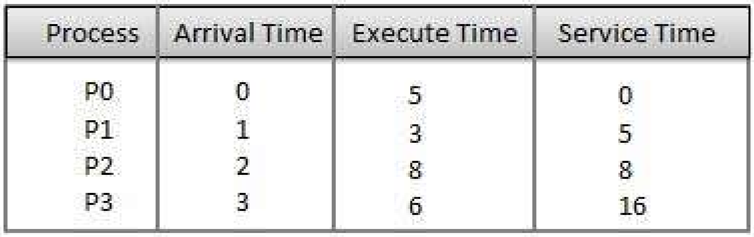
Scheduling algorithms

A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms. There are six popular process scheduling algorithms−

- **First-Come, First-Served (FCFS) Scheduling**
- Shortest-Job-Next (SJN) Scheduling
- Priority Scheduling
- Shortest Remaining Time
- **Round Robin(RR) Scheduling**
- Multiple-Level Queues Scheduling

These algorithms are either **non-preemptive or preemptive**. Non-preemptive algorithms are designed so that once a process enters the running state, it cannot be preempted until it completes its allotted time, whereas the preemptive scheduling is based on priority where a scheduler may preempt a low priority running process anytime when a high priority process enters into a ready state.

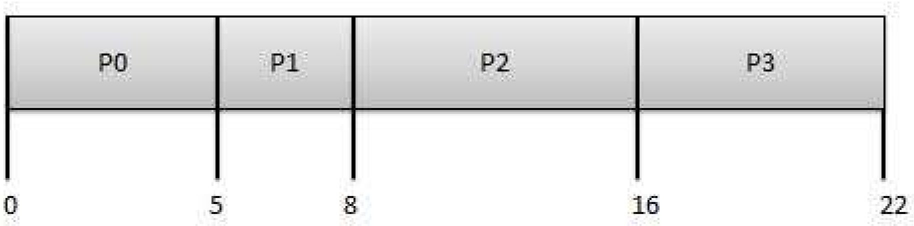
We will consider the following processes for the next two scheduling algorithms



Process	Arrival Time	Execute Time	Service Time
P0	0	5	0
P1	1	3	5
P2	2	8	8
P3	3	6	16

First Come First Serve (FCFS) [**non-preemptive**]

- Jobs are executed on first come, first serve basis.
- It is a non-preemptive, pre-emptive scheduling algorithm.
- Easy to understand and implement.
- Its implementation is based on FIFO queue.
- Poor in performance as average wait time is high.



Wait time of each process is as follows –

Process	**Wait Time : Service Time - Arrival Time**
P0	0 - 0 = 0
P1	5 - 1 = 4
P2	8 - 2 = 6
P3	16 - 3 = 13

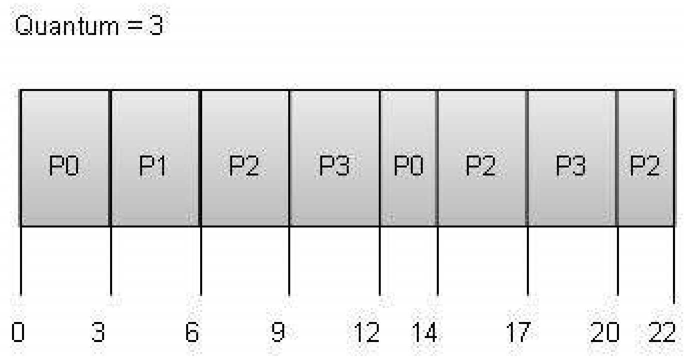
Average Wait Time: (0+4+6+13) / 4 = 5.75

Algorithm:

```
Step 1-  Input the processes along with their execution time (et).

Step 2-  Find waiting time (wt) for all processes.

Step 3-  As first process that comes need not to wait so

         waiting time for process 1 will be 0 i.e. wt[0] = 0.

Step 4-  Find waiting time for all other processes i.e. for
         process i:
         wt[i] = et[i-1] + wt[i-1] .
Step 5-  Find turnaround time = waiting_time + burst_time
         for all processes.
Step 6-  Find average waiting time = total_waiting_time / no_of_processes.
Step 7-  Find average turnaround time = total_turn_around_time / no_of_processes.
```

Round Robin Scheduling [**preemptive**]

- Round Robin is the preemptive process scheduling algorithm.
- Each process is provided a fix time to execute, it is called a **quantum**.
- Once a process is executed for a given time period, it is preempted and other process executes for a given time period.
- Context switching is used to save states of preempted processes.



Wait time of each process is as follows −

Process	Wait Time : Service Time - Arrival Time
P0	(0 - 0) + (12 - 3) = 9
P1	(3 - 1) = 2
P2	(6 - 2) + (14 - 9) + (20 - 17) = 12
P3	(9 - 3) + (17 - 12) = 11

Average Wait Time: (9+2+12+11) / 4 = 8.5

Algorithm:

```
Step 1- Create an array rem_et[] to keep track of remaining
        execution time of processes. This array is initially a
        copy of et[] (execution times array)
Step 2- Create another array wt[] to store waiting times
        of processes. Initialize this array as 0.
Step 3- Initialize time : t = 0
Step 4- Keep traversing the all processes while all processes
        are not done. Do following for i'th process if it is
        not done yet.
          If rem_et[i] > quantum
             (i)  t = t + quantum
             (ii) et_rem[i] -= quantum;
          Else // Last cycle for this process
             (i)  t = t + et_rem[i];
             (ii) wt[i] = t - et[i]
             (ii) et_rem[i] = 0; // This process is over
```