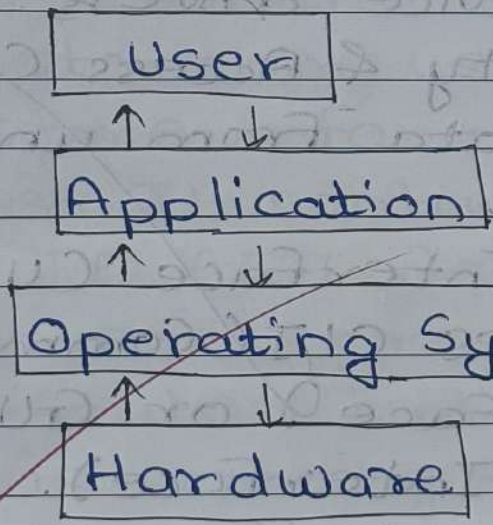
Assignment -1

Q.1. Define the Operating System.

Ans. An Operating System (OS) is system software that manages computer hardware and software resources, providing services for computer programs. It acts as an interface between users and the hardware, ensuring efficient resource allocation and system functionality.



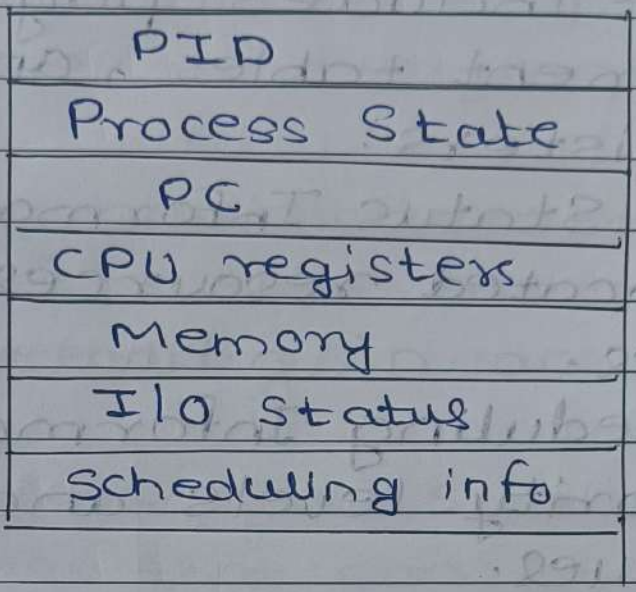
Q.2. What are the various main functions of an OS?

Ans. The main functions of an Operating System include:

1. Process Management - Controls the execution of processes, scheduling and resource allocation.
2. Memory Management - Manages primary memory allocation for processes.
3. File System Management - Organizes, stores, retrieves, and secures files.
4. Device Management - Manages input/output (I/O) devices via device drivers.
5. Security & Access Control - Protects data from unauthorized access.
6. User Interface (UI) Management - Provides CLI (Command-Line Interface) or GUI (Graphical User Interface).
7. Networking - Enables communication between devices over a network.
8. Error Detection & Handling - Identifies and responds to system errors.

Q.3. Describe the structure of the Process Control Block (PCB).

Ans.



The Process Control Block (PCB) is a data structure used by the OS to store information about a process. It contains:

1. Process ID (PID) - Unique identifier to each process.
2. Process State - Indicates if the process is running, ready, waiting, etc.
3. Program Counter (PC) - Stores the address of the next instruction.
4. CPU Registers - Holds value for

process execution.

5. Memory management information - Includes page table, segment tables, and base/limit registers.
6. I/O Status Information - Lists allocated resources and open files.
7. Scheduling Information - Contains priority levels and scheduling queues.

Q.4. What do you understand about Context Switching?

Ans. Context Switching is the process of saving the state of a running process and loading the state of another process. It allows multitasking by switching the CPU between multiple process efficiently. It involves.

1. Saving the current process state (PCB update).
2. Loading the new process state

3. Resuming execution of the new process - This process ensures fair CPU utilization but incurs overhead due to frequent switching.

Q.5. What do you understand about threads? Explain the concept of multithreading in an Operating System.

Ans. A thread is the smallest unit of a process that can be executed independently. It shares process resources like memory but has its own stack and registers. Multithreading refers to the execution of multiple threads within a single process. It improves performance by allowing parallel execution of tasks within the same program.

Types of Multithreading:

1. User-Level Threads - Managed by user applications, not the OS.

2. Kernel-Level Threads - Managed directly by the OS.

Benefits of Multithreading:
- Improved CPU utilization
- Faster execution of tasks
- Better system responsiveness

Q.6. Differentiate Multitasking and Multiprocessing.

Ans.

Multitasking	Multiprocessing
1. Running multiple processes on a single CPU using time-sharing	Running multiple processes using multiple CPUs
2. Processes share a single CPU by time slicing	Each process run on a separate CPU.
3. Less efficient than multiprocessing due to context switching.	More efficient as mutiple CPUs work simultaneously.
4. Ex. Running a web browser and	Ex. Running a database server

media player simultaneously on a single-core CPU	and web server on different CPUs.

Assignment - 2

Q. 1. Define Process Scheduling in Operating Systems, explain its importance and types.

Ans. Process scheduling is a mechanism by which an operating system (OS) decides the order in which processes are executed by the CPU. It ensures efficient CPU utilization and system responsiveness.

Importance:

- Maximizes CPU efficiency
- Reduces waiting time for process
- Ensures fairness among processes
- Improves system responsiveness
- Support multitasking.

Types of Scheduling:

1. Long-term scheduling: Decides which processes are admitted into the system for execution.
2. Short-term scheduling: Determines which process runs on the CPU next (CPU scheduler)

3. Medium-term scheduling: Temporarily removes processes from memory (swapping) to optimize resource usage.

4. Preemptive scheduling: Allows the OS to interrupt a running process to allocate CPU to another process

5. Non-preemptive scheduling: A process keeps the CPU until it completes or voluntarily releases it

Q.2. Explain and discuss the following scheduling criteria:

a) CPU Utilization - The percentage of time the CPU is actively executing processes. Higher utilization means better performance.

b) Throughput - The number of processes completed per unit time. Higher throughput is desirable.

c) Turnaround Time - The total time

taken for a process from submiss ion to completion. Lower turnarou nd time is better.

d) Waiting Time - The time a process spending waiting in the ready queue. Lower waiting time improves system efficiency.

e) Response Time - The time taken from process submission to the first response. Lower response time is crucial for interactive systems.

Q.3. Compare and contrast the performance of different sched- uling algorithm with respect to CPU Utilization, Throughput, Turnaround Time, Waiting Time, Response Time.

Ans:

	Scheduling Algo	CPU Utilization	Through put	TAT	WT	RT
1.	FCFS	Low	Moderate	High	High	High
2.	SJF	high	high	low	low	low
3.	RR	Moderate	high	Moderate	Moderate	low
4.	Priority	high	Variable	Variable	Variable	Variable

Q.4. Define Inter-Process Communication (IPC) and explain its importance in a multi-process System.

Ans. Inter-Process Communication refer to mechanism that allow process to exchange data and co-ordinate execution.

Importance:

- enable data sharing between processes.
- support synchronization in concurrent execution
- helps in modular design when processes perform independent tasks.
- essential in distributed computing and multiprocessing environment.

Q.5. What is Mutual Exclusion, and how doest it relate to the critical section problem.

Ans. Mutual exclusion ensures that one process access a shared resource at a time.
Critical section problem occurs when multiple process attempts to access a shared resource simultaneously leading to data inconsistency
Solution:
Use synchronization mechanism like Semaphores, locks, Peterson's algorithm to ensure Mutual Exclusion.

Q.6. Explain Peterson's solution to the critical section problem?

Ans. • Used for two processes to achieve mutual exclusion.
• Uses :-
1) Flag (2) - Indicates if a process wants to enter.

→ turn - Decides whose turn it is.
- Process set its flag to ~~turn~~ true and gives turn to the other process.
- Enters critical section only if the other process is not in it.
- After execution, it resets its flag to false.
- Ensures mutual exclusion, avoids deadlocks and allows fair execution.

Q.7. Explain strict alteration solution for the critical section problem?

Ans.
- Uses a turn variable to allow two process to take turns accessing the critical section.
- Process P_0 runs if turn == 0, then sets turn = 1,
- Process P_1 runs if turn == 1, then set turn = 0.
- Problem - causes unnecessary waiting if one process is slow.
- Not efficient for real-world systems.

Q.8. Explain following terms.

a) Semaphores

Ans. A counter used to control access to shared resources.

b) Event counter

Ans. Similar to semaphores but used when order matters.

c) Monitors

Ans. High level synchronization tool that allows safe access to shared resources.

d) Message passing

Ans. Processes communicate by sending and receiving messaging.

Q.9. What are potential challenges or issues faced when implementing inter-process communication mechanisms?

Ans.

- Synchronization issues
- Deadlocks
- Data consistency problems.
- Performance overhead

Q.10. Compare & contrast dIf synchronization techniques (semaphores, monitors, message passing) with respect to:
- Ease to use
- Performance
- Deadlock prevention
- Scalability in multicore system.

Ans:

	Ease of use	Performance	Deadlock	Scalability
Semaphore	Medium	High	needs extra handing	Good
Monitor	Easy	Medium	Handles deadlocks	limited
Message passing	Easy	Medium	Prevents deadlocks	Scalable

Q.11. How do semaphores help in solving the producer-consumer problem &

what is the role of the buffer in this solution?

Ans. Semaphores control access to the shared buffer to ensure smooth synchronization between producer (who adds item) & the consumer (who retrieves item)

- How semaphores help:

① Mutex semaphore - Ensure mutual exclusion so only one process modifies the buffer at a time.

② Empty Semaphore - Keeps track of empty slots in the buffer

③ Full semaphore - keeps track of filled slots in the buffer.

- Buffer Role

- Holds items produced by the producer & consumed by the consumer.
- Prevents overfilling or underflow

- Working

- Producer adds an item → decreases empty, increases full
- Consumer remove an item →

increases empty, decreases full.
- Mutex ensures only one process accesses the buffer at a time.

Q.12. How can message passing be used to solve the producer-consumer problem?

Ans. • Message passing allows the producer to send items to the consumer through a communication channel (like a queue or pipe)
• The producer sends a message (item) to the queue & the consumer retrieves it.
• How it works:
- Producer place an item in the queue (message passing)
- Consumer retrieves the item from the queue when ready.
- No direct access to the buffer. Synchronization is handled by the message-passing system.