of process states and the corresponding event queues maintained by the operating system
suggests that a process can only be in one event queue at a time.

   o Is it possible that you would want to allow a process to wait on more than one event at
     the same time? Provide an example.

   o In that case, how would you modify the queueing structure to support this new feature?

**Answer:**

Yes, it is possible that a process waits on more than one event at the same time. For example, a process
may need to transfer data from one device to another. In this case, it may request both devices at once and
wait until both are available for use. Another example is that a network application may wait on multiple
sockets until data packets arrive at any of them.
3.5 What is the purpose of the command interpreter? Why is it usually separate from the kernel?
Answer: It reads commands from the user or from a file of commands and executes them, usually by turning them into one or more system calls. It is usually not part of the kernel since the command interpreter is subject to changes.

3.12 What are the main advantages of the microkernel approach to system design?
Answer: Benefits typically include the following (a) adding a new service does not require modifying the kernel, (b) it is more secure as more operations are done in user mode than in kernel mode, and (c) a simpler kernel design and functionality typically results in a more reliable operating system.

3.13 What is the main advantage for an operating-system designer of using virtual-machine architecture? What is the main advantage for a user?
Answer: The system is easy to debug, and security problems are easy to solve. Virtual machines also provide a good platform for operating system research since many different operating systems may run on one physical system.

4.1 Palm OS provides no means of concurrent processing. Discuss three major complications that concurrent processing adds to an operating system.
Answer:
• A method of time sharing must be implemented to allow each of several processes to have access to the system. This method involves the preemption of processes that do not voluntarily give up the CPU (by using a system call, for instance) and the kernel being reentrant (so more than one process may be executing kernel code concurrently).
• Processes and system resources must have protections and must be protected from each other. Any given process must be limited in the amount of memory it can use and the operations it can perform on devices like disks.
• Care must be taken in the kernel to prevent deadlocks between processes, so processes aren’t waiting for each other’s allocated resources.

4.4 Describe the actions a kernel takes to context switch between processes.
Answer: In general, the operating system must save the state of the currently running process and restore the state of the process scheduled to be run next. Saving the state of a process typically includes the values of all the CPU registers in addition to memory allocation. Context switches must also perform many architecture-specific operations, including flushing data and instruction caches.

5.5 Describe the actions taken by a thread library to context switch between user-level threads.
Answer: Context switching between user threads is quite similar to switching between kernel threads, although it is dependent on the threads library and how it maps user threads to kernel threads. In general, context switching between user threads involves taking a user thread of its LWP and replacing it with another thread. This act typically involves saving and restoring the state of the registers.