L2 Processes Threads

1. What is a process in an operating system?
a) A static entity stored on disk
b) A running program with an address space
c) A thread of control within a program
d) A collection of open files
Answer:

2. Which of the following is NOT part of a process's memory layout?
a) Code
b) Stack
c) Heap
d) Cache
Answer:

3. What does the `fork()` system call do?
a) Creates a new thread
b) Creates a new process by cloning the parent process
c) Terminates a process
d) Suspends the execution of a process
Answer:

4. Which system call replaces the current process image with a new one?
a) `fork()`
b) `wait()`
c) `exec()`
d) `exit()`
Answer:

5. What is the purpose of the `wait()` system call?
a) To create a new process
b) To wait for I/O operations to complete
c) To suspend the parent process until a child process terminates
d) To destroy a process
Answer:

6. What is stored in the Process Control Block (PCB)?
a) Process ID, state, and parent pointer
b) Only the stack pointer and program counter
c) The entire code and data of the process

d) Kernel threads only
Answer:

7. In which state is a process when it is waiting for an I/O operation to complete?
a) READY
b) RUNNING
c) BLOCKED
d) TERMINATED
Answer:

8. What does the `execvp()` function do?
a) Creates a new thread in user space
b) Executes a program specified by its path and arguments array
c) Waits for child processes to terminate
d) Suspends a thread
Answer:

9. What is the difference between `wait()` and `waitpid()`?
a) `waitpid()` allows specifying which child process to wait for, while `wait()` does not.
b) `wait()` waits for all processes, while `waitpid()` waits only for threads.
c) Both are identical in functionality.
d) `waitpid()` suspends processes, while `wait()` terminates them.
Answer:

10. What is the primary advantage of threads over processes?
a) Threads have separate address spaces.
b) Threads are cheaper to create and manage than processes.
c) Threads cannot share resources like processes can.
d) Threads only exist in kernel mode.
Answer:

11. What does multithreading allow in modern operating systems?
a) Multiple address spaces per thread
b) Concurrent execution within the same address space
c) Execution of only one thread at any time in a system
d) Elimination of kernel threads
Answer:

12. Which type of thread is managed entirely by user-level libraries?
a) Kernel threads
b) User-level threads

c) System threads
d) Lightweight kernel threads
Answer:

13. Which API function creates a new thread in POSIX systems?
a) ` pthread_exit()`
b) ` pthread_create()`
c) ` pthread_wait()`
d) ` pthread_signal()`
Answer:

14. What happens if no system call or trap occurs in cooperative multitasking?
a) The OS forcibly preempts the process.
b) The CPU remains idle until an interrupt occurs.
c) The running process continues to execute indefinitely.
d) The OS switches to kernel mode automatically.
Answer:

15. In user-level threading, what manages thread scheduling?
a) Operating system kernel
b) User-level thread library
c) Hardware interrupts
d) System calls
Answer:

16. Which of these is NOT true about kernel threads?
a) They are managed by the operating system kernel.
b) They are more expensive than user-level threads for fine-grained tasks.
c) They require no context switching overhead.
d) They allow overlapping I/O and computation inside a process.
Answer:

17. Why are user-level threads faster than kernel threads?
a) They bypass kernel involvement for common operations like creation and synchronization.
b) They have dedicated CPU cores assigned to them directly by hardware.
c) They do not require stack memory allocation during creation.
d) They avoid all forms of context switching overheads entirely.
Answer:

18. What does the term "process tree" refer to?

a) A hierarchical representation of all running threads within an OS
b) A structure showing parent-child relationships among processes
c) A list of all processes sorted by their priority levels
d) A binary tree used for scheduling algorithms
Answer:

19. What happens during a thread context switch at user level?
a) Kernel saves and restores hardware states like PC and SP using system calls.
b) Machine state is saved on one thread's stack and restored from another's stack by user code.
c) All open files are closed before switching contexts between two threads in the same process.
d) The OS allocates additional memory for each thread switch dynamically during runtime.
Answer:

20. Which command can be used to display all processes as a flat list in Linux?
a) `pstree`
b) `ps`
c) `top`
d)`lsproc`
Answer: