1. Mark the issues (or non-issues) that are not issues for Interprocess Communication (IPC). As the MOS explains, this covers interthread communication and synchronization, etc.

- How can one process pass information to another?
- How can one thread pass information to another?
- The OS must guarantee that interregister transfer machine instructions, such as `movl %esp,%ebp` performed by one thread are not interfered with by another.
- Make sure two threads do not get into each other’s way.
- Proper sequencing when there are dependencies.
- When `exec` loads a program into virtual memory, all variables in the `.bss` section must be initialized to 0.

2. What describes a critical region best?

- The code or modules of an application that are most critical to its function.
- Pages that must be brought into memory first.
- A part of the program code where shared memory is accessed.
- A part of each thread that only that thread has access to, such as the CPU registers.
- A part of the program’s data structures that many threads can access at once.

3. What is an advantage to disabling interrupts on a single CPU system?

4. Does mutual exclusion by disabling interrupts work on multi-CPU or multicore CPU systems? Yes. No.

5. Is an ordinary shared memory variable used as a lock any good for mutual exclusion? Assume there are interrupts. Yes. No.

6. Will strict alternation work when the same process tries to enter and leave the critical section twice in succession without the other process entering and leaving it in-between? Yes. No.