Operating System

PAANA ACADEMY

Advantages of concurrency

It happens in the operating system when there are several process threads running in parallel.

There are several motivations for allowing concurrent execution

- **Physical resource Sharing:** Multiuser environment since hardware resources are limited
- **Logical resource Sharing:** Shared file (same piece of information)
- **Computation Speedup:** Parallel execution
- **Modularity:** Divide system functions into separation processes

Types of process

The Processes executing in the operating system is one of the following two types:

- Independent Processes
- Cooperating Processes

Independent Processes

Its state is not shared with any other process.

- The result of execution depends only on the input state.
- The result of the execution will always be the same for the same input.
- The termination of the independent process will not terminate any other.

Types of process

Cooperating System

Its state is shared along other processes.

- The result of the execution depends on relative execution sequence and cannot be predicted in advanced(Non-deterministic).
- The result of the execution will not always be the same for the same input.
- The termination of the cooperating process may affect other process.

Critical Section

The critical section represents the segment of code that can access or modify a shared resource.

There can be only one process in the critical section at a time.

The critical section contains shared variables or resources that need to be synchronized to maintain the consistency of data variables.

Synchronization mechanisms allow the processes to access critical section in a synchronized manner to avoid the inconsistent results.

Properties of Process Synchronization

Although there are some properties that should be followed if any code in the critical section

1. **Mutual exclusion**: If process Pi is executing in its critical section, then no other processes can be executing in their critical sections.
2. **Progress:** If no process is executing in its critical section and some processes wish to enter their critical sections, then only those processes that are not executing in their remainder sections can participate in deciding which will enter its critical section next, and this selection cannot be postponed indefinitely.
3. **Bounded Waiting:** There exists a bound, or limit, on the number of times that other processes are allowed to enter their critical sections after a process has made a request to enter its critical section and before that request is granted

Race Condition

Race condition is a situation where-

- The final output produced depends on the execution order of instructions of different processes
- Several processes compete with each other.
- A race condition is a situation that may occur inside a critical section.

Mutex

Mutex is a specific kind of binary semaphore that is used to provide a locking mechanism.

It stands for Mutual exclusion Object.

- No race condition arises, as only one process is in the critical section at a time.
- Data remains consistent and it helps in maintaining integrity.
- It’s a simple locking mechanism that can be obtained by a process before entering into a critical section and released while leaving the critical section.

Semaphores

A semaphore is a non-negative integer variable that is shared between various threads.

Semaphore works upon signaling mechanism, in this a thread can be signaled by another thread.

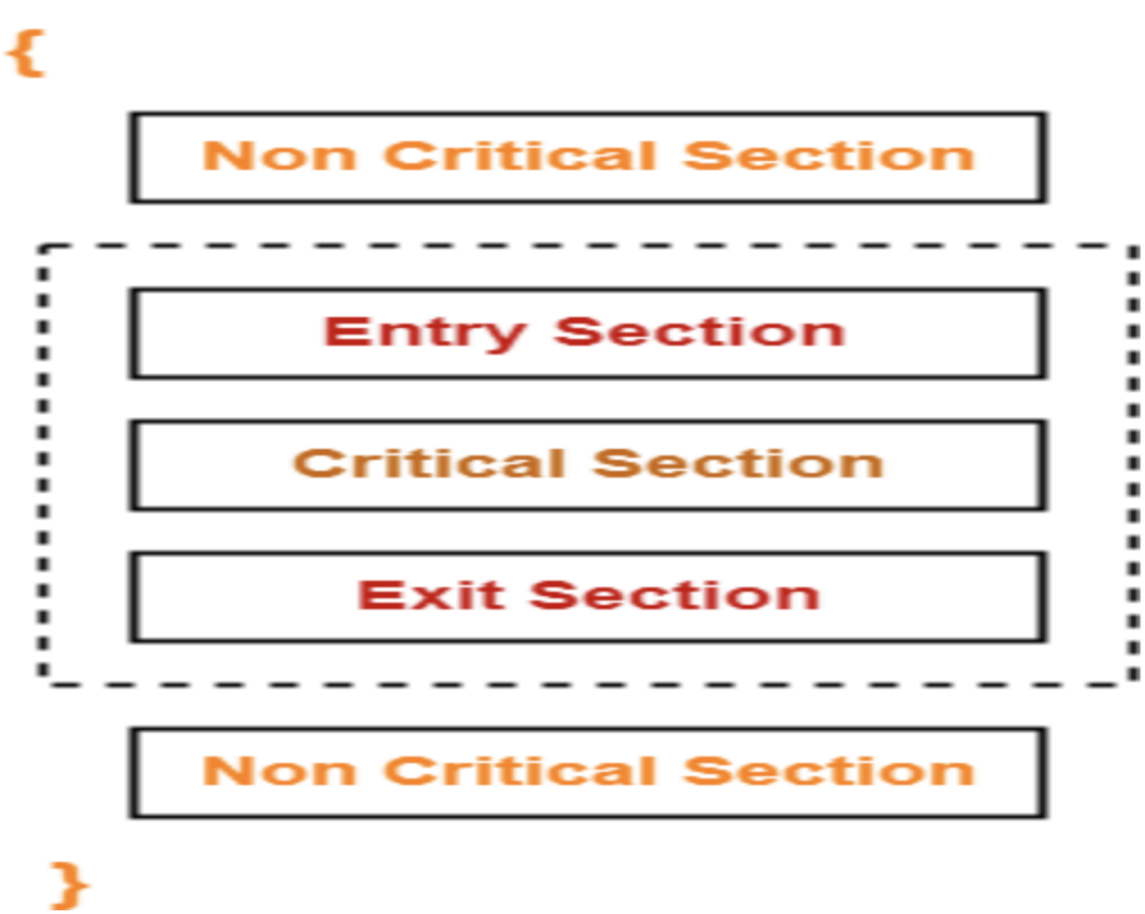
Semaphore uses two atomic operations for process synchronisation:

- Wait (P)
- Signal (V)

- Semaphores are machine-independent.
- Only one process will access the critical section at a time, however, multiple threads are allowed.

Code of wait and signal

```
wait(A)

{

while (A<=0);

A–;

}

signal(A)

{

A++;

}
```


Process
{
Non Critical Section
Entry Section
Critical Section
Exit Section
Non Critical Section
}

Message Passing

Message Passing provides a mechanism to allow processes to communicate and to synchronize their actions without sharing the same address space.

Message passing is a time consuming process because it is implemented through kernel (system calls).

In message passing the communication is slower when compared to shared memory technique.

Classical Problem of Synchronization

1. Bounded-buffer (or Producer-Consumer) Problem
2. Dining-Philosophers Problem
3. Readers and Writers Problem
4. Sleeping Barber Problem

Monitors

higher-level synchronization construct

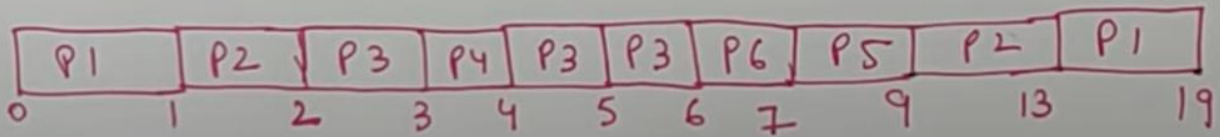
providing a high-level abstraction for data access and synchronization.

are implemented as programming language constructs, typically in object-oriented languages,

Example of SJF

Pre-emptive Mode

Process	Arrival Time	Burst Time
P1	0	7 - 6
P2	1	5 - 4 X
P3	2	3 - 2 - X
P4	3	1 X
P5	4	2 X
P6	5	1 X



C.T − A.T

Turnaround Time

P1 = 19 − 0 = 19

P2 = 13 − 1 = 12

P3 = 6 − 2 = 4

P4 = 4 − 3 = 1

P5 = 9 − 4 = 5

P6 = 7 − 5 = 2

$$\frac{19+12+4+1+5+2}{6} = \frac{43}{6} = 7.1$$

https://www.sanfoundry.com/operating-system-questions-answers-cpu-scheduling/

https://www.sanfoundry.com/operating-system-mcqs-cpu-scheduling-benefits/

https://www.sanfoundry.com/operating-system-mcqs-cpu-scheduling-algorithms-1/

https://www.sanfoundry.com/operating-system-mcqs-critical-section-problem/