Sample Questions

Note: Don't get mislead by the number of questions here. These are samples only and there would be less or more questions in each category in the actual exam.

True or False:

Mode switching from kernel to user is triggered by interrupts, traps, and system calls.

Segmentation fault is not the type of events handled by the OS.

Interrupts can be enabled or disabled by user-level processes for synchronization purpose.

In Unix-like systems, a process is created by another process except for the very first process.

Exokernel follows the microkernel architecture.

In SPIN, user-level processes can directly access the kernel code.

With virtual memory, the page size may vary at runtime for spatial efficiency.

Non-preemptive scheduling cannot be used with priority-based scheduling.

Preemptive scheduling has less overhead than non-preemptive scheduling.

A binary semaphore provides mutual exclusion.

Single choice.

Which of the following is true about the OS kernel?

A. It executes as a process.

B. It is always executing in support of other processes.

C. It should execute as little as possible.

D. A & B

E. B & C

How many of the following statements is/are correct about paging?

A. Multi-level paging can reduce the average memory access time

B. Multi-level paging can reduce the runtime overhead of page fault handling

C. Multi-level paging can reduce the spatial overhead of page table allocation.

D. With multi-level paging, threads belonging to the same process use different page tables.

Consider the following code:

```
fork();
printf("hello\n");
fork();
printf("hello\n");
fork();
printf("hello\n");
```

How many hellos will be printed?

A. 3

B. 6

C. 8

D. 14

E. 24

Multiple choices: Choose all that are correct.

Copy-on-Write (CoW) in virtual memory:

A. CoW reduces the cost of fork().

B. CoW defers allocation of physical memory until a write attempt happens.

C. Shared pages among parent and child processes are read & write protected.

D. CoW requires MMU.

E. If two or more child processes are forked, only the first child benefits from CoW.

Scheduling:

A. Unix schedulers aim to minimize overall response time.

B. Starvation may occur in priority-based scheduling.

C. Global scheduling has only one ready queue for all processors.

D. Lottery scheduling achieves deterministic fairness.

E. Compensation tickets in lottery scheduling can solve the priority inversion problem.

(f) In stride scheduling, error is independent of allocation time.

Synchronization:

A. Spinlocks can be implemented without hardware support.

B. Using spinlocks in user-level processes is never a good idea because doing so dries up CPU time.

C. Semaphores are better than spinlocks for short critical sections.

D. Interrupt disabling cannot provide mutual exclusion for multiprocessors.

E. Locks implemented with the test-and-set instruction avoid starvation.

Short answers (in 1-2 sentences):

What is the difference between CPU-bound and I/O-bound processes?

User-level threads can provide higher performance than kernel-level threads. Why?

Faults and interrupts are both unexpected events, but faults are said to be synchronous. What is the meaning of "synchronous" here?

"Microkernels are inherently slow" -- Is this statement correct? Why or why not?

Longer answers:

Many modern OSes, e.g., Linux, MS Windows, and Mac OS, rely on the privileged mode provided by computer hardware to protect against erroneous/insecure user processes. Can we protect the OS without the privileged mode? Why or why not?

In the Exokernel paper, the authors implemented application-level stride scheduler and claimed that the results are close to "ideal" by showing the following plot. Discuss whether this argument holds or not when multiple libOSes compete for a single processor.

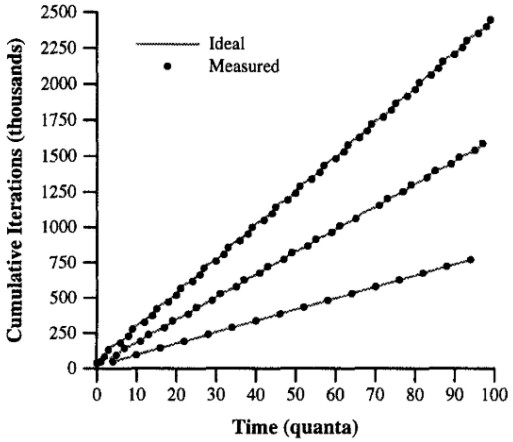

Figure 3: Application-level stride scheduler.