# **Operating System**

1. What if there is no OS?

* Apps become bulky and complex (because it’s the responsibility of app’s developer to write the all the code for resource management, memory management, etc. which are required to run the app on machine which has no OS)
* No memory protection

E.g., There are two apps in memory, so if there is no OS in picture then one app can able to read/ write the memory of some another apps. so, this occurs major security problems.

* Resource Exploitation by one app

**Goals of OS**

1. Maximum CPU Utilization
2. No Process Starvation
3. High Priority Job Execution

**Types of OS**

1. Single process OS
   1. Single CPU
   2. No maximum CPU Utilization
   3. Process Starvation Occurs
   4. High priority jobs cannot be able to run on single process OS
   5. Example: MSDOS
2. Batch OS
   1. Single CPU
   2. No maximum CPU Utilization
   3. Process Starvation and Batch Starvation Occurs.
   4. High priority jobs cannot be able to run on single process OS
   5. Example:
3. Multiprogramming OS
   1. Single CPU
   2. Jobs are in Ready Queue, when there is any job is going to performs IO operation then OS take new job from the ready queue, schedule to CPU, so CPU cannot be IDLE. so, in this CPU cannot sit idle.
      1. Context Switching: When there is any job which is going to perform any kind of IO operation then it is responsibility of OS to store the current context (state of the process) of the process into its PCB (Process Control Block) and restore context of another process which is schedule to CPU. This is known as context switching.
   3. Process starvation occurs when if current process is not going to perform IO operation. for example, if process p1 is currently executed and there are n processes after p1 in ready queue, and process p1 did not have any IO operation then the processes after the p1 in ready queues are suffer in starvation.
   4. Hi
4. Multi-Tasking OS
   1. Single CPU
   2. Context switching occurs based on assigned time Quantum.
   3. Time sharing
   4. Maximum CPU Utilization
   5. Less Process starvation
   6. High priority Process get chance for execution.
5. Multi-Processing OS
   1. multiple CPU (No of CPU >= 1)
   2. context switching
   3. time sharing
   4. Maximum CPU Utilization
   5. Less Process starvation
   6. High priority Process get chance for execution.
   7. Increased Reliability (In case of Any CPU fails)
6. Distributed OS (loosely coupled OS)
   1. multiple CPU (No of CPU >= 1)
   2. multiple memory
   3. multiple users
   4. all computer is interconnected over network
   5. context switching
   6. ex. leetcode
7. real time OS
   1. Need high Accuracy
   2. Ex. Used in ATC (Air Traffic Control), Industrial.

**Multi-Tasking vs multi-Threading**

* **Process**:
  + Process is known as” Program (.exe file) under execution”.
  + CPU does not able perform any kind of operation on program until it comes into main memory for execution.
  + When the program is in main memory it is said to be “program under execution” or Process.
* **Thread:** 
  + Lightweight process.
  + Executed Independently.
  + Used for parallel execution.
  + Asynchronous Execution.

Que. How many CPU needed in multi-threading? Or ek process ne ketala thread ma divide kari e to maximum gain male?

Ans: if there is only one CPU is available then multithreading karava thi koi performance gain nahi male km k ek j CPU upar ek karata vadhare thread context switching karase

So no of threads per process is equal to the number of cores (CPU) available in the system.

Or also say that No of CPU == No of thread per process. both side true.