**PRIORITY SCHEDULING**

In the priority scheduling, a priority is associated with each process, and the CPU is allocated to the process with the highest priority. Processes with same priority are executed in FCFS scheduling.

Priority can be static or dynamic.

**Static priority:** It does not change priority throughout the execution of the process.

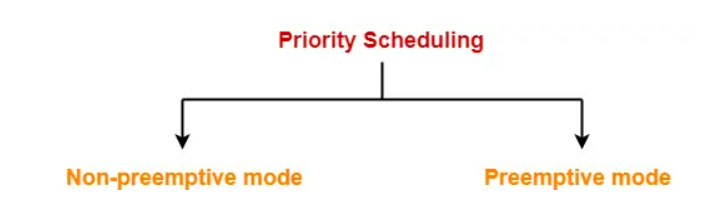
**Dynamic priority:** In this dynamic priority, priority can be changed by the scheduler at regular interval of time.

SJF algorithm is a special case of the general priority-scheduling algorithm. An SJF algorithm is simply a priority algorithm where the priority (p) is the inverse of the (predicted) next CPU burst. The larger the CPU burst, the lower the priority, and vice versa.

Priorities are generally indicated by some fixed range of numbers, such as 0 to 7 or 0 to 4,095. However, there is no general agreement on whether 0 is the highest or lowest priority. Some systems use low numbers to represent low priority; others use low numbers for high priority.

Priority scheduling can be of two types:

1. Preemptive Priority Scheduling.
2. Non-Preemptive Priority Scheduling.

****

**Non-Preemptive Priority Scheduling:**

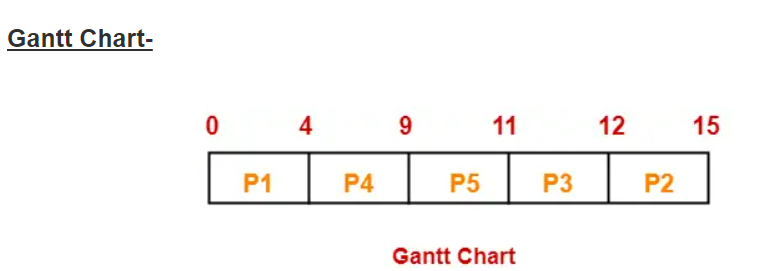
In Non-Preemptive Priority Scheduling, the process which has the highest priority is scheduled first, and if the processes have the same priority number, then it will be executed according to the First-Come-First-Serve manner. If the process is scheduled, then it will not leave the CPU until it completes the execution.

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

|  |  |  |  |
| --- | --- | --- | --- |
| 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 non-preemptive, calculate the average waiting time and average turn around time. (Higher number represents higher priority).

## **Solution-**



|  |  |  |  |
| --- | --- | --- | --- |
| **Process Id** | **Exit time** | **Turn Around time** | **Waiting time** |
| P1 | 4 | 4 – 0 = 4 | 4 – 4 = 0 |
| P2 | 15 | 15 – 1 = 14 | 14 – 3 = 11 |
| P3 | 12 | 12 – 2 = 10 | 10 – 1 = 9 |
| P4 | 9 | 9 – 3 = 6 | 6 – 5 = 1 |
| P5 | 11 | 11 – 4 = 7 | 7 – 2 = 5 |

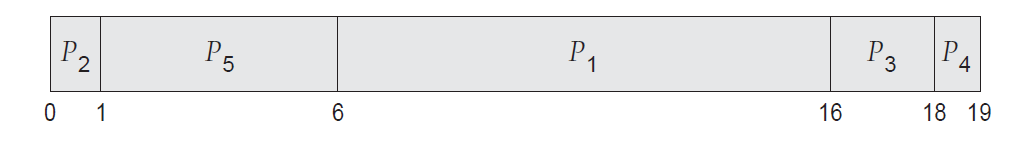
Now,

* Average Turn Around time = (4 + 14 + 10 + 6 + 7) / 5 = 41 / 5 = 8.2 milli seconds
* Average waiting time = (0 + 11 + 9 + 1 + 5) / 5 = 26 / 5 = 5.2 milliseconds.

**Example:** consider the set of processes that arrive at time 0 and CPU-burst time given in millisecond (Lower number represents higher priority)

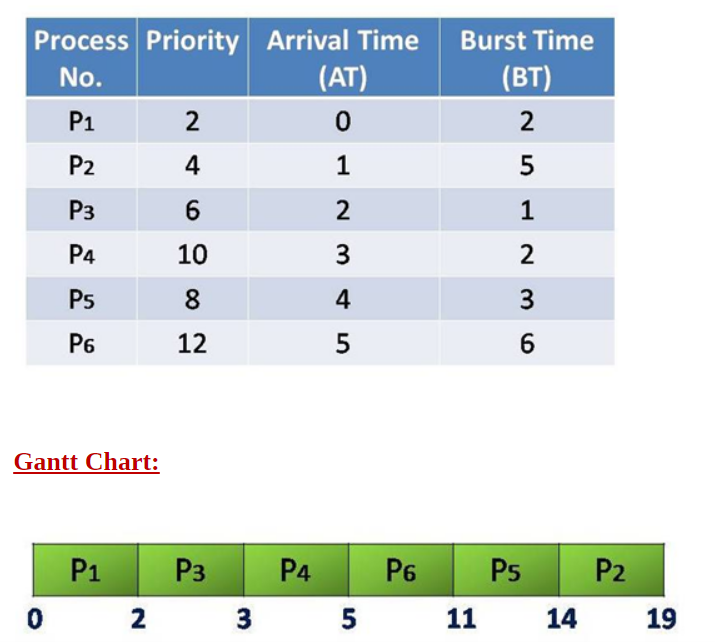
|  |  |  |
| --- | --- | --- |
| **Process** | **CPU-Burst Time** | **Priority** |
| P1 | 10 | 3 |
| P2 | 1 | 1 |
| P3 | 2 | 4 |
| P4 | 1 | 5 |
| P5 | 5 | 2 |

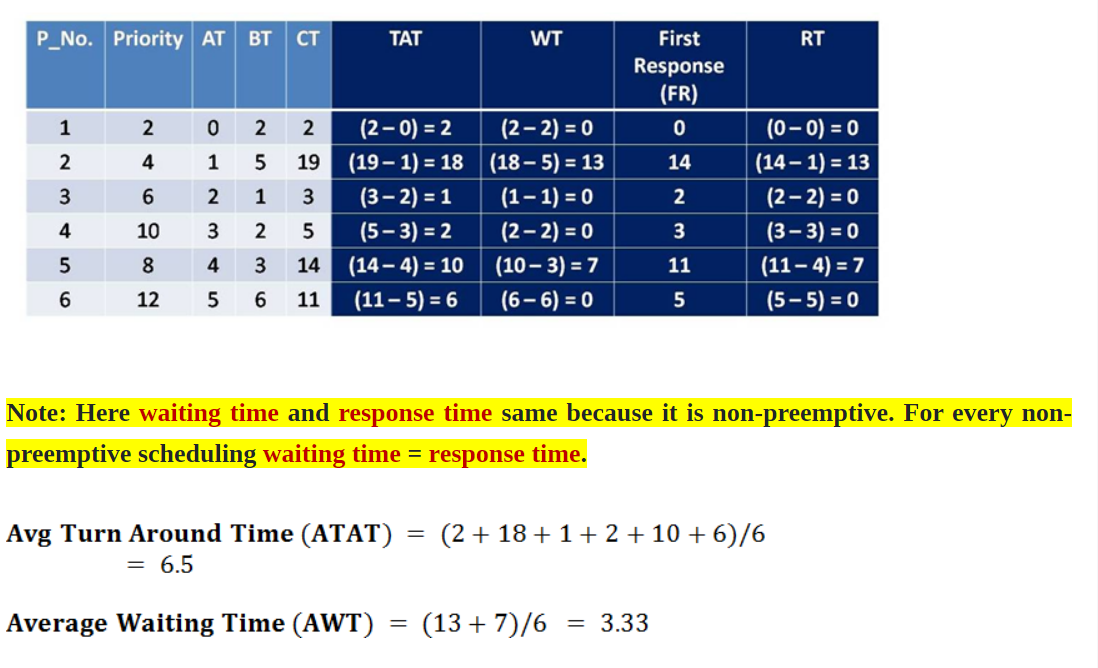
**Gantt Chart**:



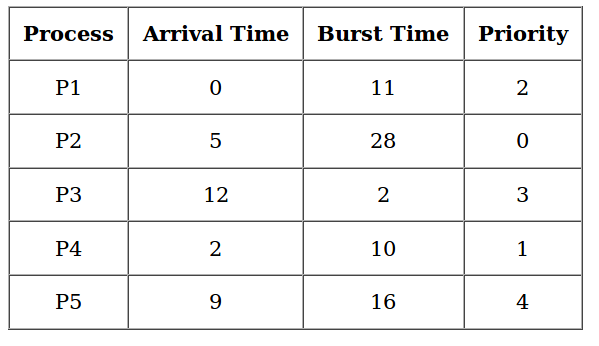
The average waiting time is 8.2 milliseconds.

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

****

****

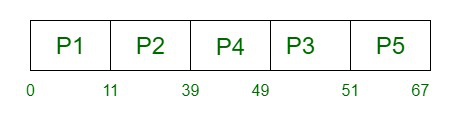
**Example:**



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

## **Solution-**

**Gantt Chart –**



Average Waiting Time is : 5.6 milli seconds

Average Turn Around time is : 8.8 milli seconds

**Preemptive Priority Scheduling:**

Preemptive Priority Scheduling is a scheduling algorithm that is used when a process enters the ready queue first. We compare the priority of the process with other processes, present in the ready queue and, which the CPU is executing at that point of time. The process which has the highest priority among all other processes will be assigned with the CPU first.

In preemptive priority scheduling, if a new process enters into the ready queue, having a higher priority than the currently running process. In such a case, the CPU is preempted to the newly arrived process, i.e., the current process processing gets stopped, and the new incoming process begins its execution.

**Example:** Consider the set of 6 processes whose arrival time and burst time are given below-

|  |  |  |  |
| --- | --- | --- | --- |
| Process | Arrival Time | Burst time | Priority |
| P1 | 0 | 4 | 4 |
| P2 | 1 | 2 | 5 |
| P3 | 2 | 3 | 6 |
| P4 | 3 | 1 | 10 (Highest) |
| P5 | 4 | 2 | 9 |
| P6 | 5 | 6 | 7 |

If the CPU scheduling policy is priority preemptive, calculate the average waiting time and average turnaround time.

**Solution:**

**Example:** Consider the set of 6 processes whose arrival time and burst time are given below-

|  |  |  |  |
| --- | --- | --- | --- |
| Process | Arrival Time | Burst time | Priority |
| P1 | 0 | 7 | 9 |
| P2 | 1 | 3 | 4 |
| P3 | 2 | 5 | 2 |
| P4 | 3 | 2 | 1(Highest) |
| P5 | 4 | 6 | 3 |
| P6 | 5 | 1 | 8 |

If the CPU scheduling policy is priority preemptive, calculate the average waiting time and average turnaround time.

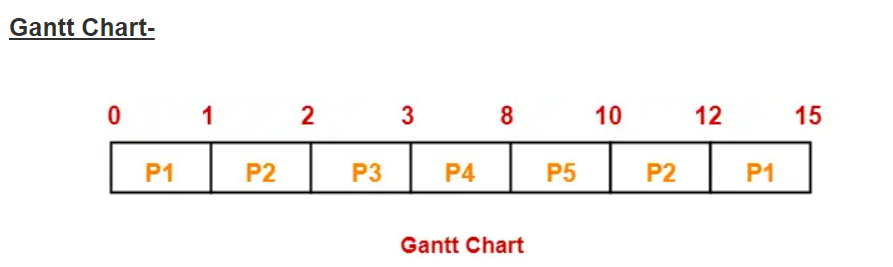
**Solution:**

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

|  |  |  |  |
| --- | --- | --- | --- |
| Process | 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).

## **Solution-**



|  |  |  |  |
| --- | --- | --- | --- |
| **Process Id** | **Exit time** | **Turn Around time** | **Waiting time** |
| P1 | 15 | 15 – 0 = 15 | 15 – 4 = 11 |
| P2 | 12 | 12 – 1 = 11 | 11 – 3 = 8 |
| P3 | 3 | 3 – 2 = 1 | 1 – 1 = 0 |
| P4 | 8 | 8 – 3 = 5 | 5 – 5 = 0 |
| P5 | 10 | 10 – 4 = 6 | 6 – 2 = 4 |

Now,

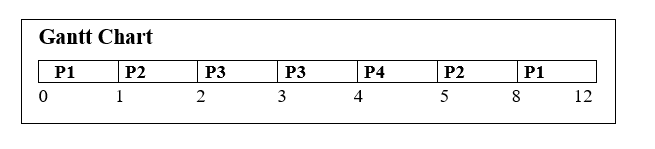
* Average Turn Around time = (15 + 11 + 1 + 5 + 6) / 5 = 38 / 5 = 7.6 unit
* Average waiting time = (11 + 8 + 0 + 0 + 4) / 5 = 23 / 5 = 4.6 unit

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

|  |  |  |  |
| --- | --- | --- | --- |
| Process | Arrival time | Burst time | Priority |
| P1 | 0 | 5 | 11 |
| P2 | 1 | 4 | 22 |
| P3 | 2 | 2 | 33 |
| P4 | 4 | 1 | 44 |

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

## **Solution-**



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **Priority** | **AT** | **BT** | **CT** | **TAT** | **WT** |
| P1 | 11 | 0 | 5 | 12 | 12 | 7 |
| P2 | 22 | 1 | 4 | 8 | 7 | 3 |
| P3 | 33 | 2 | 2 | 4 | 2 | 0 |
| P4 | 44 | 4 | 1 | 5 | 1 | 0 |

**Average Turnaround Time**= 12+7+2+1/4

                                               =22/4

                                               =5.5 milli seconds.

**Average Waiting Time=**7+3+0+0/4

                                       =10/4

                                        =2.5 milli seconds

**Advantages:**

Following are the advantages of priority scheduling method:

1) It considers the priority of the processes and allows the important processes to run first.

2) Priority scheduling in preemptive mode is best suited for real time operating system.

**Disadvantages:**

Following are the disadvantages of priority scheduling method:

1)A major problem with priority scheduling is starvation. A priority scheduling algorithm can leave some low priority processes waiting indefinitely. In a heavy loaded computer system, a steady stream of high priority processes can prevent low priority processes from ever getting the CPU.

**Aging:** A solution to the problem of indefinite blockage of low priority processes or starvation is aging. Aging involves gradually increasing the priority of processes that wait in the system for a long time.

**Example:** if priorities range from 127 (low) to 0 (high), we could increase the priority of a waiting process by 1 every 15 minutes. Eventually, even a process with an initial priority of 127 would have the highest priority in the system and would be executed. In fact, it would take no more than 32 hours for a priority-127 process to age to a priority-0 process.