## OPERATING SYSTEM

Operating system is a bridge between your user and your hardware.

Your Laptop is a hardware and to communicate with that hardware the bridge between you and the laptop is operating system.

It makes a platform over the hardware, a platform where the user can run his application software to perform a specific task.

Operating system is responsible to manage:

1. RAM
2. Processor
3. Hard Disk
   Etc.

## OPERATING SYSTEM SERVICES

1. Program Execution
2. I/O Operations
3. File-System Manipulation
4. Communications
5. Error Detection
6. Resource Allocation
7. Accounting
8. Protection and Security

## SYSTEM INTERRUPTS

Meaning: Stop the continuous progress of an activity.

The interrupt function in OS does the same.

Interrupts can be software or hardware.

### INTERRUPT HANDLER

After a user makes an interrupt, some instructors are given for further to let the system know why the interrupt was generated.

Interrupt Handler is a special type of code block, the control is directly given to interrupt handler, it then executes the code that is placed in the handler.

### INTERRUPT VECTOR

Interrupt vector are special type of codes, that contains the address to interrupts.

It prioritizes the interrupts and saves them in a queue if more than one interrupt is waiting to be handled.

## KERNAL

Whenever we say OS is responsible for doing this and that, 99% of the times we are referring to Kernal.

### TWO TYPES OF KERNAL

1. Micro Kernal: Allows basic operations.
   Example:
   Smart Watches

2. Monolithic Kernal: Allows advance options.
   Example:
   Linux, Android, Chrome, Firefox.

   Windows Kernal is a Hybrid Kernal.

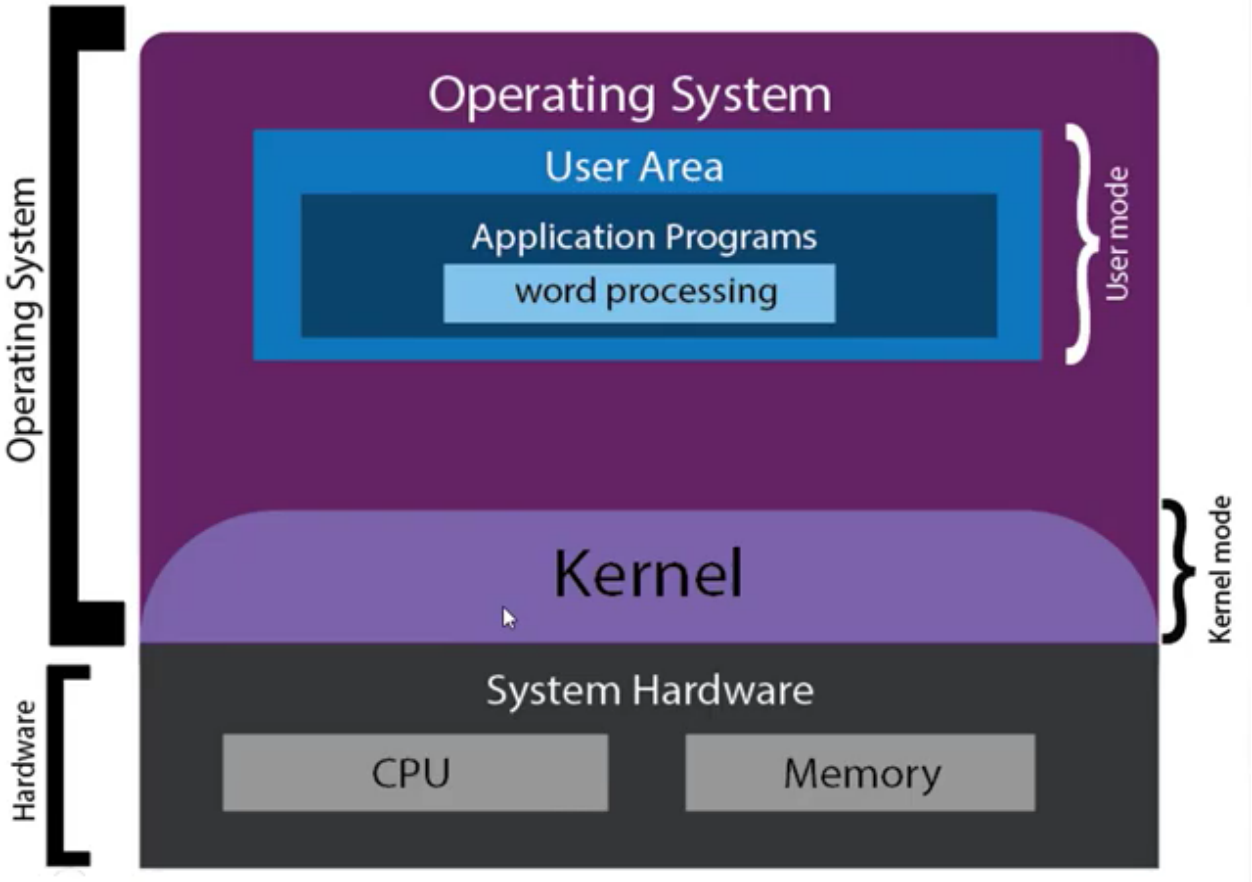
## MULTIPROGRAMMING VS. MULTITASKING

- CPU get one job out of the pool and start executing it.
- Now between the execution there are 2 case look what
- CASE 1: CPU will get the job and execute it until it finish.
- CASE 2: CPU will be keep executing the job until the JOB itself ask the CPU to wait, as I am going for some I/O operating.
  - When the job go for I/O operation, What CPU will do?
  - It will not sit idle; it will choose some other job and Execute till the old one come back.
  - Switching from one job to other is known as **Context switching**.

# Multitasking OS Concepts. . .

- Multitasking means working on more than one task at a time means you are using computer and listening song also searching some thing on internet using browser and making word file that's your assignment.
- This is you are doing multitasking it seems like all the task are happening in the same time.
  - Its not all the task happening at the same time it's the processor switching between the different tasks in so speedy manners you think that they are happening at the same time.
  - Lets go on the next slide to see how it happens behind the scene.

## USER-MODE AND KERNAL-MODE




User sends a system call, that is then executed in Kernal mode.

## MODE BIT

Mode bit is a bit which tells you which mode you are in.

Kernal Mode = 0 bitq

User Mode = 1 bit

## OPERATING SYSTEM GENERATION

In computing System generation or sysgen is the process of creating a particular unique instance of an operating system by combining user-specified options and parameters with manufacturer-supplied general-purpose program code to produce an operating system tailored for a particular hardware and software environment.

## PERFORMANCE TUNING

Proper OS tuning improves system performance by preventing the occurrence of error conditions. Operating system error conditions always degrade performance. This performance can be improved by removing bottlenecks.

An OS must provide means of measuring and identifying behavior of a system,

For Example: "Top" program or Task Manager.

## TRACING

Tracing involves a specialized use of logging to record information about a program's execution.

strace – trace system calls invoked by a process.

gdb – source-level debugger.

perf – collection of Linux performance tools

tcpdump – collects network packets.

## PROCESS

A program in execution is process.

A passive sleeping calculator is a program, but when it is executed and the user uses it, it become a process. An active program uses system resources, GPU, Memory etc.

### ATTRIBUTES OF A PROCESS

1. Program Counter: Used to keep track of the address of the next instruction to be executed.
2. Stack: A stack is used to store the data of the program. E.g. Local Variable.
3. Data Section: Used to store the global variables of the program.
4. Heap: Heap is used for Dynamic Memory Allocation.

## PROCESS STATES

It states what the process is currently doing.

A process goes through 5 number of states.

1. New
2. Ready
3. Running
4. Waiting
5. Terminated.

### NEW

At this stage it us just a program, which is newly created and not loaded into the memory.

### READY

Program is loaded into the memory and waiting to be executed.

The Ready and Running state shift depends upon the scheduling algorithms.

### RUNNING

The program got the processor to start running.

It will perform all the number of instructions written into it.

Process keeps running until it finishes its task.

### TERMINATION

Now the process has been run, it has completed the tasks for which it was running, it will now be terminated.

### WAITING

Suppose a process is in running state, but after execution of some tasks, the process requires an input for the user to execute further, so now the process will wait for an

input from the user or some other program, so at this time the process is in waiting state.

After waiting state, the program will not return to running state, it will go into ready state. Cause during the waiting state, the CPU assigned the processor to some other process since the previous one was waiting for an input.

## PCB (PROCESS CONTROL BLOCK)

PCB in simple words is a data structure, and it is an ID Card of a process.

It is design to keep the data of a process, and every process has its own PCB.

Process use its PCB to store the data such as, Process state, Program counter, CPU registers, Memory, I/O Status, etc.

### WHAT KIND OF DATA IS STORED IN PCB?

Data which is used to control the execution of the process, data about the resources allocated to process to perform any task.

## PROCESS SCHEDULING

The number of processes is more than resources. We have only one processor, but more than one process.

The processes are not sensible enough to schedule themselves one after another, so the OS take this responsibility. OS Schedule these processes using the scheduling algorithms.

### SCHEDULING QUEUE

A place where the process waits to get the resources is known as Scheduling Queue.

1. Job Queue: Queue of the processes in the secondary memory.
2. Ready Queue: Queue of the processes in the main memory.
3. Waiting Queue: If the process is waiting for I/O operations it will wait here.

### SCHEDULING ALGORITHM

The role of scheduling algorithm is to:

1. Arrange them in a queue depending on their priority.
2. Assign them to processor.

3. To swap in swap out from processor.

There are three different scheduling algorithms:

1. Long Term Schedular
   Responsible for managing the process in the job queue which is in the secondary memory.
   Check which process wants to go first and prioritizes that process in the list.
   Sends the process from job queue to ready queue.

2. Short Term Schedular
   This is responsible for managing the processes in the ready queue, which is in the main memory.
   Processor gets the processes from this queue because it is near the processor.
   Prioritize them accordingly.

3. Medium Term Schedular
   Responsible for keeping the things in smooth flow.
   Sometimes it might happen that a process come into the main memory (Ready queue) and may do nothing other than keeping the memory reserve.
   Medium term schedular takes those processes out of the main memory.

## CONTEXT – SWITCH

Processor keeps switching between process, so this is known as context switching.

A processor switch from one process to another when the one process terminates or goes into waiting state.

A processor knows nothing about a process when it comes to the processor, it reads its PCB and check if it is new, or it is coming from the waiting state and all those things. So, every time before switching the system stores all the details of the process to its PCB.

## THREADS

Thread is also known as lightweight process.

It is a sequence of instructions within a process, thread itself behaves like a process inside a process.

It does not have a PCB it relays on the PCB of the process under which it is behaving like a process.

A thread passes through 3 states:

1. Running
2. Ready
3. Blocked

## PROCESS CREATION

If a process creates a child process that child uses the resources of the parent process, Parent can also limit the resource usage of child process. A child process can further create more child and form a complete tree.

## PROCESS TERMINATION

Process termination means the closing of a process after it is done performing all the tasks it was assigned to do, or due to some other reason.

It tells the OS by issuing Exit() call that it is going to be terminated.

### CHILD PROCESS TERMINATION

Usually, child also finishes after completing its task.

One addition thing that it does is that at normal termination it returns some data to its parent process. That is the result of the tasks it was assigned to do so.

If sometimes parent wants to terminate the child process due to any reason, the parent will issue the Abort () call to its child and kill it.

### ORPAN PROCESS

Usually when a parent is terminated the children also terminate with it, but it not always the case, sometimes the child chose to continue its execution, such child processes are known as Orphan process.

### ZOMBIE PROCESS

On Unix and Unix-like computer operating systems, a zombie process or defunct process is a process that has completed execution (via the exit system call) but still has an entry in the process table: it is a process in the "Terminated state".

## THREAD VS. PROCESS

| Thread | Process |
|---|---|
| It is lightweight entity. | It is heavy weight entity |
| If a thread ends working | Process may keep working and if a process terminates all its threads will terminate also. |
| Communication b/w threads happens via memory. | Communication b/w Process happens via OS. |
| The Creation of thread and context switching is inexpensive | The creation of the process is expensive. |

## MULTI-THREADING

Multithreading allows you to execute different threads at the same time.

Web Browser is an example of multithreading, one thread is working to get the data from the server, other thread is working to display the images and UI to the user.

## ADVANTAGES OF MULTITHREADING

1. Responsive
2. Resource sharing
3. Multiprocessor Architecture Utilization

## TYPES OF THREADING

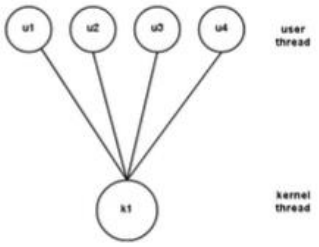
1. Kernel-Level threads
   They allow kernel to perform all kernel level tasks in parallel manner.
2. User-Level threads
   These threads are so fast, most of the times kernel do not have complete information of these threads.
   They manage themselves without interrupting operating system and kernel too much.

## MULTITHREADING MODEL

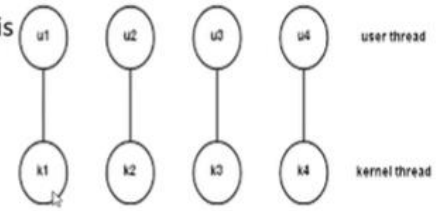
User thread must be mapped to kernel thread, so the operating system remains aware of what is going on in user mode and provide resources if needed from kernel.

We have three models to link User level thread to kernel level:

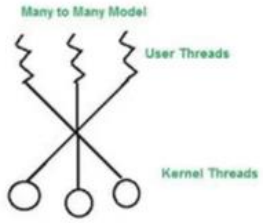
1. Many to One
   - **Many to One Model**
     - A process using this model will be blocked completely if a thread makes a blocking call.
     - Only one thread can access the kernel at a time.
     - This model can not run in parallel or multiprocessor.




2. One to One
   - **One to One Model**
     - In this model each user thread model is mapped to a kernel thread.
     - If a thread, make a blocking call still other thread can execute.
     - It facilitate the parallel processing using multiprocessor.




3. Many to Many
   - **Many to Many Model**
     - This model map many user level thread to equal or smaller level of kernel thread.
     - It also allow parallel processing and use multiprocessor.
     - The kernel can execute another thread if a thread makes blocking call.




## CHAPTER 4

## PRE-EMPTION VS. NON-PREEMPTION

## PRE EMPTIVE

From Ready Queue (Main Memory) to CPU, and the process is removed from the queue, even if it is still executing.

In Pre Emptive scheduling the CPU is allocated to a process for a limited time.

The first factor due to which we do this switch is Time Quantum, it means the CPU has already decided that it will execute the certain process for n seconds.

We use this to enhance responsiveness.

Algorithms:

1. SRTF (Shortest Remaining Time First) Also Known as SJF (PreEmptive)
2. LRTF (Longest Remaining Time First)
3. Round Robin
4. Priority Based

## NON-PRE-EMPTIVE

The CPU is allocated to the process until the process terminates or switches to waiting state.

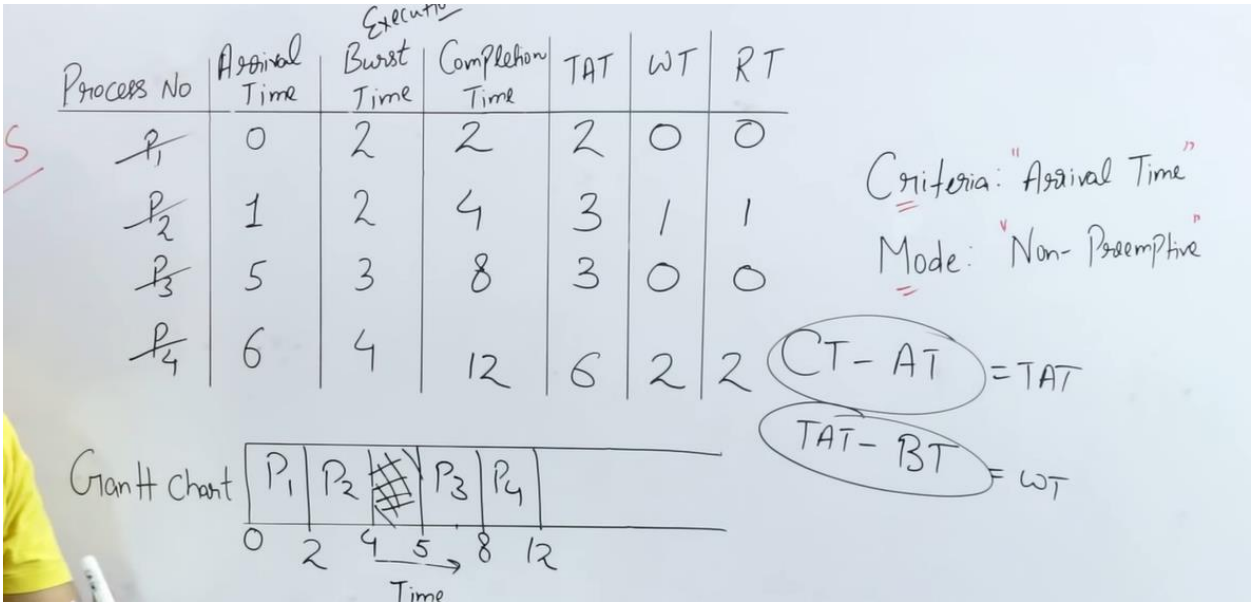
Algorithms:

1. FCFS (First Come First Serve)
2. SJF (Shortest Job First)
3. LJF (Longest Job First)
4. HRRN (Highest Responsive Ration Next)
5. Multilevel Queue

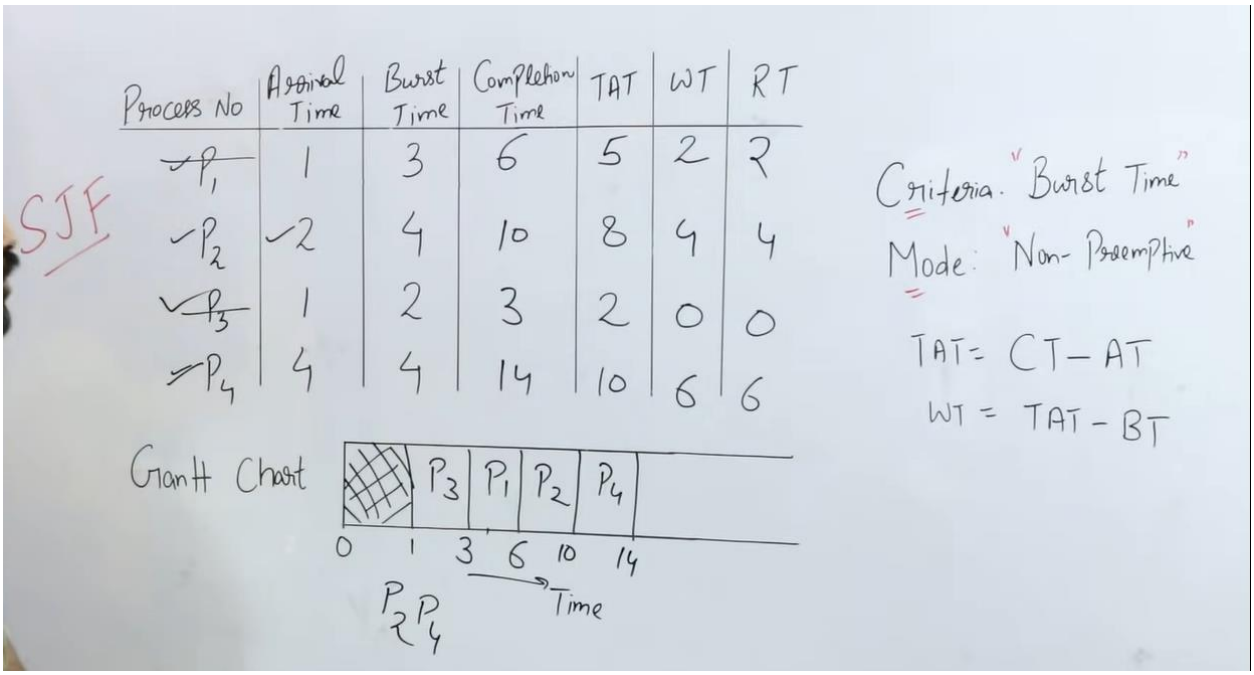
## TIMES

1. Arrival Time: The time at which the process enters ready Queue or state.
2. Burst Time: Time required by process to get executed on CPU.
3. Completion Time: The Time at which the process completes its execution.
4. Turn Around Time: (Completion Time – Arrival Time)
5. Waiting Time: (Turn Around Time – Burst Time)
6. Response Time: (The at which the process got CPU first time – Arrival Time)

## FIRST COME FIRST SERVE (FCFS)

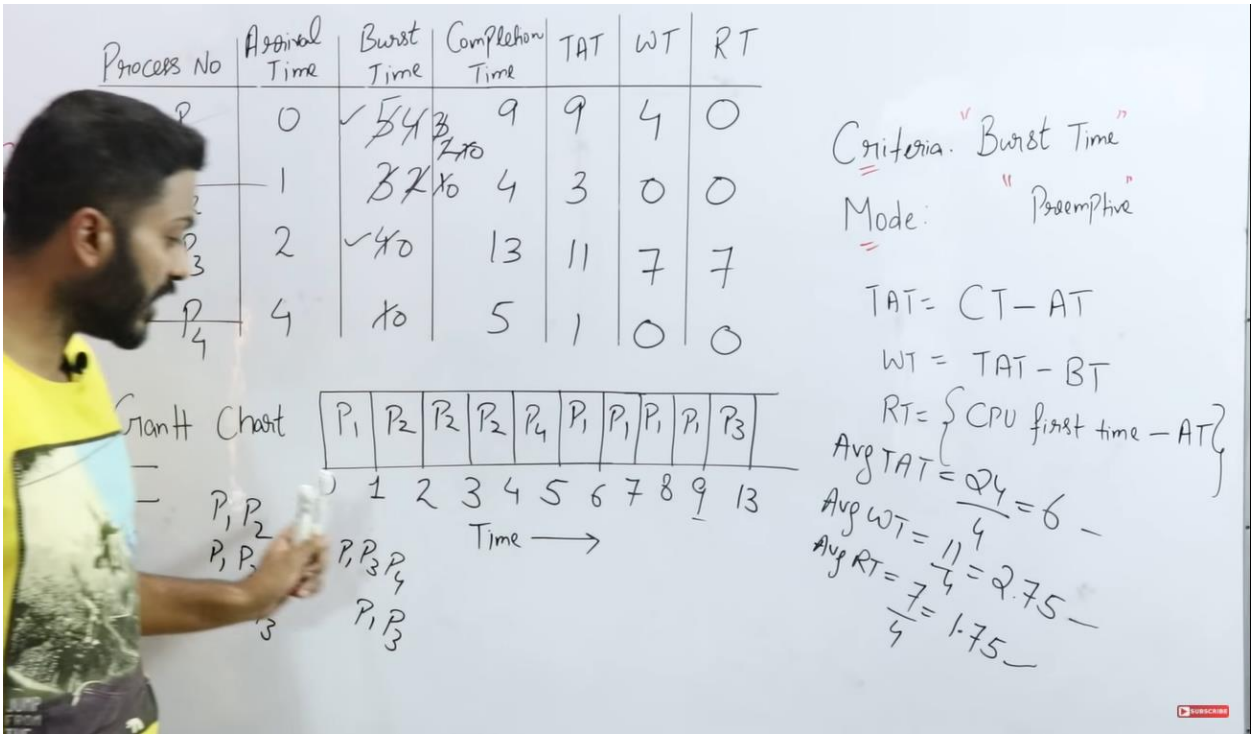



## SHORTEST JOB FIRST (SJF)

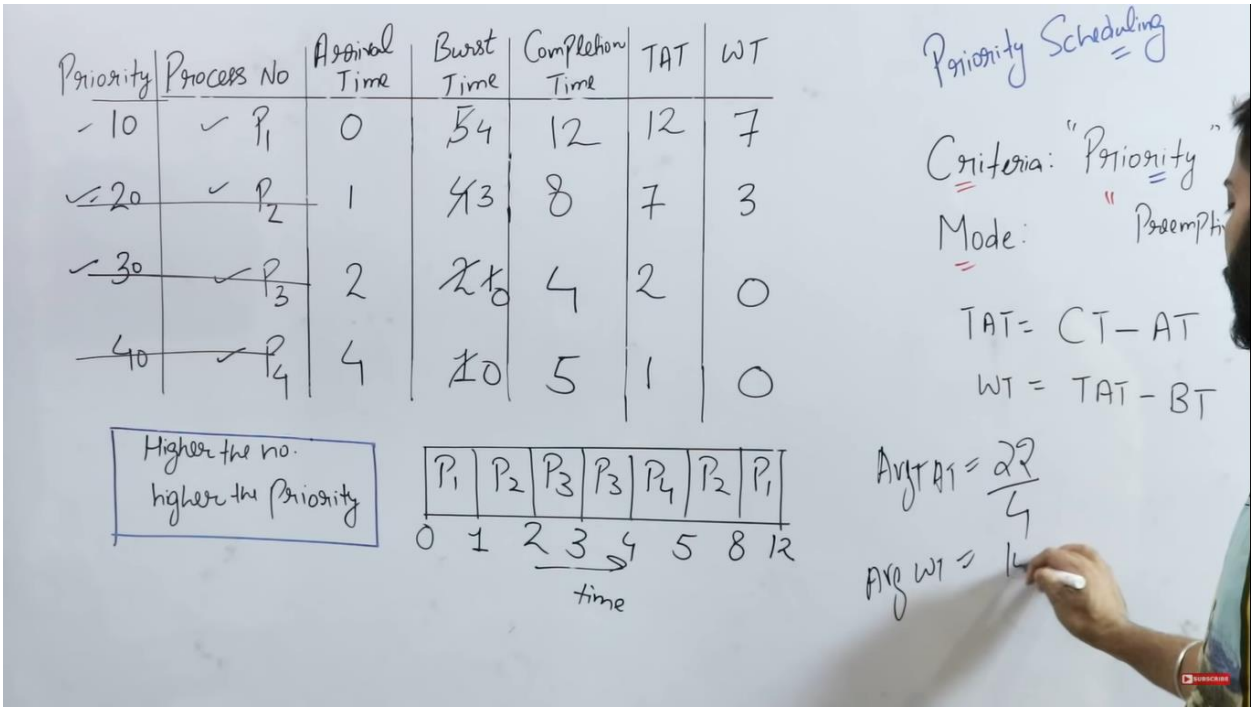



## SHORTEST JOB FIRST (PRE EMPTIVE)

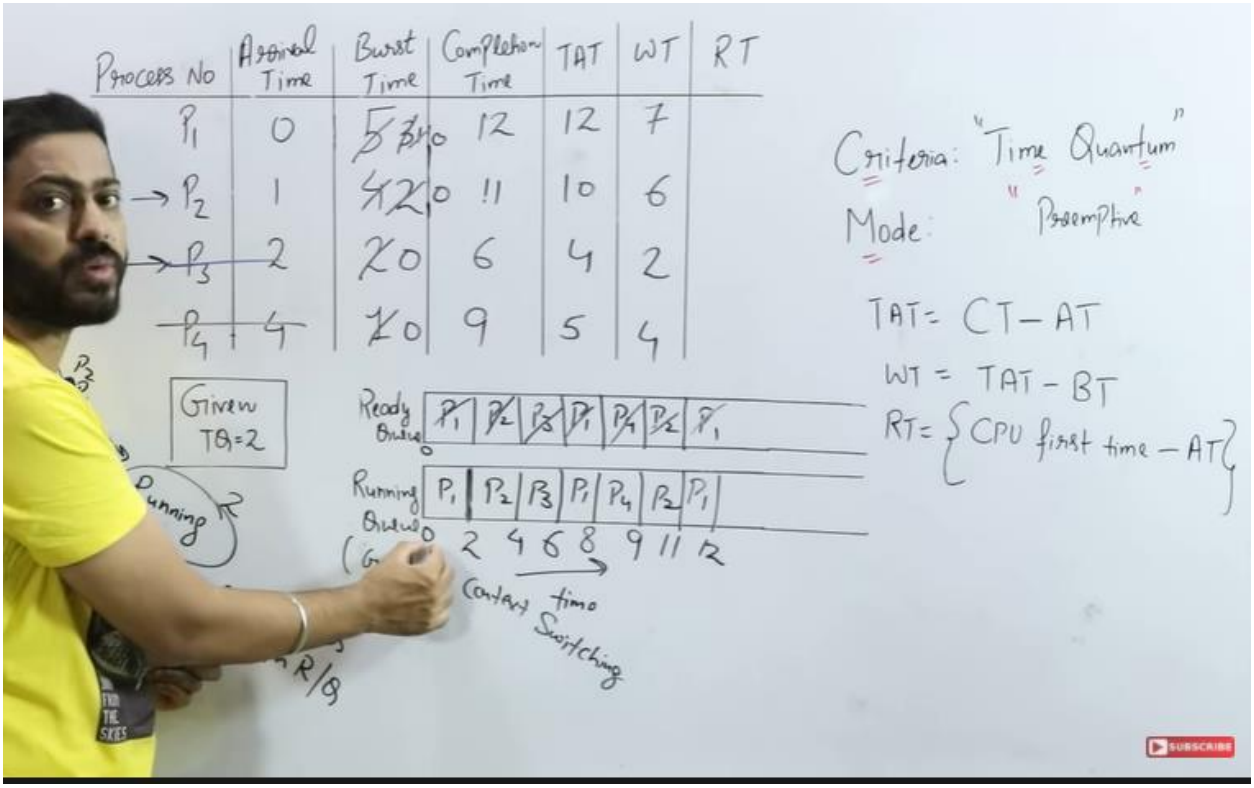
## SHORTEST REMAINING TIME FIRST (SRTF)




## PRIORITY SCHEDULING



## ROUND ROBIN




## DISPATCHER

- Dispatcher module gives control of the CPU to the process selected by the short-term scheduler, this involves:
    - switching context
    - switching to user mode
    - jumping to the proper location in the user program to restart that program.