Answers - Operating System - Sample Questions

1	B
2	D
3	D
4	A
5	D
6	A
7	D
8	D
9	A
10	D
11	C
12	A
13	B
14	A
15	B
16	C
17	C
18	B
19	C
20	A
21	D
22	D
23	B
24	C
25	A
26	B

27	False
28	False
29	False
30	False
31	True
32	True
33	False
34	False
35	True
36	False
37	False
38	True
39	False
40	True
41	False
42	True
43	A bootstrap program is the initial program that the computer runs when it is powered up or rebooted. It initializes all aspects of the system, from CPU registers to device controllers to memory contents. Typically, it is stored in read-only memory (ROM) or electrically erasable programmable read-only memory (EEPROM), known by the general term firmware, within the computer hardware.
44	Physical memory is the memory available for machines to execute operations (i.e., cache, random access memory, etc.). Virtual memory is a method through which programs can be executed that requires space larger than that available in physical memory by using disk memory as a backing store for main memory. Logical memory is an abstraction of the computer's different types of memory that allows programmers and applications a simplified view of memory and frees them from concern over memory-storage limitations.
45	System programs are not part of the kernel, but still are associated with the operating system. Application programs are not associated with the operating of the system.
46	DMA is efficient for moving large amounts of data between I/O devices and main memory. It is considered efficient because it removes the CPU from being responsible for transferring data. DMA instructs the device controller to move data

	between the devices and main memory.
47	A large reason why it is more efficient is that communication between processors on the same chip is faster than processors on separate chips.
48	The possible states of a process are: new, running, waiting, ready, and terminated. The process is created while in the new state. In the running or waiting state, the process is executing or waiting for an event to occur, respectively. The ready state occurs when the process is ready and waiting to be assigned to a processor and should not be confused with the waiting state mentioned earlier. After the process is finished executing its code, it enters the termination state.
49	The primary distinction between the two schedulers lies in the frequency of execution. The short-term scheduler is designed to frequently select a new process for the CPU, at least once every 100 milliseconds. Because of the short time between executions, the short-term scheduler must be fast. The long-term scheduler executes much less frequently; minutes may separate the creation of one new process and the next. The long-term scheduler controls the degree of multiprogramming. Because of the longer interval between executions, the long-term scheduler can afford to take more time to decide which process should be selected for execution.
50	The differences between the two types of processes stem from the number of I/O requests that the process generates. An I/O-bound process spends more of its time seeking I/O operations than doing computational work. The CPU-bound process infrequently requests I/O operations and spends more of its time performing computational work.
51	Whenever the CPU starts executing a new process, the old process's state must be preserved. The context of a process is represented by its process control block. Switching the CPU to another process requires performing a state save of the current process and a state restore of a different process. This task is known as a context switch. When a context switch occurs, the kernel saves the context of the old process in its PCB and loads the saves context of the new process scheduled to run.
52	The benefits of multithreaded programming fall into the categories: responsiveness, resource sharing, economy, and utilization of multiprocessor architectures. Responsiveness means that a multithreaded program can allow a program to run even if part of it is blocked. Resource sharing occurs when an application has several different threads of activity within the same address space. Threads share the resources of the process to which they belong. As a result, it is more economical to create new threads than new processes. Finally, a single-threaded process can only execute on one processor regardless of the number of processors actually present. Multiple threads can run on multiple processors, thereby increasing efficiency.
53	A parallel system can perform more than one task simultaneously. A concurrent system supports more than one task by allowing multiple tasks to make progress.
54	In a solution to the critical section problem, no thread may be executing in its critical section if a thread is currently executing in its critical section. Furthermore, only

	those threads that are not executing in their critical sections can participate in the decision on which process will enter its critical section next. Finally, a bound must exist on the number of times that other threads are allowed to enter their critical state after a thread has made a request to enter its critical state.
55	The scenario involves five philosophers sitting at a round table with a bowl of food and five chopsticks. Each chopstick sits between two adjacent philosophers. The philosophers are allowed to think and eat. Since two chopsticks are required for each philosopher to eat, and only five chopsticks exist at the table, no two adjacent philosophers may be eating at the same time. A scheduling problem arises as to who gets to eat at what time. This problem is similar to the problem of scheduling processes that require a limited number of resources.