KUSHAGRA AGARWAL

2018113012

Q1> Explain the importance of the "timer" in OS.

Sol> To make sure that the OS maintains control over the CPU, we must make sure that a user program returns back the control. In cases of infinite loop or errors in system calls, by user programs, OS might never get back control. Therefore OS has timers which can be set to interrupt after a specific period. This period may be fixed/variable depending on the program. The OS sets the counter and with every clock tick, the counter is decremented. When the counter is 0, an interrupt occurs.

Instructions to modify the content of the timer should be privileged to protect it from misuse.

Q2> Why "time-sharing" was included in the OS?

Sol> OS should be able to multiprogram so that the CPU never sits idle. Time sharing (or multitasking) is just a logical extension of multiprogramming where the CPU can execute multiple jobs by switching so frequently among them that the user can interact with it while it is running.

A time-shared OS allows many users to share the computer simultaneously with the impression that the entire system is dedicated to their individual use.

Q3) Difference between "device controller" & "device driver".

Sol> Device controller → i) They are in charge of a specific type of device.

ii) They maintain a local buffer storage and set of special-purpose registers

iii) Responsibility to move the data b/w the peripheral devices and the local buffer it has.

iv) It is the electrical part of the I/O device

Device driver: i) They are device-controller dependent and OS specific

ii) It provides the rest of OS with a uniform interface to the device

iii) Device driver loads the appropriate registers within a device controller to start an I/O operation.

iv) Device driver returns control back to OS when operation finishes along with the status information.

Q4) Issues of "batch programming". How multiprogramming is better than batch programming?

Sol) Batch systems processed jobs in bulk, with predetermined input from files/data sources.
i) Computer operators needed training to use batch programming systems
ii) Debugging was difficult
iii) Cost was higher
iv) If error occurs in 1 job, then other jobs also wait for unknown times
v) CPU utilisation is not maximized

Multiprogramming does away with all these problems. A multiprogramming system is cheaper and in case one user program fails then suitable interrupt is created and the next process in the queue starts execution. User interaction was not at possible in batch systems but due to time-sharing optimisations, a multiprogramming systems can interact with user in real time.

Q5) What is an "operating system"?

Sol) An operating system is a program that manages computer's hardware and provides a basis for application programs and acts as an intermediary b/w the user and hardware. It is a resource allocator and a control program. It runs at all times (kernel). Loosely everything shipped as by a vendor when we order an "OS".