Project 1: Task Scheduling

Input:

Process ID	Burst Time	Arrival Time	I/O Operations
P1	10	0	[2, 3]
P2	6	2	[1, 2]
P3	8	4	[3, 1]

I/O_Operations[0] = `io_start_time`

I/O_Operations[1] = `io_processing_time`

Let's solve this for FCFS.

Gantt Chart step by step,

|__P1__|

0 2

At this point, P2 arrives and P1 starts it's I/O operation. Remember I/O operation doesn't require CPU time/processing. So P1 is basically needs 3 sec of I/O processing time hence won't available or won't be in the ready queue until 5th sec. It will stay in the waiting_queue to complete I/O operation and will get back into ready queue at the 5th sec. Now P2 will start executing and will get executed for 1 sec.

P1 = remaining (10- 2). 8 sec

|__P1__|__P2__|

0 2 3

At this moment, p2 goes for I/O operation (as you can see in **I/O_Operations[0] = 1 for p2**). Now, p1 can't start till 5th and p2 cant start until (5 + 2) 7th sec. Why? because I/O operation is a single path execution, while I/O operator is busy with P1, P2 had to wait to perform it's I/O (only one can perform I/O operations at a time). Which means after 3 sec, the CPU doesn't have anything to execute until P3 arrives.

|__P1__|__P2__|_waste_|

0 2 3 4

After, 4 sec, P3 arrives. CPU executes it for 1 sec. (P2 can't start until 5th sec and P2 can't start until 7th)

|__P1__|__P2__|_waste_|__P3__|

0 2 3 4 5

Remaining burst time:

P1 = 8 sec, P2 = 5 sec, P3 = 7 Sec

Now we, reached at the 5th sec. At this point, P3 and P1 are available to be executed. Now which one has higher priority?

1. P1(in case of FCFC)
2. P3 (in case of SJF)
3. In case of priority scheduling (priority column will tell you which one has higher priority).
4. In the case of RR, it depends on the time quantum, Q.

As we were solving using FCFS algorithm, P1 gets higher priority. So P1 runs it's remaining 8 sec of burst time.

|__P1__|__P2__|_waste_|__P3__|__P1__|

0 2 3 4 5 13

At this point, P2 is also done with it's I/O operation and P3 needs to execute 2 more sec before it can start it's I/O operation. Priority wise, P2 came before P3. P2 executes it's remaining burst time.

|__P1__|__P2__|_waste_|__P3__|__P1__|__P2__|

0 2 3 4 5 13 18

At this point only P3 is left. P3 will run for 2 sec. And then it needs 1 sec to complete its I/O operation. Since no other process is there, CPU cycle will be wasted.

|__P1__|__P2__|_waste_|__P3__|__P1__|__P2__|__P3__|_wasted_|

0 2 3 4 5 13 18 20 21

At this point, the remaining burst time for P3 is 5 sec. So now it will be executed.

|__P1__|__P2__|_waste_|__P3__|__P1__|__P2__|__P3__|_wasted_|__P3__|

0 2 3 4 5 13 18 20 21 26

So, the summary:

1. P1 Arrival:

- P1 starts at 0 and runs for 2 seconds.

2. P2 Arrival:

- At the 2nd second, P2 arrives, and P1 initiates an I/O operation, which takes 3 seconds.
- P1 remains in the waiting queue until the 5th second.
- Meanwhile, P2 runs for 1 second.
- Remaining burst time for P1: 10 - 2 = 8 seconds.

3. P3 Arrival:

    - At the 4th second, P3 arrives, and P2 starts its I/O operation.

    - P2 remains in the waiting queue until the 7th second.

    - At the 5th second, P3 runs for 1 second.

    - Remaining burst time for each process: P1 = 8 sec, P2 = 5 sec, P3 = 7 sec.

4. Priority Decision at the 5th Second:

    - P1, being the first to arrive, gets priority in FCFS.

    - P1 runs its remaining 8 seconds of burst time.

5. Continuation:

    - P2 completes its I/O operation and runs for its remaining burst time.

    - P3, being the last, executes its remaining time.

    - CPU cycle is wasted after P3's completion due to no other processes.

6. Conclusion:

    - P3 runs again to utilize its remaining burst time.

I hope this clarifies the process handling and input requirements. Remember, each process undergoes an I/O operation once during its lifetime to simplify the analysis.