Process: unit of activity characterized by execution of sequence of instructions, current state & associated set of system resources, consists of program code & associated data at least
Trace: behavior of individual process by listing sequence of instructions
Dispatcher: switches processor from one process to another
Process Creation: assigns identifier, allocates space, initializes process control block, sets appropriate linkages, creates or expands other data structures
Process States: New, Ready, Blocked, Ready/Suspend, Blocked/Suspend, Running, Exit; process creation state is special (not in memory yet, code not executed yet); suspended process may not be removed until agest (process that sent command to suspend, ex: itself, parent, OS) orders removal explicitly
OS Control Structures: memory tables (allocation of main & secondary memory, protection attributes, info needed to manage virtual memory), I/O tables (manage devices & channels, has status of operation & location in main memory used as source/destination of transfer), file table (existence of files, location on secondary memory, current status, info may be maintained & used by file management system), process tables (has reference to memory, I/O & files, tables themselves subject to memory management)
Process Image: program, data, stack, PCB
Process Control Block (PCB): most important data structure in OS, for protection only handler is allowed to read/write
PCB parts: Consists of Identifiers, processor state (user visible registers, control & status registers (program counter, condition codes, status), stack pointer (to the top)), process control (scheduling & state info (process state, priority, scheduling info, event (waiting for)), data structuring (if linked list has pointer to next process), inter process communication, privileges, memory management (pointer to segment & page tables), resource ownership & utilization)
Kernel Functions: process management (creation & termination, scheduling & dispatching, switching, synchronization & inter process communication support, process control blocks), memory management (allocation of address space, swapping, page & segment management), I/O management (buffer management, allocation of channels & devices), support functions (interrupt handling, accounting, monitoring)
Process Switch Steps: save processor context, update process control block of running one, move process control block to appropriate queue, select another process, update selected process control block, update memory management data structures, restore processor context for selected process
OS Execution Types: non-process kernel (user processes on top of kernel), execution with user processes (user processes all have OS functions, all on top of process switching functions), process based OS (OS functions are in separate processes alongside user processes, all above process switching functions)
Thread Parts: execution state, saved thread context when not running, execution stack (user stack plus kernel stack), some static storage for local variables, and access to shared memory and resources of process
Thread Uses: foreground & background work, asynchronous processing, speed of execution, modular program structure
Thread Types: User Level Thread (ULT), Kernel Level Thread (KLT)
ULT Advantages: no need for kernel mode privileges, app specific scheduling possible, OS independent
ULT Disadvantages: can block whole process on system call but thread itself is not blocked (thread blocked status is for waiting on another thread), no multiprocessing
KLT Advantages: opposite of ULT disadvantages, kernel routines can be multithreaded
KLT Disadvantages: thread switch requires switching to kernel mode