CSC 301 - Exam 2 (Chapter 7) Study Guide

1. **Describe a scenario when using a reader-writer lock is more appropriate than anor synchronization tool such as a semaphore.**	- A tool such as a semaphore only allows one process to access shared data at a time - Reader-writer locks are useful when it is easy to distinguish if a process is only reading or reading/writing shared data. If a process is only reading shared data, it can access shared data concurrently with or readers. In case when re are several readers, a reader-writer lock may be much more efficient.
2. **Describe a wait-for graph and how it detects deadlock.**	- If all resources have only a single instance, n we can define a deadlock-detection algorithm that uses a variant of resourceallocation graph, called a wait-for graph. We obtain this graph from resource-allocation graph by removing resource nodes and collapsing appropriate edges. To detect deadlocks, system needs to maintain wait-for graph and periodically invoke an algorithm that searches for a cycle in graph.
3. **Describe four conditions that must hold simultaneously in a system if a deadlock is to occur.**	For a set of processes to be deadlocked: at least one resource must remain in a nonsharable mode, a process must hold at least one resource and be waiting to acquire additional resources held by or processes, resources in system cannot be preempted, and a circular wait has to exist between processes.
4. **Describe how a safe state ensures deadlock will be avoided.**	- A safe state ensures that re is a sequence of processes to finish ir program execution - Deadlock is not possible while system is in a safe state. - However, if a system goes from a safe state to an unsafe state, deadlock is possible - One technique for avoiding deadlock is to ensure that system always stays in a safe state - This can be done by only assigning a resource as long as it maintains system in a safe state.
5. **Describe two methods for eliminating processes by aborting a process.**	- first method is to abort all deadlocked processes - Aborting all deadlocked processes will clearly break deadlock cycle; however, deadlocked processes may have to be computed for a long time, and results of se partial computations must be discarded and will probably have to be recomputed later. second method is to abort one process at a time until deadlock cycle is eliminated - aborting one process at a time incurs considerable overhead, since, after each process is aborted, a deadlock-detection algorithm must be invoked to determine wher any processes are still deadlocked.
6. **Describe two protocols to ensure that hold-and-wait condition never occurs in a system.**	One protocol requires each process to request and be allocated all its resources before it begins execution. We can implement this provision by requiring that system calls requesting resources for a process precede all or system calls. An alternative protocol allows a process to request resources only when it has none. A process may request some resources and use m. Before it can request any additional resources, however, it must release all resources that it is currently allocated.
7. **Explain what has to happen for a set of processes to achieve a deadlocked state.**	For a set of processes to exist in a deadlocked state, every process in set must be waiting for an event that can be caused only be anor process in set. Thus, processes cannot ever exit this state without manual intervention.
8. **Name three issues that need to be addressed if a preemption is required to deal with deadlocks**	- First, order of resources and processes that need to be preempted must be determined to minimize cost. - Second, if a resource is preempted from a process, process must be rolled back to some safe state and restarted from that state - simplest solution is a total rollback - Finally, we must ensure that starvation does not occur from always preempting resources from same process.
9. **What are three general ways that a deadlock can be handled?**	A deadlock can be prevented by using protocols to ensure that a deadlock will never occur. A system may allow a deadlock to occur, detect it, and recover from it. Lastly, an operating system may just ignore problem and pretend that deadlocks can never occur.

10. **What does a claim edge signify in a resource-allocation graph?**	A claim edge indicates that a process may request a resource at some time in future. This edge resembles a request edge in direction, but is represented in graph by a dashed line.
11. **What factors influence decision of when to invoke a detection algorithm?**	- first factor is how often a deadlock is likely to occur; if deadlocks occur frequently, detection algorithm should be invoked frequently. - second factor is how many processes will be affected by deadlock when it happens; - if deadlock-detection algorithm is invoked 16 for every resource request, a considerable overhead in computation time will be incurred.
12. **What is difference between deadlock prevention and deadlock avoidance?**	Deadlock prevention is a set of methods for ensuring that at least one of necessary conditions for deadlock cannot hold. Deadlock avoidance requires that operating system be given, in advance, additional information concerning which resources a process will request and use during its lifetime.
13. **What is one way to ensure that a circular-wait condition does not occur?**	One way to ensure that this condition never holds is to impose a total ordering of all resource types, and to require that each process requests resources in an increasing order of enumeration. This can be accomplished by assigning each resource type a unique integer number to determine wher one precedes anor in ordering.