7.12 Assume a multithreaded application uses only reader —writer locks for synchronization. Applying the four necessary conditions for deadlock, is deadlock still possible if multiple reader—writer locks are used?

Answer: Yes.

- Mutual Exclusion is intact as a lock cannot be shared if there is a writer.
- Hold-and-Wait could happen if a writer is holding a lock.
- No Preemption happens because you cannot take a lock away.
- Circular Wait is still possible with a writer holding a lock.

7.19 Consider the version of the dining-philosophers problem in which the chopsticks are placed at the center of the table and any two of them can be used by a philosopher. Assume that requests for chopsticks are made one at a time. Describe a simple rule for determining whether a particular request can be satisfied without causing deadlock given the current allocation of chopsticks to philosophers.

Answer: If only one philosopher can make a request at a time then don't satisfy the request if there are two philosophers with two chopsticks already or there's only one chopstick left.

7.22 Answer:

Process	Allocation ABCD	Max ABCD
P0	3014	5117
P1	2210	3211
P2	3121	3321
P3	0510	4612
P4	4212	6325

Answer:

a) **Available = (0, 3, 0, 1)**
No, because when do P2, P1, P3 we have an Available of (5, 11, 4, 2) which is not enough to do either P0 or P4 because D will not suffice for either process.

b) Yes, (P1, P2, P0, P3, P4)

7.23

a) Order of Processes is: (P0, P3, P4, P1, P2)
b) No, because even though the resources are available, the resulting state would be unsafe.
c) No, because even though the resources are available, the resulting state would be unsafe.