Import Settings:
Base Settings: Brownstone Default
Highest Answer Letter: D
Multiple Keywords in Same Paragraph: No

Chapter: Chapter 5

Multiple Choice

1. Which of the following is true of cooperative scheduling?
A) It requires a timer.
B) A process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state.
C) It incurs a cost associated with access to shared data.
D) A process switches from the running state to the ready state when an interrupt occurs.

Ans: B
Feedback: 5.1.3
Difficulty: Medium

2. ____ is the number of processes that are completed per time unit.
A) CPU utilization
B) Response time
C) Turnaround time
D) Throughput

Ans: D
Feedback: 5.2
Difficulty: Medium

3. ____ scheduling is approximated by predicting the next CPU burst with an exponential average of the measured lengths of previous CPU bursts.
A) Multilevel queue
B) RR
C) FCFS
D) SJF

Ans: D
Feedback: 5.3.2
Difficulty: Medium

4. The ____ scheduling algorithm is designed especially for time-sharing systems.
A) SJF
B) FCFS
C) RR
D) Multilevel queue

Ans: C
Feedback: 5.3.4
Difficulty: Medium

5. Which of the following scheduling algorithms must be nonpreemptive?
A) SJF
B) RR
C) FCFS
D) priority algorithms

Ans: C
Feedback: 5.3.1
Difficulty: Medium

6. Which of the following is true of multilevel queue scheduling?
A) Processes can move between queues.
B) Each queue has its own scheduling algorithm.
C) A queue cannot have absolute priority over lower-priority queues.
D) It is the most general CPU-scheduling algorithm.

Ans: B
Feedback: 5.3.5
Difficulty: Medium

7. The default scheduling class for a process in Solaris is ____.
A) time sharing
B) system

C) interactive
D) real-time

Ans: A
Feedback: 5.7.3
Difficulty: Easy

8. Which of the following statements are false with regards to the Linux CFS scheduler?
A) Each task is assigned a proportion of CPU processing time.
B) Lower numeric values indicate higher relative priorities.
C) There is a single, system-wide value of `vruntime`.
D) The scheduler doesn't directly assign priorities.

Ans: C
Feedback: 5.7.1
Difficulty: Easy

9. The Linux CFS scheduler identifies ______________ as the interval of time during which every runnable task should run at least once.
A) virtual run time
B) targeted latency
C) `nice` value
D) load balancing

Ans: B
Feedback: 5.7.1
Difficulty: Medium

10. In Little's formula, λ, represents the ____.
A) average waiting time in the queue
B) average arrival rate for new processes in the queue
C) average queue length
D) average CPU utilization

Ans: B
Feedback: 5.7.2
Difficulty: Medium

11. In Solaris, what is the time quantum (in milliseconds) of an interactive thread with priority 35?
A) 25
B) 54
C) 80
D) 35

Ans: C
Section: 5.7.3
Difficulty: Easy

12. In Solaris, if an interactive thread with priority 15 uses its entire time quantum, what is its priority recalculated to?
A) 51
B) 5
C) 160
D) It remains at 15

Ans: B
Feedback: 5.7.3
Difficulty: Easy

13. In Solaris, if an interactive thread with priority 25 is waiting for I/O, what is its priority recalculated to when it is eligible to run again?
A) 15
B) 120
C) 52
D) It remains at 25

Ans: C
Feedback: 5.7.3
Difficulty: Easy

14. ______ allows a thread to run on only one processor.
A) Processor affinity
B) Processor set
C) NUMA
D) Load balancing

Ans: A

Feedback: 5.5.2
Difficulty: Medium

15. What is the numeric priority of a Windows   thread in the NORMAL_PRIORITY_CLASS with HIGHEST relative priority?
A) 24
B) 10
C) 8
D) 13

Ans: B
Feedback: 5.7.2
Difficulty: Easy

16. What is the numeric priority of a Windows   thread in the HIGH_PRIORITY_CLASS with ABOVE_NORMAL relative priority?
A) 24
B) 10
C) 8
D) 14

Ans: D
Feedback: 5.7.2
Difficulty: Easy

17. What is the numeric priority of a Windows   thread in the BELOW_NORMAL_PRIORITY_CLASS with NORMAL relative priority?
A) 6
B) 7
C) 5
D) 8

Ans: A
Feedback: 5.7.2
Difficulty: Easy

18. __________ involves the decision of which kernel thread to schedule onto which CPU.
A) Process-contention scope

B) System-contention scope
C) Dispatcher
D) Round-robin scheduling

Ans: B
Feedback: 5.4.1
Difficulty: Easy

19. With _______ a thread executes on a processor until a long-latency event (i.e. a memory stall) occurs.
A) coarse-grained multithreading
B) fine-grained multithreading
C) virtualization
D) multicore processors

Ans: A
Feedback: 5.5.4
Difficulty: Medium

20. A significant problem with priority scheduling algorithms is _____.
A) complexity
B) starvation
C) determining the length of the next CPU burst
D) determining the length of the time quantum

Ans: B
Feedback: 5.3.3
Difficulty: Medium

21. The ______ occurs in first-come-first-served scheduling when a process with a long CPU burst occupies the CPU.
A) dispatch latency
B) waiting time
C) convoy effect
D) system-contention scope

Ans: C
Feedback: 5.3.1
Difficulty: Medium

22. The rate of a periodic task in a hard real-time system is ____, where *p* is a period and *t* is the processing time.
A) 1/*p*
B) *p/t*
C) 1/*t*
D) *pt*

Ans: A
Section: 5.6.2
Difficulty: Medium

23. Which of the following is true of the rate-monotonic scheduling algorithm?
A) The task with the shortest period will have the lowest priority.
B) It uses a dynamic priority policy.
C) CPU utilization is bounded when using this algorithm.
D) It is non-preemptive.

Ans: C
Section: 5.6.3
Difficulty: Difficult

24. Which of the following is true of earliest-deadline-first (EDF) scheduling algorithm?
A) When a process becomes runnable, it must announce its deadline requirements to the system.
B) Deadlines are assigned as following: the earlier the deadline, the lower the priority; the later the deadline, the higher the priority.
C) Priorities are fixed; that is, they cannot be adjusted when a new process starts running.
D) It assigns priorities statically according to deadline.

Ans: A
Section: 5.6.4
Difficulty: Medium

25. The two general approaches to load balancing are __________ and ____________.
A) soft affinity, hard affinity
B) coarse grained, fine grained

C) soft real-time, hard real-time
D) push migration, pull migration

Ans: D
Section: 5.5.3
Difficulty: Medium

Essay

26. Distinguish between coarse-grained and fine-grained multithreading.

Ans: There are two approaches to multithread a processor. (1) Coarse-grained multithreading allows a thread to run on a processor until a long-latency event, such as waiting for memory, to occur. When a long-latency event does occur, the processor switches to another thread. (2) Fine-grained multithreading switches between threads at a much finer-granularity, such as between instructions.
Feedback: 5.5.4
Difficulty: Medium

27. Explain the concept of a CPU–I/O burst cycle.

Ans: The lifecycle of a process can be considered to consist of a number of bursts belonging to two different states. All processes consist of CPU cycles and I/O operations. Therefore, a process can be modeled as switching between bursts of CPU execution and I/O wait.
Feedback: 5.1.1
Difficulty: Medium

28. What role does the dispatcher play in CPU scheduling?

Ans: The dispatcher gives control of the CPU to the process selected by the short-term scheduler. To perform this task, a context switch, a switch to user mode, and a jump to the proper location in the user program are all required. The dispatch should be made as fast as possible. The time lost to the dispatcher is termed dispatch latency.
Feedback: 5.1.4
Difficulty: Medium

29. Explain the difference between response time and turnaround time. These times are both used to measure the effectiveness of scheduling schemes.

Ans: Turnaround time is the sum of the periods that a process is spent waiting to get into memory, waiting in the ready queue, executing on the CPU, and doing I/O. Turnaround time essentially measures the amount of time it takes to execute a process. Response time, on the other hand, is a measure of the time that elapses between a request and the first response produced.
Feedback: 5.2
Difficulty: Medium

30. What effect does the size of the time quantum have on the performance of an RR algorithm?

Ans: At one extreme, if the time quantum is extremely large, the RR policy is the same as the FCFS policy. If the time quantum is extremely small, the RR approach is called processor sharing and creates the appearance that each of *n* processes has its own processor running at 1/*n* the speed of the real processor.
Feedback: 5.3.4
Difficulty: Medium

31. Explain the process of starvation and how aging can be used to prevent it.

Ans: Starvation occurs when a process is ready to run but is stuck waiting indefinitely for the CPU. This can be caused, for example, when higher-priority processes prevent low-priority processes from ever getting the CPU. Aging involves gradually increasing the priority of a process so that a process will eventually achieve a high enough priority to execute if it waited for a long enough period of time.
Feedback: 5.3.3
Difficulty: Difficult

32. Explain the fundamental difference between asymmetric and symmetric multiprocessing.

Ans: In asymmetric multiprocessing, all scheduling decisions, I/O, and other system activities are handled by a single processor, whereas in SMP, each processor is self-scheduling.
Feedback: 5.5.1
Difficulty: Medium

33. Describe two general approaches to load balancing.

Ans: With push migration, a specific task periodically checks the load on each processor and —if it finds an imbalance—evenly distributes the load by moving processes from overloaded to idle or less-busy processors. Pull migration occurs when an idle processor pulls a waiting task from a busy processor. Push and pull migration are often implemented in parallel on load-balancing systems.
Feedback: 5.5.3
Difficulty: Medium

34. In Windows, how does the dispatcher determine the order of thread execution?

Ans: The dispatcher uses a 32-level priority scheme to determine the execution order. Priorities are divided into two classes. The variable class contains threads having priorities from 1 to 15, and the real-time class contains threads having priorities from 16 to 31. The dispatcher uses a queue for each scheduling priority, and traverses the set of queues from highest to lowest until it finds a thread that is ready to run. The dispatcher executes an idle thread if no ready thread is found.
Feedback: 5.7.2
Difficulty: Difficult

35. What is deterministic modeling and when is it useful in evaluating an algorithm?

Ans: Deterministic modeling takes a particular predetermined workload and defines the performance of each algorithm for that workload. Deterministic modeling is simple, fast, and gives exact numbers for comparison of algorithms. However, it requires exact numbers for input, and its answers apply only in those cases. The main uses of deterministic modeling are describing scheduling algorithms and providing examples to indicate trends.
Feedback: 5.8.1
Difficulty: Medium

36. What are the two types of latency that affect the performance of real-time systems?
Ans: Interrupt latency refers to the period of time from the arrival of an interrupt at the CPU to the start of the routine that services the interrupt. Dispatch latency refers to the amount of time required for the scheduling dispatcher to stop one process and start another.
Section: 5.6.1
Difficulty: Medium

37. What are the advantages of the EDF scheduling algorithm over the rate-monotonic scheduling algorithm?
Ans: Unlike the rate-monotonic algorithm, EDF scheduling does not require that processes be periodic, nor must a process require a constant amount of CPU time per burst. The appeal of EDF scheduling is that it is theoretically optimal - theoretically, it can schedule processes so that each process can meet its deadline requirements and CPU utilization will be 100 percent.
Section: 5.6.4
Difficulty: Medium

True/False

38. In preemptive scheduling, the sections of code affected by interrupts must be guarded from simultaneous use.

Ans: True
Feedback: 5.1.3
Difficulty: Medium

39. In RR scheduling, the time quantum should be small with respect to the context-switch time.

Ans: False
Feedback: 5.3.4
Difficulty: Medium

40. The most complex scheduling algorithm is the multilevel feedback-queue algorithm.

Ans: True
Feedback: 5.3.6
Difficulty: Medium

41. Load balancing is typically only necessary on systems with a common run queue.

Ans: False
Feedback: 5.5.3
Difficulty: Medium

42. Systems using a one-to-one model (such as Windows, Solaris , and Linux) schedule threads using process-contention scope (PCS).

Ans: False
Feedback: 5.4.1
Difficulty: Easy

43. Solaris and Windows assign higher-priority threads/tasks longer time quantums and lower-priority tasks shorter time quantums.

Ans: False
Feedback: 5.7
Difficulty: Medium

44. A Solaris interactive thread with priority 15 has a higher relative priority than an interactive thread with priority 20

Ans: False
Feedback: 5.7.3
Difficulty: Easy

45. A Solaris interactive thread with a time quantum of 80 has a higher priority than an interactive thread with a time quantum of 120.

Ans: True
Feedback: 5.7.3
Difficulty: Easy

46. SMP systems that use multicore processors typically run faster than SMP systems that place each processor on separate cores.

Ans: True
Feedback: 5.5.4
Difficulty: Easy

47. Windows 7 User-mode scheduling (UMS) allows applications to create and manage thread independently of the kernel

Ans: True
Feedback: 5.7.2
Difficulty: Medium

48. Round-robin (RR) scheduling degenerates to first-come-first-served (FCFS) scheduling if the time quantum is too long.

Ans: True
Feedback: 5.3.4
Difficulty: Easy

49. Load balancing algorithms have no impact on the benefits of processor affinity.

Ans: False
Feedback: 5.5.3
Difficulty: Medium

50. A multicore system allows two (or more) threads that are in compute cycles to execute at the same time.

Ans: True
Feedback: 5.5.4
Difficulty: Easy

51. Providing a preemptive, priority-based scheduler guarantees hard real-time functionality.

Ans: False
Section: 5.6
Difficulty: Difficult

52. In hard real-time systems, interrupt latency must be bounded.

Ans: True
Section: 5.6.1
Difficulty: Medium

53. In Pthread real-time scheduling, the SCHED_FIFO class provides time slicing among threads of equal priority.

Ans: False
Section: 5.6.6
Difficulty: Medium

54. In the Linux CFS scheduler, the task with smallest value of `vruntime` is considered to have the highest priority.

Ans: True
Section: 5.7.1
Difficulty: Medium

55. The length of a time quantum assigned by the Linux CFS scheduler is dependent upon the relative priority of a task.

Ans: False
Section: 5.7.1
Difficulty: Medium

56. The Completely Fair Scheduler (CFS) is the default scheduler for Linux systems.

Ans: True
Section: 5.7.1
Difficulty: Medium