Unit 2

Process Management: Process Model, Creation, Termination, States & Transitions, Hierarchy, Context Switching, Process Implementation, Process Control Block, Basic System calls- Linux & Windows. Basic concepts, classification, CPU and I/O bound, CPU scheduler- short, medium, long-term, dispatcher, scheduling:- preemptive and non-preemptive, Static and Dynamic Priority Criteria/Goals/Performance Metrics, scheduling algorithms- FCFS, SJFS, shortest remaining time, Round robin, Priority scheduling, multilevel queue scheduling, multilevel feedback queue scheduling

Process Management Introduction

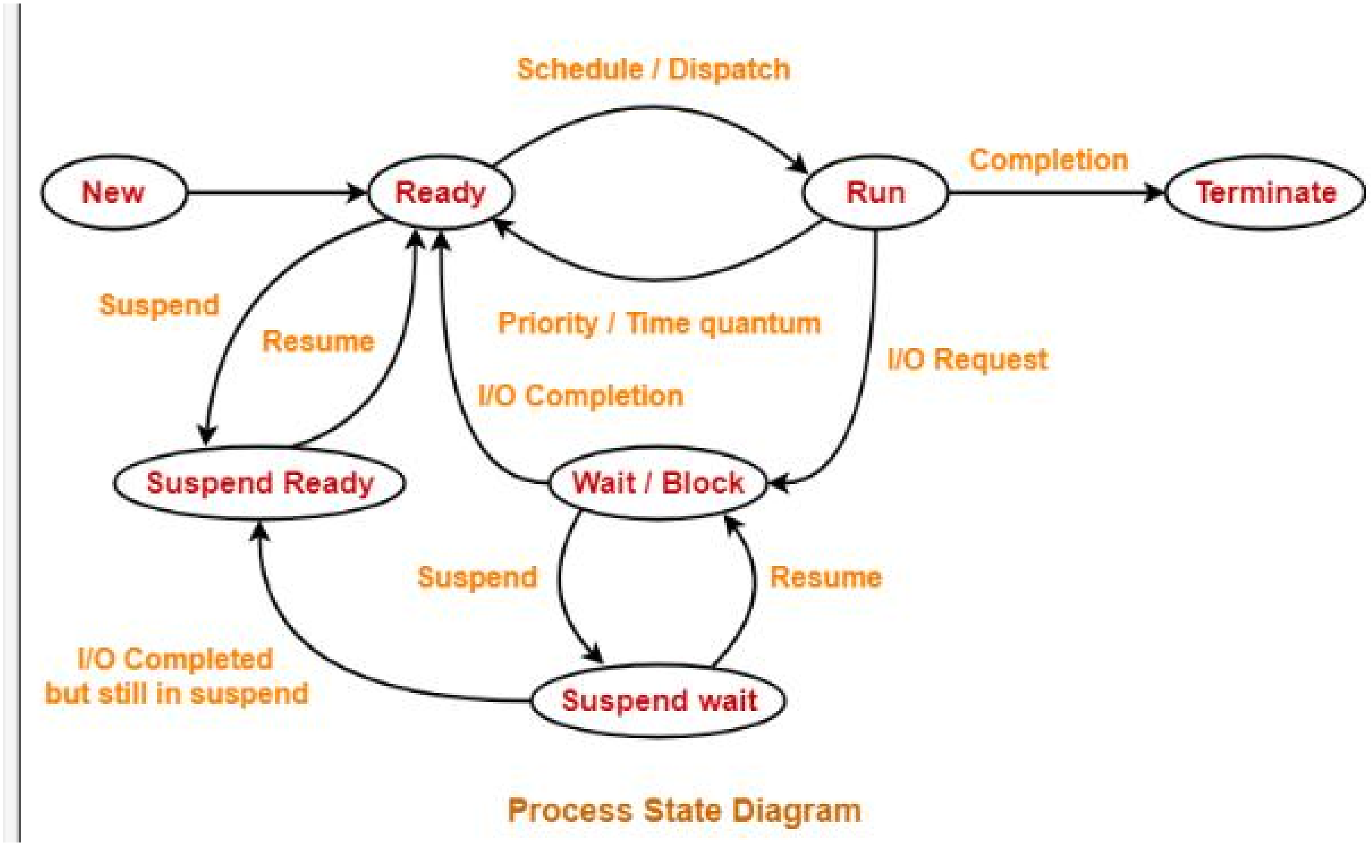
- A Program does nothing unless its instructions are executed by a CPU. A program in execution is called a process. In order to accomplish its task, process needs the computer resources.
- There may exist more than one process in the system which may require the same resource at the same time. Therefore, the operating system has to manage all the processes and the resources in a convenient and efficient way.

- The operating system is responsible for the following activities in connection with Process Management:

1. Scheduling processes and threads on the CPUs.
2. Creating and deleting both user and system processes.
3. Suspending and resuming processes.
4. Providing mechanisms for process synchronization.
5. Providing mechanisms for process communication.

Process States in Operating System-

Each process goes through different states in its life cycle-



Process State Diagram

1. New State-

A process is said to be in new state when a program present in the secondary memory is initiated for execution.

2. Ready State-

A process moves from new state to ready state after it is loaded into the main memory and is ready for execution.

In ready state, the process waits for its execution by the processor.

In multiprogramming environment, many processes may be present in the ready state.

3. Run State-

A process moves from ready state to run state after it is assigned the CPU for execution.

4. Terminate State-

A process moves from run state to terminate state after its execution is completed.

After entering the terminate state, context (PCB) of the process is deleted by the operating system.

5. Block Or Wait State-

A process moves from run state to block or wait state if it requires an I/O operation or some blocked resource during its execution.

After the I/O operation gets completed or resource becomes available, the process moves to the ready state.

Suspend Ready State-

A process moves from ready state to suspend ready state if a process with higher priority has to be executed but the main memory is full.

Moving a process with lower priority from ready state to suspend ready state creates a room for higher priority process in the ready state.

The process remains in the suspend ready state until the main memory becomes available.

When main memory becomes available, the process is brought back to the ready state.

6. Suspend Wait State-

A process moves from wait state to suspend wait state if a process with higher priority has to be executed but the main memory is full.

Moving a process with lower priority from wait state to suspend wait state creates a room for higher priority process in the ready state.

After the resource becomes available, the process is moved to the suspend ready state.

After main memory becomes available, the process is moved to the ready state.

State	Present in Memory
New state	Secondary Memory
Ready state	Main Memory
Run state	Main Memory
Wait state	Main Memory
Suspend wait state	Secondary Memory
Suspend ready state	Secondary Memory
Terminate state	–

- Operations on the Process
- 1. Creation

  Once the process is created, it will be ready and come into the ready queue (main memory) and will be ready for the execution.
- 2. Scheduling

  Out of the many processes present in the ready queue, the Operating system chooses one process and start executing it. Selecting the process which is to be executed next, is known as scheduling.
- 3. Execution

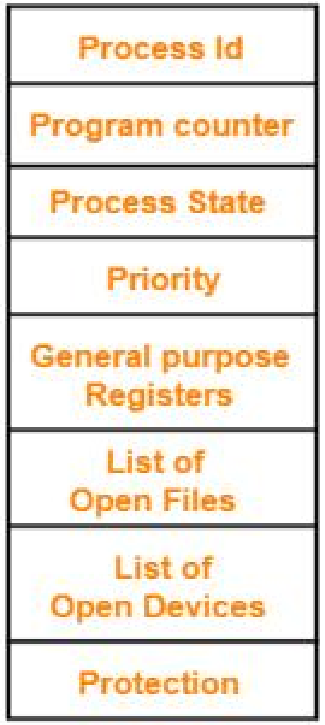
  Once the process is scheduled for the execution, the processor starts executing it. Process may come to the blocked or wait state during the execution then in that case the processor starts executing the other processes.
- 4. Deletion/killing

  Once the purpose of the process gets over then the OS will kill the process. The Context of the process (PCB) will be deleted and the process gets terminated by the Operating system

- **Process Control Block-**
- Process Control Block (PCB) is a data structure that stores information about a particular process.

This information is required by the CPU while executing the process.

-



Process Control Block (PCB)

Process Attributes-

The various attributes of process stored in the PCB are-

1. Process Id-

Process Id is a unique Id that identifies each process of the system uniquely.
A process Id is assigned to each process during its creation.

2. Program Counter-

Program counter specifies the address of the instruction to be executed next.
Before execution, program counter is initialized with the address of the first instruction of the program.
After executing an instruction, value of program counter is automatically incremented to point to the next instruction.
This process repeats till the end of the program.

3. Process State-

Each process goes through different states during its lifetime.
Process state specifies the current state of the process.

4. Priority-

Priority specifies how urgent is to execute the process.
Process with the highest priority is allocated the CPU first among all the processes.

5. General Purpose Registers-

General purpose registers are used to hold the data of process generated during its execution.
Each process has its own set of registers which are maintained by its PCB.

6. List of Open Files-

Each process requires some files which must be present in the main memory during its execution.
PCB maintains a list of files used by the process during its execution.

7. List of Open Devices-

PCB maintains a list of open devices used by the process during its execution.

Schedulers in OS-

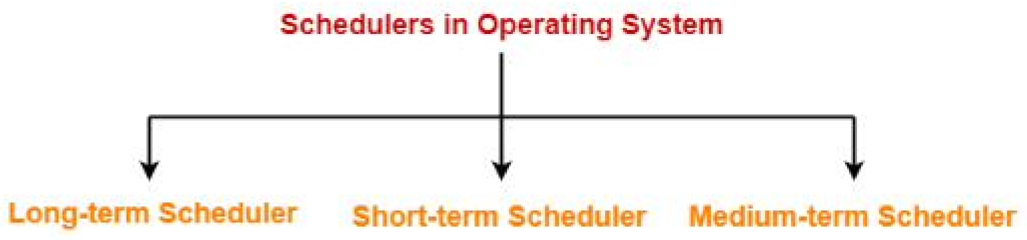
Schedulers in OS are special system software.

They help in scheduling the processes in various ways.

They are mainly responsible for selecting the jobs to be submitted into the system and deciding which process to run.

Types of Schedulers-

There are 3 kinds of schedulers-



1. Long-term scheduler
2. Short-term scheduler
3. Medium-term scheduler

Long-term Scheduler-

The primary objective of long-term scheduler is to maintain a good degree of multiprogramming.

Long-term scheduler is also known as **Job Scheduler**.

It selects a balanced mix of I/O bound and CPU bound processes from the secondary memory (new state).

Then, it loads the selected processes into the main memory (ready state) for execution.

Short-term Scheduler-

The primary objective of short-term scheduler is to increase the system performance.

Short-term scheduler is also known as **CPU Scheduler**.

It decides which process to execute next from the ready queue.

After short-term scheduler decides the process, **Dispatcher** assigns the decided process to the CPU for execution.

- **Medium-term Scheduler-**

The primary objective of medium-term scheduler is to perform swapping.

Medium-term scheduler swaps-out the processes from main memory to secondary memory to free up the main memory when required.

Thus, medium-term scheduler reduces the degree of multiprogramming.

After some time when main memory becomes available, medium-term scheduler swaps-in the swapped-out process to the main memory and its execution is resumed from where it left off.

Swapping may also be required to improve the process mix.

Context switching

- The Context switching is a technique or method used by the operating system to switch a process from one state to another to execute its function using CPUs in the system. When switching perform in the system, it stores the old running process's status in the form of registers and assigns the CPU to a new process to execute its tasks. While a new process is running in the system, the previous process must wait in a ready queue. The execution of the old process starts at that point where another process stopped it. It defines the characteristics of a multitasking operating system in which multiple processes shared the same CPU to perform multiple tasks without the need for additional processors in the system.

- Example of Context Switching

- Suppose that multiple processes are stored in a Process Control Block (PCB). One process is running state to execute its task with the use of CPUs. As the process is running, another process arrives in the ready queue, which has a high priority of completing its task using CPU. Here we used context switching that switches the current process with the new process requiring the CPU to finish its tasks. While switching the process, a context switch saves the status of the old process in registers. When the process reloads into the CPU, it starts the execution of the process when the new process stops the old process. If we do not save the state of the process, we have to start its execution at the initial level. In this way, context switching helps the operating system to switch between the processes, store or reload the process when it requires executing its tasks.

what is preemptive and non-preemptive scheduling

In Operating Systems, **Preemptive Scheduling** is a type of CPU scheduling method in which the CPU is allocated for a limited time to a given process. In contrast, **Non-Preemptive Scheduling** is the scheduling technique in which the CPU is allocated to a process and hold by it till the process gets terminated.

- Scheduling is the operating system's process to decide which process should be allocated to the CPU to execute several processes.

  What is Dispatcher in OS

- A dispatcher is a special program that comes into play after the scheduler. When the short term scheduler selects from the ready queue, the Dispatcher performs the task of allocating the selected process to the CPU. A running process goes to the waiting state for IO operation etc., and then the CPU is allocated to some other process. This switching of CPU from one process to the other is called **context switching**.

what is Static and Dynamic Priority

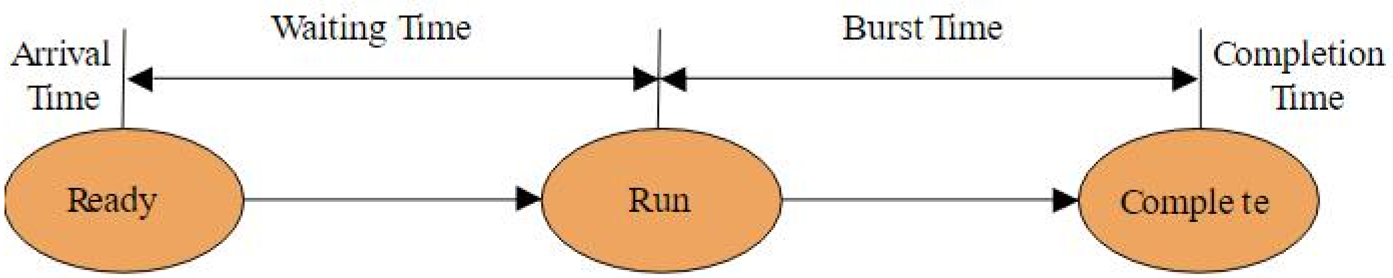
- The priority number assigned to each of the process may or may not vary. If the priority number doesn't change itself throughout the process, it is called static priority, while if it keeps changing itself at the regular intervals, it is called dynamic priority.

- A system is said to be I/O bound if the time taken to complete a computation is dependent on the period spent waiting for input/output operations to be completed. This is the complete opposite of a system being CPU bound where the tasks are determined solely on the performance of the CPU.

Various Times related to the Process

- 1. Arrival Time
- The time at which the process enters into the ready queue is called the arrival time.
- 2. Burst Time
- The total amount of time required by the CPU to execute the whole process is called the Burst Time.
- 3. Completion Time
- The Time at which the process enters into the completion state or the time at which the process completes its execution, is called completion time.

- . Turnaround time
- The total amount of time spent by the process from its arrival to its completion, is called Turnaround time.
- 5. Waiting Time
- The Total amount of time for which the process waits for the CPU to be assigned is called waiting time.
- 6. Response Time
- The difference between the arrival time and the time at which the process first gets the CPU is called Response Time.



CT - AT = WT + BT

TAT = CT - AT

Waiting Time = TAT - BT

TAT	⟶	Turn around time
BT	⟶	Burst time
AT	⟶	Arrival time

Scheduling Algorithms

- There are various algorithms which are used by the Operating System to schedule the processes on the processor in an efficient way.

The Purpose of a Scheduling algorithm

- Maximum CPU utilization
- Fare allocation of CPU
- Maximum throughput
- Minimum turnaround time
- Minimum waiting time
- Minimum response time

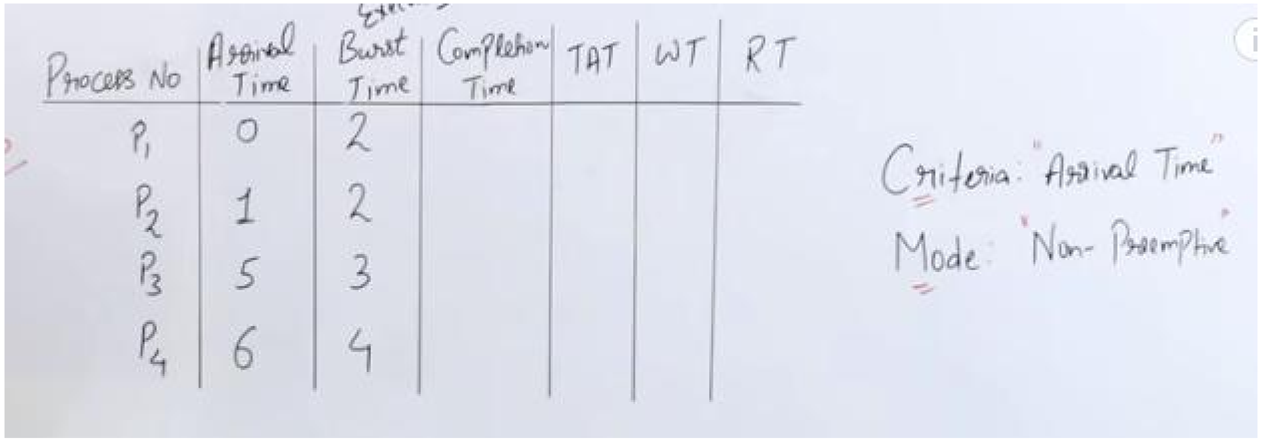
There are the following algorithms which can be used to schedule the jobs.

- 1. First Come First Serve
- It is the simplest algorithm to implement. The process with the minimal arrival time will get the CPU first. The lesser the arrival time, the sooner will the process gets the CPU. It is the non-preemptive type of scheduling.
- 2. Round Robin
- In the Round Robin scheduling algorithm, the OS defines a time quantum (slice). All the processes will get executed in the cyclic way. Each of the process will get the CPU for a small amount of time (called time quantum) and then get back to the ready queue to wait for its next turn. It is a preemptive type of scheduling.

- 3. Shortest Job First
- The job with the shortest burst time will get the CPU first. The lesser the burst time, the sooner will the process get the CPU. It is the non-preemptive type of scheduling.
- 4. Shortest remaining time first
- It is the preemptive form of SJF. In this algorithm, the OS schedules the Job according to the remaining time of the execution.
- 5. Priority based scheduling
- In this algorithm, the priority will be assigned to each of the processes. The higher the priority, the sooner will the process get the CPU. If the priority of the two processes is same then they will be scheduled according to their arrival time.

1. FCFS

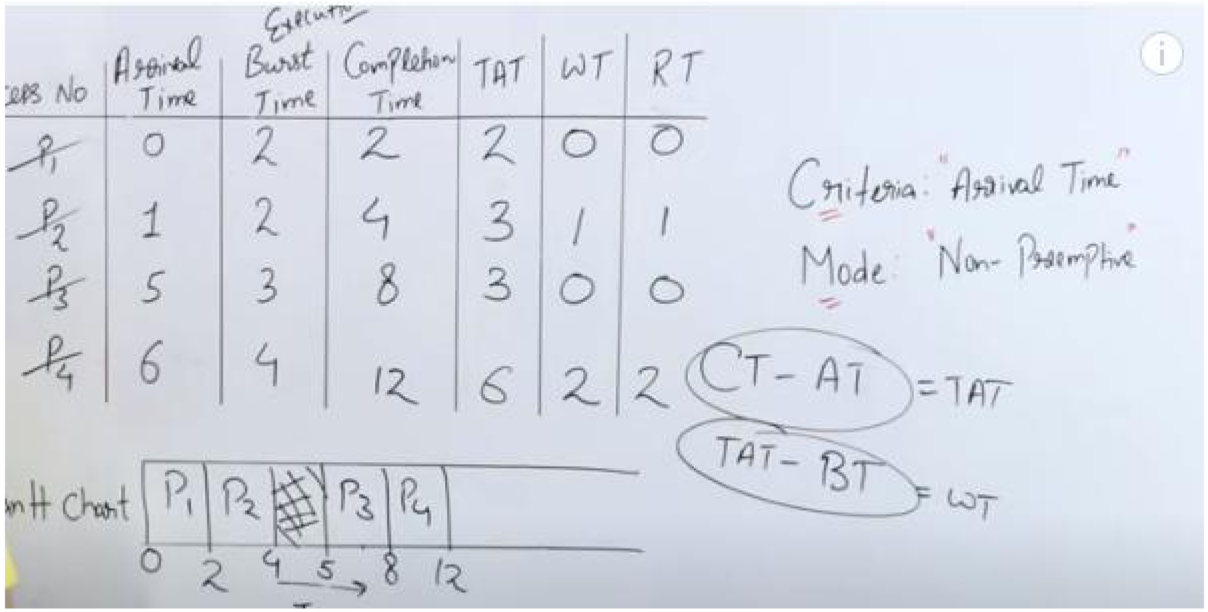
TAT=CT-AT
WT=TAT-BT



Process No	Arrival Time	Execution Burst Time	Completion Time	TAT	WT	RT
P_1	0	2				
P_2	1	2				
P_3	5	3				
P_4	6	4				

Criteria: "Arrival Time"

Mode: "Non-Preemptive"



cess No	Arrival Time	Execution Burst Time	Completion Time	TAT	WT	RT
~~P_1~~	0	2	2	2	0	0
~~P_2~~	1	2	4	3	1	1
~~P_3~~	5	3	8	3	0	0
~~P_4~~	6	4	12	6	2	2

Criteria: "Arrival Time"

Mode: "Non-Preemptive"

CT - AT = TAT

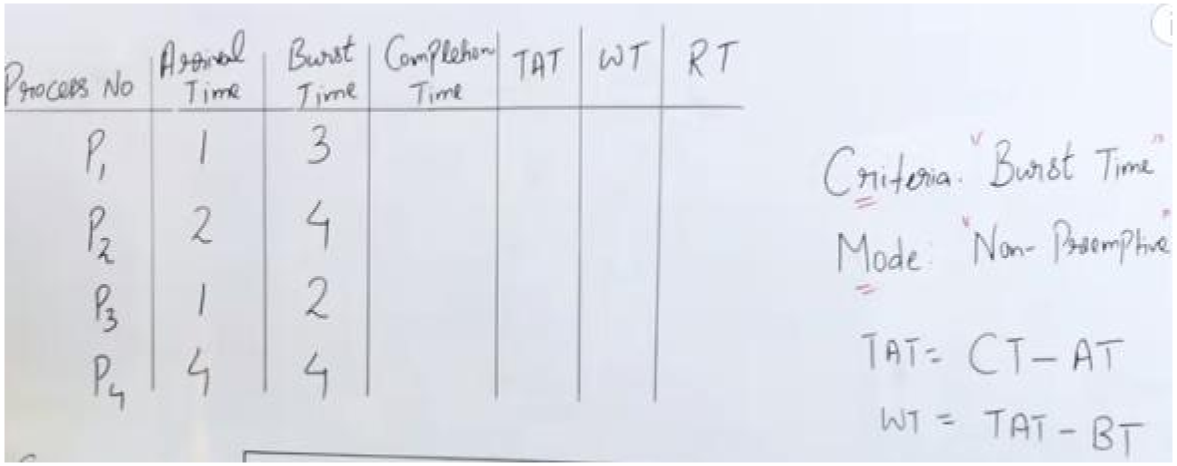
TAT - BT = WT

ntt Chart

P_1	P_2	▨	P_3	P_4

0 2 4 5 8 12

Sortest Job First (SJF)



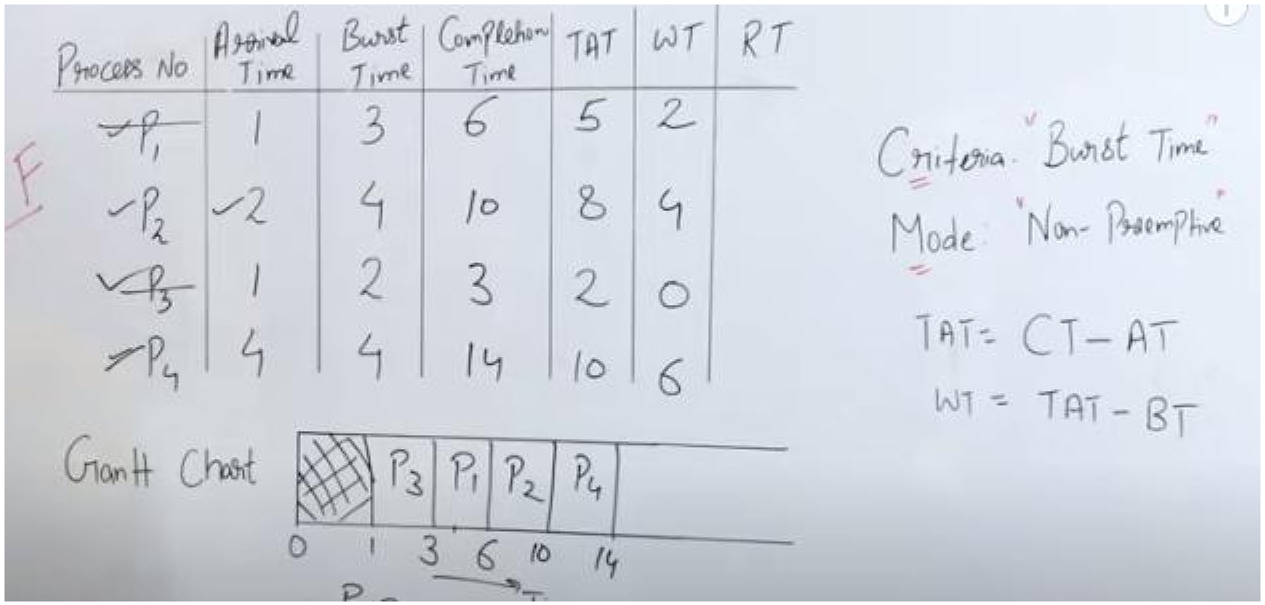
Process No	Arrival Time	Burst Time	Completion Time	TAT	WT	RT
P_1	1	3				
P_2	2	4				
P_3	1	2				
P_4	4	4				

Criteria: "Burst Time"

Mode: "Non-Preemptive"

TAT = CT − AT

WT = TAT − BT



F

Process No	Arrival Time	Burst Time	Completion Time	TAT	WT	RT
P_1	1	3	6	5	2	
P_2	2	4	10	8	4	
P_3	1	2	3	2	0	
P_4	4	4	14	10	6	

Criteria: "Burst Time"

Mode: "Non-Preemptive"

TAT = CT − AT

WT = TAT − BT

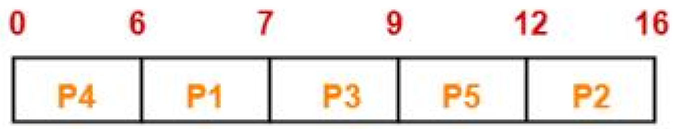
Gantt Chart

▨	P_3	P_1	P_2	P_4

0 1 3 6 10 14

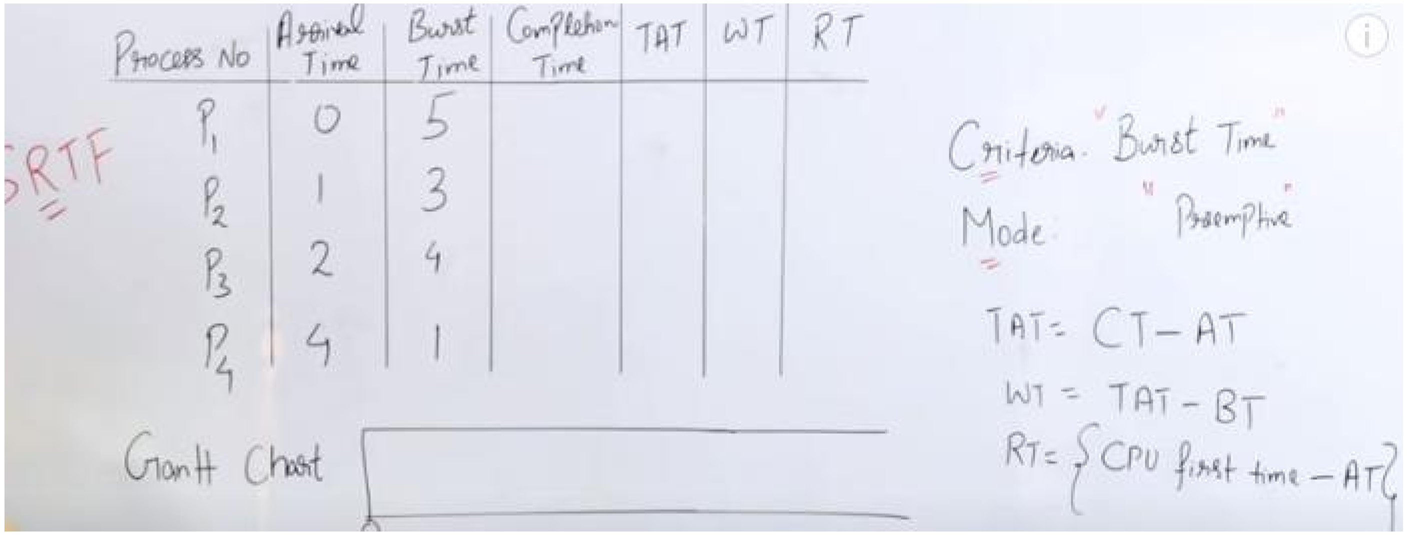
Example: SJF

Process Id	Arrival time	Burst time
P1	3	1
P2	1	4
P3	4	2
P4	0	6
P5	2	3



Gantt Chart

Shortest remaining time first/ sortest job first with preemtive



SRTF

Process No	Arrival Time	Burst Time	Completion Time	TAT	WT	RT
P_1	0	5				
P_2	1	3				
P_3	2	4				
P_4	4	1				

Criteria "Burst Time"

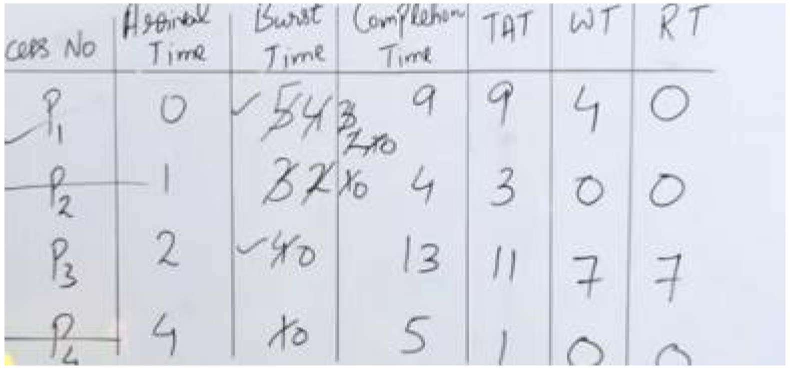
Mode "Preemptive"

TAT = CT − AT

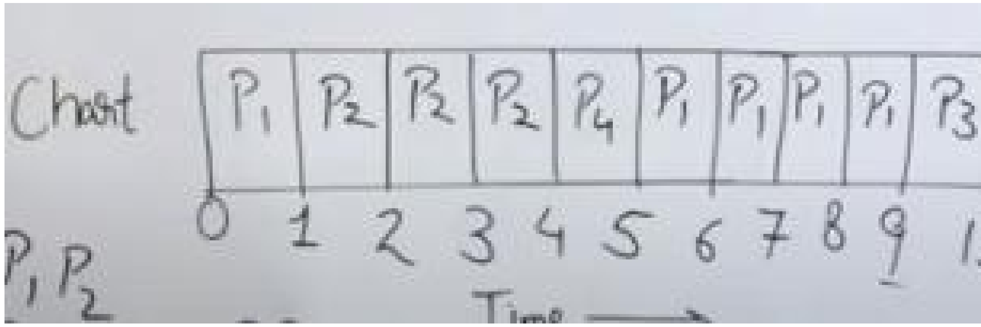
WT = TAT − BT

RT = {CPU first time − AT}

Gantt Chart



Process No	Arrival Time	Burst Time	Completion Time	TAT	WT	RT
P_1	0	✓ ~~5~~ ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0	9	9	4	0
P_2	1	~~3~~ ~~2~~ ~~1~~ 0	4	3	0	0
P_3	2	✓ ~~4~~ 0	13	11	7	7
P_4	4	~~1~~ 0	5	1	0	0



Chart

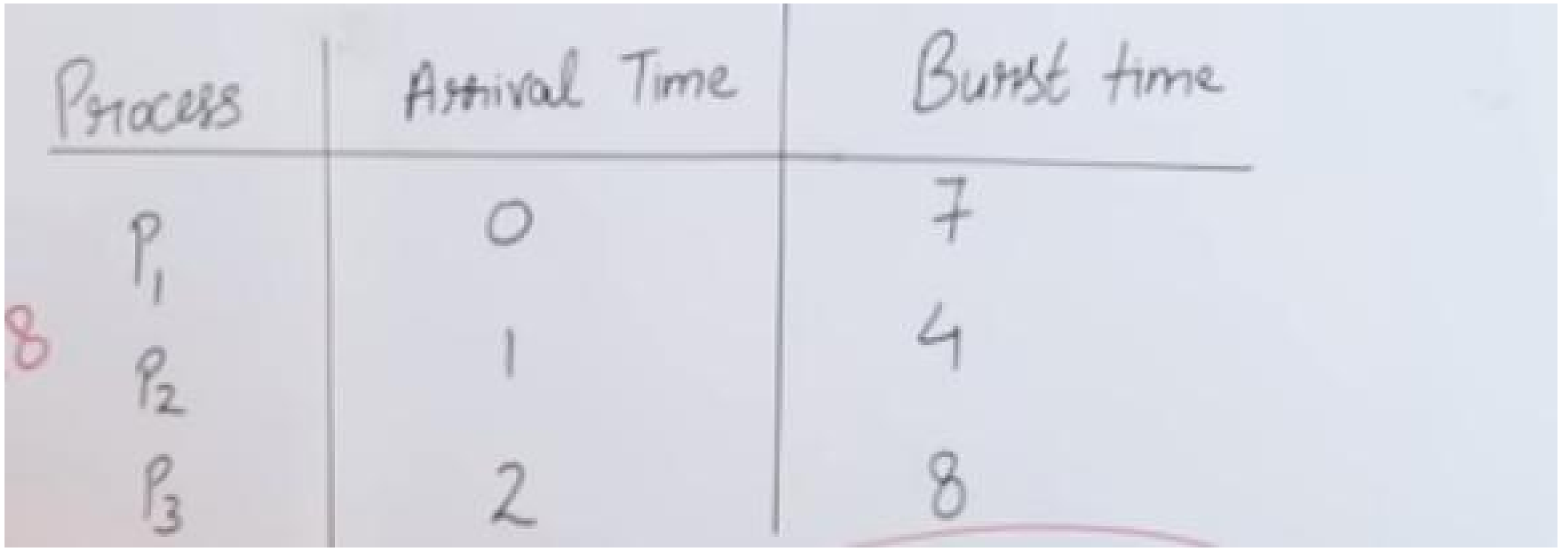
P_1	P_2	P_2	P_2	P_4	P_1	P_1	P_1	P_1	P_3

0 1 2 3 4 5 6 7 8 9 1

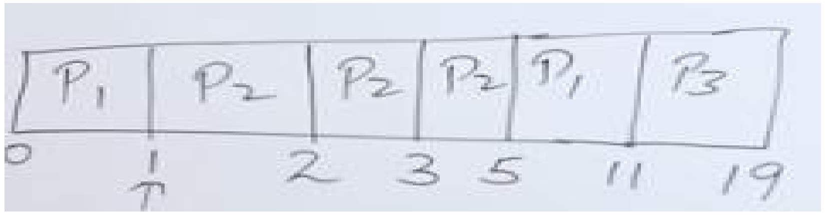
P_1, P_2

Time →

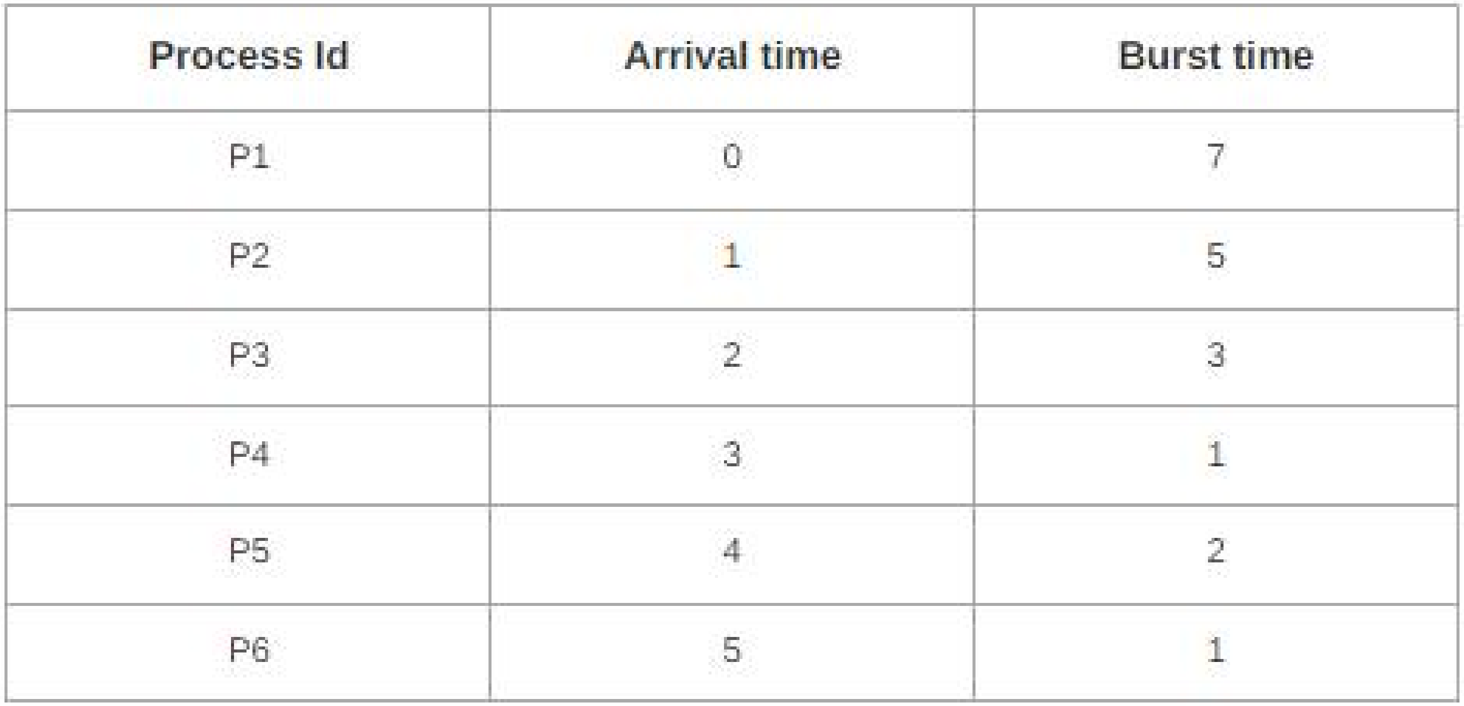
Example of Shortest remaining time first



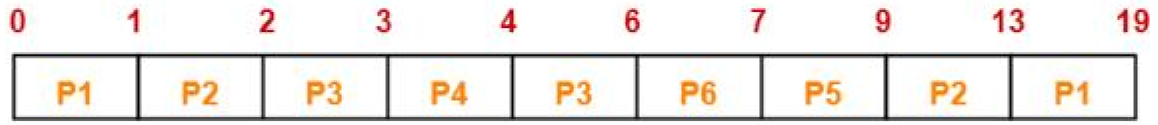
Process	Arrival Time	Burst time
P_1	0	7
P_2	1	4
P_3	2	8



Example:SRTF



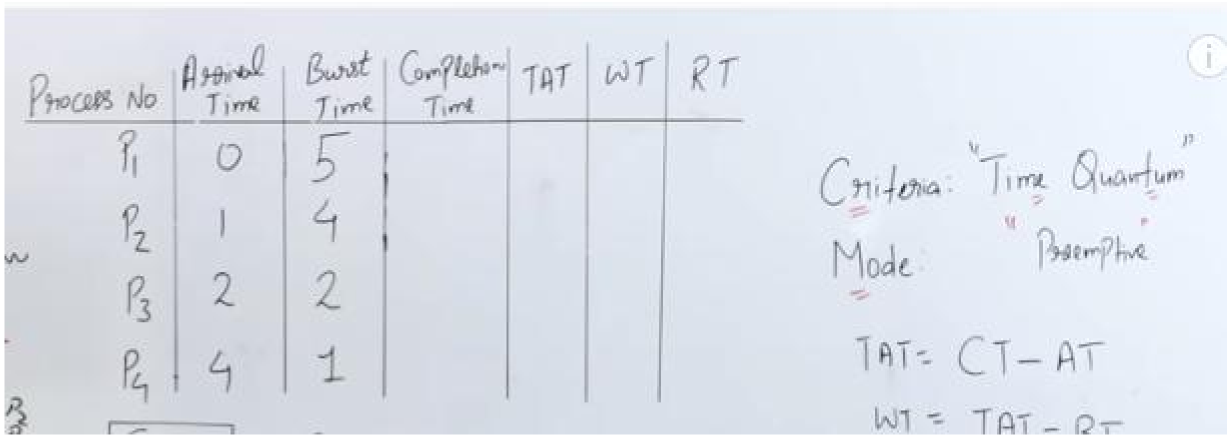
Process Id	Arrival time	Burst time
P1	0	7
P2	1	5
P3	2	3
P4	3	1
P5	4	2
P6	5	1



Gantt Chart

Round Robin Scheduling-

- CPU is assigned to the process on the basis of FCFS for a fixed amount of time.
- This fixed amount of time is called as **time quantum** or **time slice**.
- After the time quantum expires, the running process is preempted and sent to the ready queue.
- Then, the processor is assigned to the next arrived process.
- It is always preemptive in nature.



Process No	Arrival Time	Burst Time	Completion Time	TAT	WT	RT
P_1	0	5				
P_2	1	4				
P_3	2	2				
P_4	4	1				

Criteria: "Time Quantum"

Mode "Preemptive"

TAT = CT − AT

WT = TAT − BT

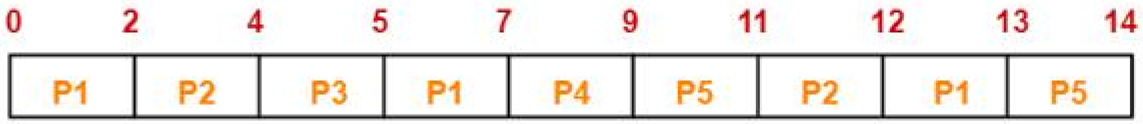
TQ=2

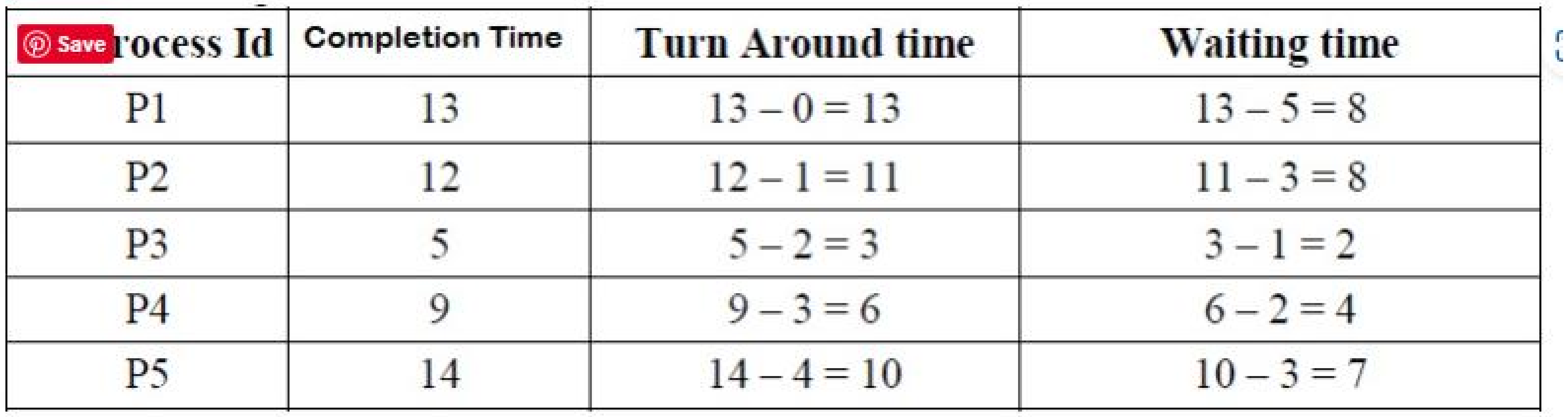
PRACTICE PROBLEMS BASED ON ROUND ROBIN SCHEDULING-

Problem-01:

Process Id	Arrival time	Burst time
P1	0	5
P2	1	3
P3	2	1
P4	3	2
P5	4	3

If the CPU scheduling policy is Round Robin with time quantum = 2 unit, calculate the average waiting time and average turn around time.





rocess Id	Completion Time	Turn Around time	Waiting time
P1	13	13 – 0 = 13	13 – 5 = 8
P2	12	12 – 1 = 11	11 – 3 = 8
P3	5	5 – 2 = 3	3 – 1 = 2
P4	9	9 – 3 = 6	6 – 2 = 4
P5	14	14 – 4 = 10	10 – 3 = 7

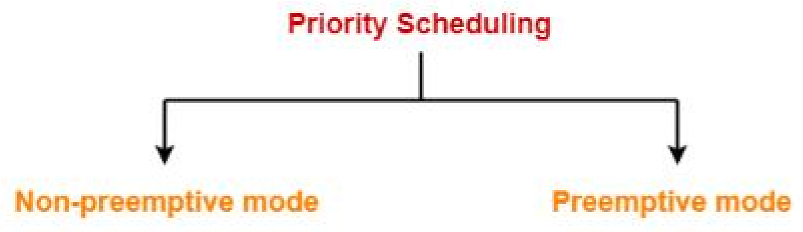
Average Waiting time = (13+11+3+6+10)/5= 8.6 ms
Average turnaround time =(8+8+2+4+7)/5= 5.8 ms

Process Id	Arrival time	Burst time
P1	0	4
P2	1	5
P3	2	2
P4	3	1
P5	4	6
P6	6	3

If the CPU scheduling policy is Round Robin with time quantum = 2, calculate the average waiting time and average turn around time.

Priority Scheduling-

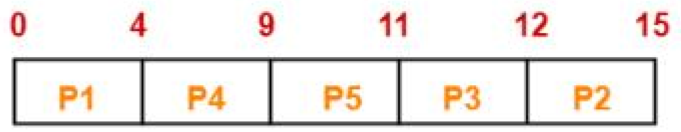
Out of all the available processes, CPU is assigned to the process having the highest priority.



- Priority Scheduling can be used in both preemptive and non-preemptive mode.

Process Id	Arrival time	Burst time	Priority
P1	0	4	2
P2	1	3	3
P3	2	1	4
P4	3	5	5
P5	4	2	5

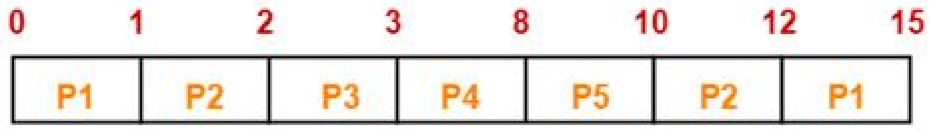
f the CPU scheduling policy is priority non-preemptive, calculate the average waiting time and average turn around time. (Higher number represents higher priority)



Gantt Chart

Process Id	Arrival time	Burst time	Priority
P1	0	4	2
P2	1	3	3
P3	2	1	4
P4	3	5	5
P5	4	2	5

If the CPU scheduling policy is priority preemptive, calculate the average waiting time and average turn around time. (Higher number represents higher priority)



Multilevel Queue (MLQ) CPU Scheduling

- Each algorithm supports a different process, but in a general system, some processes require scheduling using a priority algorithm. While some processes want to stay in the system (interactive processes), others are background processes whose execution can be delayed.
- The number of ready queue algorithms between queues and within queues may differ between systems. A round-robin method with various time quantum is typically utilized for such maintenance. Several types of scheduling algorithms are designed for circumstances where the processes can be readily separated into groups. There are two sorts of processes that require different scheduling algorithms because they have varying response times and resource requirements. The foreground (interactive) and background processes (batch process) are distinguished. Background processes take priority over foreground processes.

- The ready queue has been partitioned into seven different queues using the multilevel queue scheduling technique. These processes are assigned to one queue based on their priority, such as memory size, process priority, or type. The method for scheduling each queue is different. Some queues are utilized for the foreground process, while others are used for the background process. The foreground queue may be scheduled using a round-robin method, and the background queue can be scheduled using an FCFS strategy.

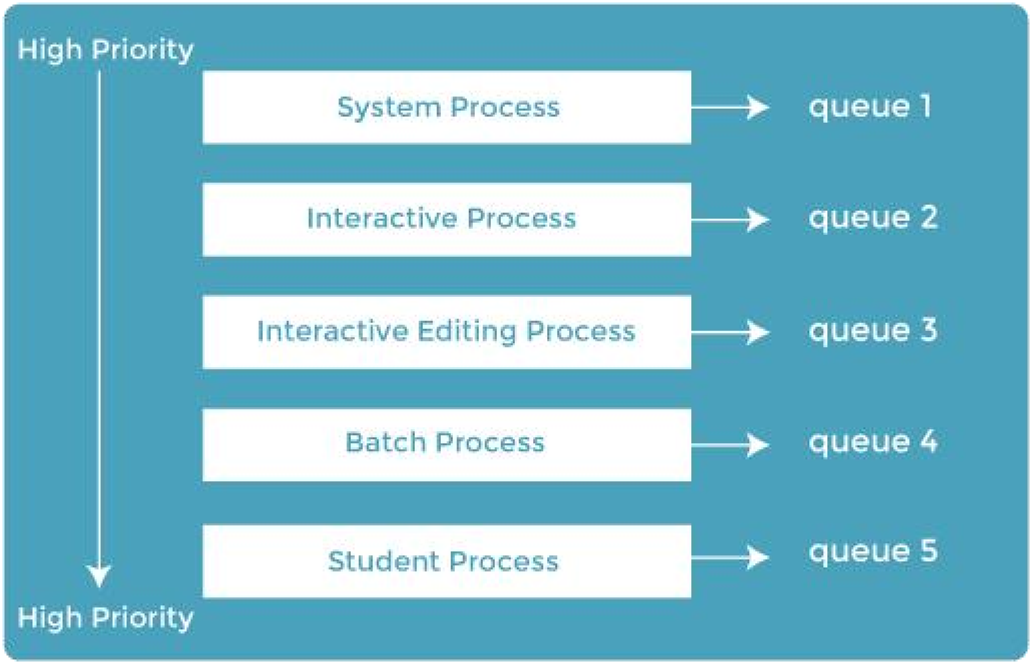
- Advantages and Disadvantages of Multilevel Queue Scheduling
- There are various advantages and disadvantages of multilevel queue scheduling. Some of the advantages and disadvantages of the multilevel queue scheduling are as follows:
- Advantages

1. You can use multilevel queue scheduling to apply different scheduling methods to distinct processes.
2. It will have low overhead in terms of scheduling.

- Disadvantages

1. There is a risk of starvation for lower priority processes.
2. It is rigid in nature.

Let's take an example of a multilevel queue-scheduling algorithm with five queues to understand how this scheduling works:



1.System process
2.Interactive processes
3.Interactive editing processes
4.Batch processes
5.Student processes

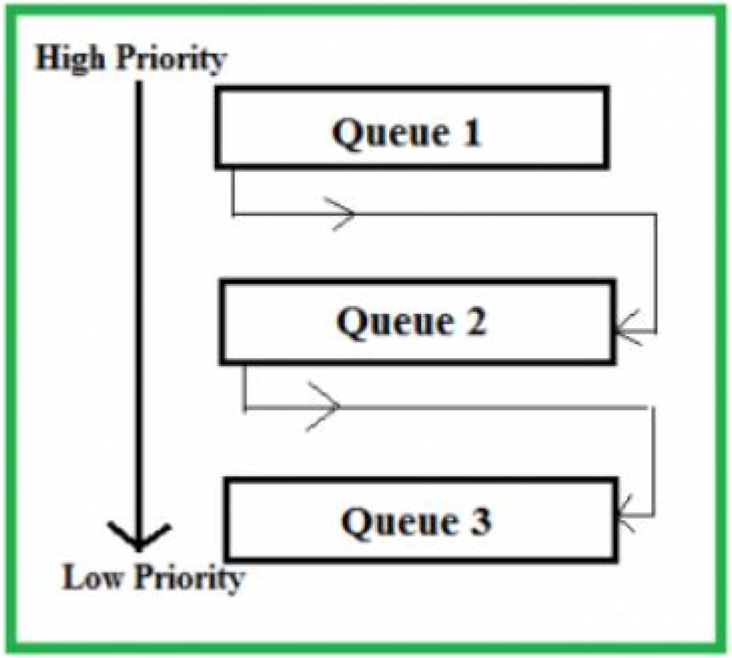
Multilevel Feedback Queue Scheduling (MLFQ) CPU Scheduling

- Multilevel Feedback Queue Scheduling (MLFQ) CPU Scheduling is like Multilevel Queue(MLQ) Scheduling but in this process can move between the queues. And thus, much more efficient than multilevel queue scheduling.
- Characteristics of Multilevel Feedback Queue Scheduling:
- In a multilevel queue-scheduling algorithm, processes are permanently assigned to a queue on entry to the system, and processes are allowed to move between queues.
- As the processes are permanently assigned to the queue, this setup has the advantage of low scheduling overhead,

Features of Multilevel Feedback Queue Scheduling (MLFQ) CPU Scheduling:

- **Multiple queues:** Similar to MLQ scheduling, MLFQ scheduling divides processes into multiple queues based on their priority levels. However, unlike MLQ scheduling, processes can move between queues based on their behavior and needs.
- **Priorities adjusted dynamically:** The priority of a process can be adjusted dynamically based on its behavior, such as how much CPU time it has used or how often it has been blocked. Higher-priority processes are given more CPU time and lower-priority processes are given less.
- **Time-slicing:** Each queue is assigned a time quantum or time slice, which determines how much CPU time a process in that queue is allowed to use before it is preempted and moved to a lower priority queue.
- **Feedback mechanism:** MLFQ scheduling uses a feedback mechanism to adjust the priority of a process based on its behavior over time. For example, if a process in a lower-priority queue uses up its time slice, it may be moved to a higher-priority queue to ensure it gets more CPU time.
- **Preemption:** Preemption is allowed in MLFQ scheduling, meaning that a higher-priority process can preempt a lower-priority process to ensure it gets the CPU time it needs.

- **Multilevel feedback queue scheduling**, however, allows a process to move between queues. Multilevel Feedback Queue Scheduling **(MLFQ)** keeps analyzing the behavior (time of execution) of processes and according to which it changes its priority.


High Priority
Queue 1
Queue 2
Queue 3
Low Priority