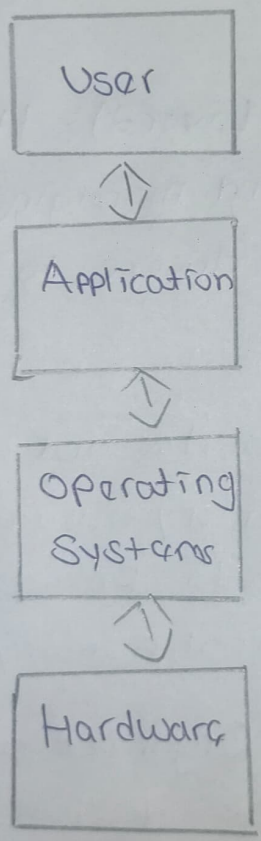
Name - Rakesh Suthar
Registration No - 22BCA10178
Subject - Operating Systems
Faculty - Dr. Chour Singh Rajpoot

Mid Term Exam

1) The structure of Computer Sytems



The layered approach is a design paradigm where an operating system is divided into distinct layers, each performing specific functions and relying on the services of the layers below it.

* Hardware Layer (Layer 0) - Directly interacts with the system physical components (CPU, memory, disk drives, etc). Provides basic services like memory allocation, device I/O, and interrupt handling.

* Device Drivers Layer (Layer 1) - Manages specific hardware devices, translating high level requests into low level commands. Hides device specific components.

* I/O Management Layer (Layer 2) - Handles Input/Output operations, buffering data and managing device queues. Coordinates with device drivers to ensure efficient data transfer.

* Memory Management - Allocates and deallocates memory to processes, optimizing memory usage. Implements virtual memory techniques to provide a larger address space than physical memory.

* File System (Layer 4) - Organizes and manages files and directories on storage devices. Provide services for creating, deleting, reading, and writing files.

* Kernel Layer (Layer 5) - Core of the Operating System Responsible for process Scheduling, Inter Process Communication and System Security. Manages system resources and handles system calls.

* User Interface Layer (Layer 6) - Provides the user interface, including command line interfaces, graphical user interfaces, and application programming interfaces (APIs). Interacts with the user and translates user commands into system calls.

Advantages -

* Modularity - Each layer is independent, making it easier to develop, test and modify.

* Debugging - Issues can be isolated to specific layers, simplifying troubleshooting.

* Security - Layers can be designed with varying levels of privilege, enhancing system security

* Flexibility - New features can be added or modified by modifying layers.

Disadvantages -

* Performance Overhead: Communication between layer cand introduce overhead due to context switching and data copying.

2)

(i) MultiProgramming V/S Multitasking

Multi Programming -

* Multiple Programs are loaded into main memory simultaneously.

* It increases CPU utilization by organizing jobs so that the CPU always has one job to execute

* Multiprogramming operating systems use the mechanism of job scheduling and CPU scheduling

Ex - A batch processing system where multiple jobs are executed sequentially, but the OS switches between them to prevent CPU idle time.

Multitasking -

* A single Program is divided into multiple task

* The OS switches between these tasks to give the illusion of concurrent execution

Ex - A user can run multiple applications simultaneously on single computer.

Process V/s Program

Program

- * A static entity, a sequence of instructions stored in secondary memory.
- * It doesn't consume system resources until its executed
- * It doesn't perform any action on its own

Ex - A .exe file or a Python Script

Process -

- * A dynamic entity, an instance of a program being executed.
- * It consumes system resources (CPU, memory I/O devices) during its execution.
- * Multiple processes can be instances of the same program running simultaneously

Ex - When you run a Python Script, the OS creates a process to execute its instructions

Time Sharing V/S Parallel Processing System

Time Sharing -

* Multiple users share a single computer system
* The OS allocates time slices to each user's process giving the illusion of simultaneous execution
* The time that each task gets to execute is called quantum

Ex - A mainframe computer used by multiple users for different tasks.

Parallel Processing System -

Multiple processors or cores execute different parts of a program simultaneously.

This is used to speed up execution time for computationally intensive tasks.

Ex - A supercomputer with multiple processors solving complex scientific simulations.

3)

Inter Process Communication - It is a type of mechanism usually provided by the operating system. The main aim or goal of this mechanism is to provide communication in between several processes. This communication could involve a process letting another process know that some event has occured or the transferring of data from one process to another.

Need of Inter Process -

* It helps to speed up modularity
* Computational
* Privelege Separation
* Convenience
* Helps Operating System communicate with each other Synchronize their actions

Approaches to Inter Process Communication

* Pipes
* Shared memory
* Message Passing
* Direct Communication
* Message Queue
* Indirect Communication
* FIFO
* Socket
* File
* Signal

Mutual Exclusion - It is generally required that only one process thread can enter the critical section at a time. This also helps in synchronization and creates a stable state to avoid the race condition

Race Condition - A situation that occurs when two or more processes access and manipulate a shared resource concurrently, and the outcome of the execution depends on the specific timing of their actions. Race Conditions can lead to unpredictable and often incorrect results.

Critical Section - It is a code segment that accesses a shared resource and must be executed by only one process at a time to avoid race condition.

The main goal of managing a critical section is to ensure data integrity by allowing only one process to execute in that section at a time, typically using mutual exclusion techniques like semaphores or mutex

Key Points

* Entry Section
* Exit Section
* Remainder Section

4) Necessary Conditions For Deadlock

* Mutual Exclusion - Atleast one resource can be held in a non-sharable mode, meaning only one process can use it at a time.

* Hold and Wait - Atleast one process must be holding atleast one resource and waiting to acquire additional resources held by other processes.

* Non Preemption - Resources cannot be forcibly taken away from a process holding them, they must be released voluntarily.

* Circular Wait - A set of processes must exist each of which is waiting time for a resoure held by the next process in set.

Resource Graph model -

A Resource graph model is a directed graph that depicts the current state of resource allocation among processes.

Process Nodes - Represent Process in System

Resource Nodes - Represent Resource in System.

* Request Edge - From a process node to a resource node, indicating that the process is requesting the resource.

* Assignment Edge - From a resource node to a process node, indicating that the resource has been allocated to the process.

Safe and Unsafe State

A system is in a safe state if there exists a sequence of process executions that allows each process to finish without causing a deadlock. In other words, there exists a safe sequence of resource allocation and deallocation that avoids deadlock.

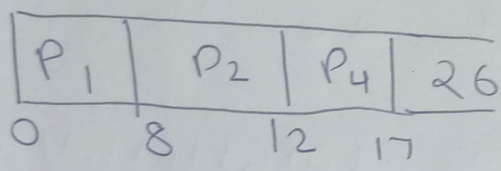
Key Points -

* A system can be both safe and unsafe, depending on the order of process execution
* Deadlock avoidance and detection techniques are employed to prevent or detect unsafe states.
* Resource allocation strategies, such as banker's algorithm, can be used to ensure system safely.

5)

Process	Arrival Time	Burst Time	Priority
P_1	0	8	3
P_2	1	4	2
P_3	2	9	1
P_4	3	5	4

Gantt Chart



TAT = CT - AT

WT = TAT - BT

Process	AT	BT	CT	TAT	WT
P_1	0	8	8	8	0
P_2	1	4	12	11	7
P_3	3	5	17	14	9
P_4	2	9	26	24	5

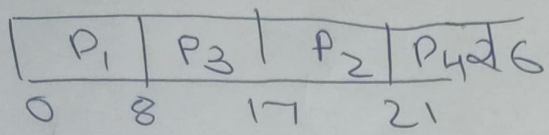
Average Waiting Time = (0+7+9+15) / 4

= 7.75

Average TAT = (8+11+14+24) / 4 = 14.25

Prority

Gant Chartt



Process	AT	BT	CT	TAT	WT
P_1	0	8	8	8	0
P_3	2	9	17	15	6
P_2	1	4	21	20	16
P_4	3	5	26	23	18

Average Waiting Time = (0+6+16+18)/4 = 6

Average Turn Around Time = (8+15+20+23)/4
= 16.5