**Quiz 1 Review**

**Systems**

1. **Multiprogramming** – 1 CPU, many processes are ready for execution. In multiprogramming operating system, programs often in competition for resources such as memory, CPU, etc.
2. **Multiprocessing** – More than 1 CPU, many processes are ready for execution. Process shares the computer bus, the clock, memory and peripheral devices.
3. **Batch Processing** – a group of similar task are batched together as a group. The operating system’s major task was to automatically transfer from one job to another.
4. **Timesharing** **system** – A multiprogramming systems that support multiple terminals, one for each active of the system.
5. **Multimode** **System** – includes both, batch and timesharing process.
6. **Network Operating System** – Each computer has its own operating system. The user is aware that there are multiple independent computers and must deal with them explicitly. Communication is done by message passing.
7. **Distributed Operating System** – one operating system shared by a network of computers.
8. **Real Time System** – real time system has a fixed time constraint, process must be completely within a time limit.
9. **Hard Real Time** – Task must be done within a fixed time. Example, the Airbag system in the car requires the system to perform the task in a fixed time. Example, Robotic, Medical Field.
10. **Soft Real Time** – a critical real time task gets priority over other tasks. Example, sound system, video streaming, Multimedia.
11. **Embedded System** – a computer system with dedicated function that is built into a larger machine. Example, Television, portable game device, DVD, Printers

**Batch File –** An executable file that contains the commands (to the OS) executed with interaction. Examples, command prompt for windows.

**Computer Boot up**

1. POST: Power-On Self-Test
2. Hardware is checked
3. Go to **MBR (Master Boot Record)** – Location of full Bootstrap Process.
4. Bootstrap Process will initialize parts of the system (main memory, drivers)
5. Load the Kernel of the OS.
   1. **Kernel** – Most used part of the OS interface with Hardware. A computer program that manages input/output requests from software and translates them into data processing instructions for the CPU and other hardware of a computer.

**Bootstrap** **Program** – initializes all aspects of the system, which include locating the operating system kernel and loading it into memory. This program is stored in the ROM (read only memory).

**Synchronous Interrupt –** are produced by the CPU control unit while executing instructions and are called synchronous because the control unit issues them only after terminating the execution of an instruction. Example, overflow, underflow, illegal reference, and system call.

**Asynchronous Interrupt (from hardware, timer)** – Can occur in the middle of an instruction execution. Hardware interrupts such as mouse clicks or pressing a key on the keyboard. It will interrupt the operating system to read the keystrokes or mouse position.

**System Call** – how a program requests a service from an operating system’s kernel. Provides an interface to the services made available by an operating system.

**Process Control Block (PCB)** – OS takes control over CPU, saves minimal information: CPU REG. Check causes of the interrupt.

1. Interrupts have IDS
2. Interrupts have priorities
3. Only one interrupt can be served at a given time.

**Dual-Mode (Hardware)** – A mode bit is added to the hardware of computer to indicate the current mode. User-mode: 1, the operation is done on behalf of the user. System (monitor) – mode: 0, the operation is done on behalf of the OS.

**Privileged instructions –** Instructions that can be executed only in monitor mode. Examples, exit(), wait(), fork().

**Command Interpreter** – Interface between user & OS (Kernel)

**Terminate and Stay Resident** **(TSR)** – a computer system call in DOS(Disk Operating System) computer operating system that returns control to the system as if the program has quit, but keeps the program in memory to be revived later by a hardware or software interrupt. It is faster to access the applications and have small address space.

**Shell** – a command line interpreter for UNIX operating system and for UNIX like systems. Users direct the operation of the computer by entering commands as text for a command line interpreter to execute or by creating text scripts of one or more such commands. Shell runs in user mode in UNIX.

**Two types of shell**

1. **Internal (Built in)** – executable code of command is part of shell. Commands are loaded at the time of booting. Example, cd, pwd command
2. **External (most of command)** – outside shell as an executable file. External commands are not included as a part of command interpreter, they are a separate binary files kept in specific path. Command Interpreter runs this binary program whenever needed. Example, rmdir.

**Modern Architecture**

* **Layered Structure** – lowest is hardware, highest is user interface. Each layer can only use functions and services of lower level layer
* **Microkernel** – Kernel as small as possible. Most of the responsibilities are moved above kernel.

**Windows**

* **Microkernel** – First introduced in Windows NT. It functions as a message exchange: validates messages, passes them between components, grants access to the hardware, and performs a protection function.
* **HAL (Hardware Abstraction Layer)** – Isolates the OS from platform specific hardware differences. Provides the support for symmetric Multiprocessing. Most of the upper level modules can access the hardware only through the HAL.

**Process** – a program in execution. A program by itself is not a process.