Module 1.2.1 OS

- Introduction tot he O/S:
- Intro to linux

Silberschatz Galvin book for O/S :

OS: Controls the hardware and co ordinates its use among the various application programs for the various users!

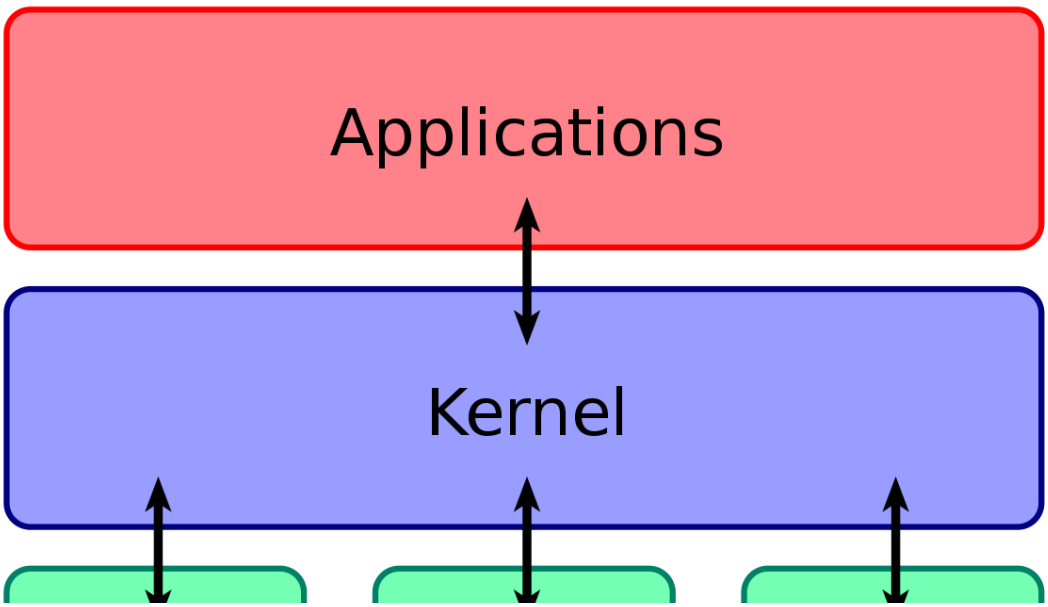
The layers in system:

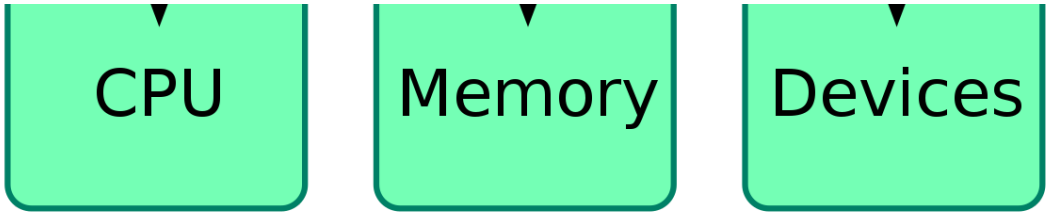
- Application
- OS
- Computer Orgnaisation
- VLSI - Very large-scale integration (VLSI) is the process of creating an integrated circuit (IC) by combining millions of MOS transistors onto a single chip
- Transistors - A transistor is a device that regulates current or voltage flow and acts as a switch or gate for electronic signals. Transistors consist of three layers of a semiconductor material, each capable of carrying a current.

HARDWARE ABSTRACTION :
RESOURCE MANAGEMENT :

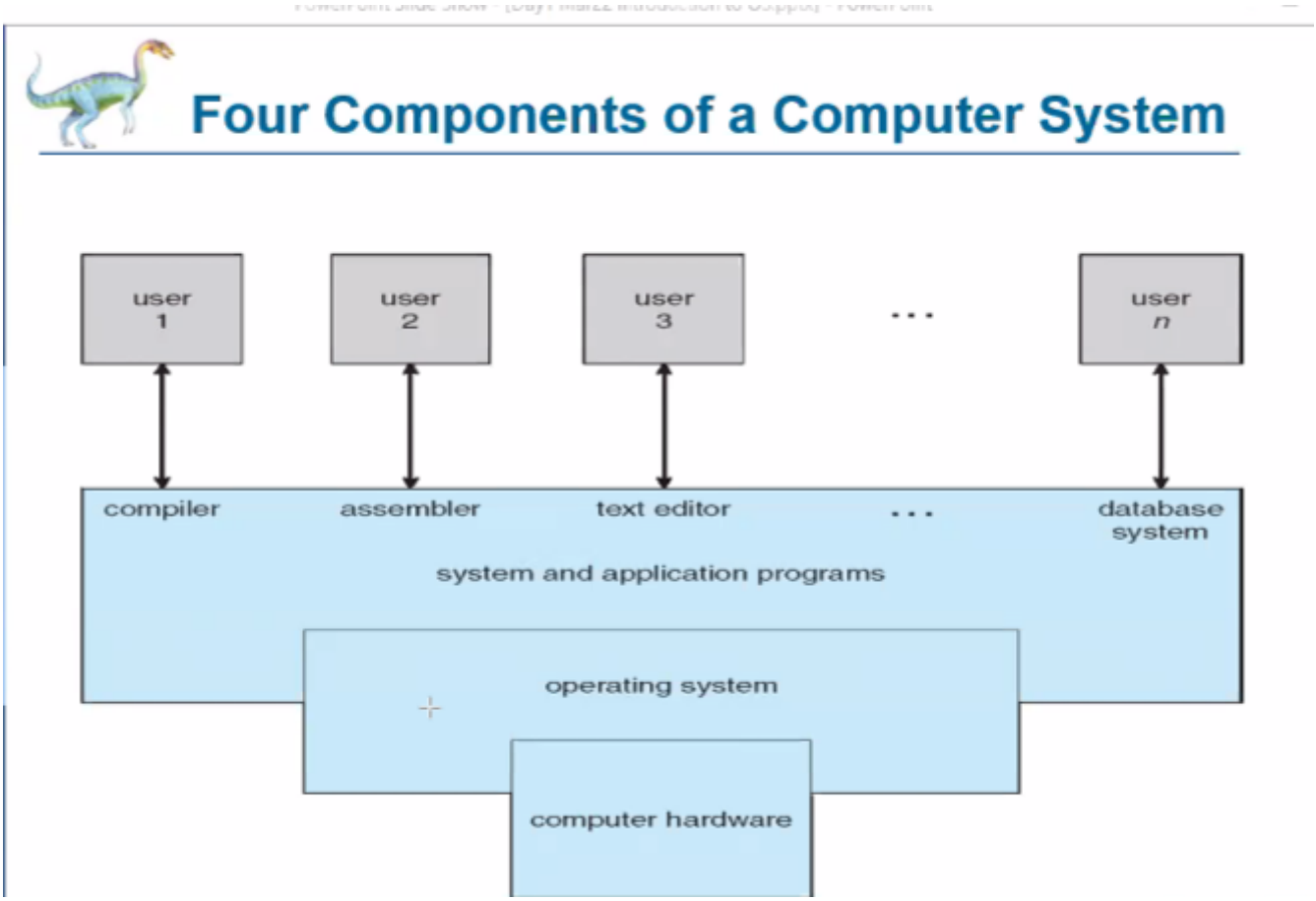
Kernal : The kernel is the essential center of a computer operating system (OS). It is the core that provides basic services for all other parts of the OS. It is the main layer between the OS and hardware, and it helps with process and memory management, file systems, device control and networking.

It's the program which runs all the time in computer:
Computer program and the core of OS with the complete control over everthing in the system.





OS is also called as Resource allocator OR Control Program :



System Boot :
Booting is the process of starting a computer. It can be initiated by hardware such as a button press or by a software command. After it is switched on, a CPU has no software in its main memory, so some processes must load software into memory before execution. This may be done by hardware or firmware in the CPU or by a separate processor in the computer system.

- When switch on systrem : bootstrap program will be loaded
- Volatile(RAM) and (ROM)Non-Volatile memory(permanent memory) our system has
-

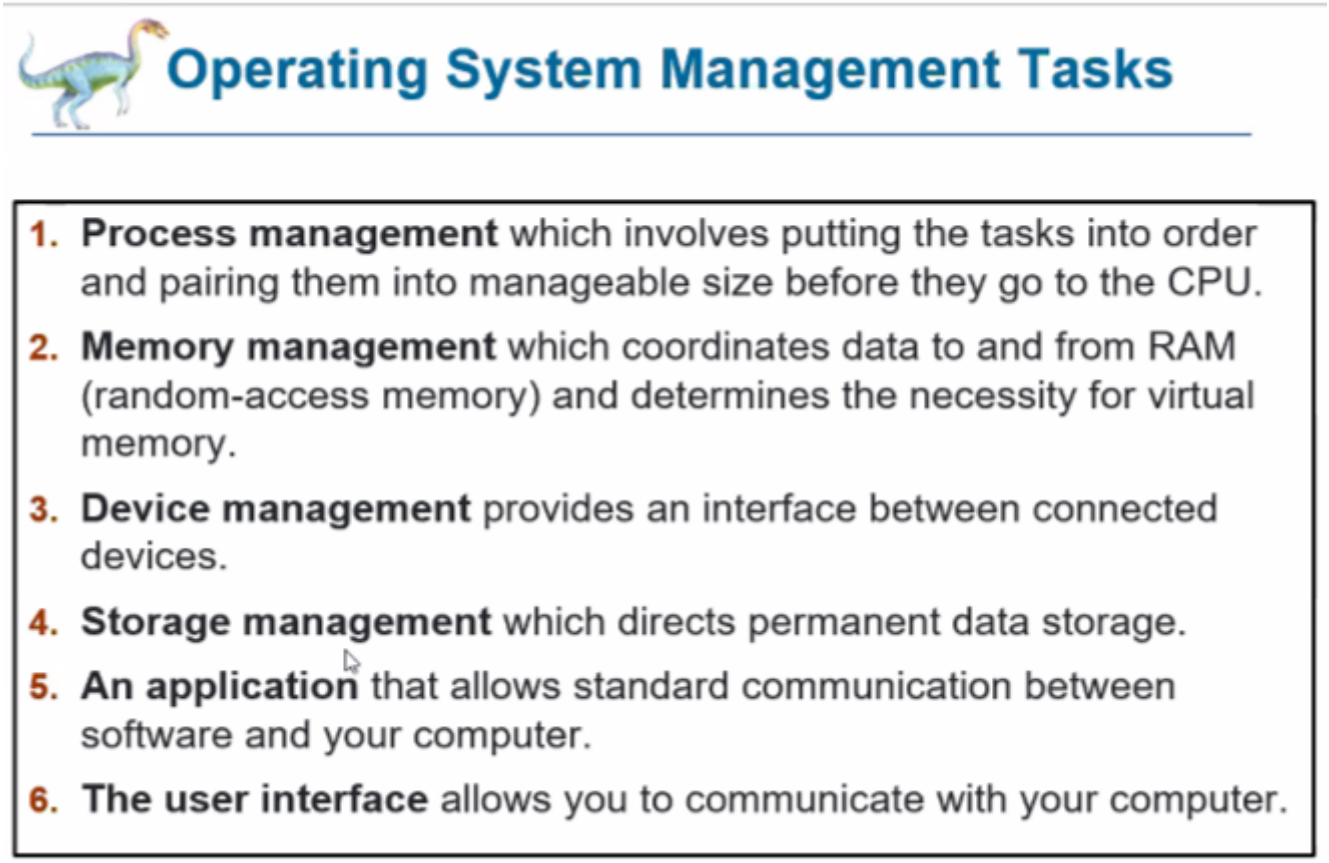
TYPES OF MEMORY IN COMPUTER :

Von-Neumann Model :

PROGRAM VS PROCESS :

program is the set of instructions

When the porgram is loaded into memory and waiting for the execution is the process!



What are the different operating systems for mobile phones?

Android (Google)

iOS (Apple)

Bada (Samsung)

Blackberry OS (Research in Motion)

Windows OS (Microsoft)

Symbian OS (Nokia)

Tizen (Samsung)

A real-time operating system (RTOS) is an operating system with two key features: predictability and determinism. In an RTOS, repeated tasks are performed within a tight time boundary, while in a general-purpose operating system, this is not necessarily so.

OS From Utube :

- Basic concepts:
  - Types of OS, System calls, Kernal, Process diagram
- Process Scheduling (Easy and very important topic)
  - FIFO, SJF, PR, RoundRobin Alogorithms
- Process Synchronisation :
  - Semaphore
- Deadlock adn thread
  - Bankers alogortihms
- Memory management
  - Paging
  - segmentation
  - fragmentation
  - Virtual memory
  - Page replacement algorithm
- Disk Scheduling
  - sCAN, cscan aLGO,fcfs
- UNIX Commands
  - ls, mkdir, chmod, cd etc.
  - Open System calls
- File Management and security
  - attacks and it's types

An operating system is a piece of software that manages all the resources of a computer system, both hardware and software, and provides an environment in which the user can execute his/her programs in a convenient and efficient manner by hiding underlying complexities of the hardware and acting as a resource manager.

An Operating System can be defined as an interface between user and hardware. It is responsible for the execution of all the processes, Resource Allocation, CPU management, File Management and many other tasks.
The purpose of an operating system is to provide an environment in which a user can execute programs in convenient and efficient manner.

What does an Operating system do?

1. Process Management
2. Process Synchronization
3. Resource Management
4. Memory Management
5. Ram Management
6. CPU Scheduling
7. File Management
8. Security

8. Security

shorthand : PP MRF CS

Types of OS :

- Single processing OS

Eg. MS-DOS

- Batch eg ATLAS
  - similar kinds of job given to the pc for the execution
  - one batch at one time
  - its very old system
  - no longer used now as the complexities raised
  - Job is given to the operator, he converts that into the batches and B1 B2 B3 and it will be given to the CPU and CPU will be idle this was the major disadvantage of this system
  - CPU will execute the next task only after the completion of the one job
  - Monitors were introduced to replace the opeartos
  - CPu utilisation was not full
  - Process starvation problem was still there.

Fortran : is a general-purpose, compiled imperative programming language that is especially suited to numeric computation and scientific computing.
First appeared: 1957; 65 years ago

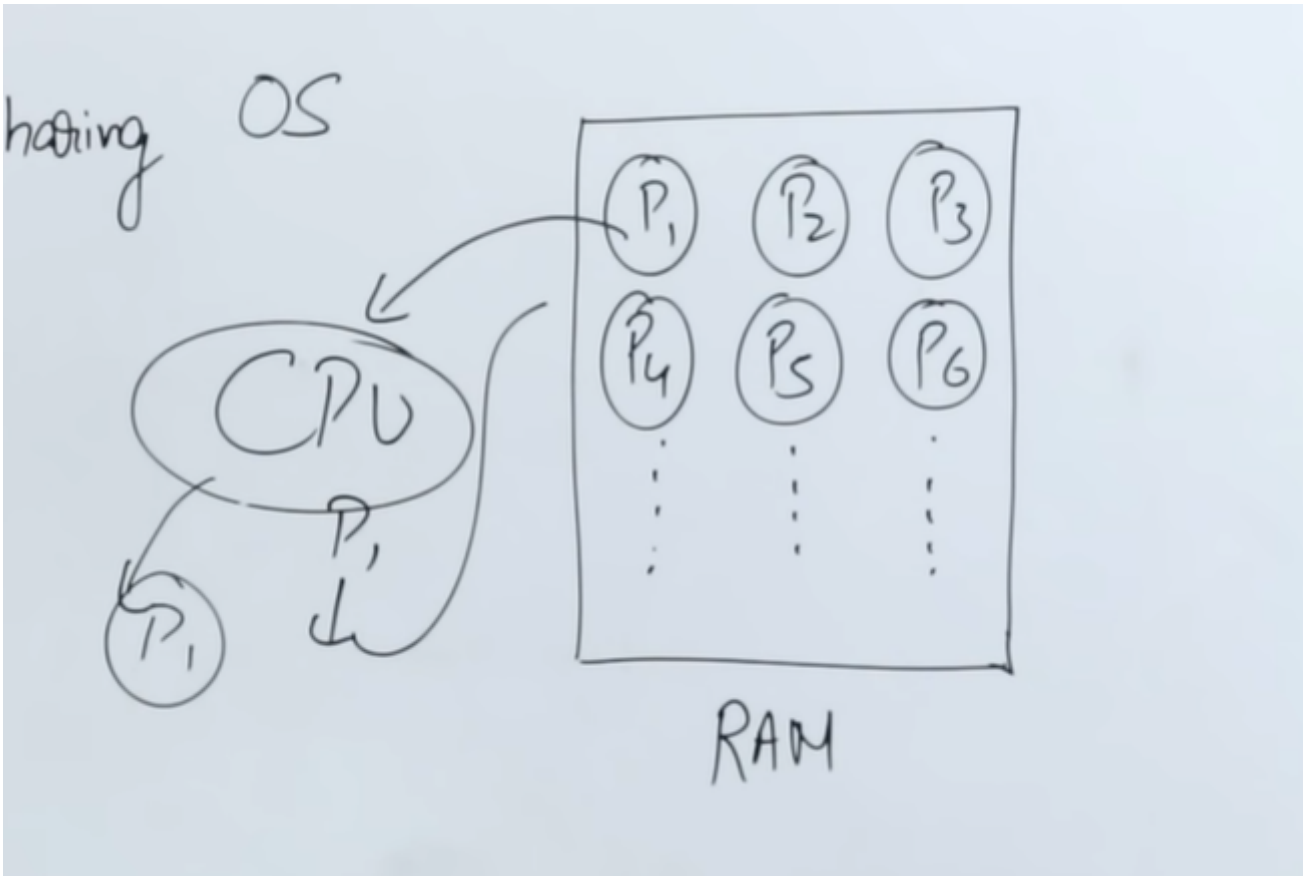
IBM 7090/94 IBSYS
IBSYS itself is a resident monitor program, that reads control card images placed between the decks of program and data cards of individual jobs. An IBSYS control card[a] begins with a "$" in column 1, immediately followed by a Control Name that selects the various IBSYS utility programs needed to set up and run the job. These card deck images are usually read from magnetic tapes prepared offline, not directly from the card reader.

- Multi Programmed eg THE
  - CPU ideal time was reduced with this process
  - Non preemptive - will finish the once process only after that will go to the next one
  - Multiple processes were brought into the ram and then P sent to CPU for processing and CPU will execute one program at a time till its end until and unless it ask for any i/p or o/p
  - came ebefore multitasking

- Multi tasking/ time sharing eg CTSS
  - Preemptive - will switch on fixed time intervals
  - In this case the CPU will go to every process without ending the complete process, here CPU has set the time that it will give the fixed amout of time to each and everyt process and even if

it didnt get completed then also it will go to the next process.

- In this case the major advang=tage is this that the last process will have to wait less as compared with the multiprogramming OS and its turn will come early and after that ti will go to the next process.
- Responsiveness is high here
- After specific time quntum (Applicable in Round Robin Algo) the process wil go back to the ready queue and wait in the ready queue.
- Context switching is applicable here which means the process after specific time quntum will back in the queue

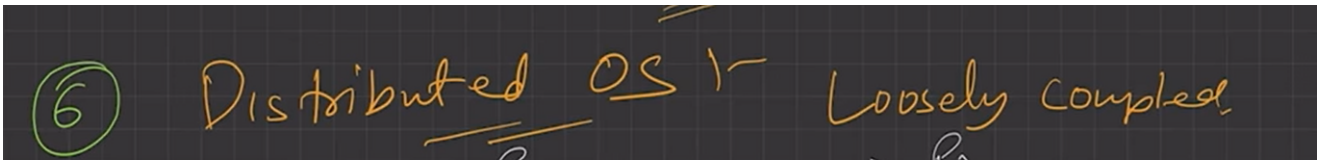


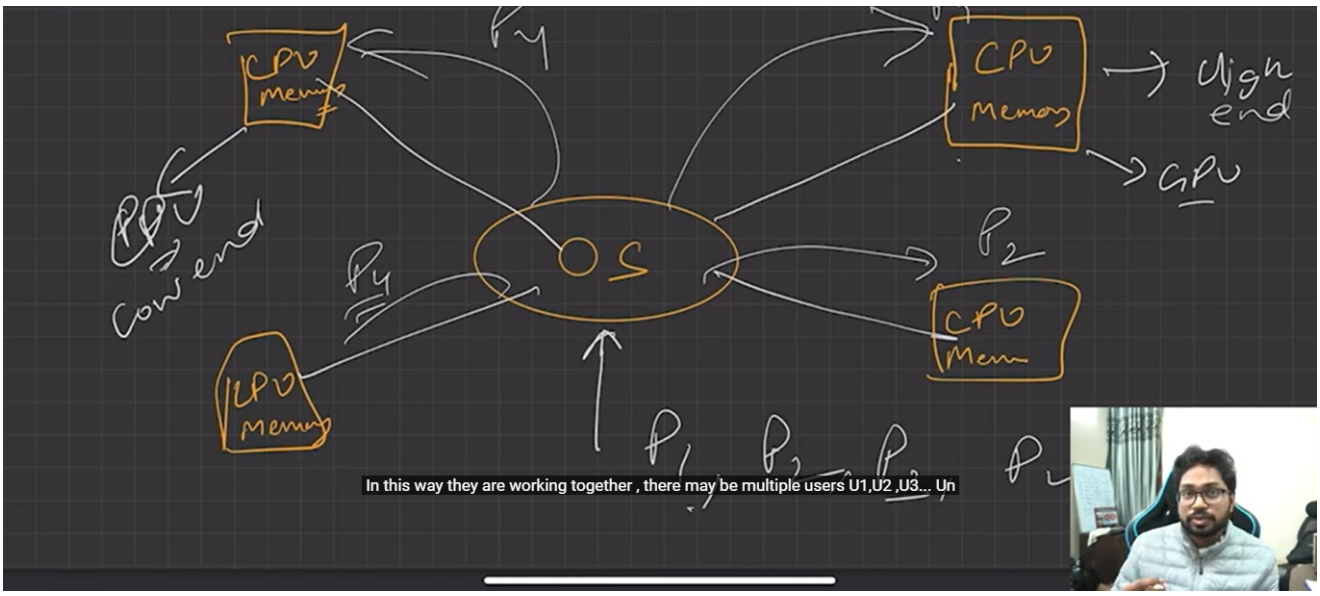
- MultiProcessing eg Windows

No of CPU>1

LETS ONE CPU HAS 4 CORE then 8 cores is like 2 CPU so it's multiprocessing OS.

- Real time OS
  - Live system
  - time is very important
  - hard real time - no delay accepted
  - soft real time - small delay is a accepted

- Distributed
  - Geographically seperated but connected with the network
  - each and every system has got its own process and environment
- 



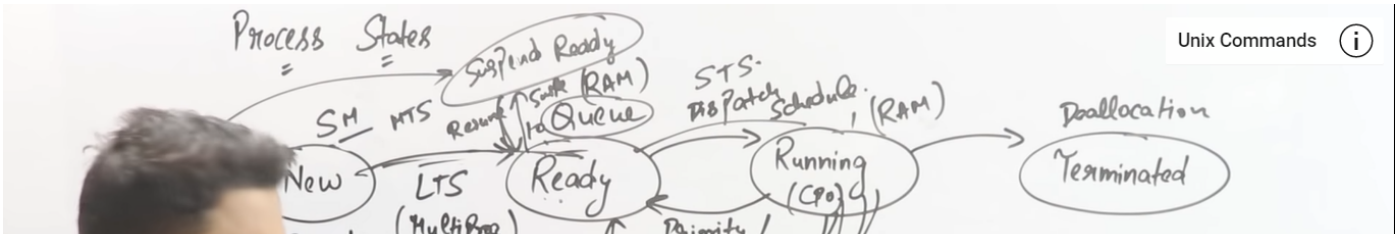
- Lossely coupled autonomous systems.
- Clustered
  - its simliar to the distributed but its clustered!
  - its power increases
  - it acts as cuper computer

- Embeded
  - works on a fixed system
  - e.g. washing machine, microwave etc. they perofrm fixed functions

program: Set of instruction stored in file on
process : Program under execution is called process in RAM
thread : light weight process or small part of process

Process states:

new : Start of the program eg.g. program is created and stored on ROM
ready (queue) : various processes send to the RAM by LTS
Running: Here the program is still in the ram but now its send to CPU for the operation and its donw by the STS and if ends the program then its called as multiprogramming and if sent back after time quantum then its called as an multiprogramming and the program here is sent backl to the ready queue and the next propgram is taken for the execution in cpu.
Terminated: and if the progam is complted then it fo this stage and deallocation is done in this stage all the repspources allocated =to this stage are taken back from the program.
wait/ block :if the prgram needs any kind od i/p or o/p then it goes for the wait block



PS command is used to check the process stage in linux OS

Important linux commands:
chmod ugo=r file_name.ext => used to give read acces to all the three groups

lseek command:

System calls in OS:
WINDOWS HAS 700+ SYSTEM CALLS
Linux has 300+ System calls
System calls are used to interact with the hardware thru kernel
API are used inprogramming languages such as printf(), to print the o/p on screen and here it's called as API's.

File related :
open(), Read(),Write(),close(),CreateFile()

Device Related:
Read, Write,Reposition,ioctl,fcrtl

Information calls:
get pid(process ID), PPid (Parent process ID),attributes,get system time and data, to get teh metadata ;

Process control:
Load, Execute, Abort,Fork,wait,signal,allocate etc.

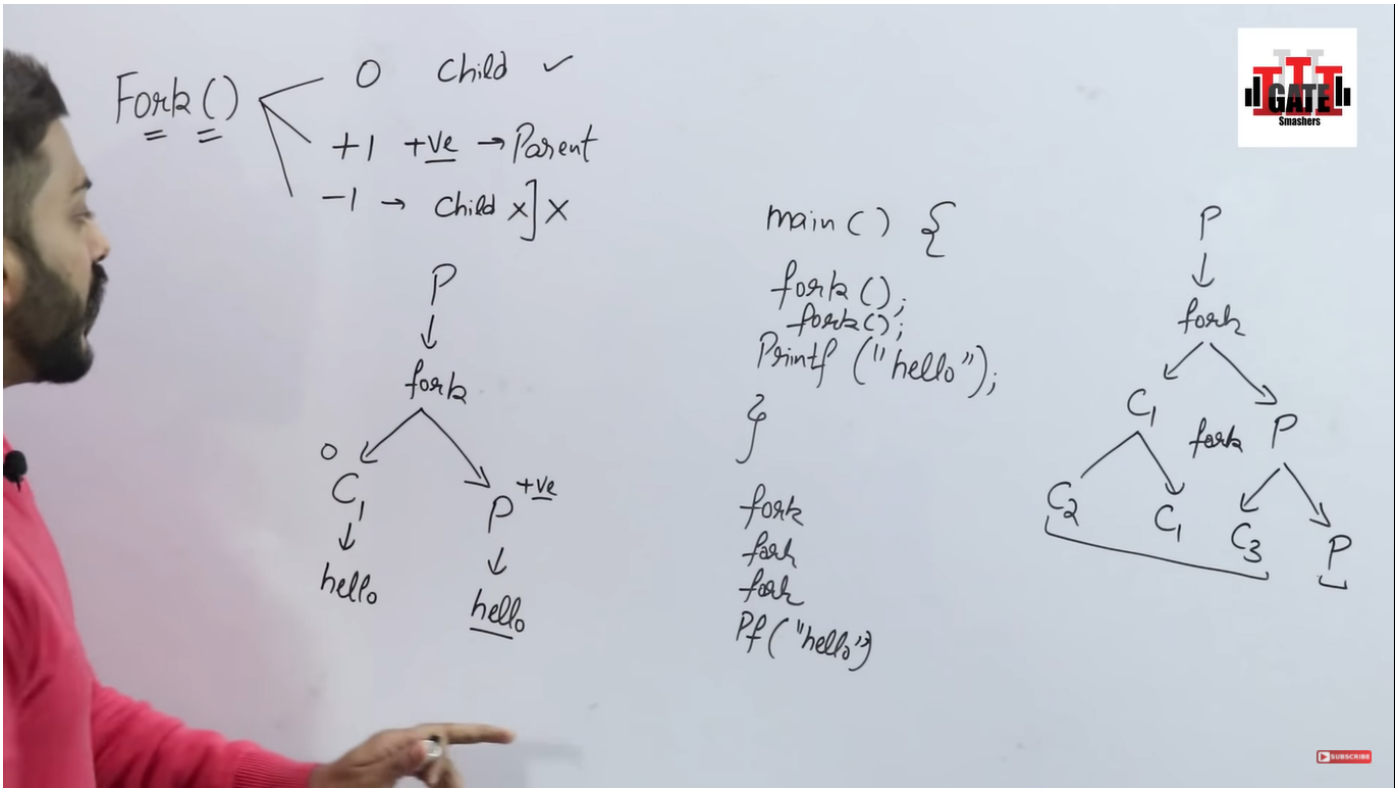
Communication:
pipe(),create/delete connection,

Fork system call;
fork creates chiuld to perform some task
0 CHILD PROCESS CREATED

0 CHILD PROCESS CREATED

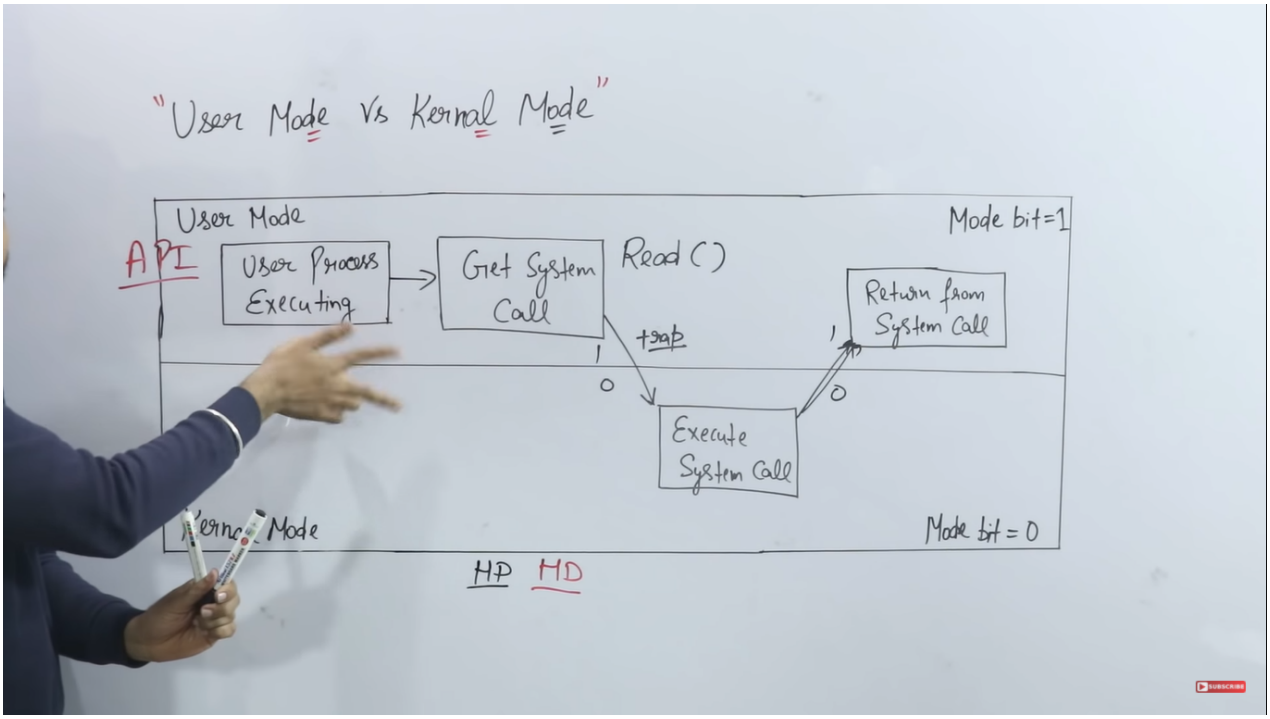
+1 is parent process

-1 child process doesnt created

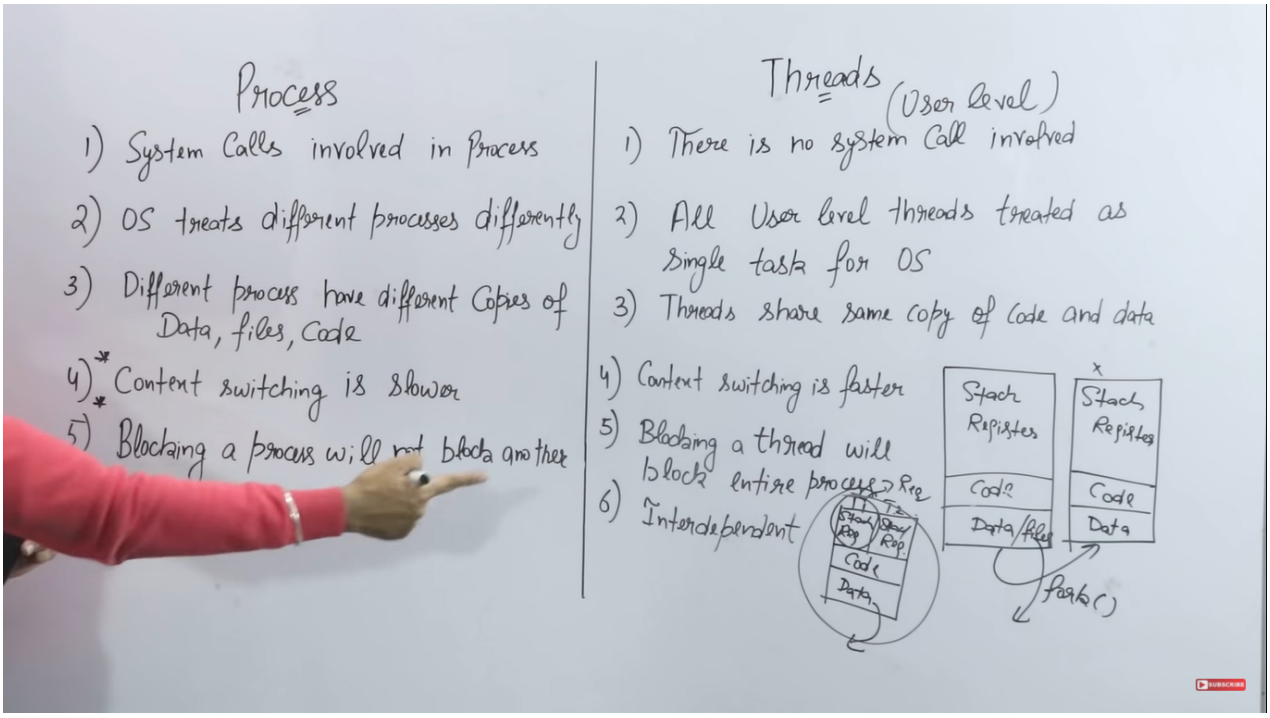


Total number of child process generated = 2**n -1

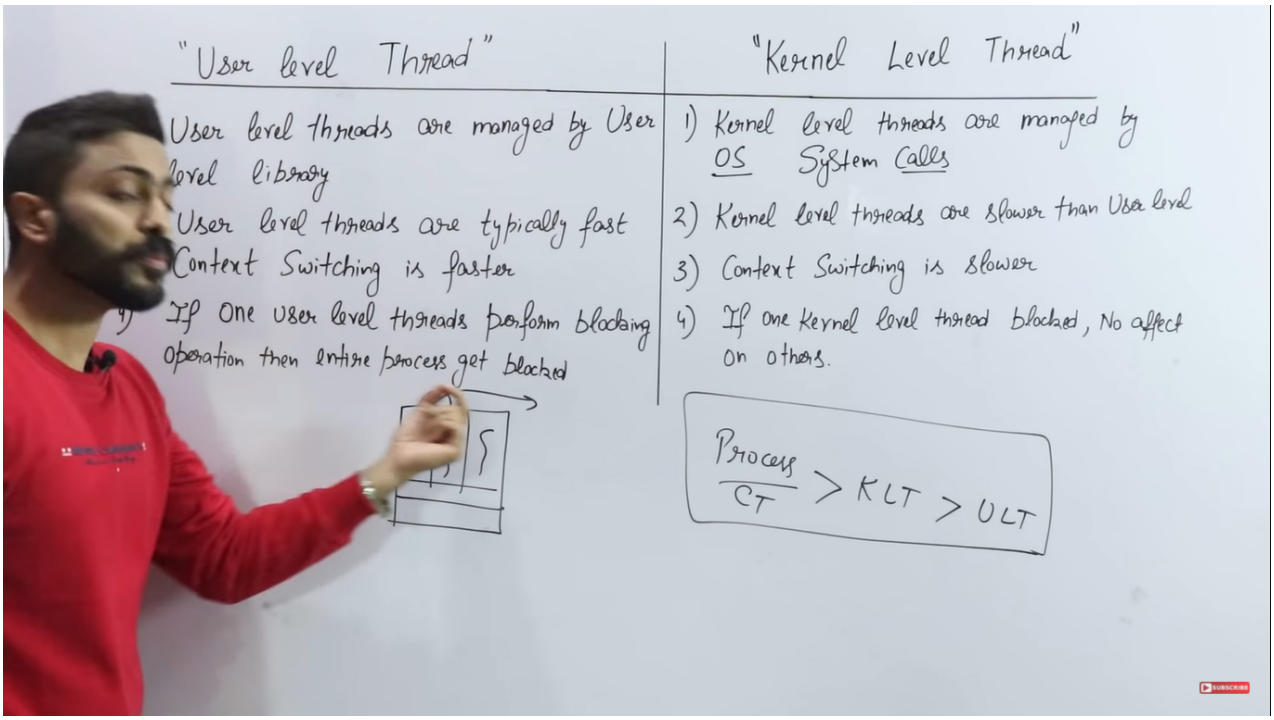
here n is the no of times fork has created



Difference between process and threads!



uSER LEVEL THREADS vS KERNEL LEVELS THREADS:



Process scheduling algoeithms:

CPU Scheduling alorithms:

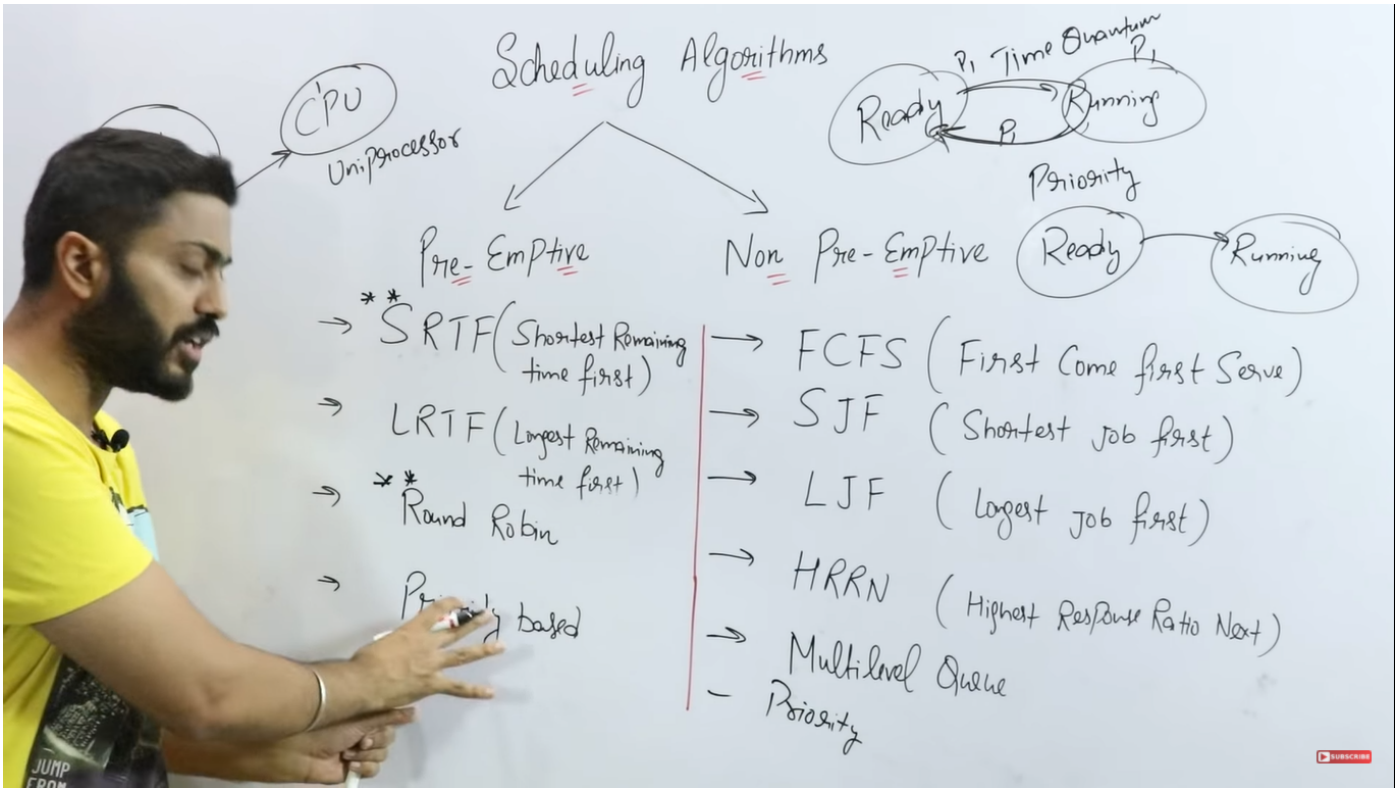
Non-Premeptive:

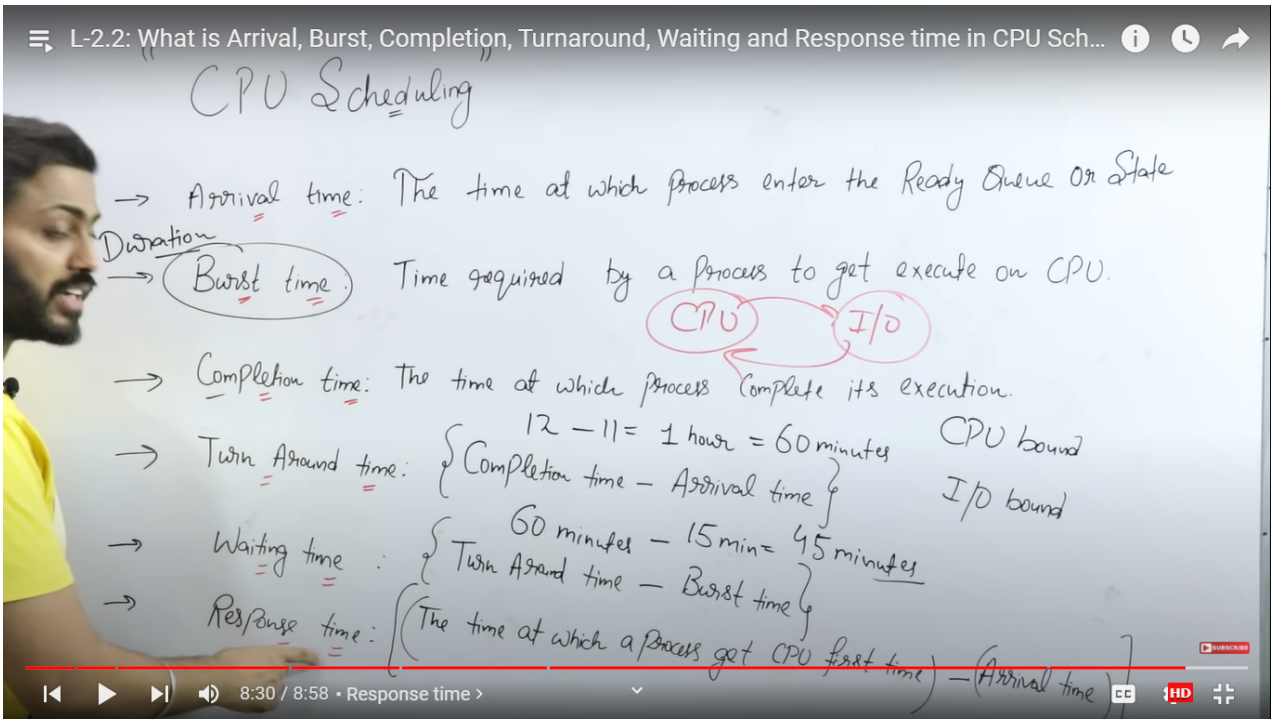
- FCFS
- SJF
- LJF
- HRRN : Highest Response Ration First

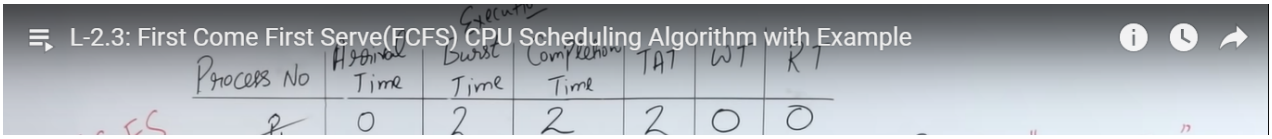
- Multi level Queue
- Priority

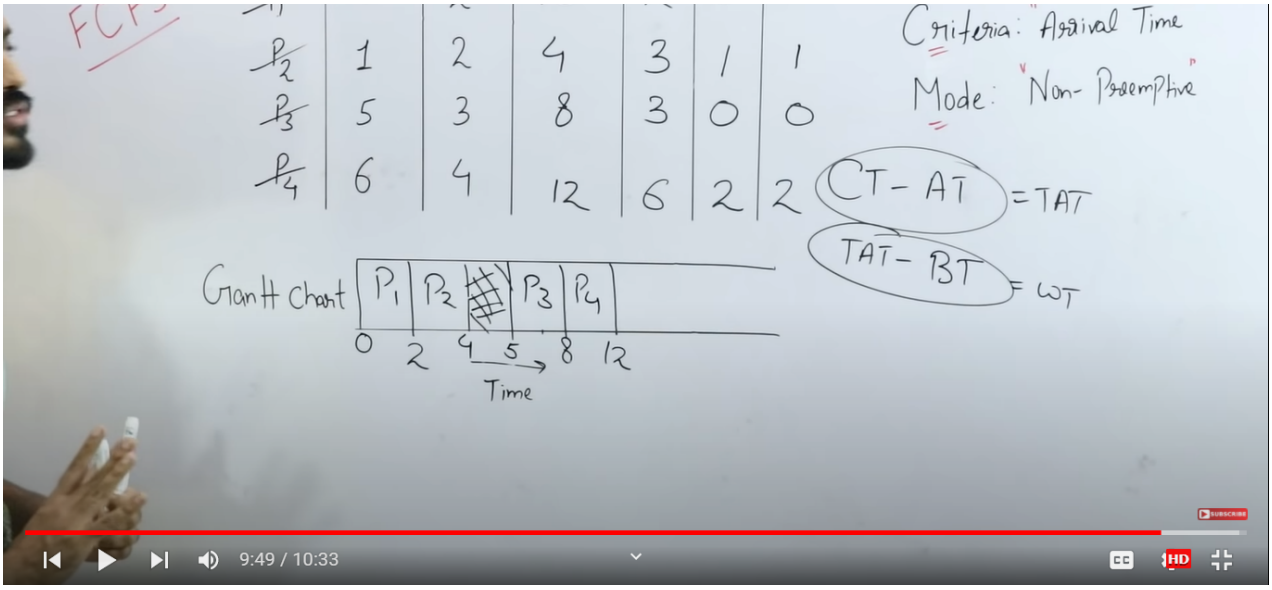
Preemptive Alogo's:

- SRTF
- LRTF
- Round Robin
- Priority Based



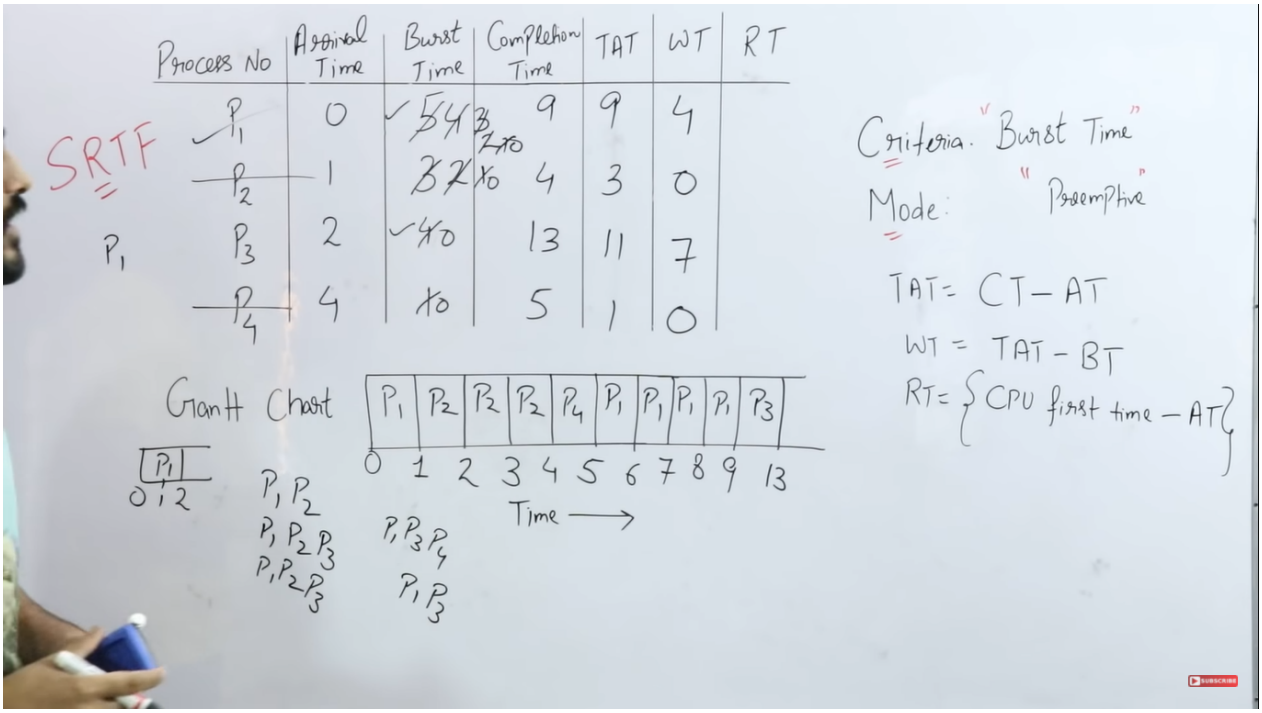






Response time is same as Waiting time in case of non premptive mode:

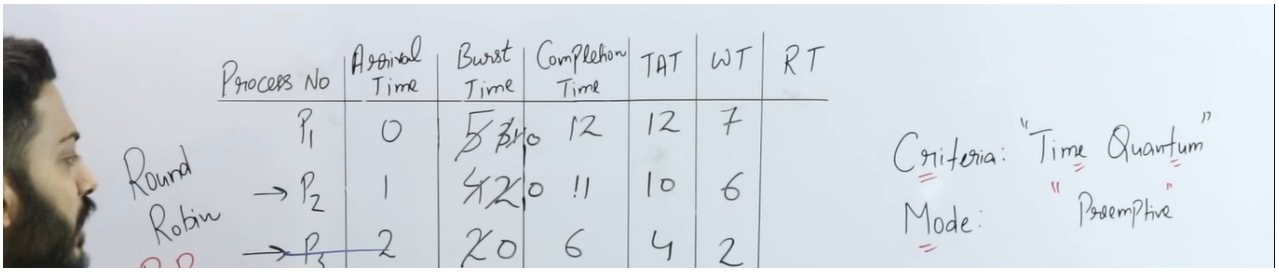
SRJF : Shortest Running Job First (It's a preemptive mode but time quntum is not applicable here process with the less burst time is selected)

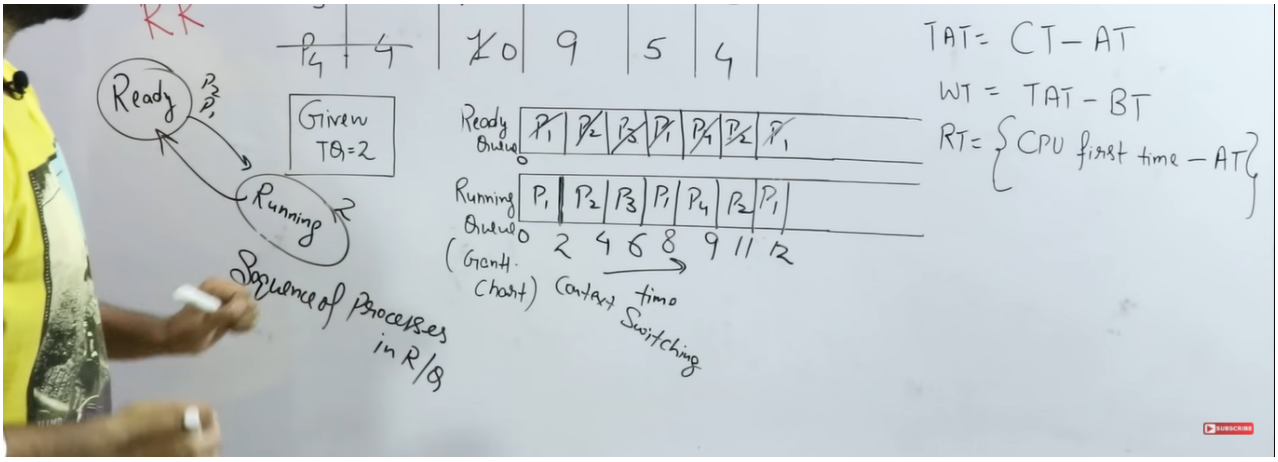


in case of preemptive Response time is very important in above e.g.:

Preemptive means the process after sertain time quntum will go back to the ready queue!

Round Robin : TQ is applicable after TQ process will be sent back to the



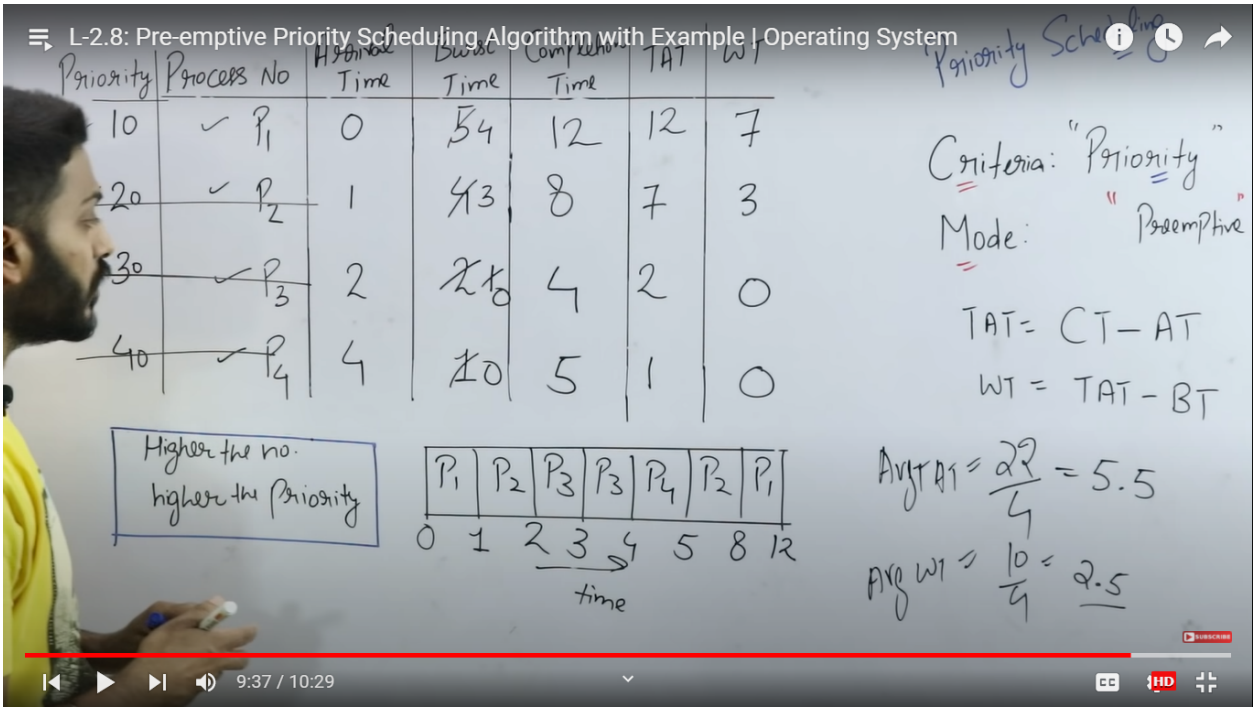


Note: In above eg check the P4 CPU scheduling here burst time is less than TQ (2) i.e. 1 so it will take only 1 unit of time for execution and will take the next process for execution

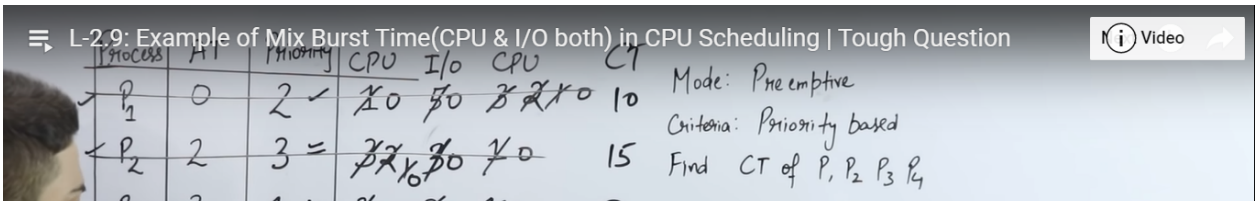
Context Switching : SAVE THE RUNNING PROCESS and sent back to the ready queuye after TQ/ This can be calculated by the no of vertical line betwen the process on the gantt chart!

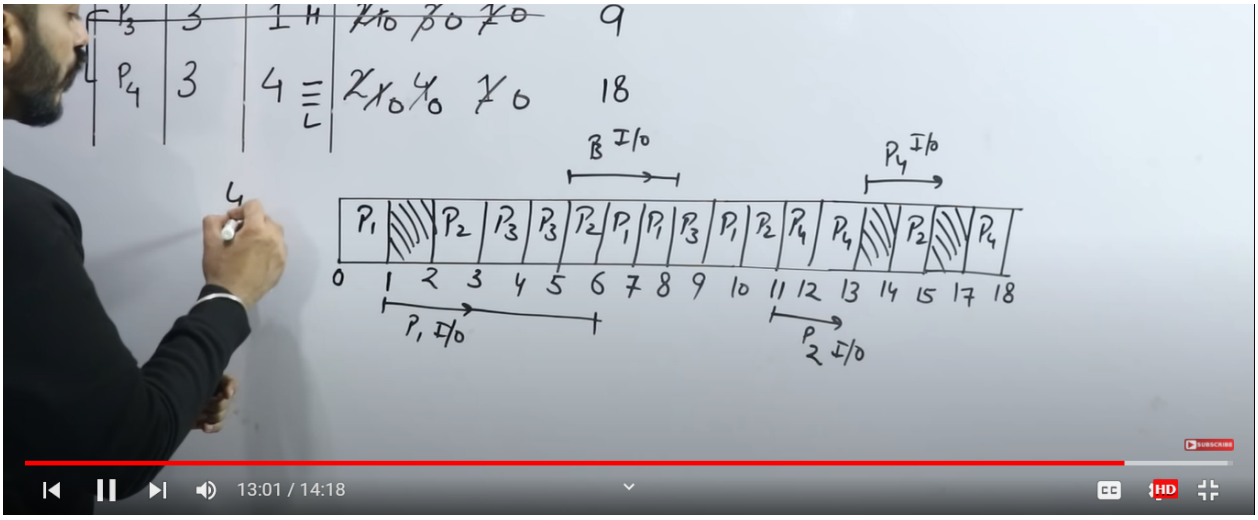
Pre-emptive Priority Scheduling Algorithm with Example | Operating System

In this case value of priority is checked! first check is arrival time and then run it for 1 unit time and check the priority after every 1 unit!

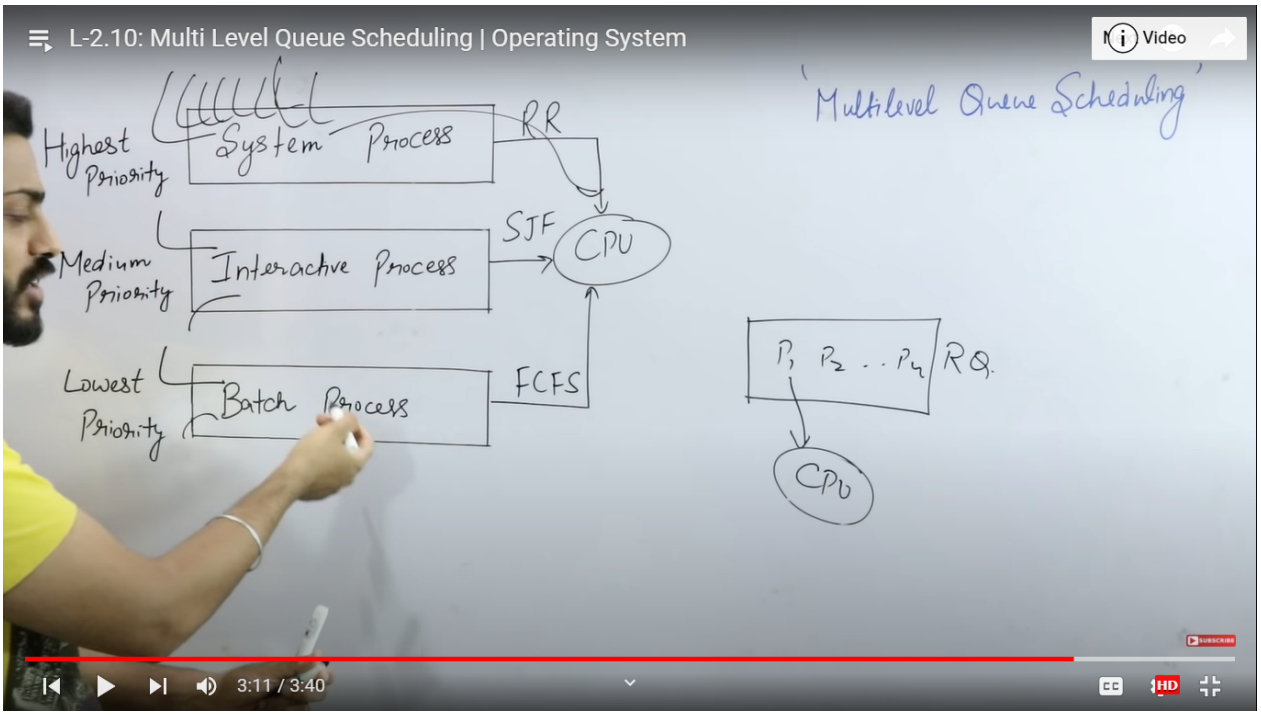


If the priority is same then run it as per the arrival time and if the arrival time is also the same then run it as per the sequence given.





Multi level Queue Scheduling :



Advantages :

- Different ready queues are created as per the real life examples because every process is diffenrent
- for every single queue diff algorithm can be set

Disadvantages:

- Starvation : as if high no of system process came then low prioriy process wont get time for execution

Multilevel Feedback Queue Scheduling | Operating System

Process Synchronisation:

Independent process:

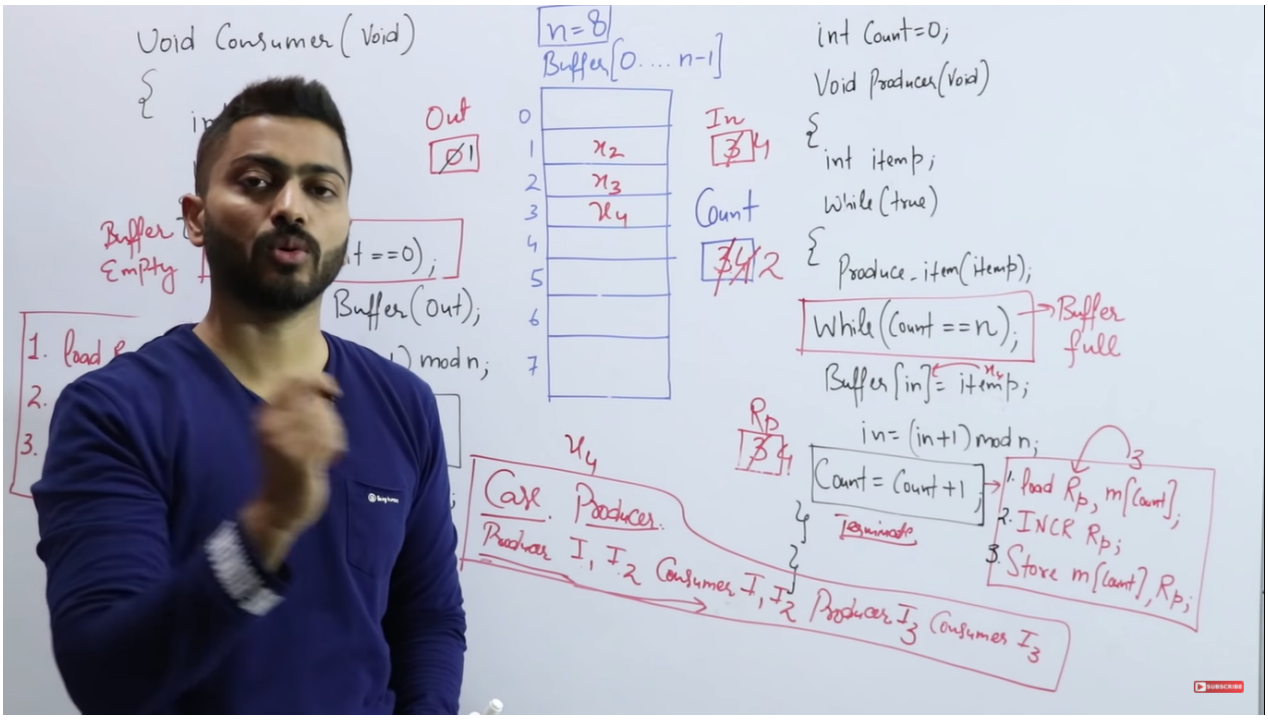
Co operative process:

- Share variable, Memory, code Resources etc

- Race Condition for using the common resources or data which is responsible for the loss of data and deadlock!
- These co operative processes creates a problem and these needs to be synchronised
- Deadlock situation may occur due to this co operative process

Uniprocessor : Single CPU system

Producer and consumer problem:

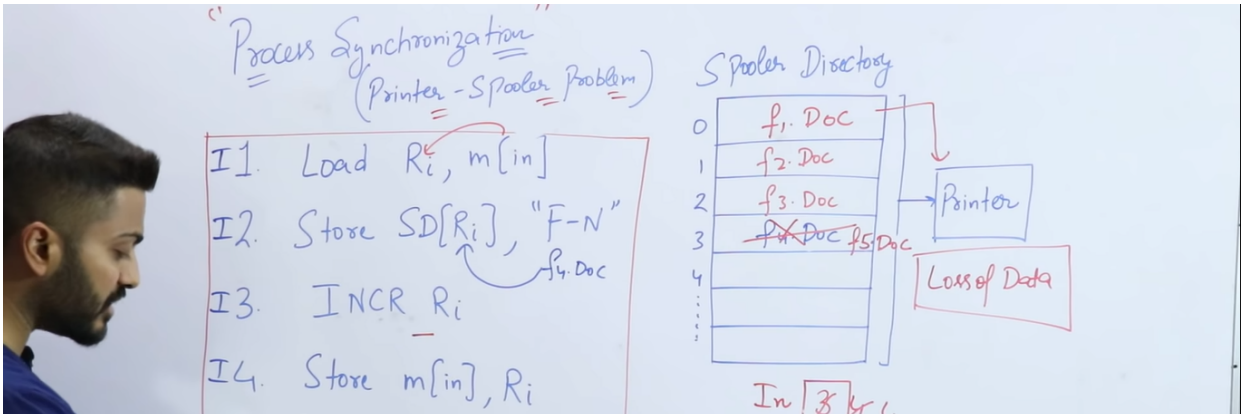


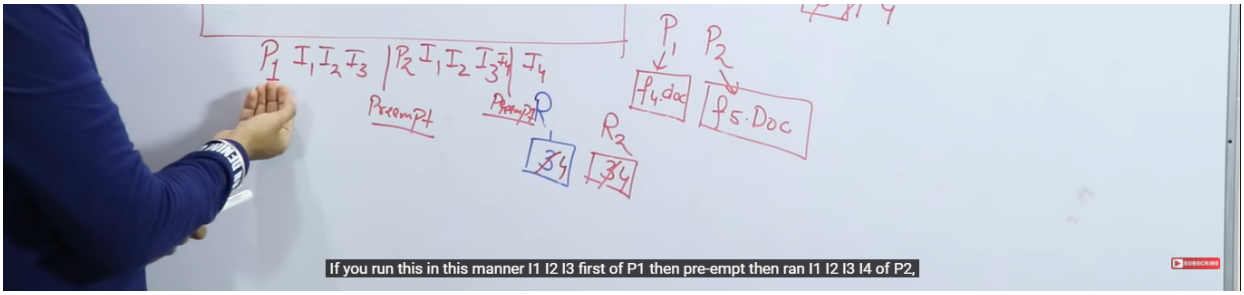
To avoid such issue below methods can be applied:

- Binary Semaphores
- Monitors
- locks
- etc.

Printer Spooler problem:

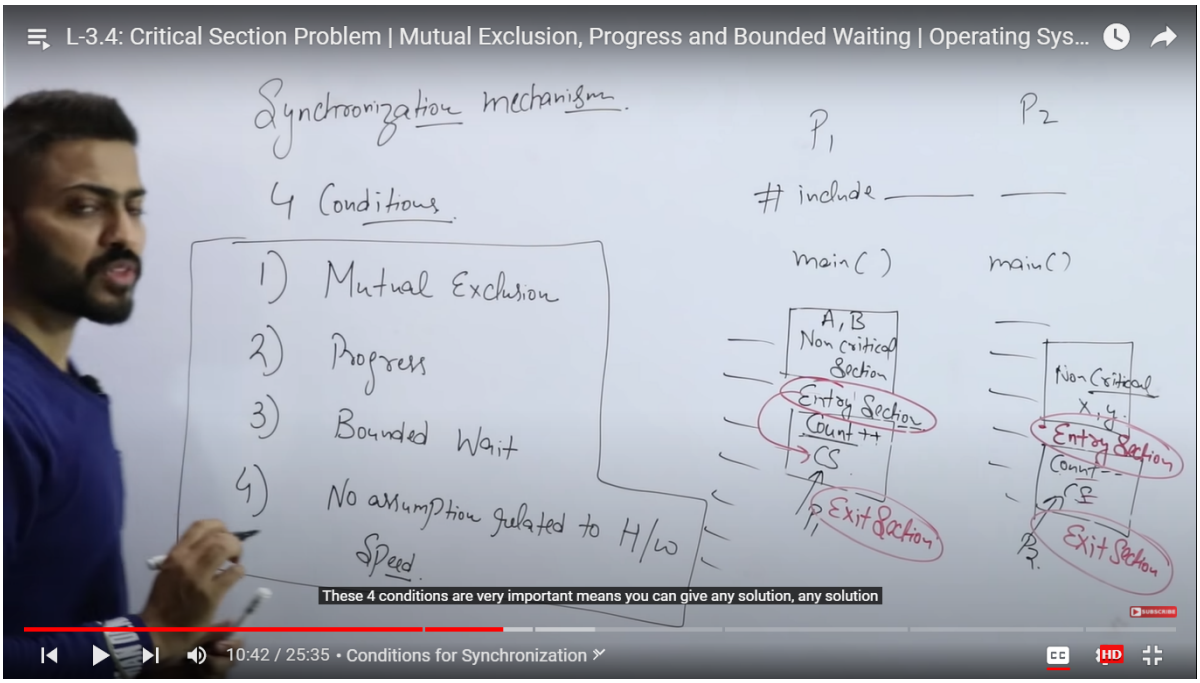
Printer is basically a very slow device as compared with the CPU,RAM etc. and Spooler is the software in printer which store the no of file sent back to back for printing,





Critical Section:

Place or the portion of the process where shared variables or resources are placed!



Bounded Wait:

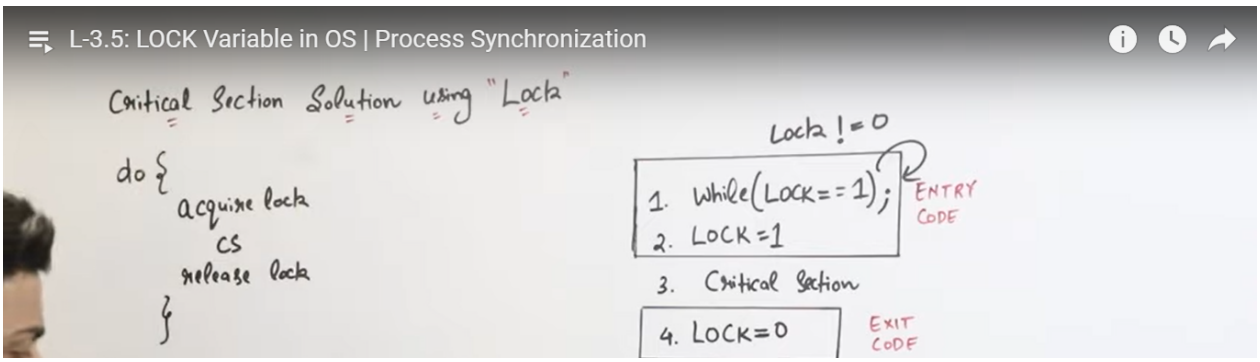
- When p1 in insdie the CS, P2 WILL BE OUT SIDE But this should not happend that p1 is always inside and p2 is not getting a chance to get inside the CS (Critical Section)

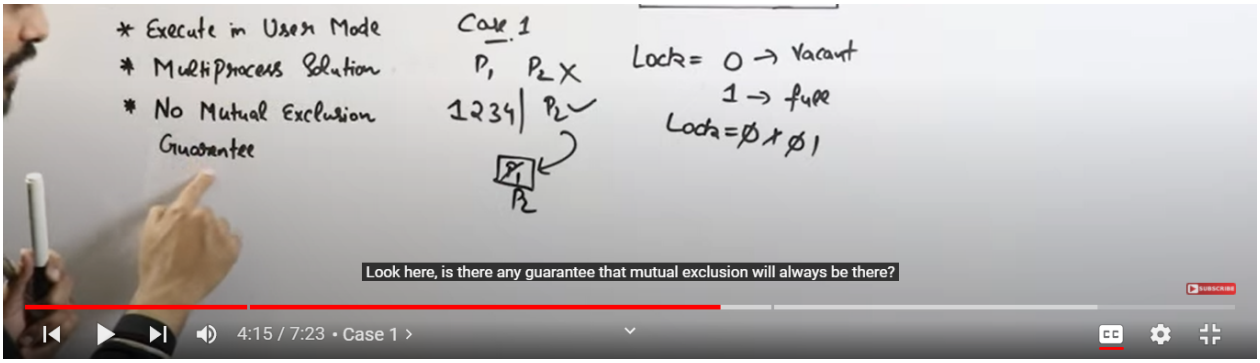
No assumtpion related to Hware or Speed:

- if synch is made for 32 bit system it should work for 64 bit too.
- All synch code must be universal, irrespective of hardware and speed of the proicess.

Lock Variable in OS :

- Oldest method :

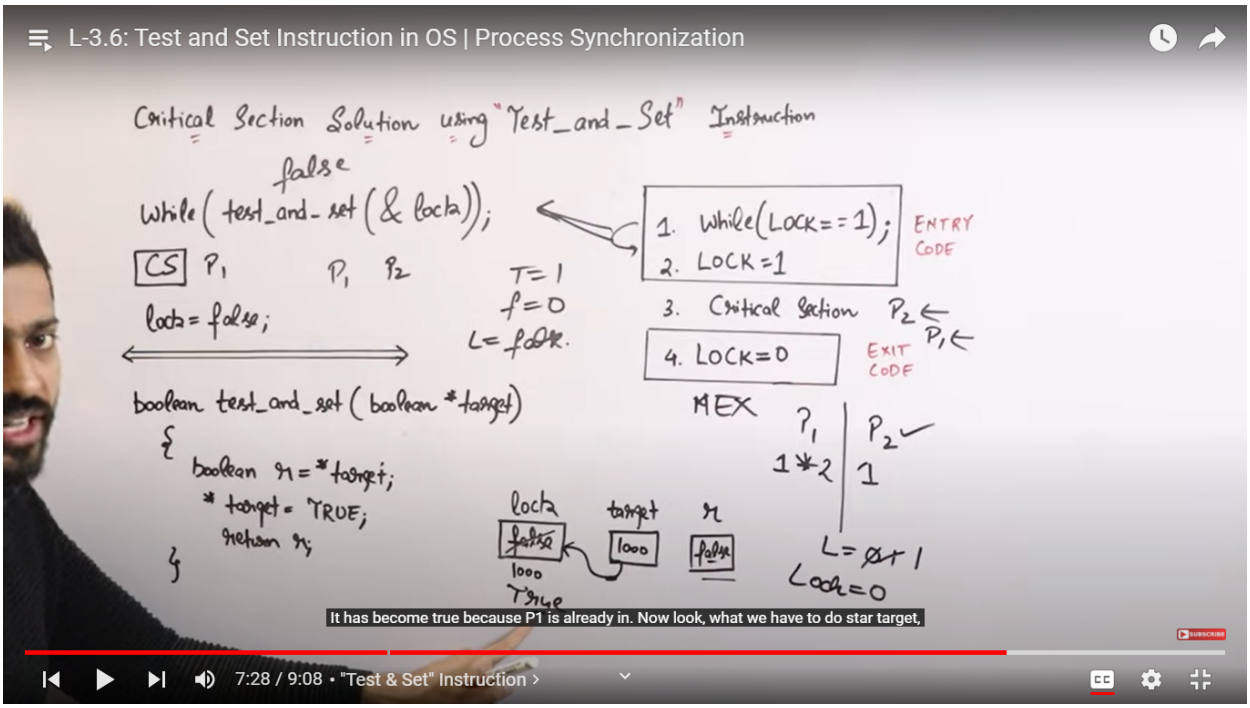




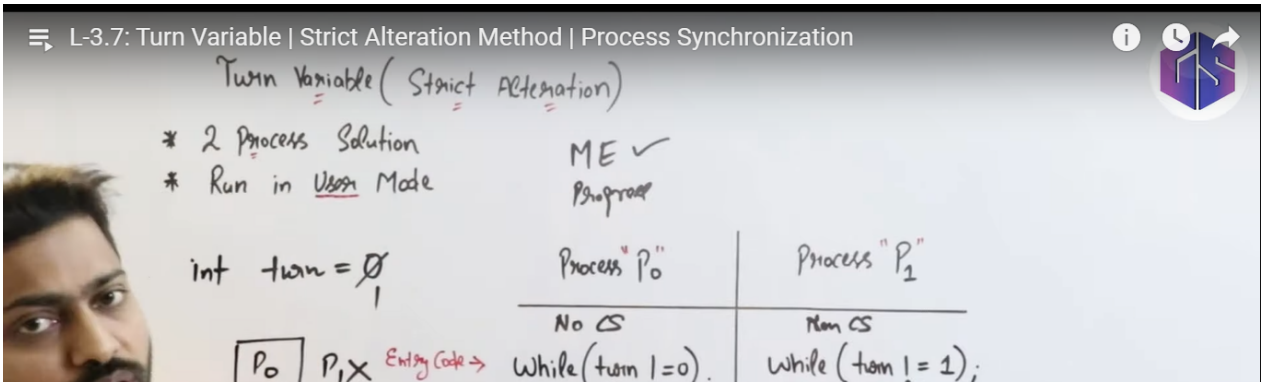
This process doent gaurantees the mutual exlusiveness, as if P1 got preemptive immidietly after while whichout making lock==1 so lock =0 and P2 will enter into the CS and will execute the code and will get out with lock =o and now when p1 will again come back so it will take lock ==1 and enter the CS but by that time it's p2 who is already executed from the CS.

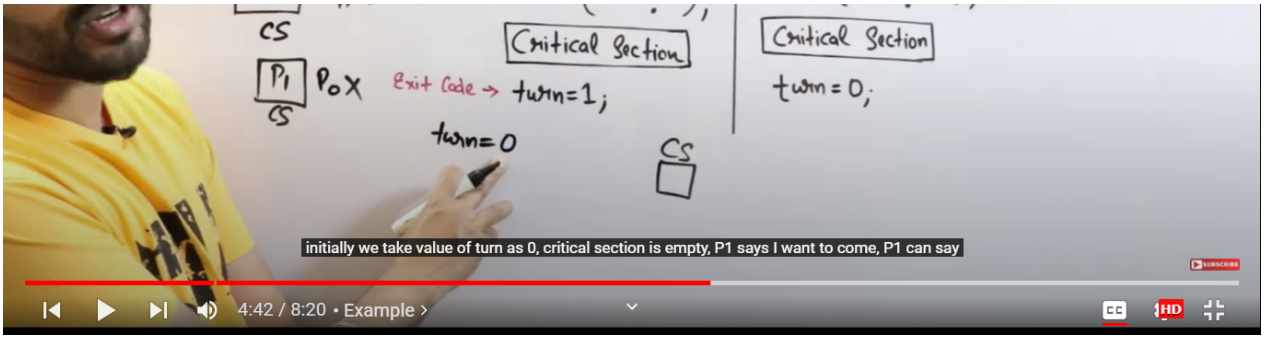
Test and Set instructions in OS:

To avoid the above error, two line of code were mergered inn to one i.e. test_and_set and now its wont give any error!



Turn Variable (Strict Alteration): This follow all the condition to make process synchronise except the process condition:



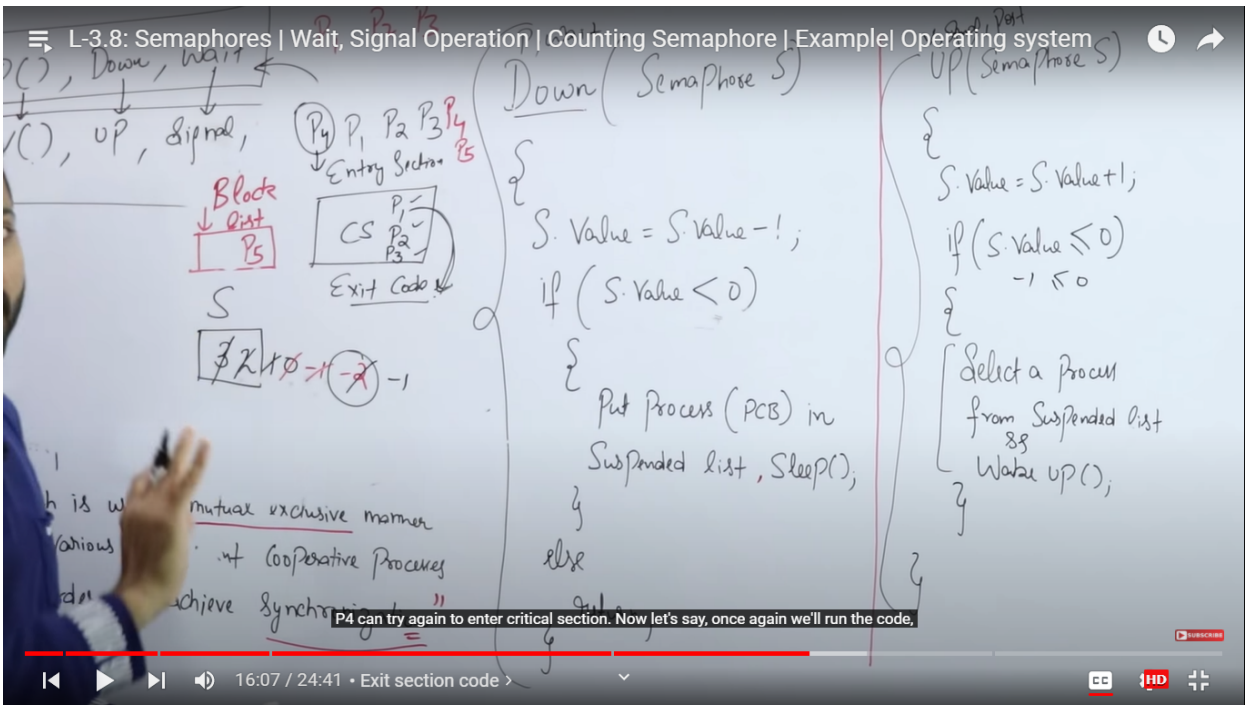


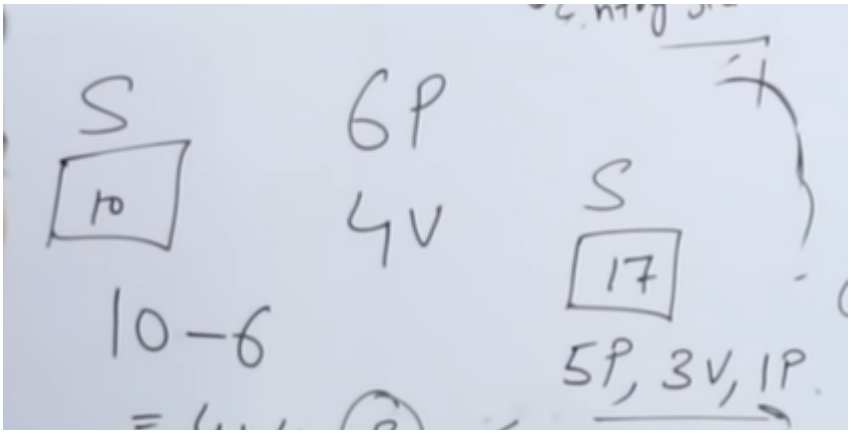
Semaphores | Wait, Signal Operation | Counting Semaphore | Example

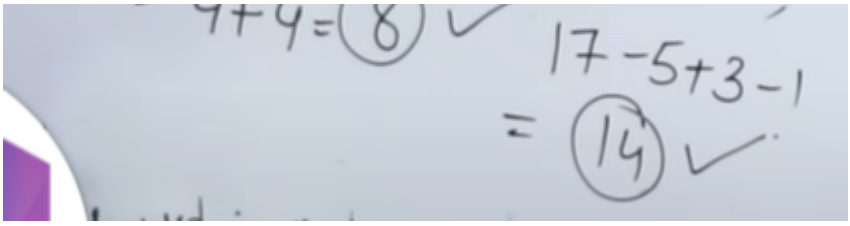
entry :p() Down wait,

exit: v(), up, signal, post, Release

- It's an integer variable
- which is used in an mutual exclusive way by concurrent co operative maner to achieve synchronisation
- value frpm -infy to +infy and in binary from (0 ,1)
- Common code is inside the Critical section and entry code and the exit coide written OUTSIDE the CS

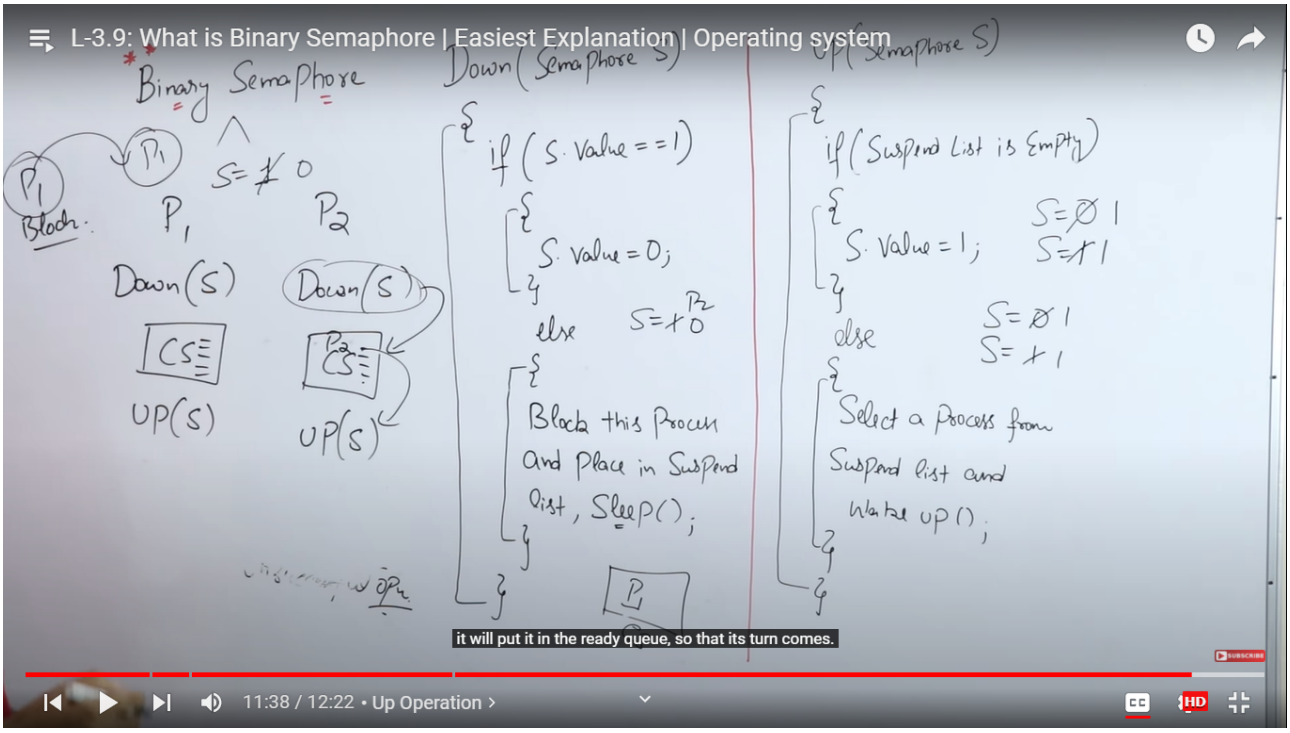


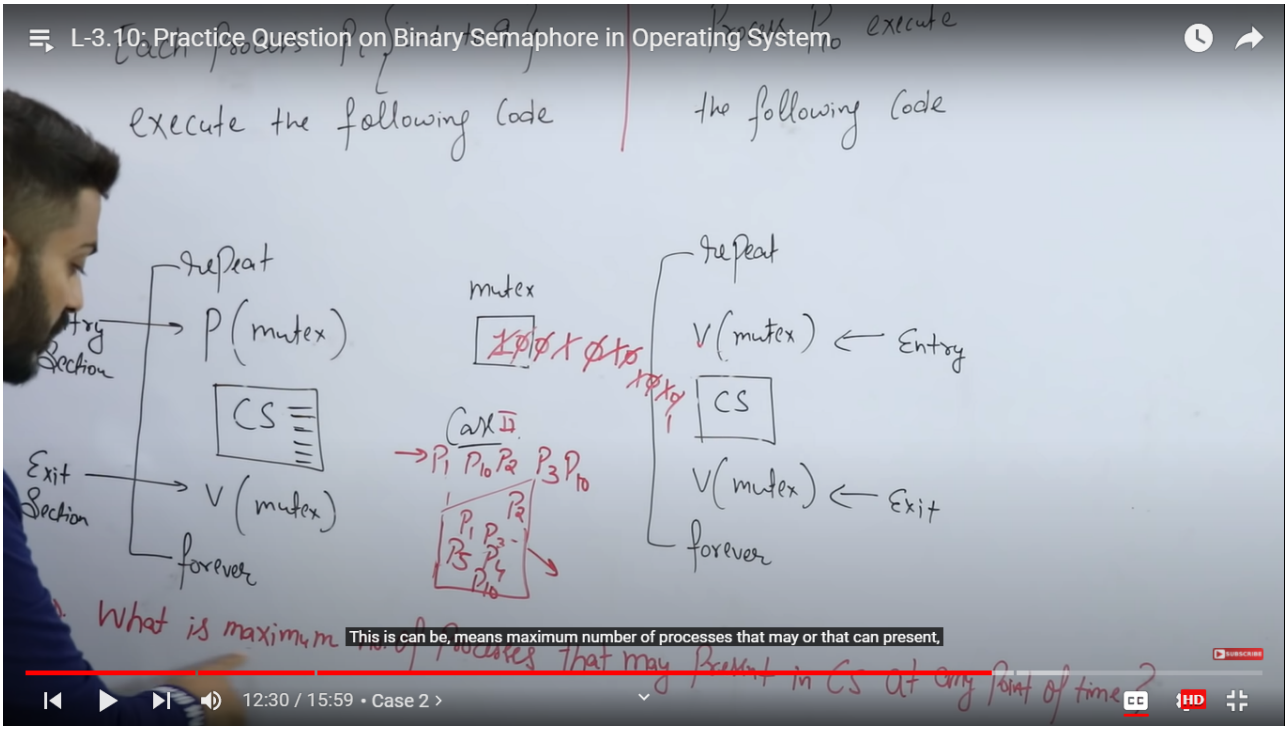


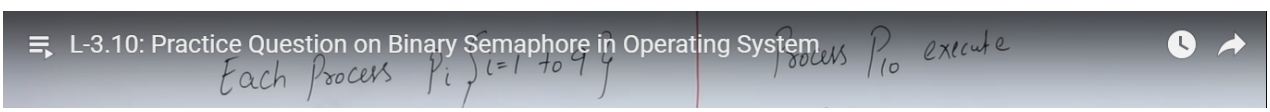


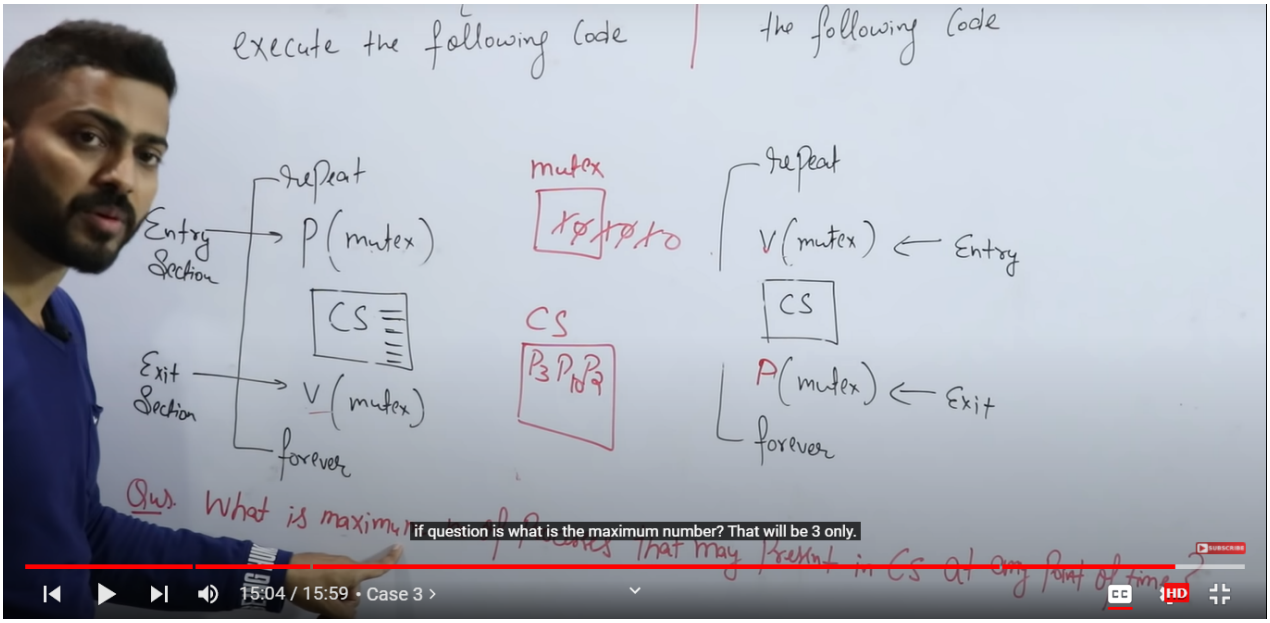
What is Binary Semaphore

- Here s deals with only 0 and 1

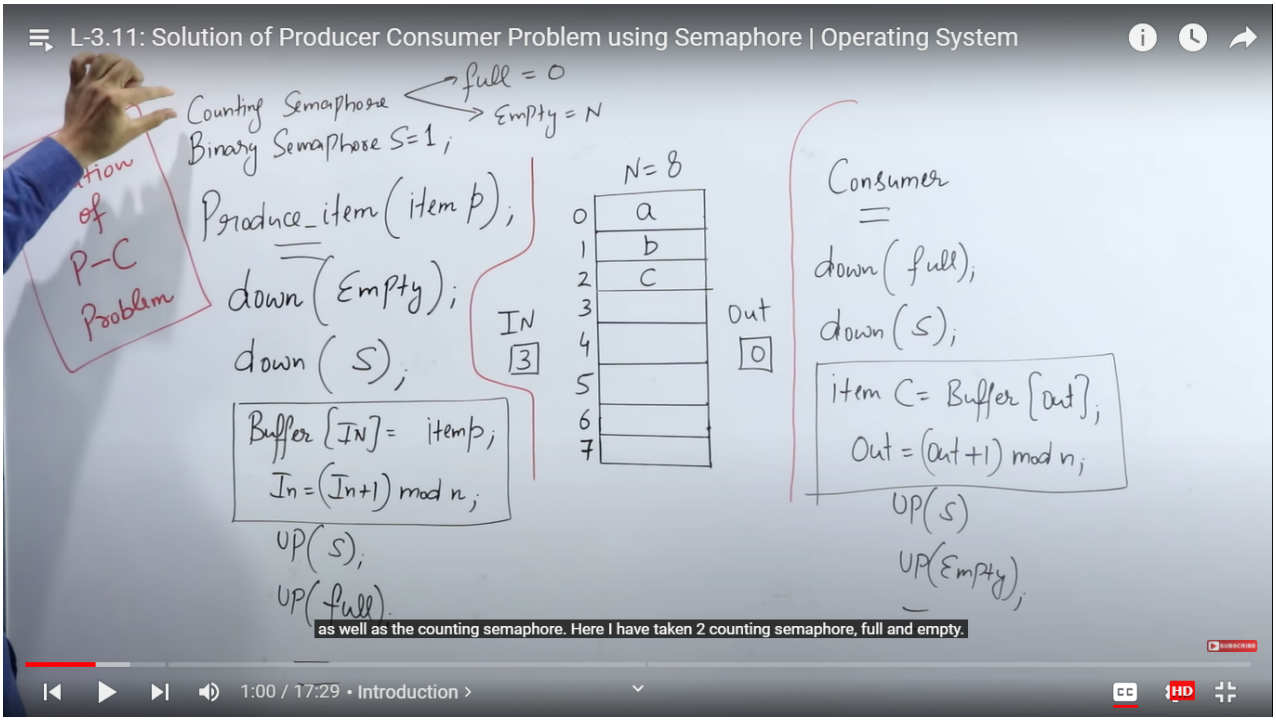


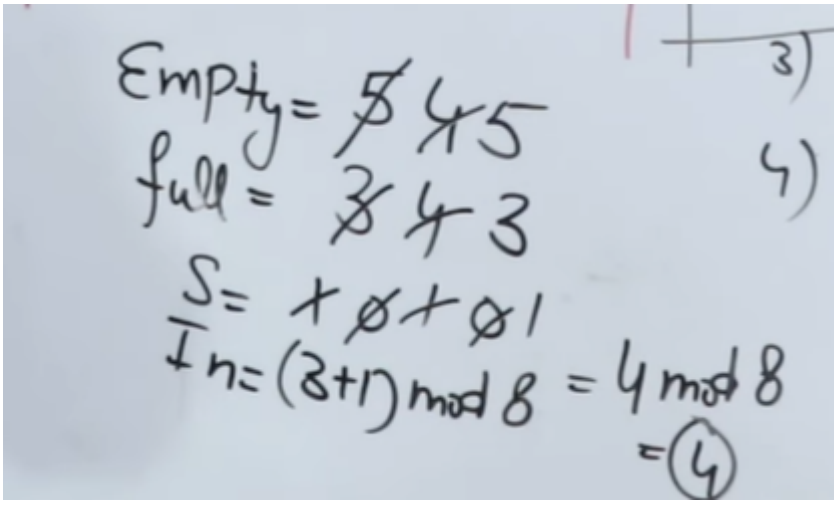






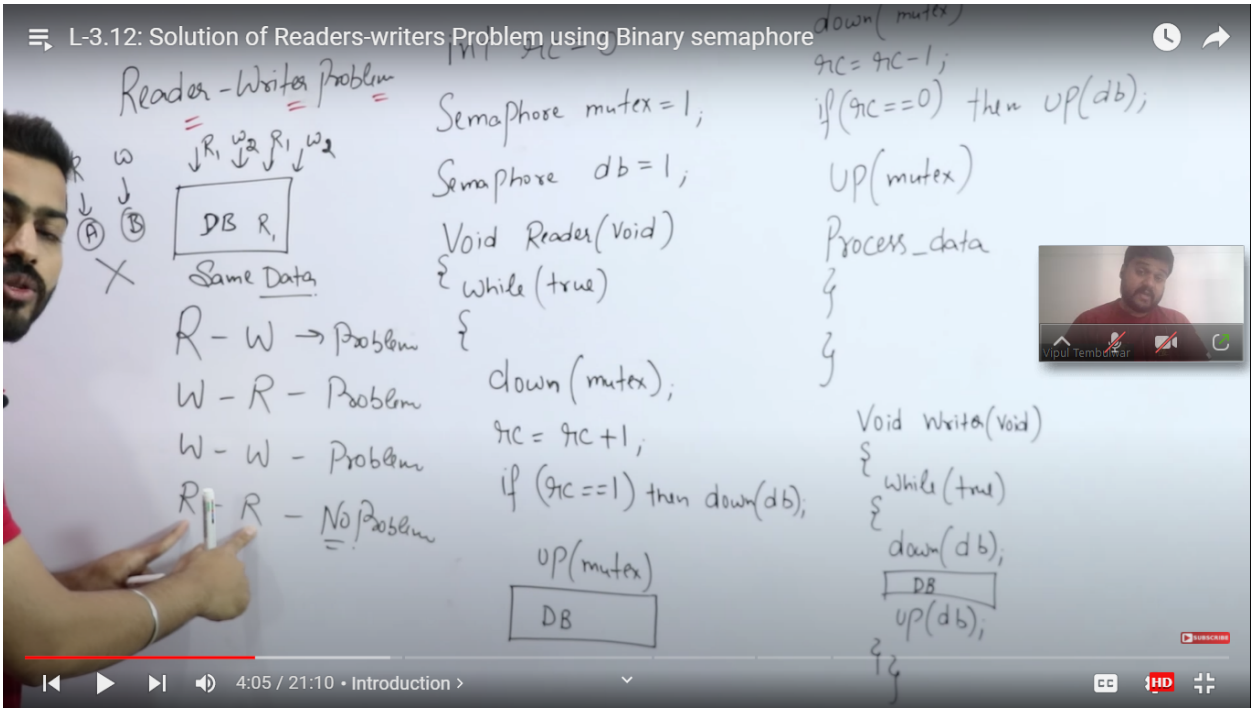
Solution of producer Consumer problem:





Reader and Writer problem using the semaphores:

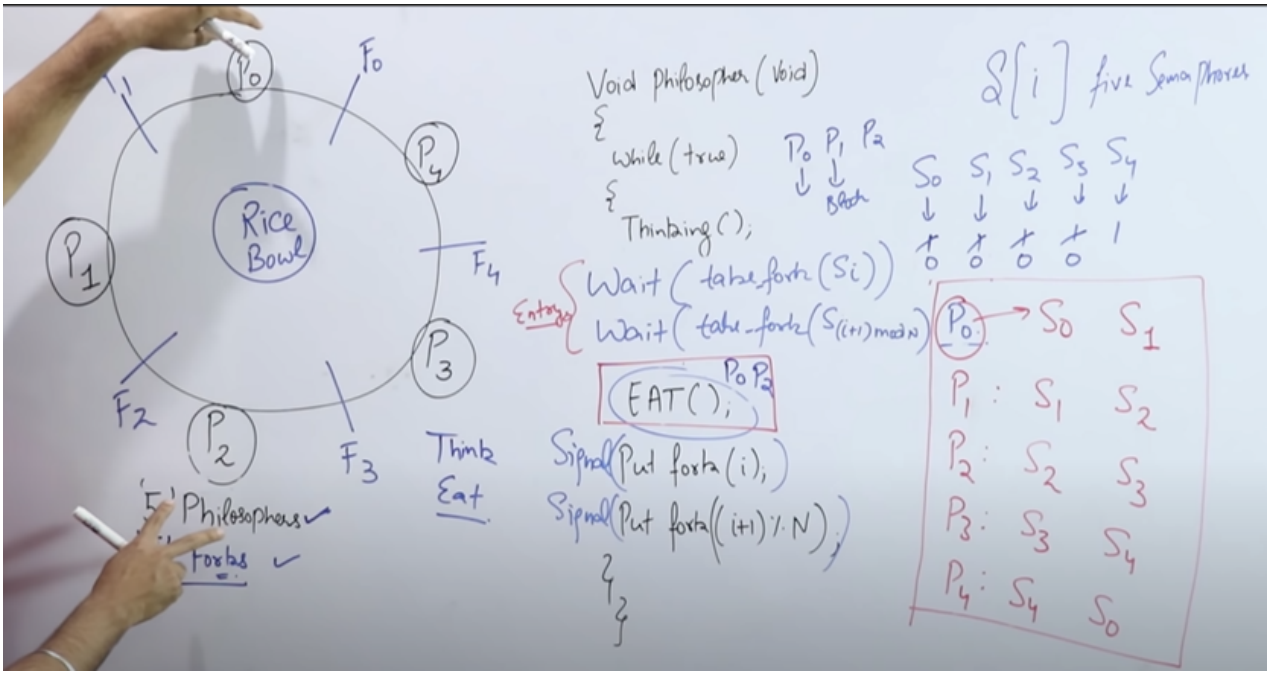
In this problme reader and write can not enter the database or it will create the problem with the data! to avoid this process is synchronised with the semaphores as below:



Dinning Philosopher problem:

5 : Philosophers

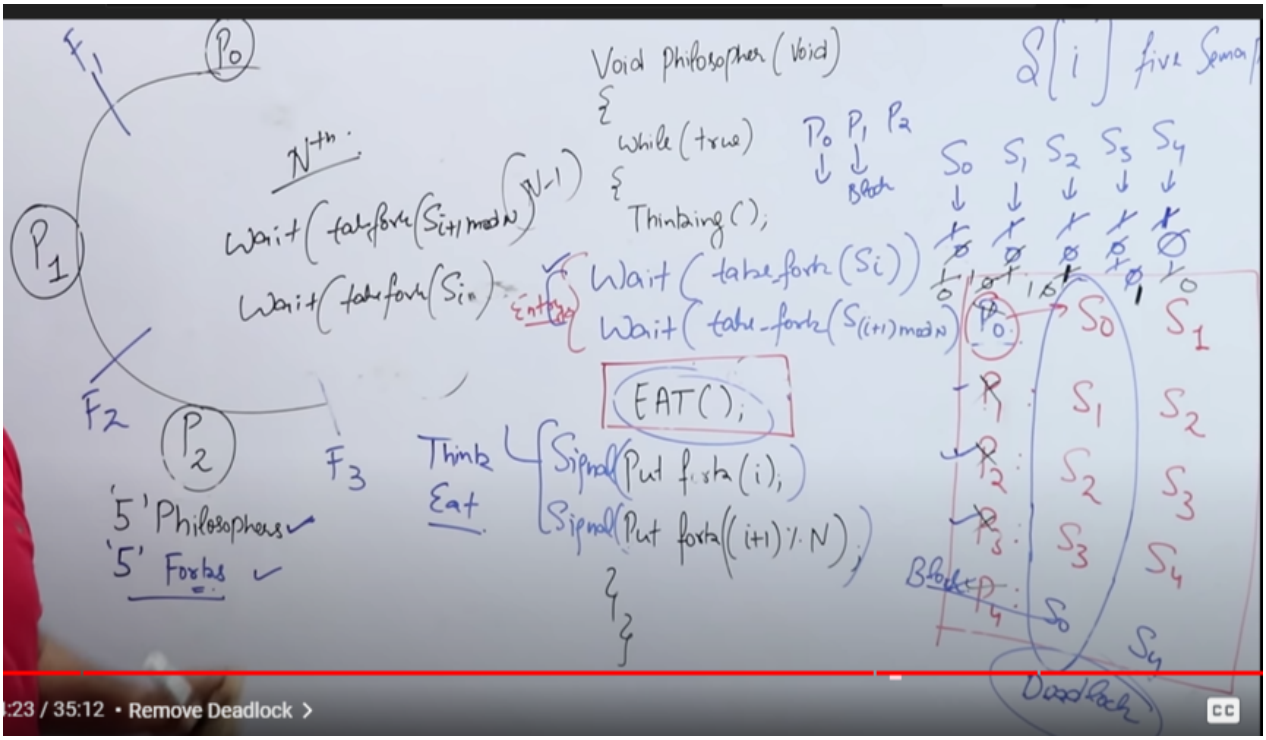
5 : Fork



Even with the semaphore also it might get blocked and deadlock will occur if in case process gets preempt after first semaphore condition

so to avoid the deadlock we will reverse the condition for the last philasopher in which he will go to pick up the right fork first instead of left and it will get blocked and then P3 can eat() with fork 4 and then cycles continues and finally p4 will come out and will eat!

Note : We can apply above condtion to any one philosopher!

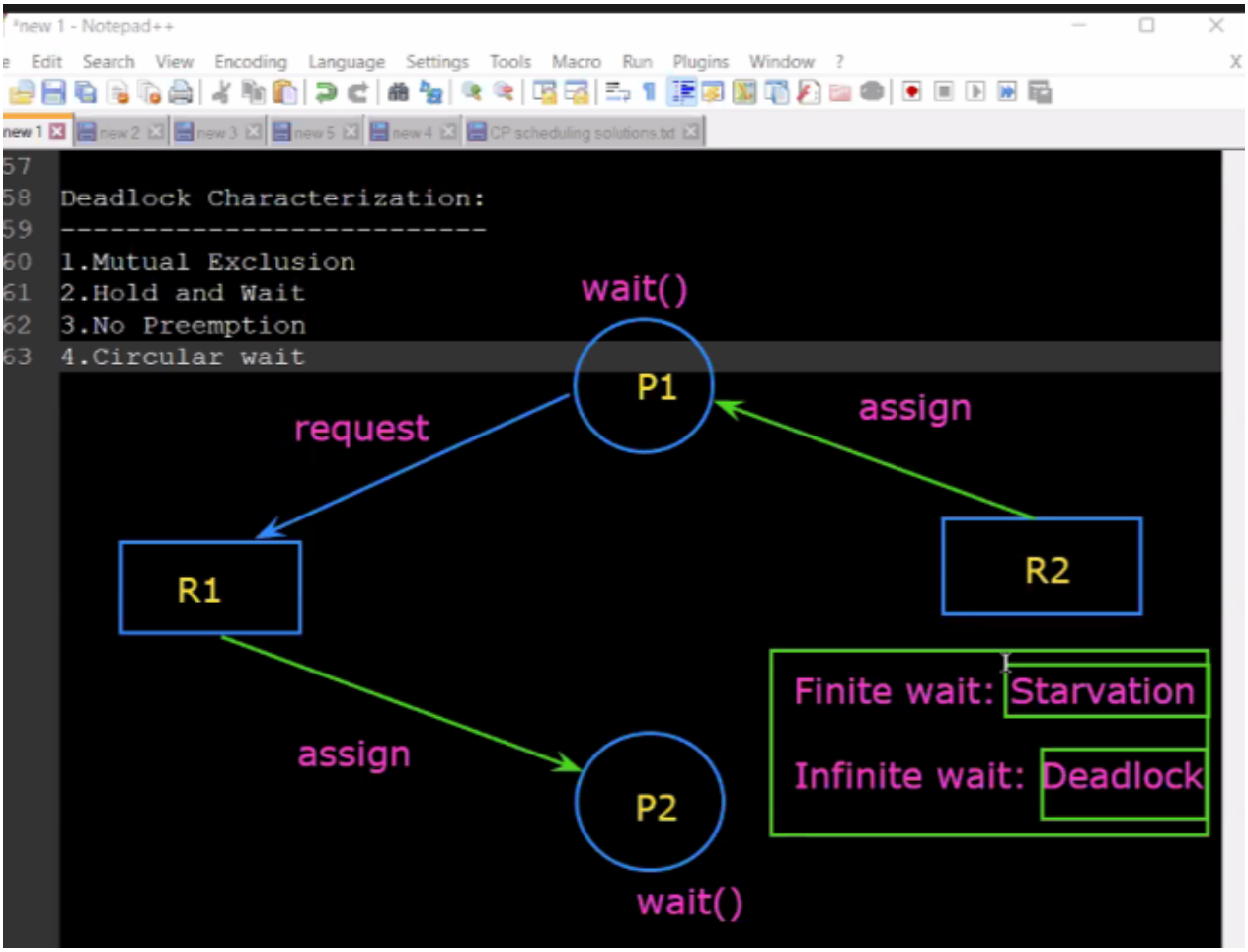


Deadlock :

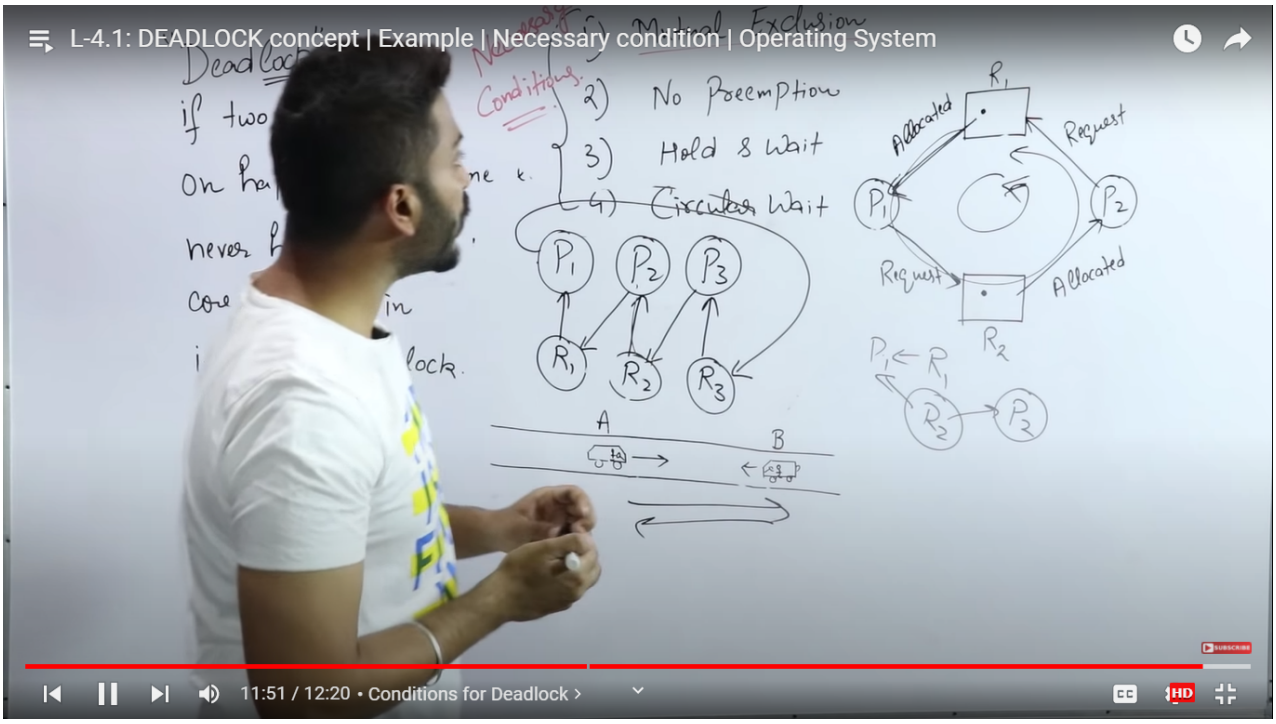
If two or more process are on happening some event which never happens :

A process in OS uses different reources and uses repurces in following way:

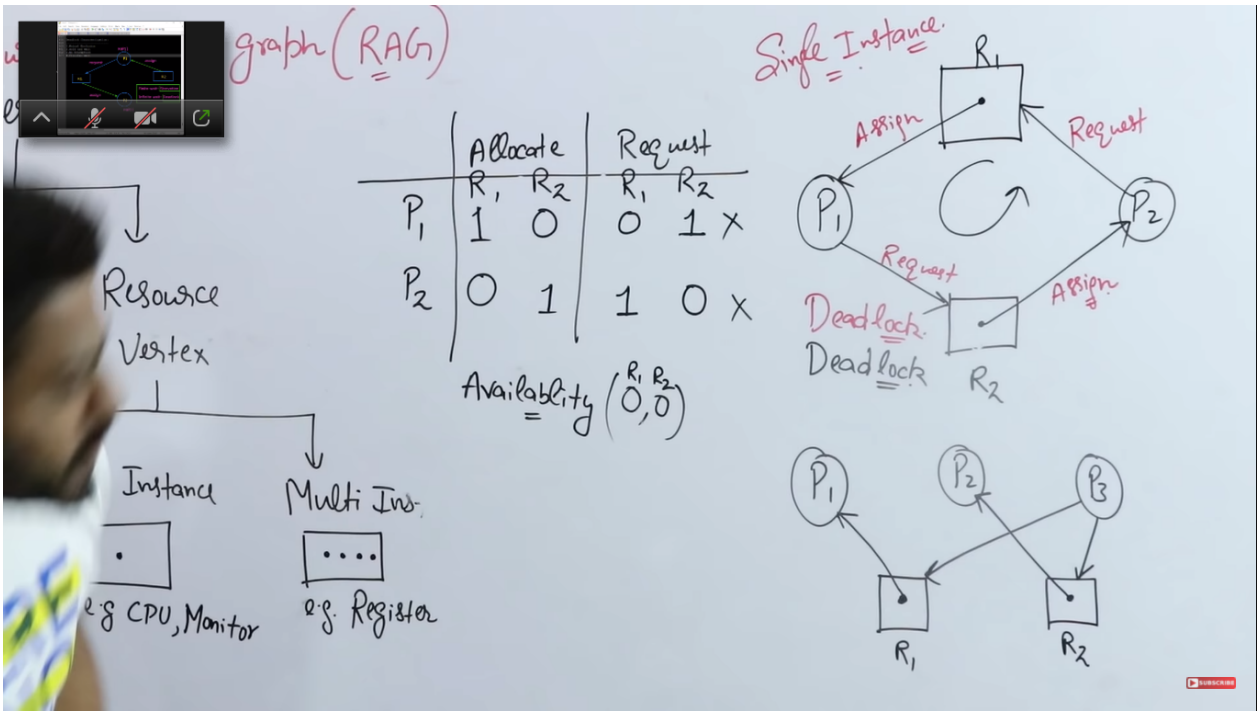
- Request a R
- Use R
- Release R



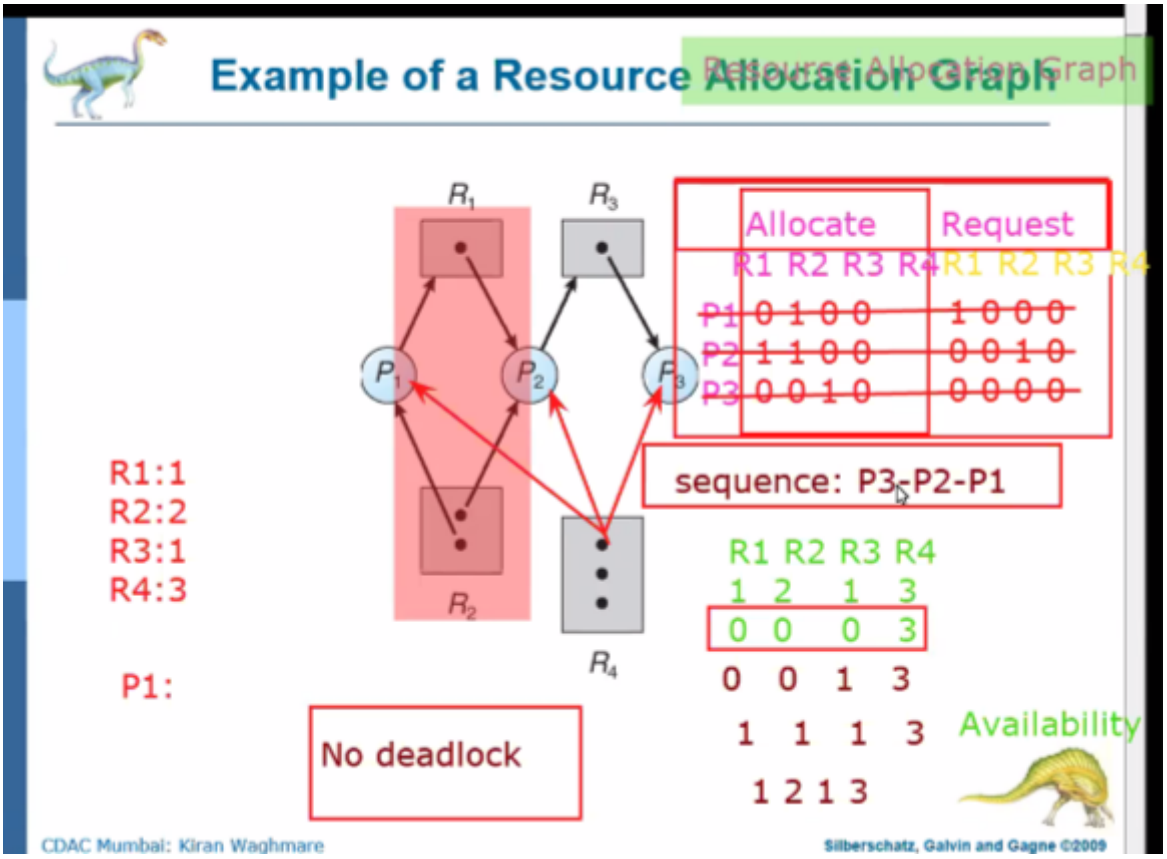
Four neccesary condition for deadlock:

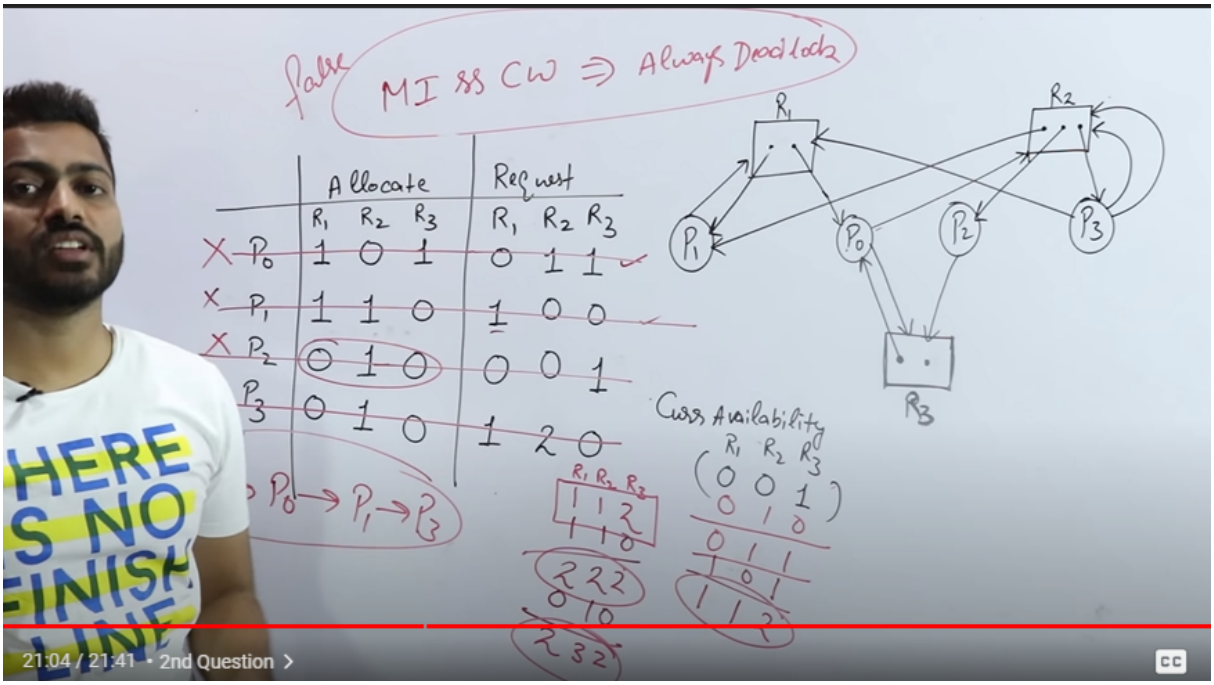


RAG : Resource allocation graph:
Most efficient and convenient way to represent the state of the system.
representation of how the resources have been alocated to the processes and how the processes has been assigned the multiple processes!
RAG is used to check if we have deadlock in the system or not

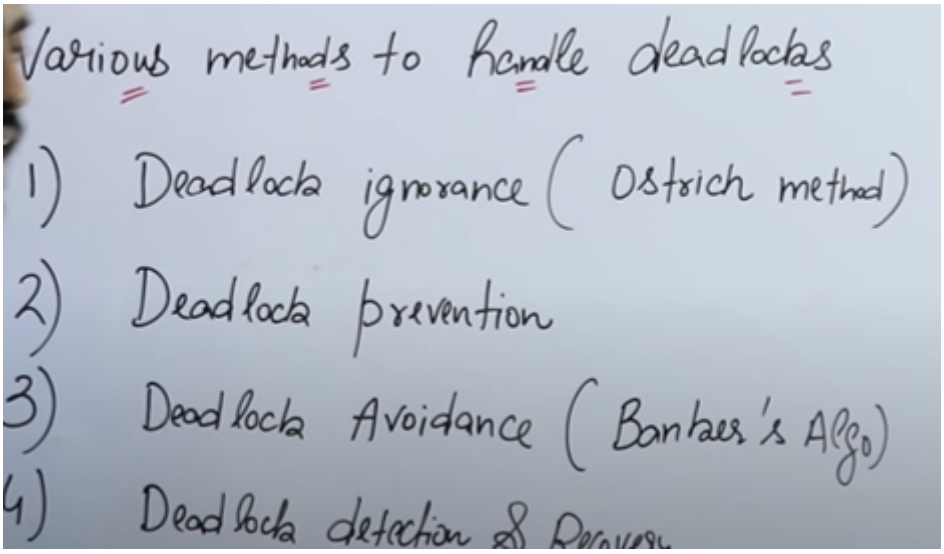


In single instance verted for cyclic wait there will be always deadlock.



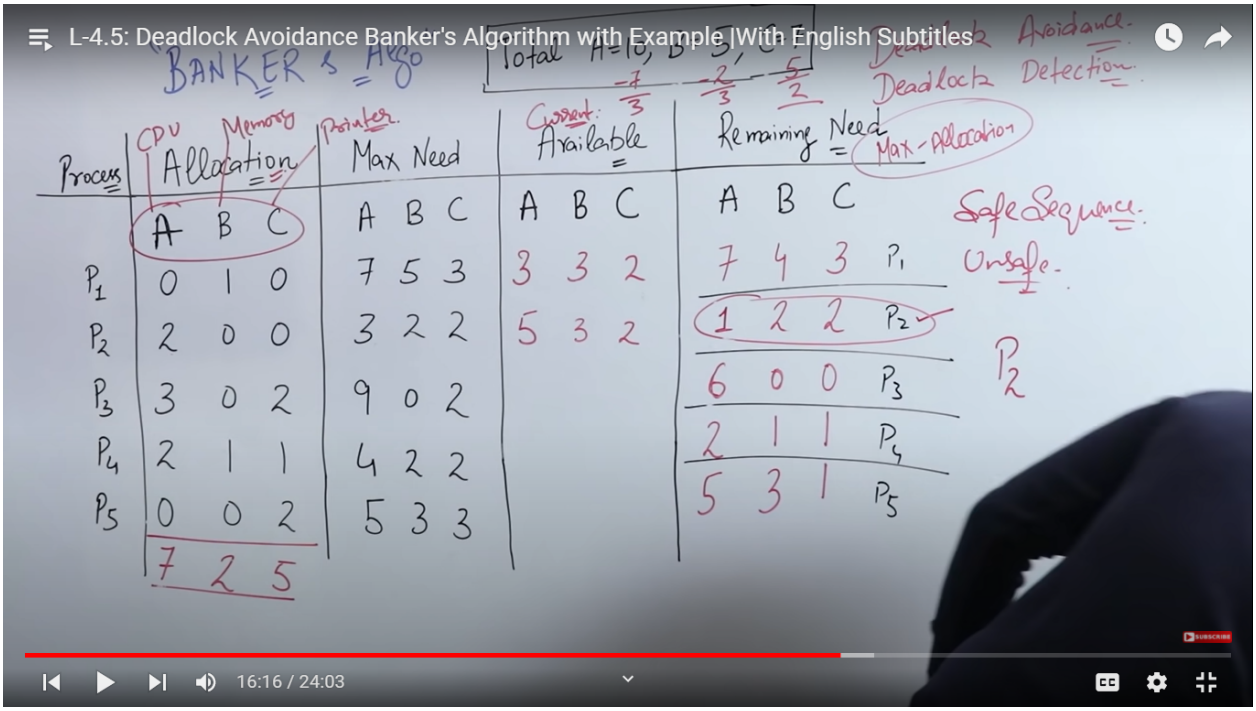


Various methods to deal with the deadlock:



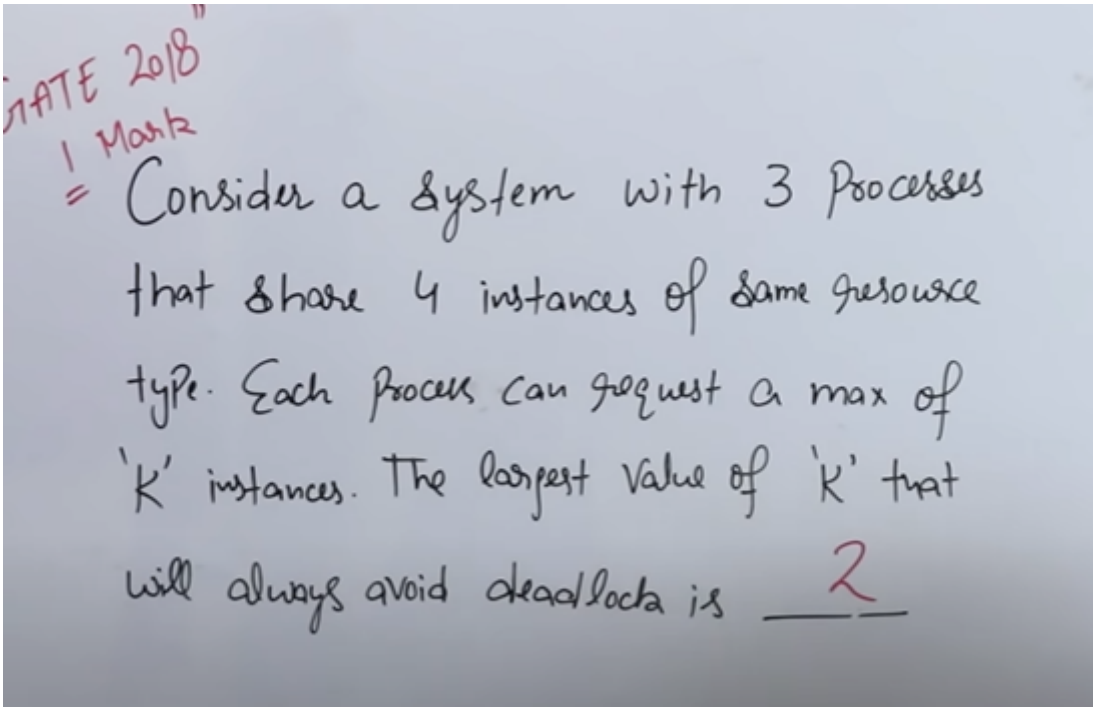
1. Deadlock occurs very rare i.e once in a 5-6 years so developers have decided to ignore the deadlock instead of writing the heavy code for it and to avoid performance damage its better to ignore it e.g. sometime the deadlock is resolved only when we restart our PC
2. DP : In this case any process can access the resources in increasing order only with this circular wait situation is avoided
3. Banker's algo resolves the deadlock issue
4. Deadlock detection and recovery : It's bit complex system in which

Deadlock avoidance with the banker's algorithm:



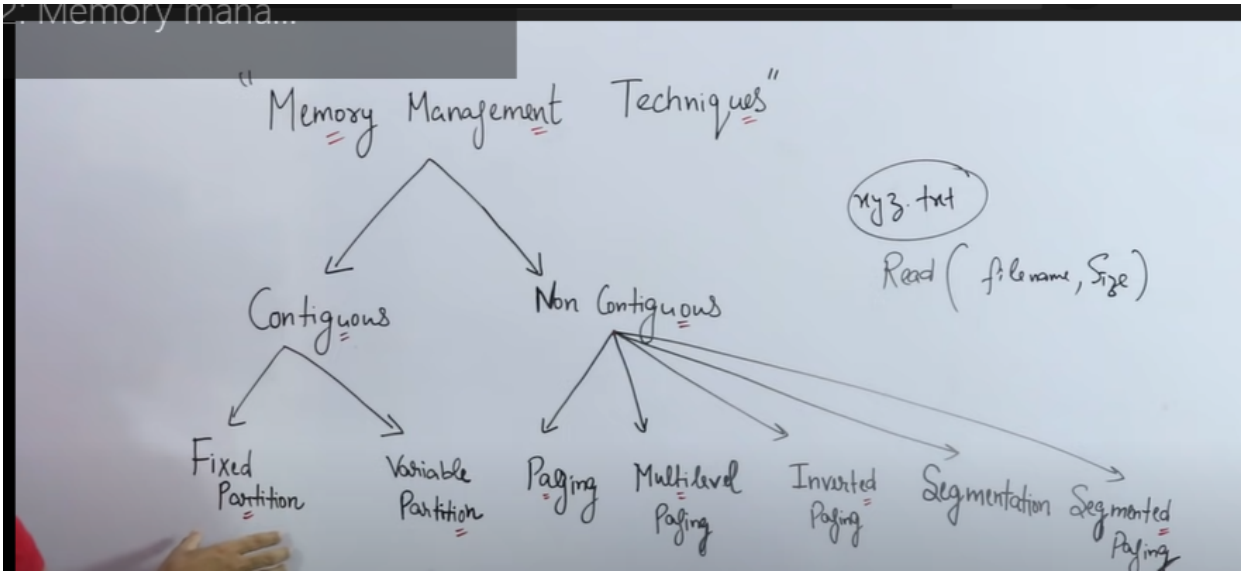
This is not possible in real life as in reality processes demand for the resources in dynamic manner so it becomes difficult to manage the things!

Safe state / sequence : In this case all proicess gets the required resources and all process gets terminated one after the other, in this case there is deadlock.

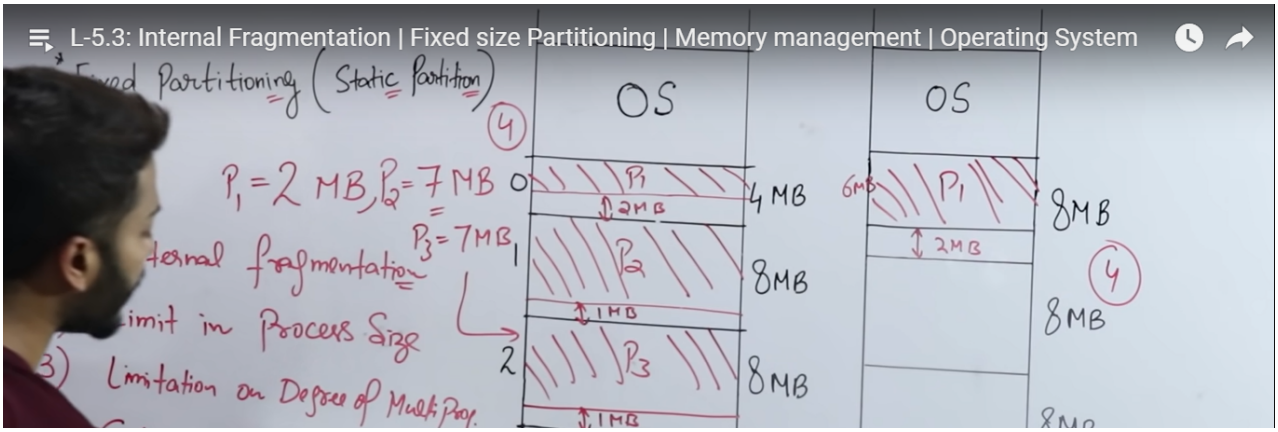


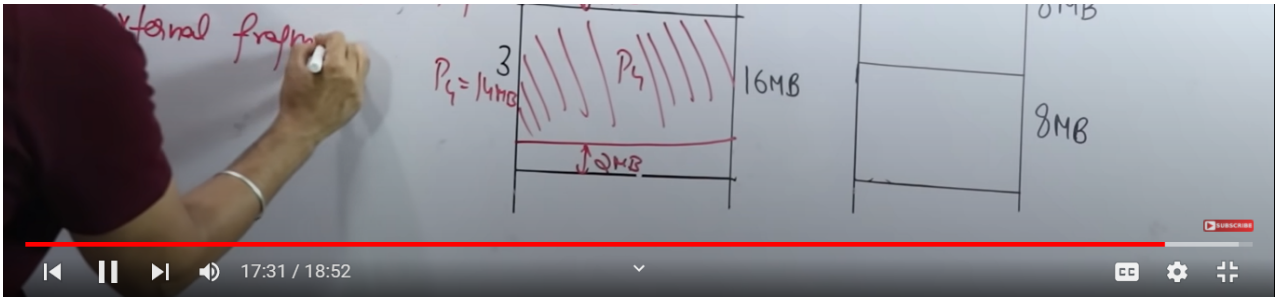
Memory Management and Degree of Multiprogramming

Degree if multiprogramming



Internal Fragmentation | Fixed size Partitioning





Variable size Partitioning