CPU Scheduling

Fatemeh Masoodi| Negin Khalifat | 1/20/2020

FCFS

First Come First Serve (FCFS) is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.

ADVANTAGES OF FCFS

Here, are pros/benefits of using FCFS scheduling algorithm:

- The simplest form of a CPU scheduling algorithm
- Easy to program
- First come first served

DISADVANTAGES OF FCFS

Here, are cons/ drawbacks of using FCFS scheduling algorithm:

- It is a Non-Preemptive CPU scheduling algorithm, so after the process has been allocated to the CPU, it will never release the CPU until it finishes executing.
- The Average Waiting Time is high.
- Short processes that are at the back of the queue have to wait for the long process at the front to finish.
- Not an ideal technique for time-sharing systems.
- Because of its simplicity, FCFS is not very efficient.

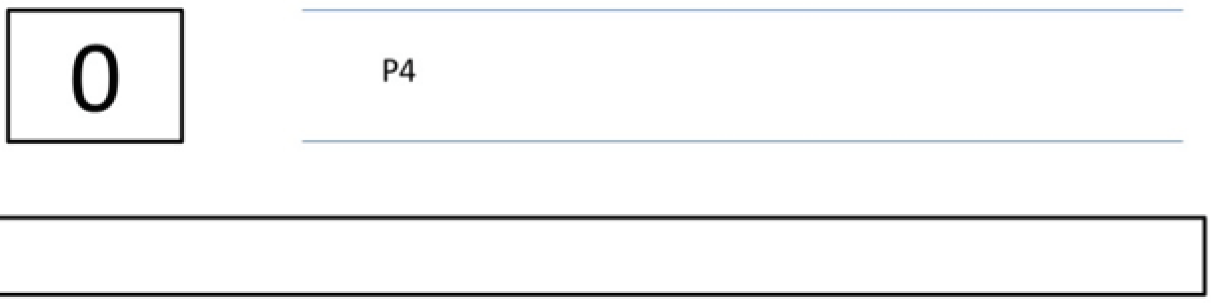
HOW FCFS WORKS? CALCULATING AVERAGE WAITING TIME

Here is an example of five processes arriving at different times. Each process has a different burst time.

Process	Burst time	Arrival time
P1	6	2
P2	3	5
P3	8	1
P4	3	0
P5	4	4

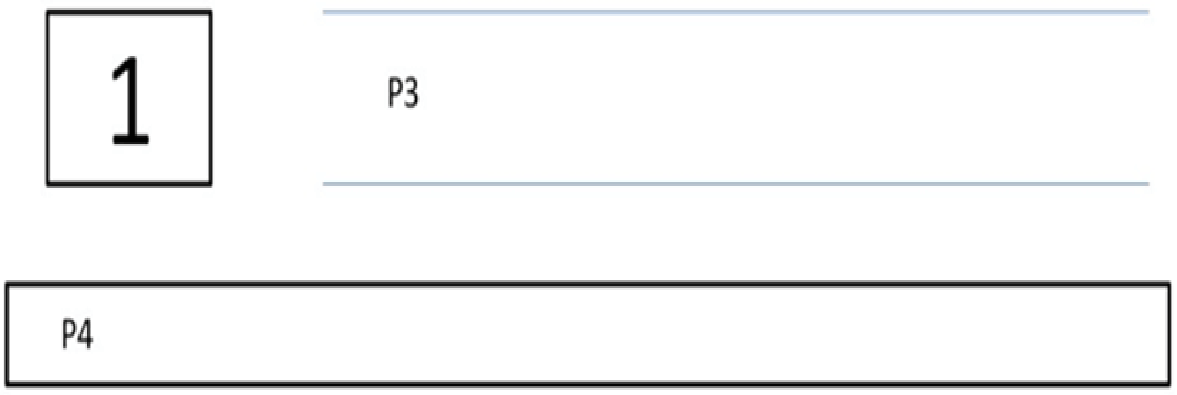
Using the FCFS scheduling algorithm, these processes are handled as follows.

Step 0) The process begins with P4 which has arrival time 0



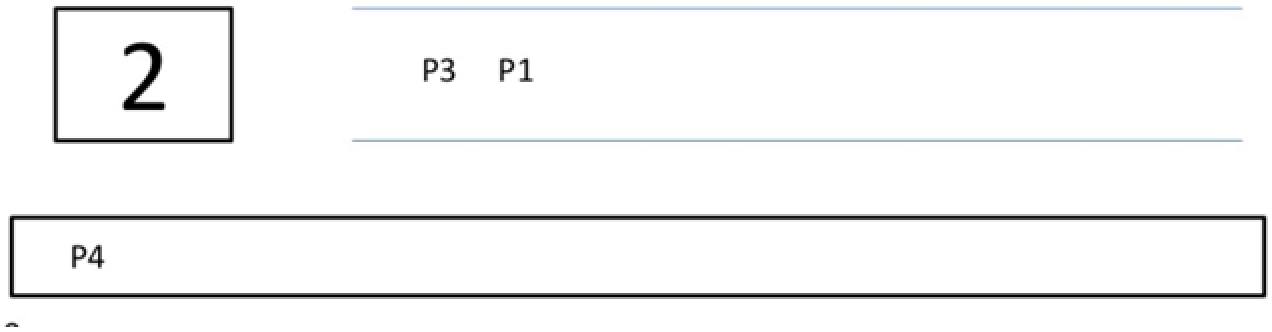
Step 1) At time=1, P3 arrives. P4 is still executing. Hence, P3 is kept in a queue.

Process	Burst time	Arrival time
P1	6	2
P2	3	5
P3	8	1
P4	3	0
P5	4	4

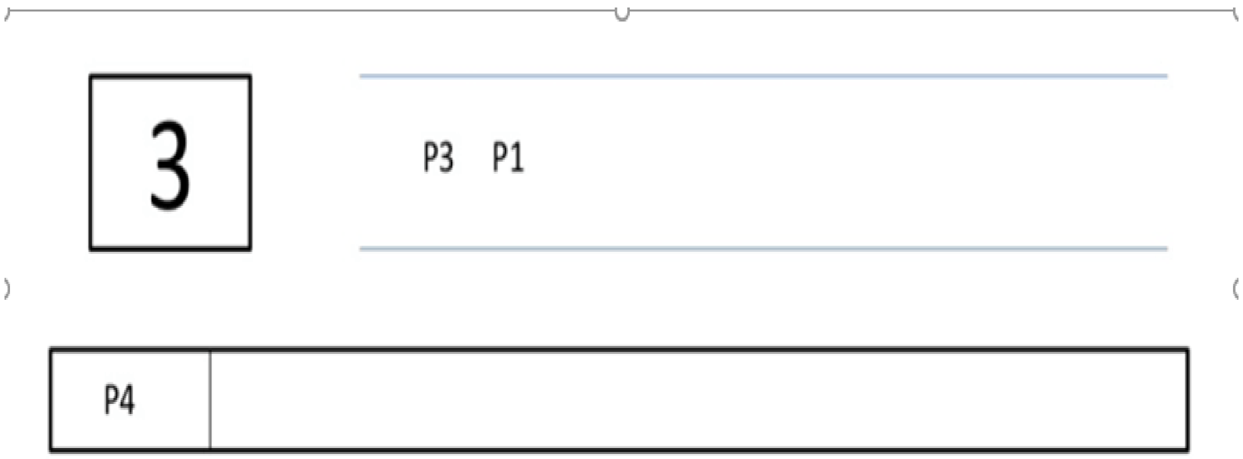


Step 2) At time= 2, P1 arrives which is kept in the queue.

Process	Burst time	Arrival time
P1	6	2
P2	3	5
P3	8	1
P4	3	0
P5	4	4

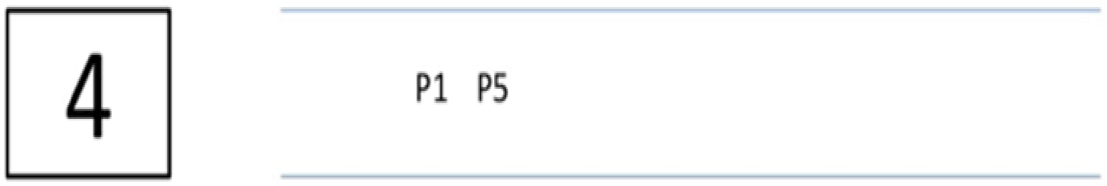


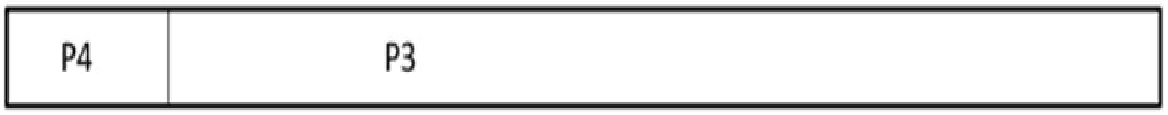
Step 3) At time=3, P4 process completes its execution.



Step 4) At time=4, P3, which is first in the queue, starts execution.

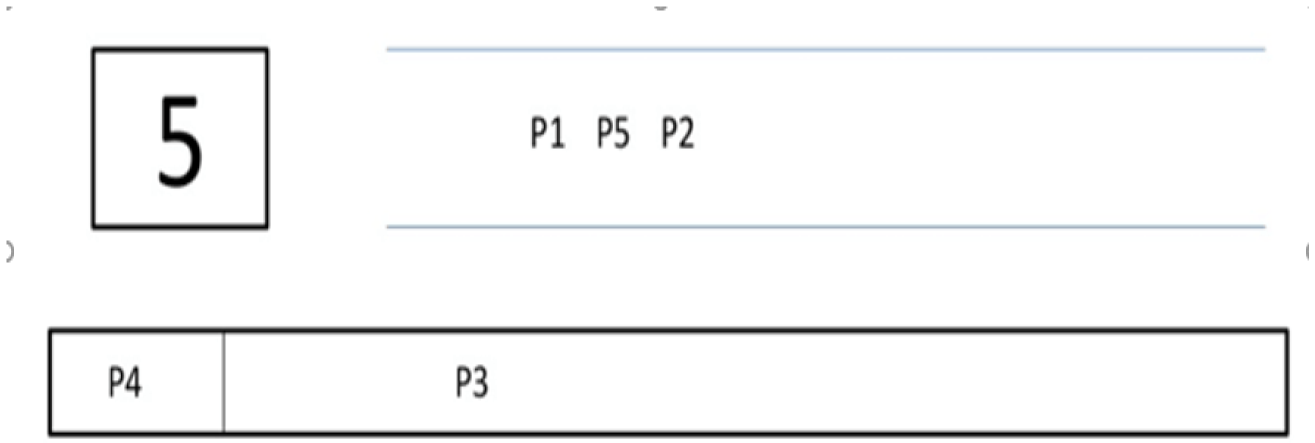
Process	**Burst time**	**Arrival time**
P1	6	2
P2	3	5
P3	8	1
P4	3	0
P5	4	4



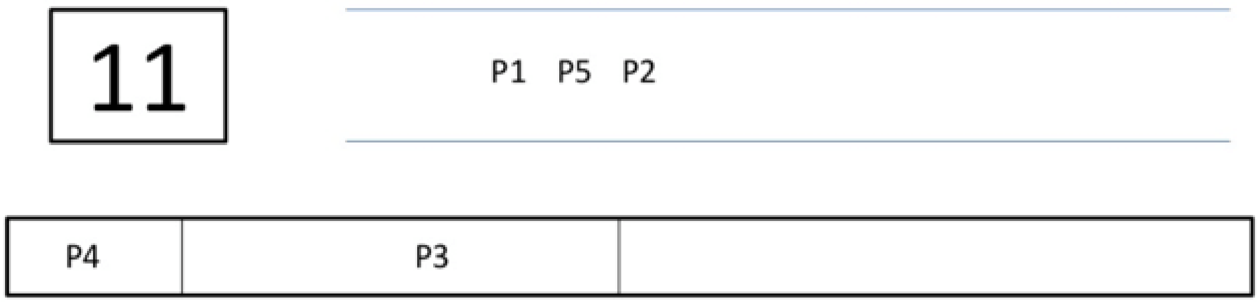


Step 5) At time =5, P2 arrives, and it is kept in a queue.

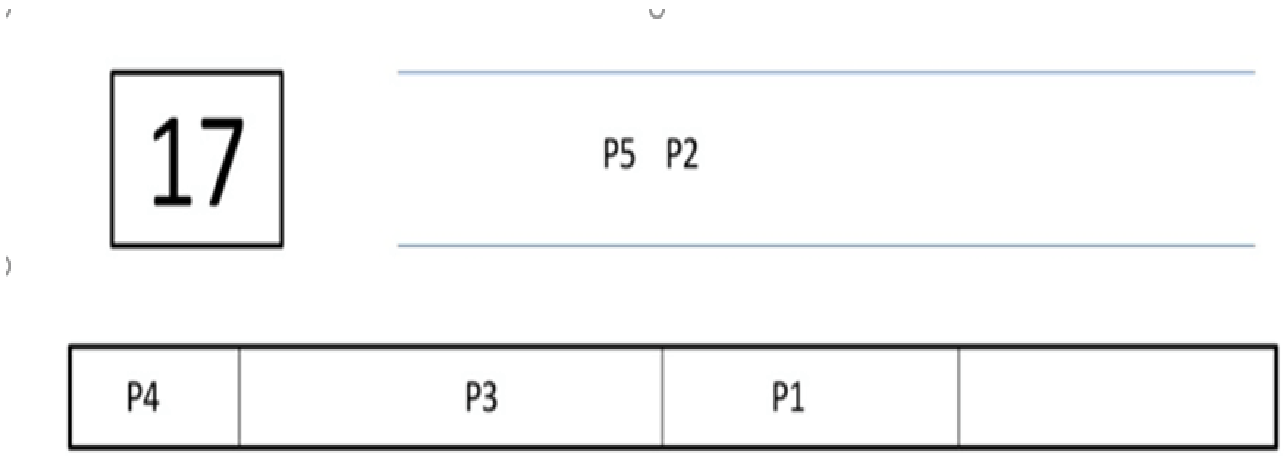
Process	Burst time	Arrival time
P1	6	2
P2	3	5
P3	8	1
P4	3	0
P5	4	4



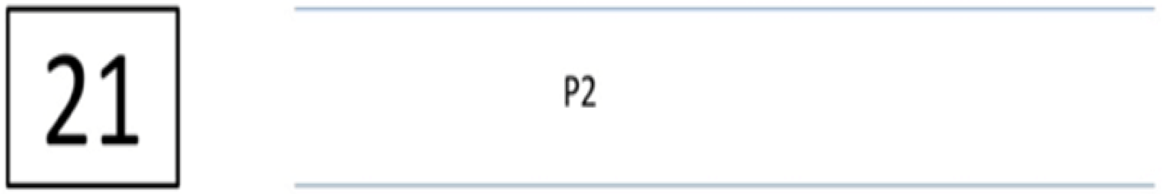
Step 6) At time 11, P3 completes its execution.

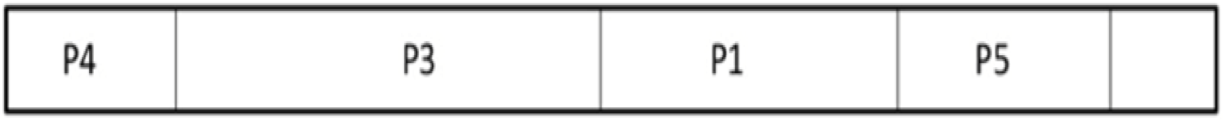


Step 7) At time=11, P1 starts execution. It has a burst time of 6. It completes execution at time interval 17

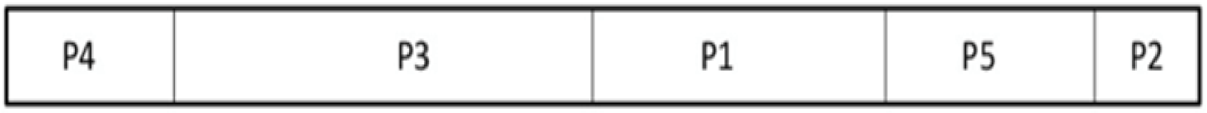


Step 8) At time=17, P5 starts execution. It has a burst time of 4. It completes execution at time=21

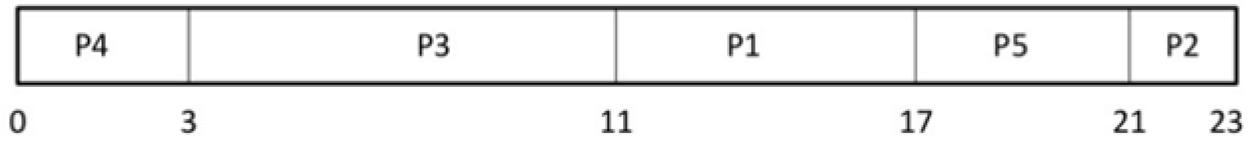




Step 9) At time=21, P2 starts execution. It has a burst time of 2. It completes execution at time interval 23



Step 10) Let's calculate the average waiting time for above example.



```
Waiting time = Start time - Arrival time
```

P4 = 0-0 = 0

P3 = 3-1 = 2

PI = 11-2 = 9

P5= 17-4 = 13

P2= 21-5= 16

Average Waiting Time=

$$\frac{0+2+9+13+16}{5}$$

= 40/5= 8

SJF

Shortest Job First (SJF) is an algorithm in which the process having the smallest execution time is chosen for the next execution. This scheduling method can be preemptive or non-preemptive. It significantly reduces the average waiting time for other processes awaiting execution. The full form of SJF is Shortest Job First.

There are basically two types of SJF methods:

- Non-Preemptive SJF
- Preemptive SJF

CHARACTERISTICS OF SJF SCHEDULING

- It is associated with each job as a unit of time to complete.
- This algorithm method is helpful for batch-type processing, where waiting for jobs to complete is not critical.
- It can improve process throughput by making sure that shorter jobs are executed first, hence possibly have a short turnaround time.
- It improves job output by offering shorter jobs, which should be executed first, which mostly have a shorter turnaround time.

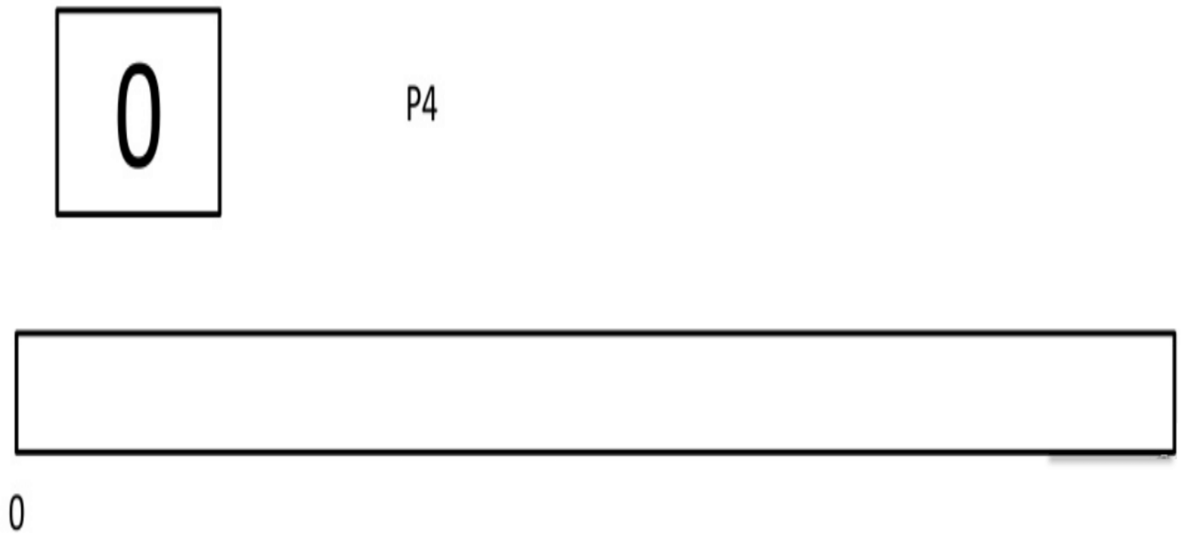
NON-PREEMPTIVE SJF

In non-preemptive scheduling, once the CPU cycle is allocated to process, the process holds it till it reaches a waiting state or terminated.

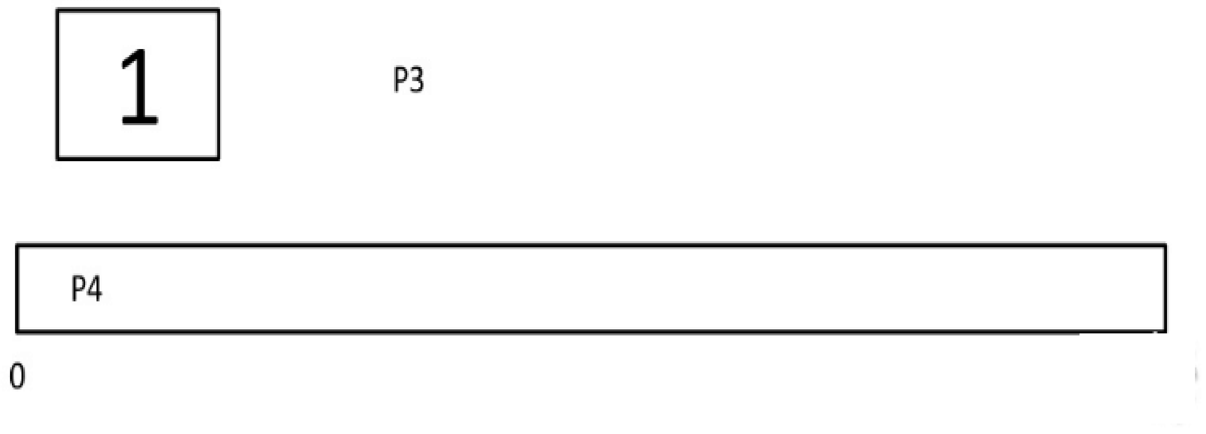
Consider the following five processes each having its own unique burst time and arrival time.

Process Queue	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

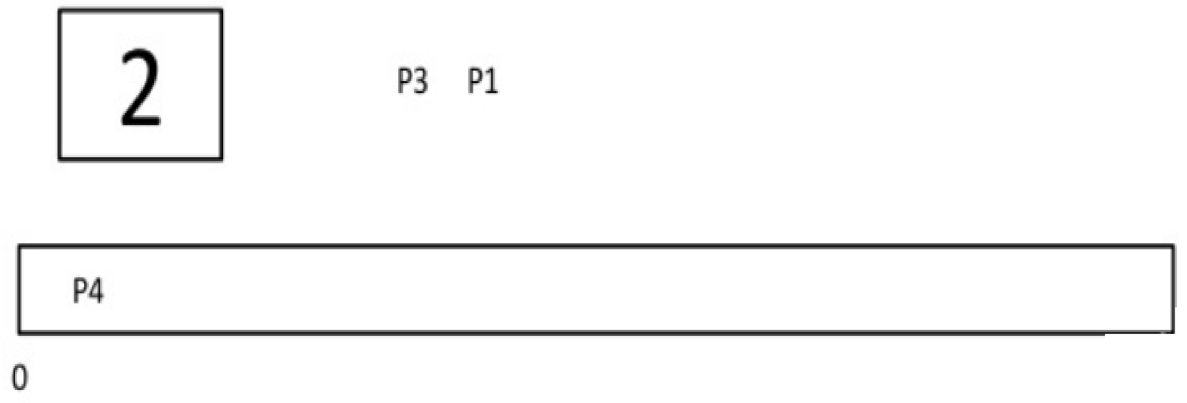
Step 0) At time=0, P4 arrives and starts execution.



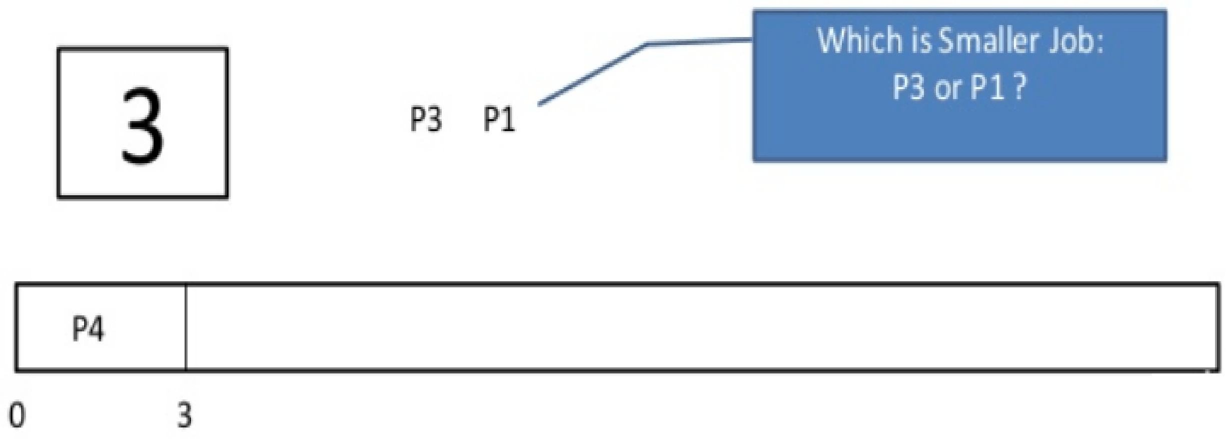
Step 1) At time= 1, Process P3 arrives. But, P4 still needs 2 execution units to complete. It will continue execution.



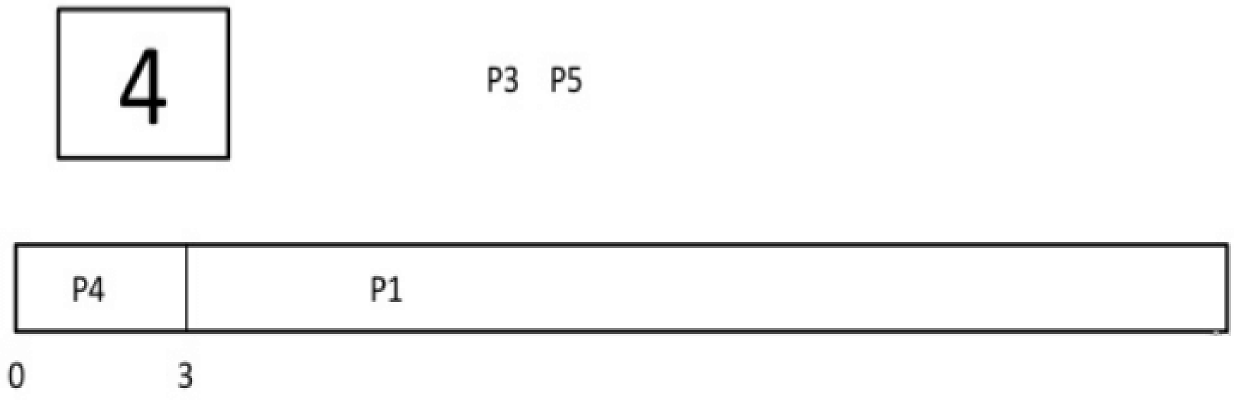
Step 2) At time =2, process P1 arrives and is added to the waiting queue. P4 will continue execution.



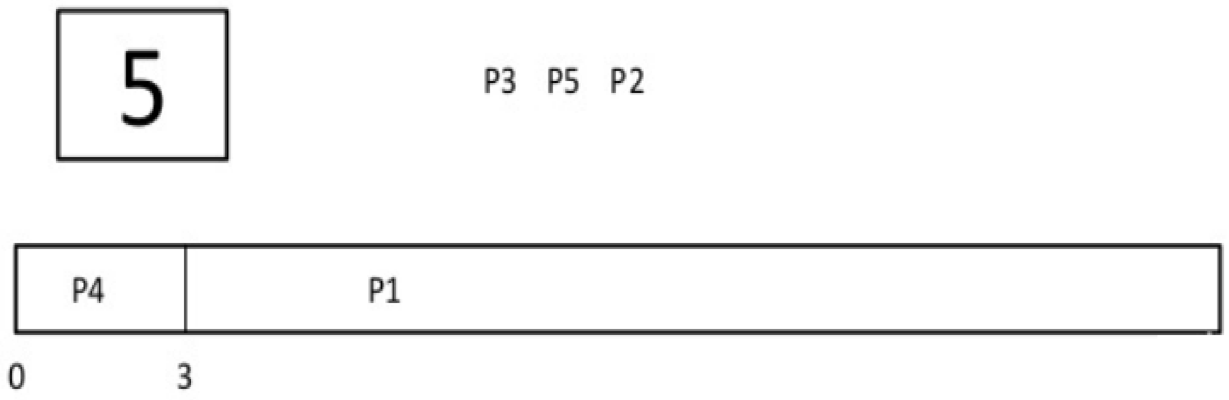
Step 3) At time = 3, process P4 will finish its execution. The burst time of P3 and P1 is compared. Process P1 is executed because its burst time is less compared to P3.



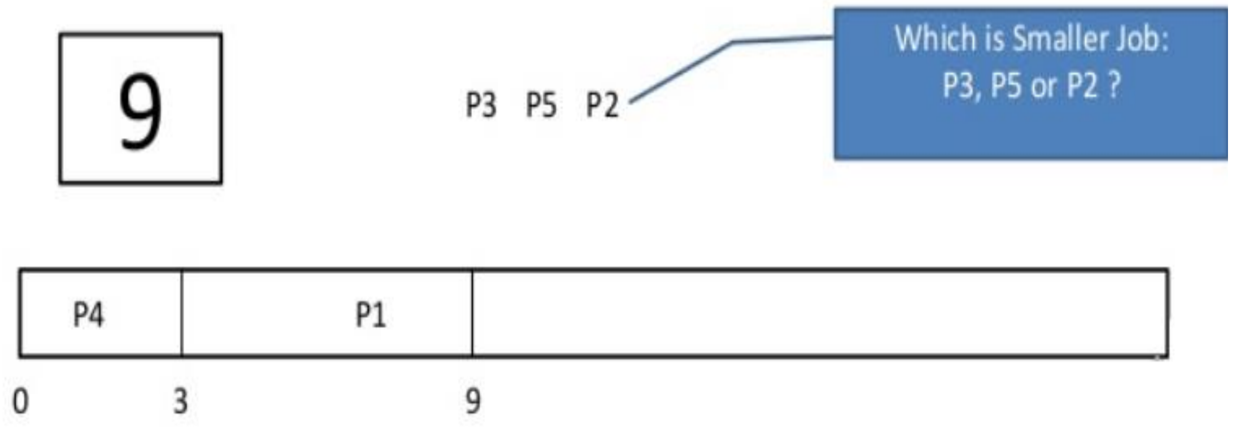
Step 4) At time = 4, process P5 arrives and is added to the waiting queue. P1 will continue execution.



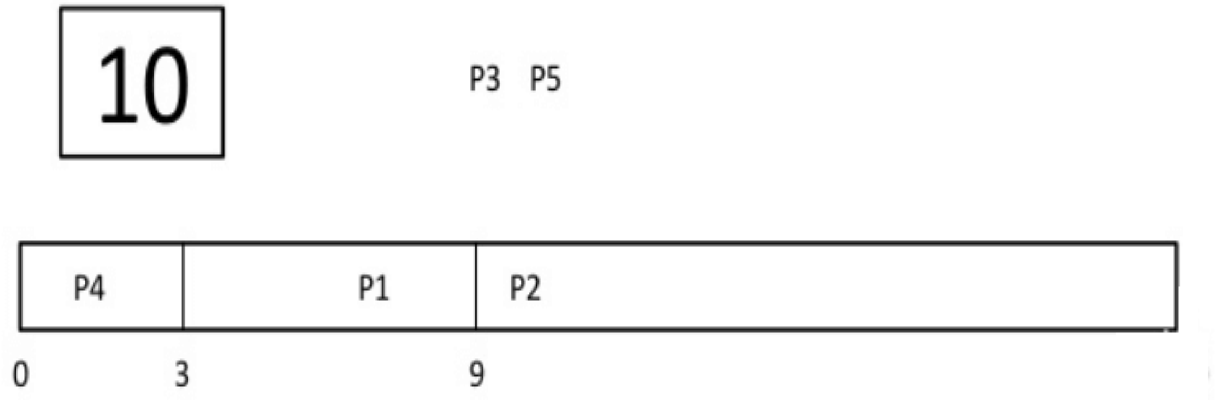
Step 5) At time = 5, process P2 arrives and is added to the waiting queue. P1 will continue execution.



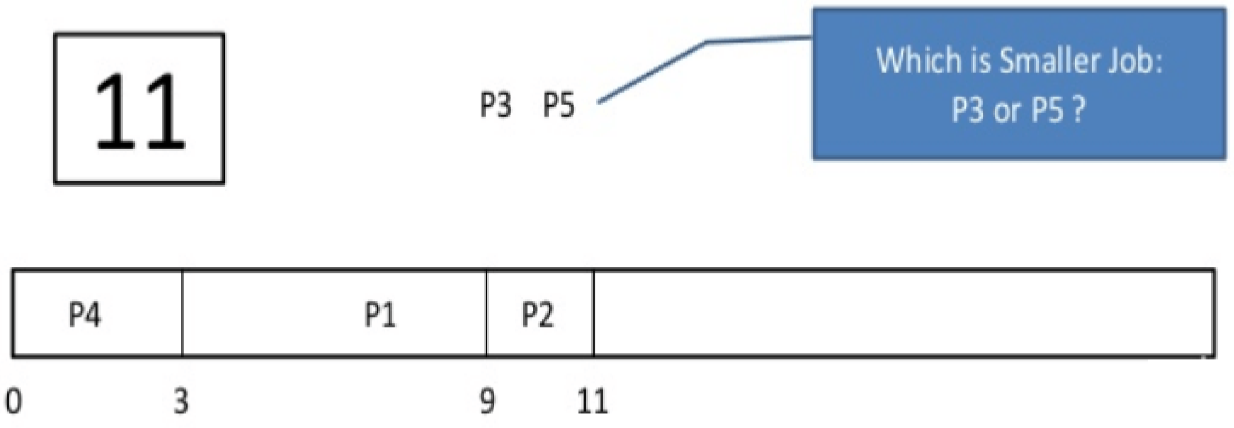
Step 6) At time = 9, process P1 will finish its execution. The burst time of P3, P5, and P2 is compared. Process P2 is executed because its burst time is the lowest.



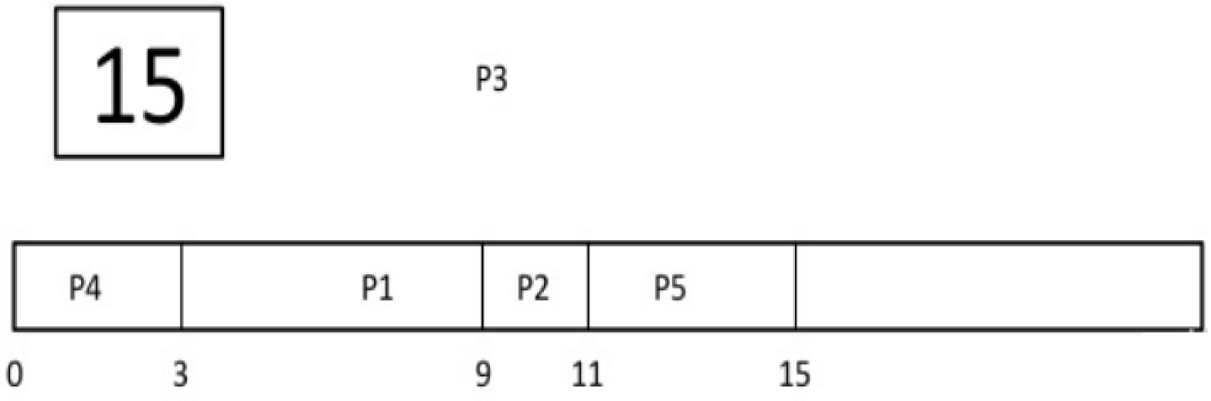
Step 7) At time=10, P2 is executing and P3 and P5 are in the waiting queue.



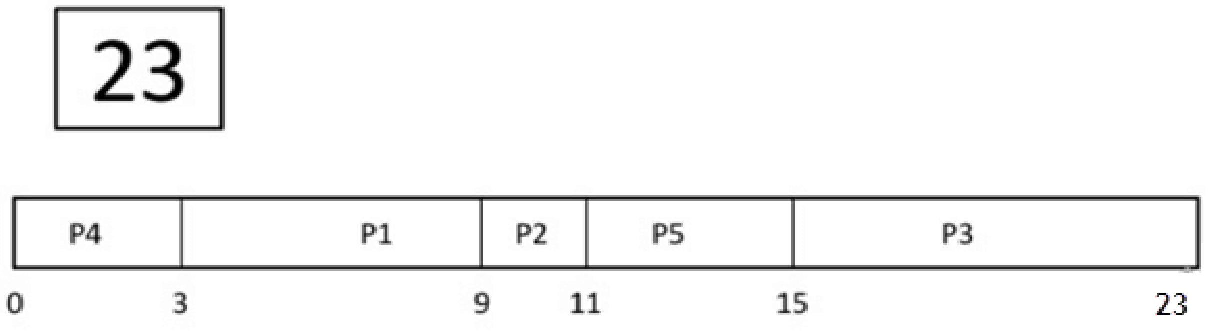
Step 8) At time = 11, process P2 will finish its execution. The burst time of P3 and P5 is compared. Process P5 is executed because its burst time is lower.



Step 9) At time = 15, process P5 will finish its execution.



Step 10) At time = 23, process P3 will finish its execution.



Step 11) Let's calculate the average waiting time for above example.

```
Waiting time
P4= 0-0=0
P1= 3-2=1
P2= 9-5=4
P5= 11-4=7
P3= 15-1=14
Average Waiting Time= 0+1+4+7+14/5 = 26/5 = 5.2
```

PREEMPTIVE SJF CALLED SRT

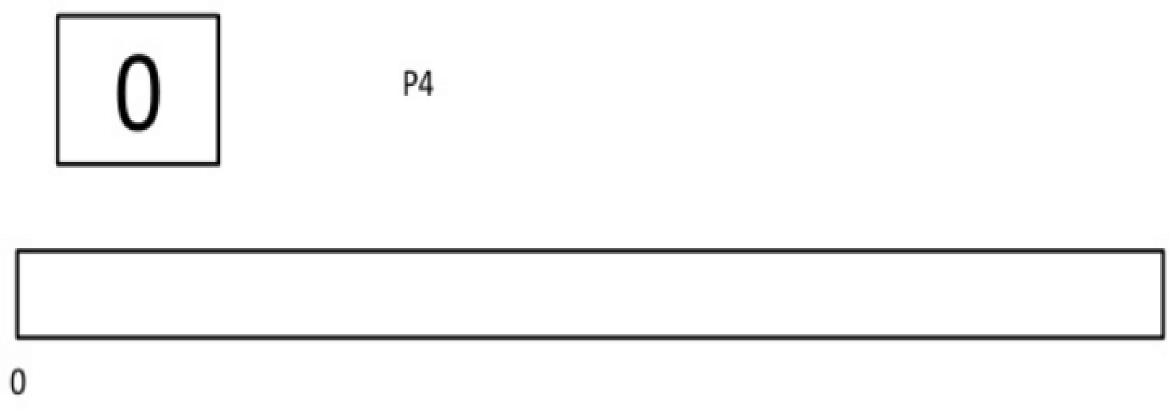
In Preemptive SJF Scheduling, jobs are put into the ready queue as they come. A process with shortest burst time begins execution. If a process with even a shorter burst time arrives, the current process is removed or preempted from execution, and the shorter job is allocated CPU cycle.

Consider the following five process:

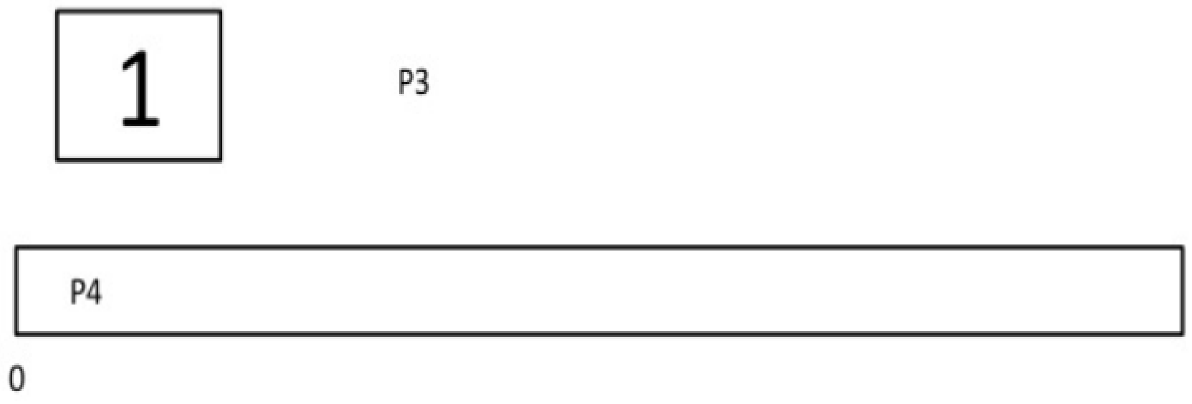
Process Queue	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

Step 0) At time=0, P4 arrives and starts execution.

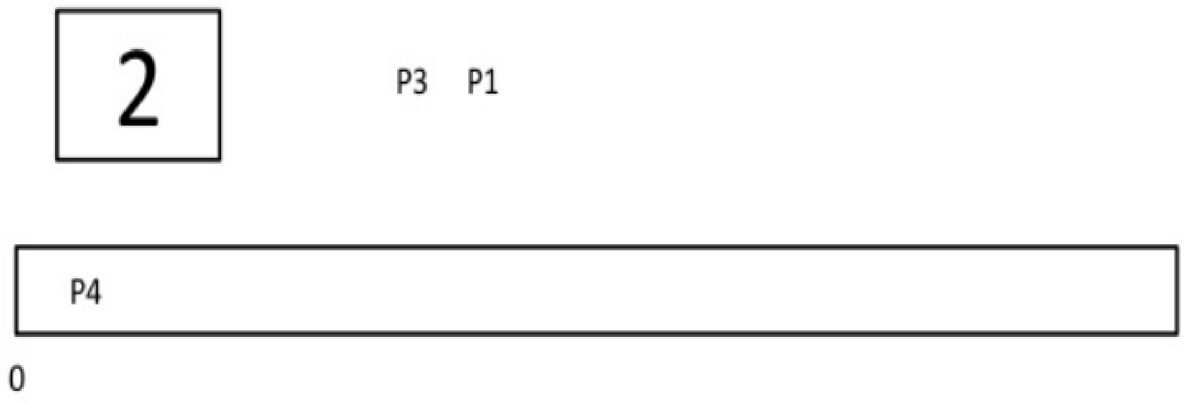
Process Queue	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4



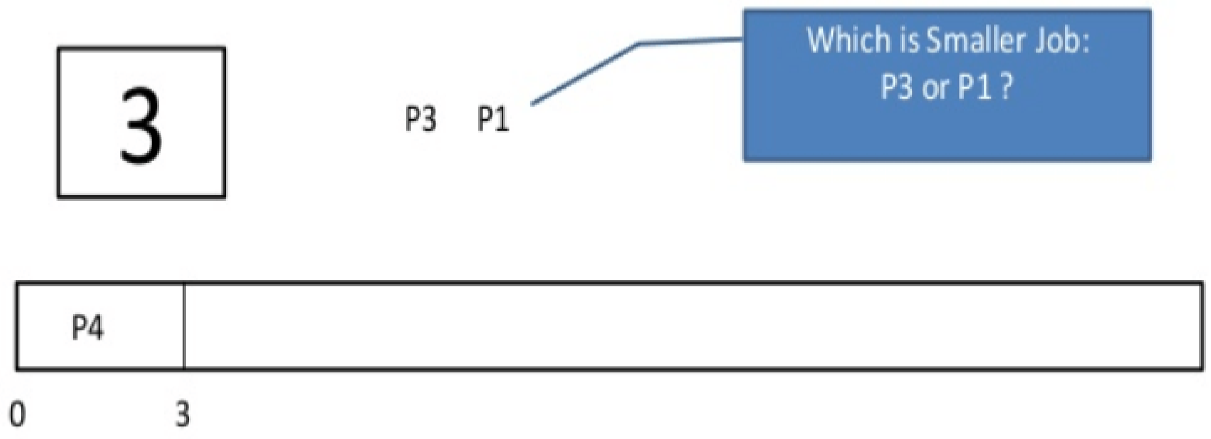
Step 1) At time= 1, Process P3 arrives. But, P4 has a shorter burst time. It will continue execution.



Step 2) At time = 2, process P1 arrives with burst time = 6. The burst time is more than that of P4. Hence, P4 will continue execution.

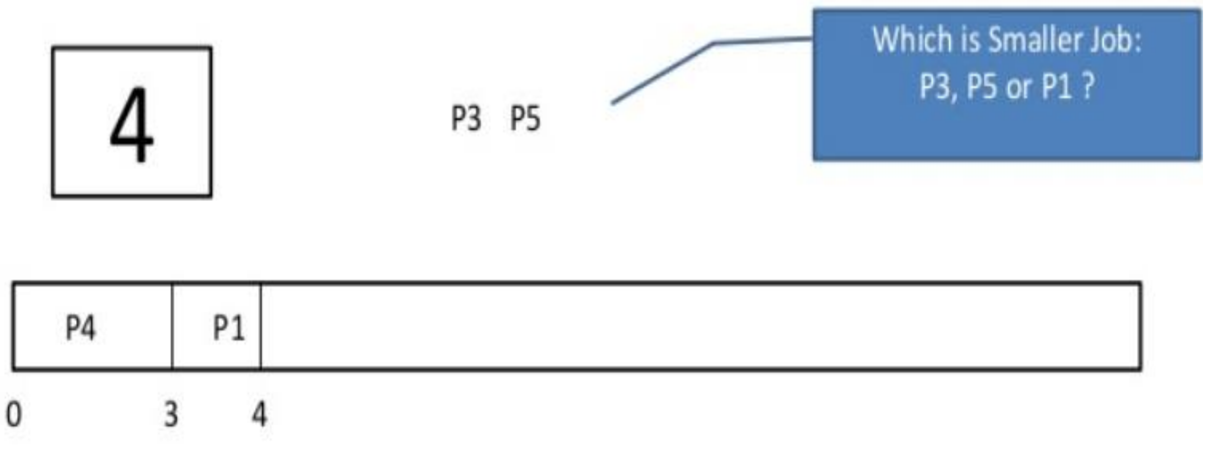


Step 3) At time = 3, process P4 will finish its execution. The burst time of P3 and P1 is compared. Process P1 is executed because its burst time is lower.



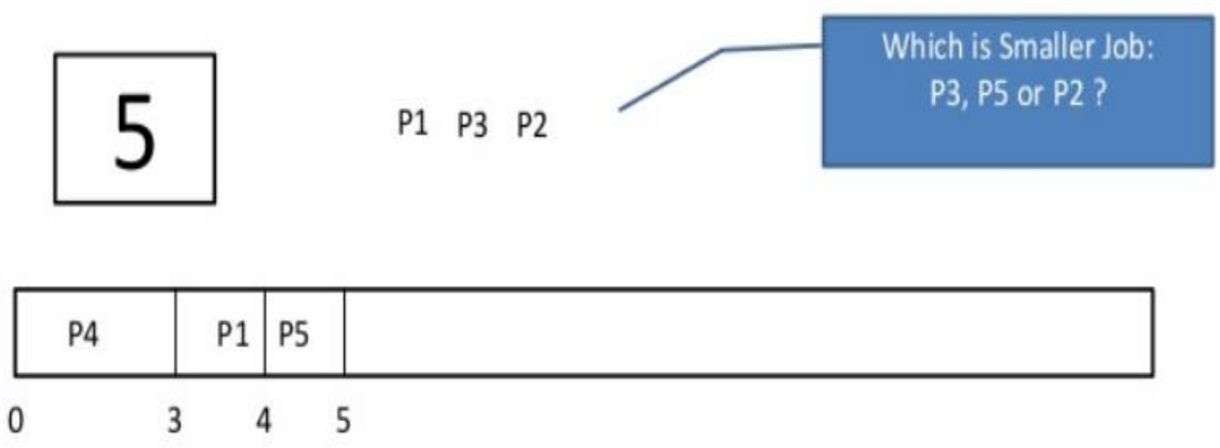
Step 4) At time = 4, process P5 will arrive. The burst time of P3, P5, and P1 is compared. Process P5 is executed because its burst time is lowest. Process P1 is preempted.

Process Queue	Burst time	Arrival time
P1	5 out of 6 is remaining	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

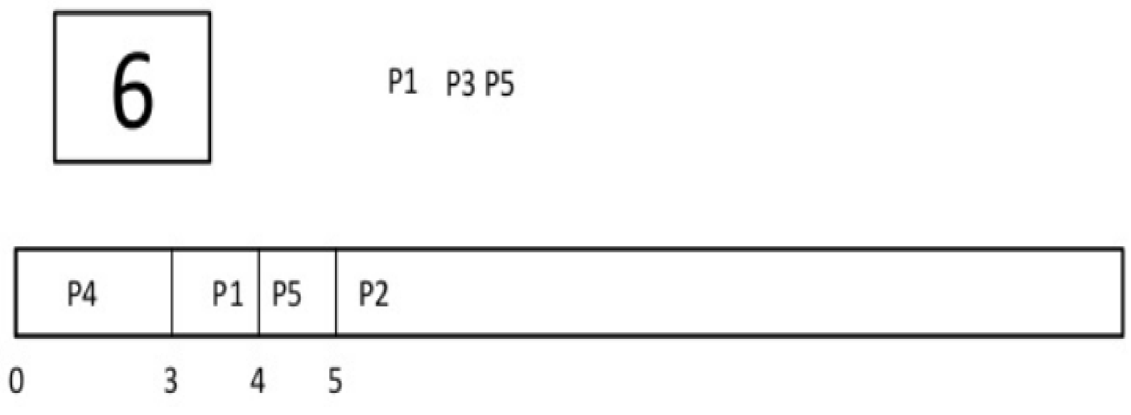


Step 5) At time = 5, process P2 will arrive. The burst time of P1, P2, P3, and P5 is compared. Process P2 is executed because its burst time is least. Process P5 is preempted.

Process Queue	Burst time	Arrival time
P1	5 out of 6 is remaining	2
P2	2	5
P3	8	1
P4	3	0
P5	3 out of 4 is remaining	4

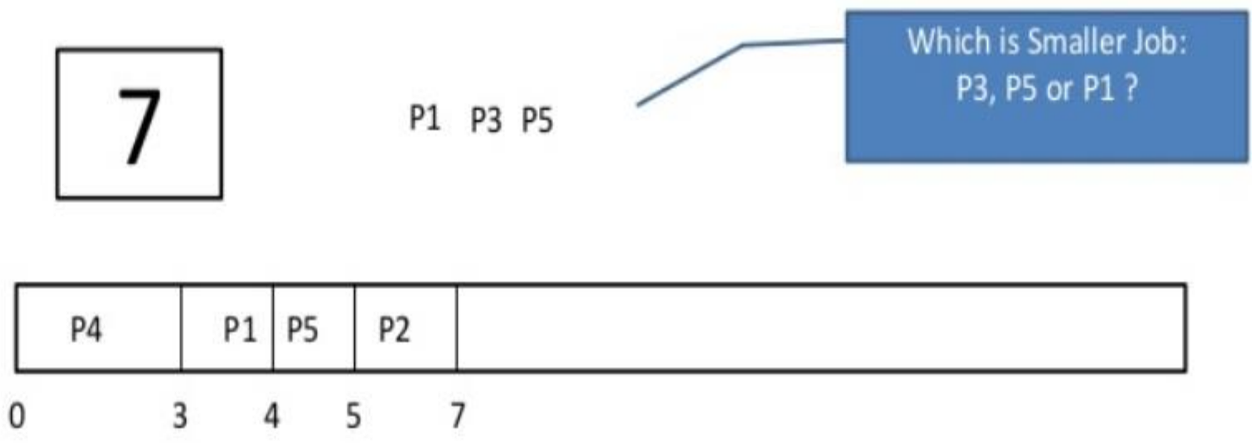


Step 6) At time =6, P2 is executing.

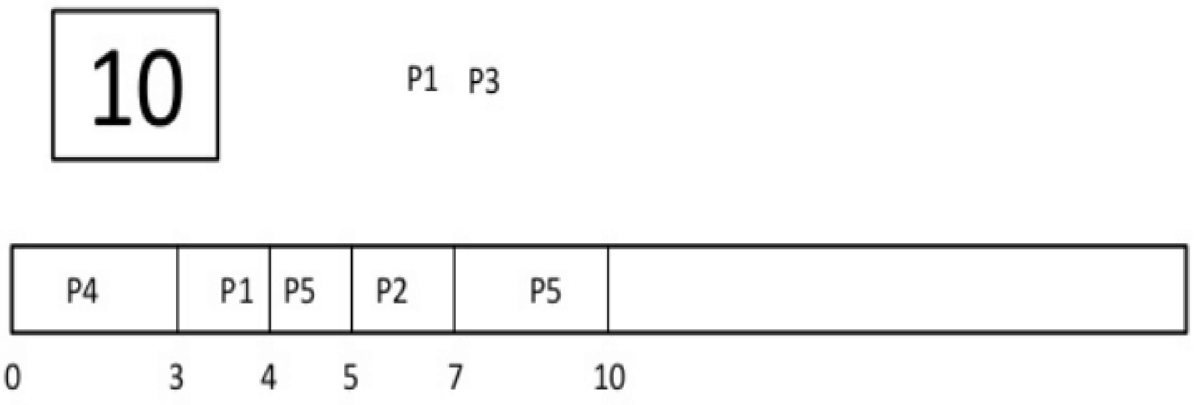


Step 7) At time =7, P2 finishes its execution. The burst time of P1, P3, and P5 is compared. Process P5 is executed because its burst time is lesser.

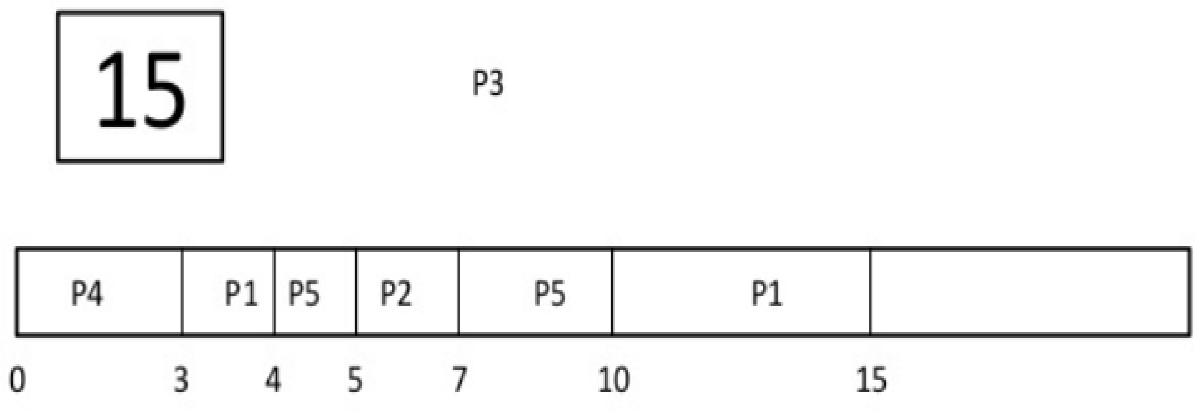
Process Queue	Burst time	Arrival time
P1	5 out of 6 is remaining	2
P2	2	5
P3	8	1
P4	3	0
P5	3 out of 4 is remaining	4



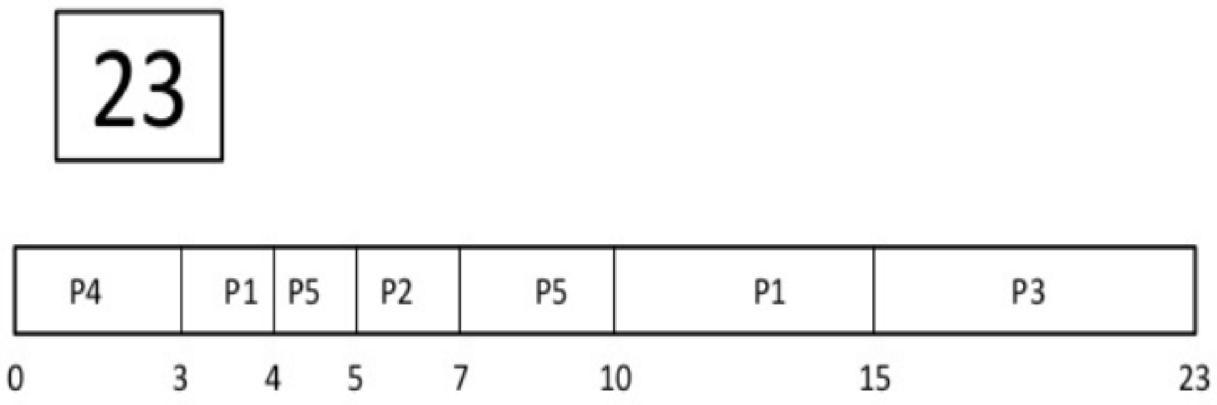
Step 8) At time =10, P5 will finish its execution. The burst time of P1 and P3 is compared. Process P1 is executed because its burst time is less.



Step 9) At time =15, P1 finishes its execution. P3 is the only process left. It will start execution.



Step 10) At time =23, P3 finishes its execution.



Step 11) Let's calculate the average waiting time for above example.

```
Wait time
P4= 0-0=0
P1= (3-2) + 6 =7
P2= 5-5 = 0
P5= 4-4+2 =2
P3= 15-1 = 14
Average Waiting Time = 0+7+0+2+14/5 = 23/5 =4.6
```

ROUND-ROBIN SCHEDULING

Round robin is the oldest, simplest scheduling algorithm. The name of this algorithm comes from the round-robin principle, where each person gets an equal share of something in turn. It is mostly used for scheduling algorithms in multitasking. This algorithm method helps for starvation free execution of processes.

Characteristics of Round-Robin Scheduling

- Round robin is a hybrid model which is clock-driven
- Time slice should be minimum, which is assigned for a specific task to be processed. However, it may vary for different processes.
- It is a real time system which responds to the event within a specific time limit.

- In our project we implement these CPU scheduling algorithm in python.
- Our purpose is to show which algorithm is worked efficiently.
- Steps of project:
    - ✓ Read from test.csv which is conclude information about process.
    - ✓ Get context switch time and time quantum from user
    - ✓ Implement each algorithm(FSFS,SJF,SRT,RR)
    - ✓ Show gantt_chart and I/O time
    - ✓ Calculate average of turnaround time , waiting time ,response time , cpu utilization , throughput
    - ✓ Save result
    - ✓ DONE!