Agenda

- First Come First Serve Scheduling (FCFS)
- Longest Job First Scheduling Algorithm(LJF)
- Longest Remaining Time First (LRTF).
- Round Robin Scheduling(RR)

First Come First Serve (FCFS) Scheduling

The process which arrives first in the ready queue is firstly assigned the CPU.

- In case of a tie, process with smaller process id is executed first.
- It is always non-preemptive in nature.

Advantages

- It is simple and easy to understand.
- It can be easily implemented using queue data structure.
- It does not lead to starvation.

Disadvantages

- It does not consider the priority or burst time of the processes.
- It suffers from convoy effect.

In the "First come first serve" scheduling algorithm, as the name suggests, the process which arrives first, gets executed first, or we can say that the process which requests the CPU first, gets the CPU allocated first.

- First Come First Serve, is just like FIFO(First in First out) Queue data structure, where the data element which is added to the queue first, is the one who leaves the queue first.
- This is used in Batch Systems.

- It's easy to understand and implement programmatically, using a Queue data structure, where a new process enters through the tail of the queue, and the scheduler selects process from the head of the queue.
- A perfect real life example of FCFS scheduling is buying tickets at ticket counter.

Calculating Average Waiting Time

For every scheduling algorithm, Average waiting time is a crucial parameter to judge it's performance.

AWT or Average waiting time is the average of the waiting times of the processes in the queue, waiting for the scheduler to pick them for execution.

Lower the Average Waiting Time, better the scheduling algorithm.

Consider the processes P1, P2, P3, P4 given in the below table, arrives for execution in the same order, with Arrival Time 0, and given Burst Time, let's find the average waiting time using the FCFS scheduling algorithm.

Problems with FCFS Scheduling

Below we have a few shortcomings or problems with the FCFS scheduling algorithm:

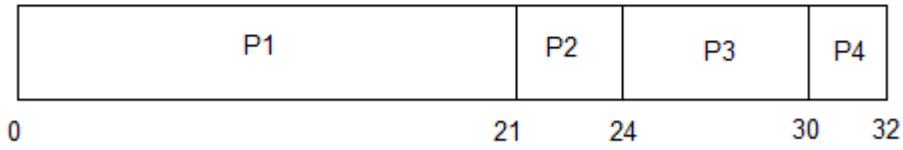
1. It is **Non Pre-emptive** algorithm, which means the **process priority** doesn't matter.

   If a process with very least priority is being executed, more like **daily routine backup** process, which takes more time, and all of a sudden some other high priority process arrives, like **interrupt to avoid system crash**, the high priority process will have to wait, and hence in this case, the system will crash, just because of improper process scheduling.

2. Not optimal Average Waiting Time.
3. Resources utilization in parallel is not possible, which leads to **Convoy Effect**, and hence poor resource(CPU, I/O etc) utilization

PROCESS	BURST TIME
P1	21
P2	3
P3	6
P4	2

The average waiting time will be = (0 + 21 + 24 + 30)/4 = 18.75 ms



This is the GANTT chart for the above processes

Consider the set of 5 processes whose arrival time and burst time are given below-

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

If the CPU scheduling policy is FCFS, calculate the average waiting time and average turn around time.

Gantt Chart

Here, black box represents the idle time of CPU.

Now, we know-

- Turn Around time = Exit time – Arrival time
- Waiting time = Turn Around time – Burst time

Process Id	Exit time	Turn Around time	Waiting time
P1	7	7 – 3 = 4	4 – 4 = 0
P2	13	13 – 5 = 8	8 – 3 = 5
P3	2	2 – 0 = 2	2 – 2 = 0
P4	14	14 – 5 = 9	9 – 1 = 8
P5	10	10 – 4 = 6	6 – 3 = 3

Now,

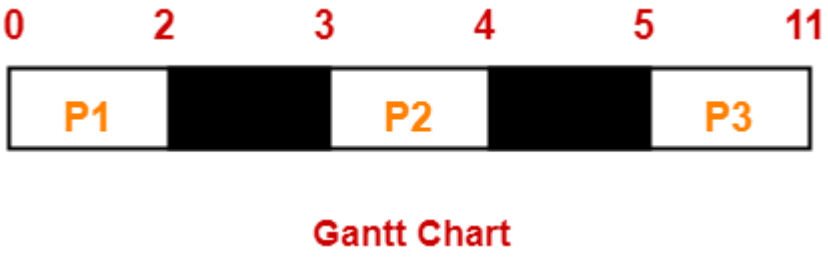
- Average Turn Around time = (4 + 8 + 2 + 9 + 6) / 5 = 29 / 5 = 5.8 unit
- Average waiting time = (0 + 5 + 0 + 8 + 3) / 5 = 16 / 5 = 3.2 unit

Consider the set of 3 processes whose arrival time and burst time are given below-

Process Id	Arrival time	Burst time
P1	0	2
P2	3	1
P3	5	6

If the CPU scheduling policy is FCFS, calculate the average waiting time and average turn around time.

Gantt Chart-



Gantt Chart

Here, black box represents the idle time of CPU.

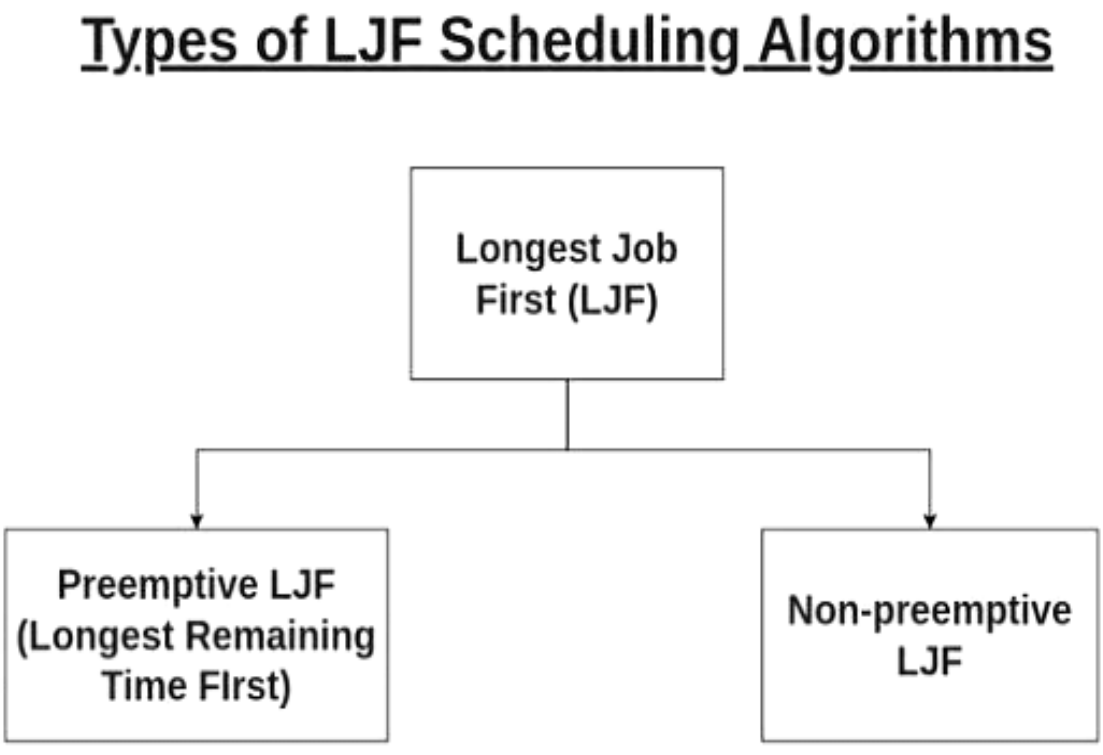
Now, we know-

- Turn Around time = Exit time – Arrival time
- Waiting time = Turn Around time – Burst time

Process Id	Exit time	Turn Around time	Waiting time
P1	2	2 – 0 = 2	2 – 2 = 0
P2	4	4 – 3 = 1	1 – 1 = 0
P3	11	11- 5 = 6	6 – 6 = 0

Now,

- Average Turn Around time = (2 + 1 + 6) / 3 = 9 / 3 = 3 unit
- Average waiting time = (0 + 0 + 0) / 3 = 0 / 3 = 0 unit



Longest Job First Scheduling Algorithm

- LJF Scheduling can be used in both preemptive and non-preemptive mode.
- Out of all the available processes, CPU is assigned to the process having largest burst time.
- Preemptive mode of Longest Job First is called as **Longest Remaining Time First (LRTF).**

Longest Job First **(LJF)** scheduling comes under the category of **the non-preemptive scheduling** algorithm. This algorithm mainly keeps the track of Burst time of all processes that are available at the arrival time itself and then it will assign the processor to the process having the longest burst time. In this algorithm, once a process starts its execution then it cannot be interrupted in between its processing. Any other process can be **executed only after** the assigned process has completed its processing and has been terminated.

This scheduling is similar to **the SJF** scheduling algorithm. But, in this scheduling algorithm, the priority is given to the process having the longest burst time.

Although this scheduling algorithm is not considered to be an efficient way of scheduling the processes

because there are many drawbacks of it:

- The first one is the Convoy effect is displayed by it
- The second one is this algorithm has a very large average turn-around time and average waiting time. Due to these two, the effectiveness of the system decreases and processing becomes slow.

In a situation if two processes having the same burst time then the tie is broken using FCFS i.e., the process that arrived first is processed first.

Let us take a look at the Procedure used in LJF scheduling:

- In the First step, the algorithm sort the processes according to the increasing order of their Arrival Time.
- In the second step, it will choose the process having the highest Burst Time among all the processes that have arrived till that time.
- After that, it will process it for its given burst time. The LJF also checks if any other process arrives until this process completes its execution.

Longest Remaining Time First (LRTF).

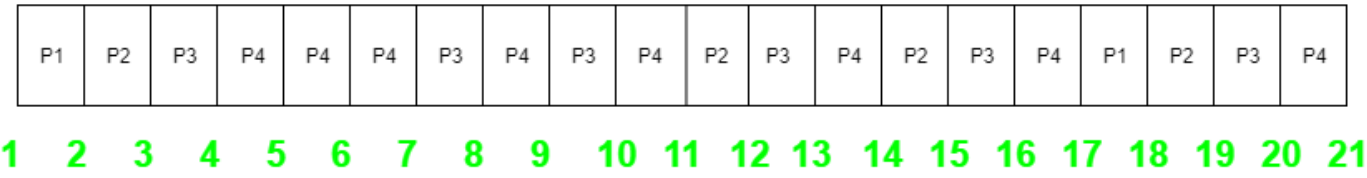
This is a pre-emptive version of Longest Job First (LJF) scheduling algorithm. In this scheduling algorithm, we find the process with maximum remaining time and then process it. We check for the maximum remaining time after some interval of time (say 1 unit each) to check if another process having more Burst Time arrived up to that time.

Procedure

- **Step-1**: First, sort the processes in increasing order of their Arrival Time.
- **Step-2**: Choose the process having least arrival time but with most Burst Time. Then process it for 1 unit. Check if any other process arrives up to that time of execution or not.
- **Step-3**: Repeat the above both steps until execute all the processes. Example: Consider the following table of arrival time and burst time for four processes P1, P2, P3 and P4.

Process	Arrival time	Burst Time
P1	1 ms	2 ms
P2	2 ms	4 ms
p3	3 ms	6 ms
p4	4 ms	8 ms

Gantt chart

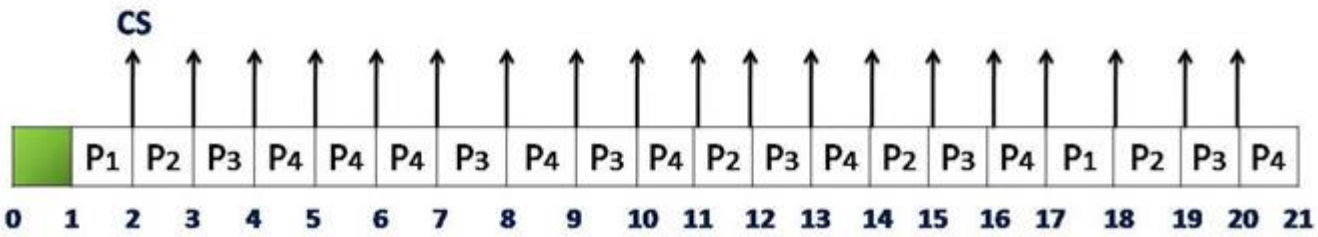


Output: *Total Turn Around Time = 68 ms So, Average Turn Around Time = 68/4 = 17.00 ms*

And, Total Waiting Time = 48 ms So, Average Waiting Time = 12.00 ms

Process No.	Arrival Time (AT)	Burst Time (BT)
P_1	1	2
P_2	2	4
P_3	3	6
P_4	4	8

Gantt Chart:



Advantages-

- No process can complete until the longest job also reaches its completion.
- All the processes approximately finishes at the same time.

Disadvantages-

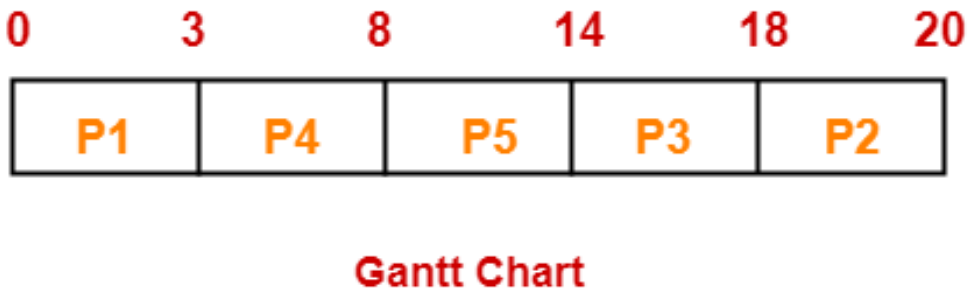
- The waiting time is high.
- Processes with smaller burst time may starve for CPU.

Problem-01:

Consider the set of 5 processes whose arrival time and burst time are given below-

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

If the CPU scheduling policy is LJF non-preemptive, calculate the average waiting time and average turn around time.



Gantt Chart

- Turn Around time = Exit time – Arrival time
- Waiting time = Turn Around time – Burst time

Process Id	Exit time	Turn Around time	Waiting time
P1	3	3 – 0 = 3	3 – 3 = 0
P2	20	20 – 1 = 19	19 – 2 = 17
P3	18	18 – 2 = 16	16 – 4 = 12
P4	8	8 – 3 = 5	5 – 5 = 0
P5	14	14 – 4 = 10	10 – 6 = 4

Average Turn Around time = (3 + 19 + 16 + 5 + 10) / 5 = 53 / 5 = 10.6 unit

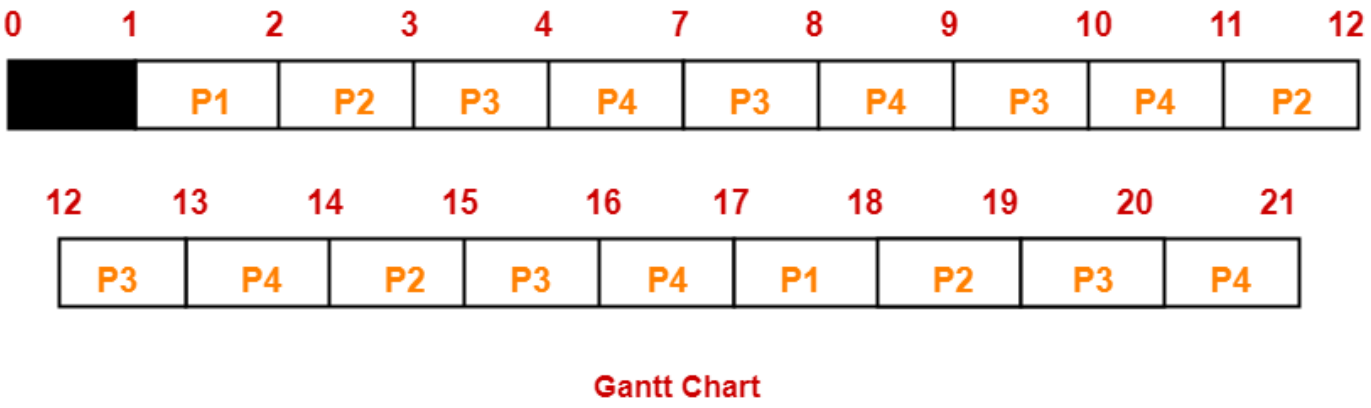
- Average waiting time = (0 + 17 + 12 + 0 + 4) / 5 = 33 / 5 = 6.6 unit

Problem:

Consider the set of 4 processes whose arrival time and burst time are given below-

Process Id	Arrival time	Burst time
P1	1	2
P2	2	4
P3	3	6
P4	4	8

If the CPU scheduling policy is LJF preemptive, calculate the average waiting time and average turn around time.



Gantt Chart

Now, we know-

- Turn Around time = Exit time – Arrival time
- Waiting time = Turn Around time – Burst time

Process Id	Exit time	Turn Around time	Waiting time
P1	18	18 – 1 = 17	17 – 2 = 15
P2	19	19 – 2 = 17	17 – 4 = 13
P3	20	20 – 3 = 17	17 – 6 = 11
P4	21	21 – 4 = 17	17 – 8 = 9

Now,

- Average Turn Around time = (17 + 17 + 17 + 17) / 4 = 68 / 4 = 17 unit
- Average waiting time = (15 + 13 + 11 + 9) / 4 = 48 / 4 = 12 unit

Round Robin Scheduling

Round Robin(RR) scheduling algorithm is mainly designed for time-sharing systems. This algorithm is similar to FCFS scheduling, but in Round Robin(RR) scheduling, preemption is added which enables the system to switch between processes.

- A fixed time is allotted to each process, called a quantum, for execution.
- Once a process is executed for the given time period that process is preempted and another process executes for the given time period.
- Context switching is used to save states of preempted processes.
- This algorithm is simple and easy to implement and the most important is thing is this algorithm is starvation-free as all processes get a fair share of CPU.
- It is important to note here that the length of time quantum is generally from 10 to 100 milliseconds in length.

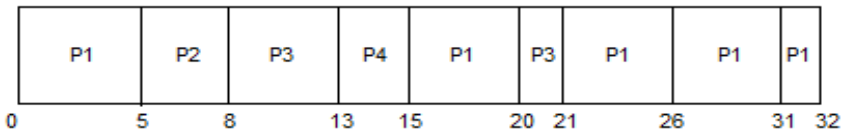
Some important characteristics of the Round Robin(RR) Algorithm are as follows:

1. Round Robin Scheduling algorithm resides under the category of Preemptive Algorithms.
2. This algorithm is one of the oldest, easiest, and fairest algorithm.
3. This Algorithm is a real-time algorithm because it responds to the event within a specific time limit.
4. In this algorithm, the time slice should be the minimum that is assigned to a specific task that needs to be processed. Though it may vary for different operating systems.
5. This is a hybrid model and is clock-driven in nature.
6. This is a widely used scheduling method in the traditional operating system.

Let us now cover an example for the same:

PROCESS	BURST TIME
P1	21
P2	3
P3	6
P4	2

The GANTT chart for round robin scheduling will be,



The average waiting time will be, 11 ms.

Examples:

1. S. No Process ID Arrival Time Burst Time
2. _
3. 1 P 1 0 7
4. 2 P 2 1 4
5. 3 P 3 2 15
6. 4 P 4 3 11
7. 5 P 5 4 20
8. 6 P 6 4 9

Assume Time Quantum TQ = 5

Ready Queue:

1. P1, P2, P3, P4, P5, P6, P1, P3, P4, P5, P6, P3, P4, P5

Gantt chart:

P1	P2	P3	P4	P5	P6	P1	P3	P4	P5	P6	P3	P4	P5
0	5	9	14	19	24	29	31	36	41	46	50	55	66

Process ID	Arrival Time	Burst Time	Completion Time	Turn Around Time	Waiting Time
P1	0	7	31	31	24
P2	1	4	9	8	4
P3	2	15	55	53	38
P4	3	11	56	53	42
P5	4	20	66	62	42
P6	4	9	50	46	37

Average Completion Time

1. Average Completion Time = (31 +9 + 55 +56 +66 + 50) / 6
2. Average Completion Time = 267 / 6
3. Average Completion Time = 44.5

Average Waiting Time

1. Average Waiting Time = (5 + 26 + 5 + 42 + 42 + 37) / 6
2. Average Waiting Time = 157 / 6
3. Average Waiting Time = 26.16667

Average Turn Around Time

1. Average Turn Around Time = (31 + 8 + 53 + 53 + 62 + 46) / 6
2. Average Turn Around Time = 253 / 6
3. Average Turn Around Time = 42.16667

Example of Round-robin Scheduling

Example-1: Consider the following table of arrival time and burst time for four processes **P1, P2, P3, and P4** and given **Time Quantum = 2**

Process	Burst Time	Arrival Time
P1	5 ms	0 ms
P2	4 ms	1 ms
P3	2 ms	2 ms
P4	1 ms	4 ms

Gantt chart for Round Robin Scheduling Algorithm