

CPU Scheduling



Review Questions

Section 5.1

- 5.1 What are the two bursts that CPU schedulers are designed around?
- 5.2 True or False? Under preemptive scheduling, when a process switches from the running to the ready state, it may lose control of the CPU.

Section 5.2

- 5.3 List at least three different criteria for designing a CPU scheduling algorithm.

Section 5.3

- 5.4 What scheduling algorithm assigns the CPU to the process with the highest priority?
- 5.5 True or False? The multilevel feedback queue scheduling algorithm allows processes to migrate between different queues.
- 5.6 What scheduling algorithm assigns the CPU to the process that first requested it?
- 5.7 What scheduling algorithm assigns the CPU to a process for only its time slice (or time quantum?)
- 5.8 What scheduling algorithm assigns the CPU to the process with the shortest burst?

Section 5.4

- 5.9 What are the two types of contention scope for thread scheduling?
- 5.10 What are the two general hardware instructions that can be performed atomically?

Section 5.5

- 5.11 What is more common on current systems, asymmetric or symmetric multiprocessing?
- 5.12 What are the two forms of processor affinity?
- 5.13 What are the two general approaches for load balancing?
- 5.14 What are the two ways to multithread a processing core?

Section 5.6

- 5.15 What are the two general types of real-time scheduling?
- 5.16 What real-time scheduling algorithm uses deadline as its scheduling criteria?
- 5.17 What real-time scheduling algorithm is used for scheduling periodic tasks with static priorities?

Section 5.7

- 5.18 What is the name of the default scheduling algorithm for current Linux systems?
- 5.19 True or False? A Windows thread is assigned both a priority class and a relative priority within that class.
- 5.20 If a thread on a Solaris system exhausts its time quantum, will it later be assigned a higher or lower priority?

Section 5.8

- 5.21 True or False? Deterministic modeling and simulations are similar strategies for evaluating scheduling algorithms.