**Q1:** Fill in the blanks with right word/words:

1. Any mechanism for controlling access of processes or users to resources defined by the OS is called ***Protection***
2. Multiprocessor environment must provide ***Cache coherency*** in hardware such that all CPUs have the most recent value in their cache
3. Some instructions designated as ***Privileged***, which only executable in kernel mode
4. Program is a ***Passive entity***, while process is an ***Active entity***
5. ***Timesharing*** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running.

**Q2:** Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU’s execution load. The CPU is allowed to execute other programs while the DMA controller is transferring data. Does this process interfere with the execution of the user programs? If so, describe what forms of interference are caused.

*Yes, it can interfere with the execution of other user programs. If a user program wants to modify some memory that is being used for DMA I/O, the program must be blocked until the DMA completes. In addition, since the I/O device is accessing memory, it can interfere directly with the CPU’s memory access, since both uses the same memory bus (in single channel memory, at least). This might cause the CPU to operate at a lower speed*.

**Q3:** Fill in the blanks with right word/words:

1. ***Security*** defense of the system against internal and external attacks.
2. Single-threaded process has one ***Program Counter*** specifying location of next instruction to execute.
3. ***Dual-mode*** operation allows OS to protect itself and other system components.
4. If processes don’t fit in memory, ***Swapping*** moves them in and out to run.
5. The occurrence of an event is usually signaled by an ***Interrupt*** from either the hardware or the software

**Q4:** Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU’s execution load. How does the CPU interface with the device to coordinate the transfer? Then how does the CPU know when the memory operations are complete?

*The CPU sets up a series of buffers in memory where the I/O device will copy the data to/from, along with a pointer or pointers to the buffers, and counters indicating the number of bytes to be transferred. It then notifies the I/O device to begin the transfer.*

*The I/O device interrupts the CPU.*