* System Programming and Operating System (SPOS) - Assignment Number - 5

Name :- Kaustubh Shrikant Kabra.
Class :- Third Year Engineering
Div :- A
Roll Number :- 38.
Batch :- T-2
Department :- Computer Department
College :- AISSMS's IOIT.

★ Explain the following concurrency management techniques.
1. Reader Writer Problem
2. Producer Consumer Problem
3. Dining Philosopher Problem

Also explain the following Deadlock strategies
1. Deadlock Prevention
2. Deadlock avoidance
3. Deadlock Detection

→ Reader Writer Problem -

The reader-writers problem related to an objects such as a file that is shared between multiple processes. Some of these processes are reader i.e. they only want to read the data from the object and some of the processes are writers. i.e. they want to write into the object.

The reader-writer problem related to is used to manage synchronization so that there are no problems with the object data.

Producer - Consumer Problem -

The producer - consumer problem is a classical multi-process synchronization problem, that is we are trying to achieve synchronization between more that one process.

There is one Producer in the producer - consumer problem, Producer is producing some items, whereas there is one Consumer that is consuming the items produced by the Producer. The same memory buffer is shared by both producers and consumers which is of fixed size.

Dining Philosophers Problem -

The dining philosophers's problem is the classical problem of synchronization which says that five philosophers are sitting around a circular table and their job is to think and eat alternatively. A bowl of noodles is placed at the center of the table along with five chopsticks for each of the philosophers. To eat a philosopher needs both their right chopstick and a left chopsticks. A philosopher can only eat if both immediate left and right chopstick of the philosopher is available. In case if both immediate left and right chopsticks of the philosopher are not available. then the philosopher puts down their chopstick and starts thinking again.

Deadlock Prevention -

If we simulate deadlock with a table which is standing on its four legs then we can also simulate four legs with four conditions which occurs simultaneously, cause deadlock.

However, if we break one of the legs of the table then the table will fall

definitely. The same happens with deadlock, if we can be able to violate one of the four necessary conditions and dont let them occur together then we can prevent the deadlock.

Deadlock Avoidance-

In deadlock avoidance, the request for any resource will be granted if the resulting state of the system doesn't cause deadlock in the system.

In order to avoid deadlock, The process must tell OS, the maximum number of resources a process can request to complete its execution.

Deadlock Detection-

The main task of the OS is detecting the deadlocks. The OS can detect the deadlock with the help of Resource allocation graph.

In single instanced resource types, if a cycle is being formed in the system then there will definitely be a deadlock. On other hand, in multiple instanced resource type graph, detecting a cycle is not just enough. We have to apply the safety algorithm on the system by converting the resource allocation graph into the allocation matrix and request matrix.