__/__/____

"Operating System"

★ Why :- i) For Resource Management. {To divide Resources available}

ii) It acts as a Interface for Resource allotment.

iii) Prevent Bulkiness of files. (DRY → Donot Repeat yourself).

iv) Resource Exploitation by 1 app.

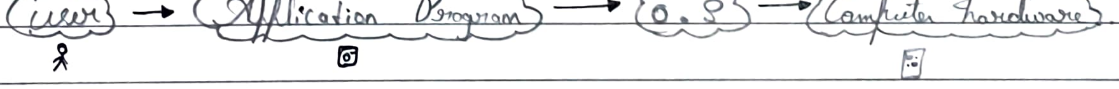
v) No Memory Protection and ⟨isolation + Protection⟩

⊙ Resources → Memory, device, file, Security, Process etc.

★ only o.s has Access to Computer Hardware [RAM, ROM, Motherboard, GPU, CP

→ An o.s is a System Software that act as an intermediary between a user of a Computer and Computer Hardware.

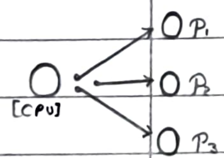
→ An o.s is a piece of Software that manages all the resources of a Computer System, both hardware & Software and Provide an environment in which users Can Execute their Program in Convinent & efficient Manner by hiding underlaying Complexity of the hardware and acting as a Resource Manager (allocator) (Management).

→ 

→ O.S Goals :-

i) Maximum CPU Utilization.

ii) make Computer System Convenient to use by hiding dificulty in managing the hardware



iii) use Computer hardware in efficient Manner.

iv). It is a Resource allocator.

v) avoid Process Starvation.

vi) High Priority Execution. (Anti-Virus).

__/__/____

"Computer System & Components"

↓

i) Hardware → Provide Basic Computer Resources. Eg→ CPU, Memory, I/o devic

ii) O.S → Resource allocation & Management.

iii) Application Programs → way in which System resources are used to Solve & Computing Problems.

Eg→ Compiler, IDE, web browser, Database Systems, games.

iv) Users → People, Machines & Computers.

→ ## Types of O.S :-

i) Single Processing O.S :- only 1 Process Executes at a time from Ready queue (Ready to Execute).

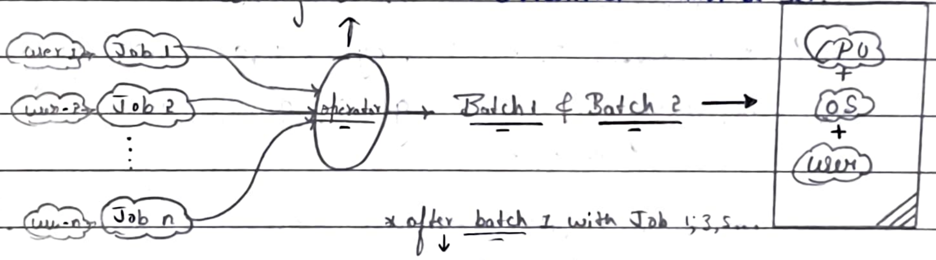
* "No fulfilment of basic O.S goals" * No Maximum utilization of CPU.

* Eg MS DOS. * Process Starvation occur.

ii) Batch Processing O.S :- * No Maximum utilization of CPU

(Sorting Jobs) * Priorities Cannot be Set.

* Batch ↓ 'with Similar needs' ↓ 'Punch Card' → "Jobs are in this form".



* after batch 1 with Job 1,3,5...
↓
* We Execute batch 2 with Job 2,4,6..

* Eg ATLAS, Manchester univ

iii) Multiprogramming O.S :-

* Increase CPU utilization by keeping multiple jobs in the memory (ready Q) So that CPU always has one to Execute in Case Some job gets busy with I/O.

* Single CPU. * I/o operation → Copy from USB

* Context Switching for Process

* Switch happens when Current Process goes to wait state

* CPU idle time reduced.

Eg→ THE, Dijkstra

(Continued) →

*) The o.s allocates the resources among the Programs Such that the hardware is used efficiently.

*) The o.s is the Program running at all the times on Computer.

↓

"It is usually Called as Kernel".

*) Eg of o.s :-

i) Windows (GUI -based, PC).

ii) GNU/Linux (Personal, workstations, Three - tier client/server

iii) macOS (Macintosh), used for Apple's Pc and Workstations

iv) Android (Google's o.s for Smartphones/watch/tablet)

v) iOS (Apple's o.s for iphone, ipad, iPod).

O.S { Non- Kernel → user necessary fun^n → Stored in harddisk (runs when called)

Kernel. → System necessary fun^n → Stored in memory (always running)

Features	O.S	Kernel.
1) Definition :	A Software that manages all hardware and Software resources.	Core part of o.s that handles Communication b/w hardware & Software.
2) user Interaction :	user directly interacts with o.s through interfaces like GUI's (or) Command lines.	Kernel operate in the background, with no direct user interaction.
3) Resource Management:	o.s manages high level resource allocation & distribution among Various Processes & Applications	Kernel Performs low-level tasks like memory allocation, Process Scheduling and device Management.

⊙ O.S is a Complete Package that includes Kernel as a key Component. While O.S allows users to interact with Computer & run Programs, the kernel is responsible for low level operations like Managing memory and interacting with hardware.

*→ Views of O.S :-

i) User View →

i) Standalone System (ease of use + high Performance).

ii) users at different terminals (Maximize Resource utilization).

iii) Workstation users (ease of use + Resource availability).

iv) Users of handheld System (ease of use & Performance Per amount of battery life).

v) embeded Systems (washing m/c) → No user interaction but Some LED's to Show Status of work.

ii) System View →

i) Resource allocator → CPU time, memory Space, file - Storage Space, I/o devices, Shared files etc (Conflict is Solved).

ii) O.S is a Control Program and Manage the Execution of user Program to Prevent errors and improper use of Computer.

⊙ PCB → Process Control Block. (Process Identification Number)

* While Creating a Process → O.S performs Several operation
* To identify each Process, it assigns PID to each Process
* As O.S performs Multiprogramming, it needs to keep track of all Process
* Thus, PCB is used to track the Process's Execution State.
* PCB is a data Structure used to manage info about a process

*) Registers :- They are Small Storage areas in Computer's CPU that Store and manipulate data during instruction Execution.

* They are made up of flip-flops [Store a Single bit of info].

⇒ 1) Stack Point Reister (SP)
2) Base Point Reister (BP)
3) CR → Control Registers

⇒

(Store Process State) ←

(Save CPU Register so it can be restored) ←

Open file list ⊙

open device list ⊙

Process ID	→ unique identification number alloted
Process Status	→ New, Ready, Running, Wait, Terminate etc
Pointers	→ Pointers to Parent / Child Process
Program Counter	→ Address to next Instruction
CPU Register	→ Accumulator, General Purpose R, Base R
CPU Scheduling Info	→ Priority of Process
I/o Status Info	→ List of I/o devices allocated, List of open files
Memory Mgmt Info	→ Value of Base & Limit Registers, Page table
Misc Info	→ Amount of CPU used, Time Constrains

iv) **Multitasking :** ⟨Time Quantum⟩

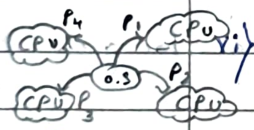
* Single CPU * Context Switching and time sharing used
* InCreased Responsiveness * high Priority Process Possible
* Logical Extension of Multiprogramming.
* able to run more than 1 task Simultaneously.
* CPU idle time further reduced.

* Eg → CTSS, MIT

v) **Multiprocessing :** * Better throughput

* Context Switching * Time Sharing
* More than 1 CPU in a Single Computer (Core → Processor)
* Least Process Starvation. * InCreases Reliability. (if 1 is damaged other works)

* Eg → Windows



vi) **Distributed O.S. :**

* O.S Manages Many bunch of Resources, >= 1 CPU, >= 1 GPU, >= memory etc ⊙ It works over Network,
* Collection of Independent, networked, Communicating and physically Seperate Computational nodes.

⊙ Loosely Coupled + Autonomous.

* Eg → Locus

vii) **RTOS :**

* Real time error free, Computations within tight-time boundaries.
* fastest Execution of Jobs. Air Traffic control System.

* Eg → ATCS

{Core → CPU}

Multi Threading

__/__/____

*) **Program:** Program is an Executable file which Contains set of instructions to Complete a Specific Job.
* It is a Compiler's Code * Stored in DISK.
* Ready to be Executed

*) **Process:** Program under Execution, Resides in Computer's memory (RAM).

*) **Thread:**
* Light-weight Process.
* An independent Path of Execution in a Process.
* Single Sequence Stream within a Process.

MultiThread: → (JPEG to PNG)
* used to achieve Parallelism by dividing a Process's tasks which are independent Path of Execution.
* Eg → Multiple tabs in a browser.
  * Text editor {when you are typing in a editor, Spell Checking, formatting of text and Saving the text are done Concurrently by multiple threads}.

Imp Interview

M Threading	MultiTasking
<More CPU> <Multiprocessing environment>	
*) A Process is divided into Several different Sub-tasks Called "Threads", which has its own Path of Execution. It is Called Multithreading.	*) Execution of more than 1 task Simultaneously is Called Multitasking. Time Sharing b/w P1, P2....
	(ulta)
*) more than 1 Process being Context Switched.	*) more than 1 Thread being Context Switched
*) CPU 1	*) Cpu >= 1.
*) Isolation & memory Protection Exists	*) No Isolation & memory Protection.
*) Process are Scheduled.	*) Threads are Scheduled.
*) Resources are differently allocated	*) Resources are Shared

__/__/____

*) **Thread Scheldiing** :-

- * Threads are Scheduled for Execution based on their Priority.
- * Even though threads are Executing within the Runtime, all threads are assigned Processor's time Slice by O.S

→

Thread Context Switching	Process Context Switching
* O.S Save Current State of thread & Switch to another thread of Same Process.	* O.S Saves Current State of Process and Switches to another by restoring its State.
* Fast Switching	* Slow Switching
* CPU's Cache State is Preserved	* CPU's Cache State is flushed
* doesnot include Switching of memory address Space. (PC, Stack, register are Switched)	* Include Switching of memory address Space.