Faculty of Engineering & Technology
Electrical & Computer Engineering Department

Operating Systems

ENCS 339

Project #1

"Simulation of CPU scheduling algorithms"

PREPARED BY:
Waseem sayara ID: (#1182733).
Anas Barakat ID: (#1180180).

INSTRUCTOR: Dr. Adnad H. Yahya.

SECTION: 2

DATE: 18-11-2020

❖ Abstract:

Computer systems supporting multiprogramming or multitasking execute multiple programs or tasks concurrently. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has something to execute. To have several jobs ready to run, the system must keep all of them in memory at the same time for their selection one-by-one. This work involves development of a simulator for CPU scheduling. It has been developed as a comprehensive tool, which runs a simulation in real time and generates useful data to be used for evaluation. This simulator can be used for measuring performance of different scheduling algorithms. It simulates First Come First Serve (FCFS) scheduling, Shortest Job First (SJF) scheduling, Shortest Remaining Time First(SRTF) scheduling, priority scheduling with preemption (PRI) , priority scheduling without preemption (PRI) , and Round Robin (RR) scheduling. A user-friendly and mouse- driven graphical user interface has been integrated.

Table of Contents

❖ Theory:

CPU scheduling is an operation which allows one process to use the CPU while the execution of another process is on hold due to unavailability of any resource like I/O etc, thereby making full use of CPU. The aim of CPU scheduling is to make the system efficient, fast and fair. There are many different CPU-scheduling algorithms:

1. **FCFS:** It is a non-preemptive, and the simplest CPU-scheduling algorithm. With this scheme, the process that requests the CPU first is allocated the CPU first. The implementation of the FCFS policy is easily managed with a FIFO queue.

2. **SJF**: It is non-preemptive, ready queue treated as a priority queue based on smallest CPU time requirement, arriving jobs inserted at proper position in queue, dispatcher selects shortest job (1st in queue) and runs to completion.

3. **RR**: It is similar to FCFS scheduling, but preemption is added to enable the system to switch between processes. A small unit of time, called a time quantum or time slice, is defined. The ready queue is treated as a circular queue. The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum.

4. **SRTF**: A preemptive version of SJF algorithm, which preempts the currently executing process when a less remaining time process becomes available.

5. **Priority Scheduling**: A priority is associated with each process, and the CPU is allocated to the process with the highest priority.

    a) **Priority scheduling with preemption:** the task with higher priority than the current task being executed arrives then the control of the CPU is taken from the current task and given to the higher priority task.

    b) **Priority scheduling without preemption:** the task with higher priority does arrive, it has to wait for the current task to release the CPU before it can be executed.

There are many different criteria's to check when considering the "best" scheduling algorithm, they are:

- CPU Utilization: To make out the best use of CPU and not to waste any CPU cycle, CPU would be working most of the time(Ideally 100% of the time). Considering a real system, CPU usage should range from 40% (lightly loaded) to 90% (heavily loaded.).

- Turnaround Time: It is the amount of time taken to execute a particular process,
i.e. The interval from time of submission of the process to the time of completion of the process(Wall clock time).

- Waiting Time: The sum of the periods spent waiting in the ready queue amount of time a process has been waiting in the ready queue to acquire get control on the CPU.

❖ Design Philosophy and Program Implementation:

When you start the program a window looks like figure 1 opens.

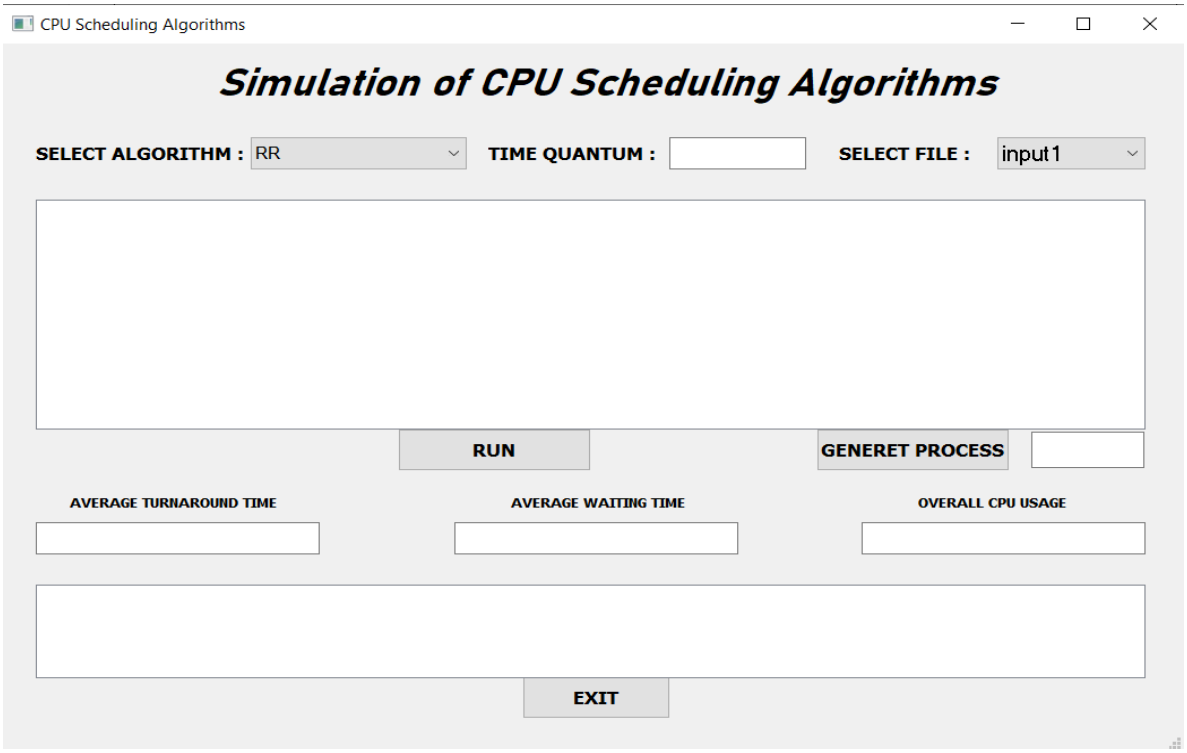


Figure 1

The window gives the user the option to choose the scheduling algorithm using a combo box (the RR algorithm is default), and to choose from 3 input files also in a combo box (input1.txt is default), after the user chooses the algorithm and the input file he wants, he have to click in the RUN button to start the simulation.

The output of the simulation is printed in two tables, the upper one has the values of each process data, while the bottom one prints which process runs at every time unit in the CPU (X stands for none).

In case of user choosing the RR algorithm, he must enter the Time Quantum he wants to use and it must be an integer, else an error massage looks like Figure2 will pop up.

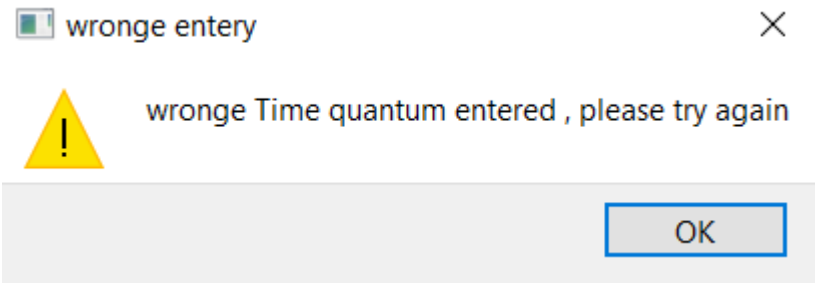


Figure 2

The program also gives the user the ability to simulate random generated processes by just entering the count of them, and then click on the Generate button.

The constrains on the random generated processes are:

- The names for the process starts from 1.
- The arrival time for each process is an integer between 1 and 100.
- The burst time for each process is an integer between 1 and 1000.
- The priority for each process is an integer between 1 and 100.
- The repeat is always 1.
- The interval and the deadline are not used so they are assigned to X.

If the user entered a wrong number format in the generate box a massage will pop up like the one in figure 3, and then the program will close.

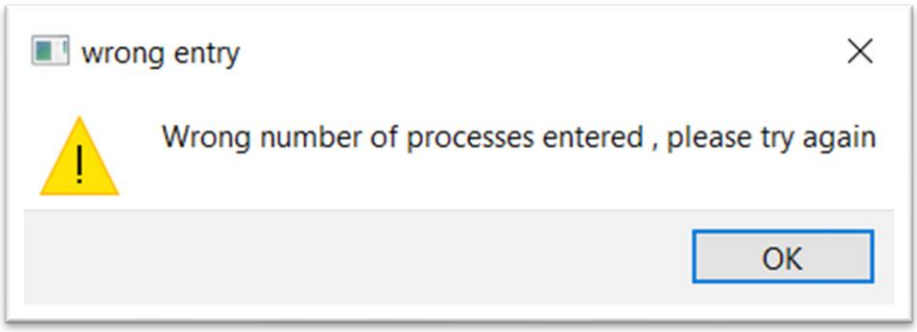


Figure 3

❖How the program work:

We created a distinct code file for every schedule algorithm that can be called from the main file, which organize the program, and makes sure that it works correct and errors free.

When the user clicks on the RUN button the program will get the value of algorithm combo box and the input file too, then according to the value of the algorithm it will call the wanted file of code to run in the screen.

In non-preemptive algorithms like FCFS, SJF and PRI without preemption, at first the program reads the input file and store it two lists, one to keep the original data and other is to modified as the simulation goes, then the program enters a loop which presents the CPU that breaks only when all the processes are finished executing when the loop starts, first, it checks the available processes at the time it reached using a special function that compares the time using the arrival time for each process, the available processes enters the queue, then the program calls another function to choose the process to be excused according the algorithm chosen, once the CPU gets the process, it keeps running until it finished all its burst, then the current time increases by the process burst and start over the loop.

In preemptive algorithms like SRTF and PRI with preemption, at first the program reads the input file and store it two lists, one to keep the original data and other is to modified as the simulation goes, then the program enters a loop which presents the CPU that breaks only when all the processes are finished executing when the loop starts, first, it checks the available processes at the time it reached using a special function that compares the time using the arrival time for each process, the available processes enters the queue, then the program calls another function to choose the process to be excused according the algorithm chosen, once the CPU gets the process, it runs for only one time unit, then recheck if it's the turn of another process to run or not, if there is another process at that time with higher priority then it enters the CPU instead of the running one, else it keeps running.

Aging:

The priority for the waiting processes is increases every some time that is random value for every time the program runs, we used this method to solve the starvation in PRI algorithms.

❖ Assumptions:

- **Priority:** the higher number presents the higher priority.
- **Aging:** the aging time is random.
- **Priority value:** the process priority value is an input from the file.

❖Extra work:

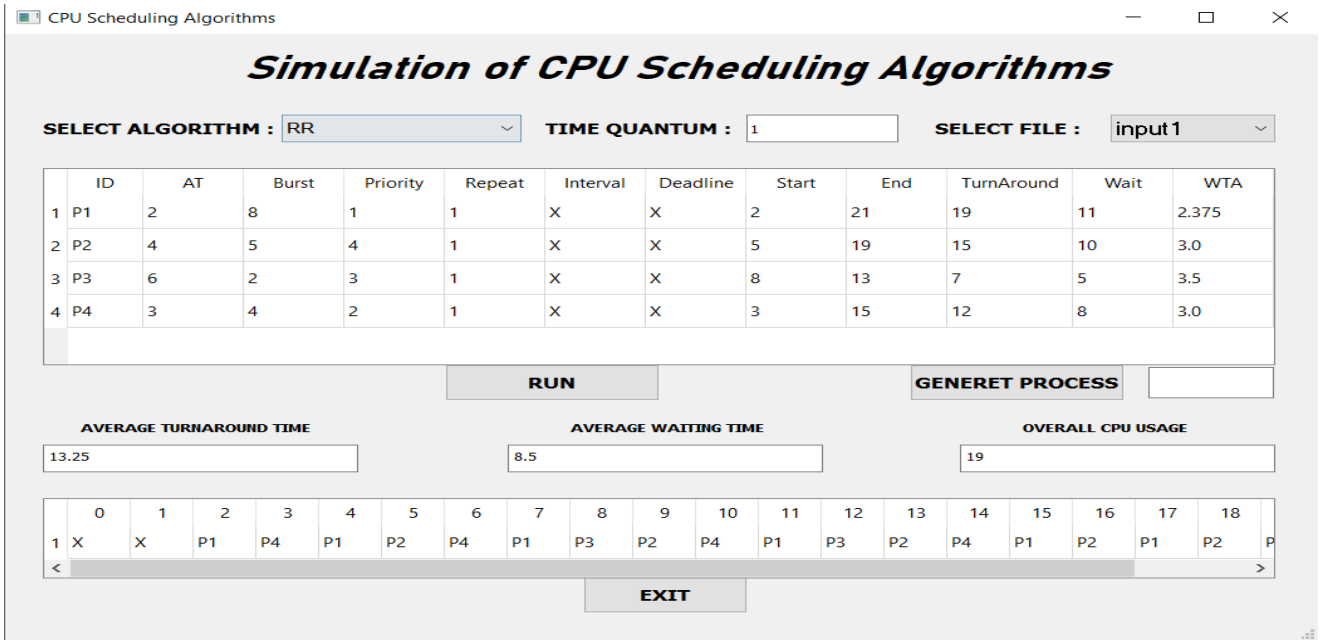
- Friendly user interface.
- Extra algorithms (6 instead of 4).

❖ Results

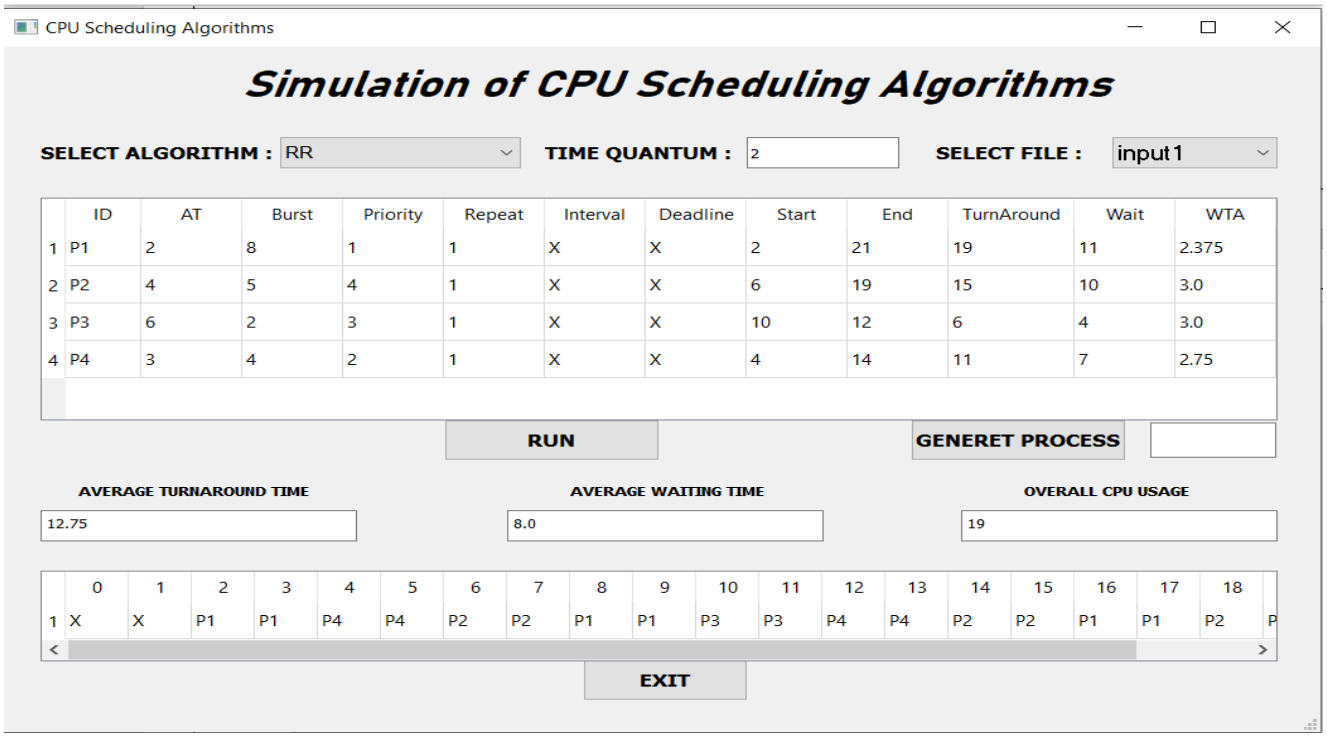
We created 3 different input files each file contains different processes were generated for each algorithm.

- **Input file 1**:

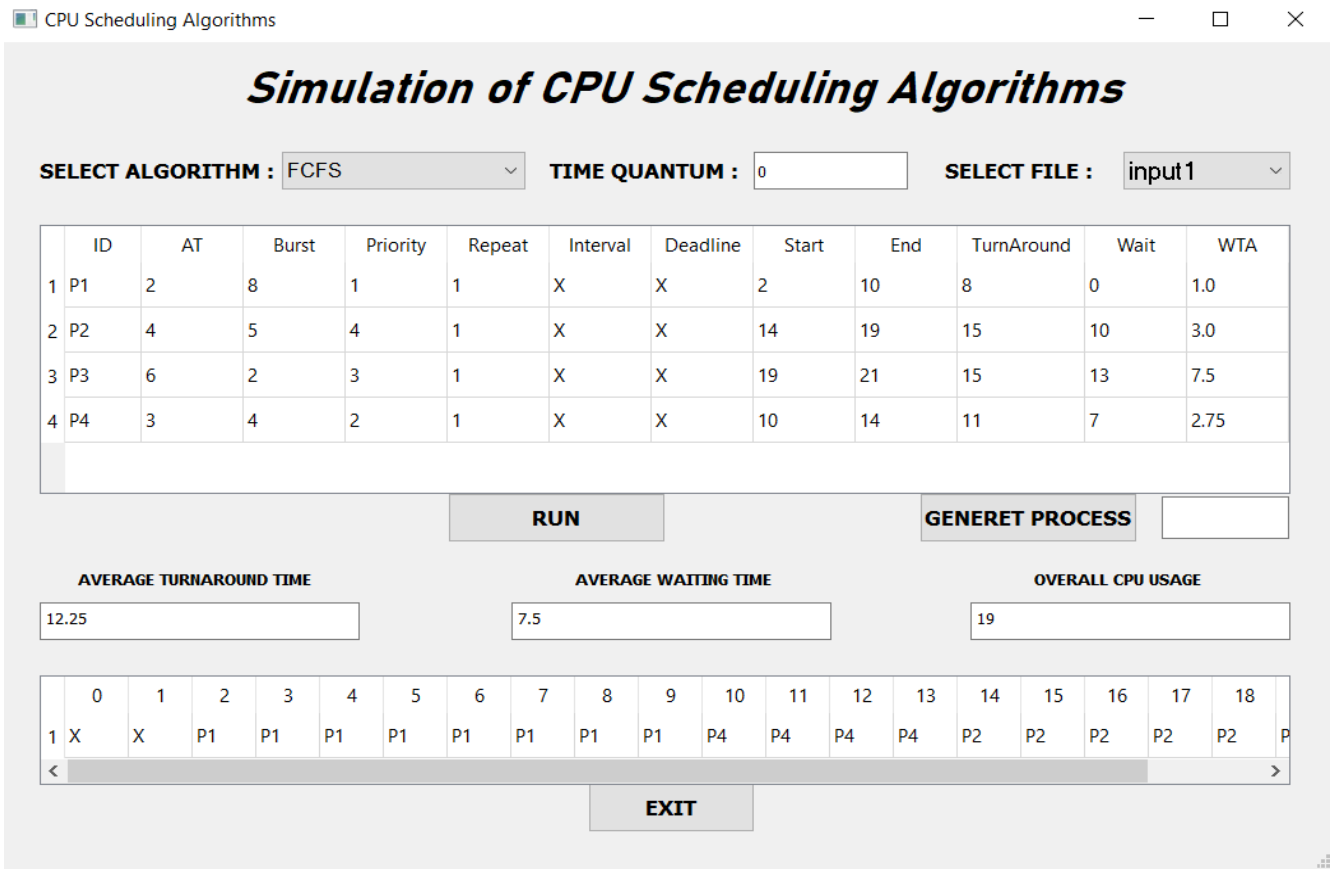
1) RR (with time quantum=1):



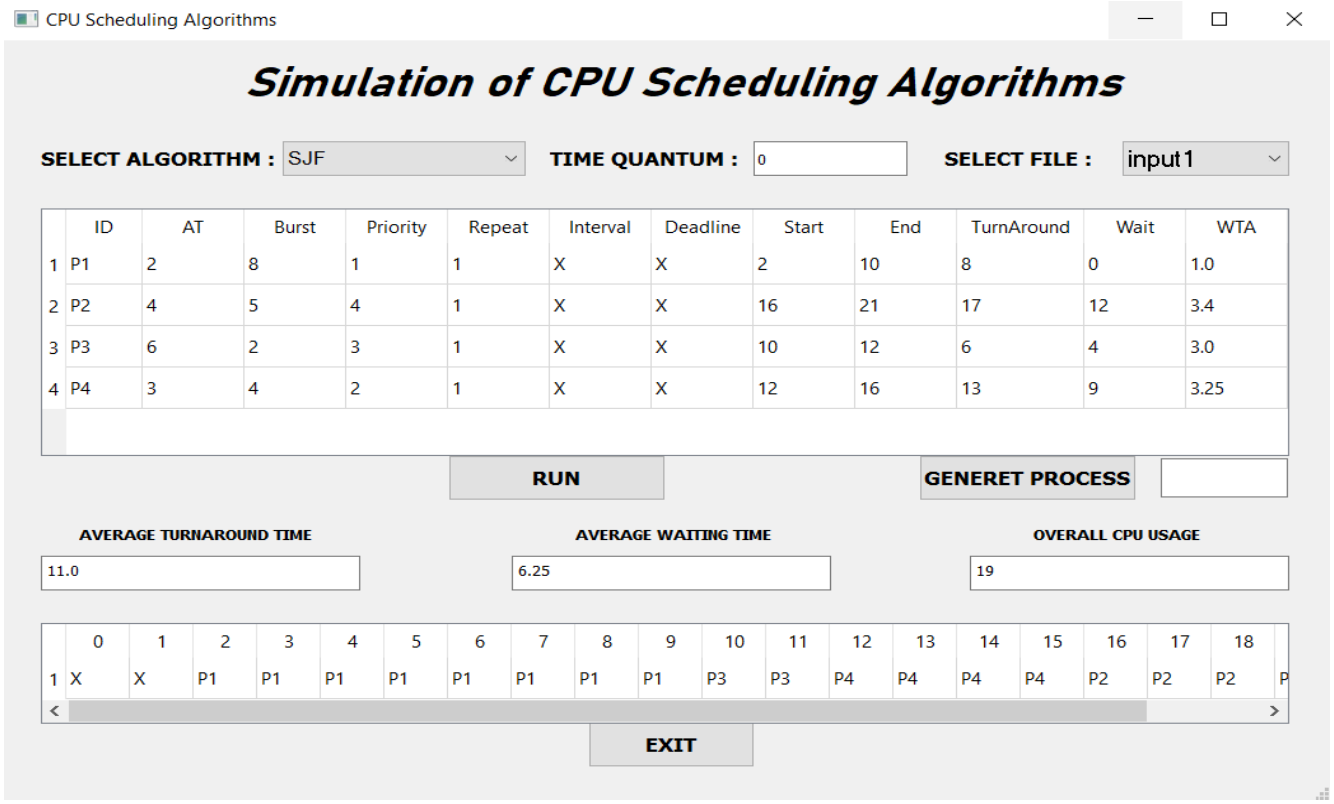
2) RR (with time quantum=2):



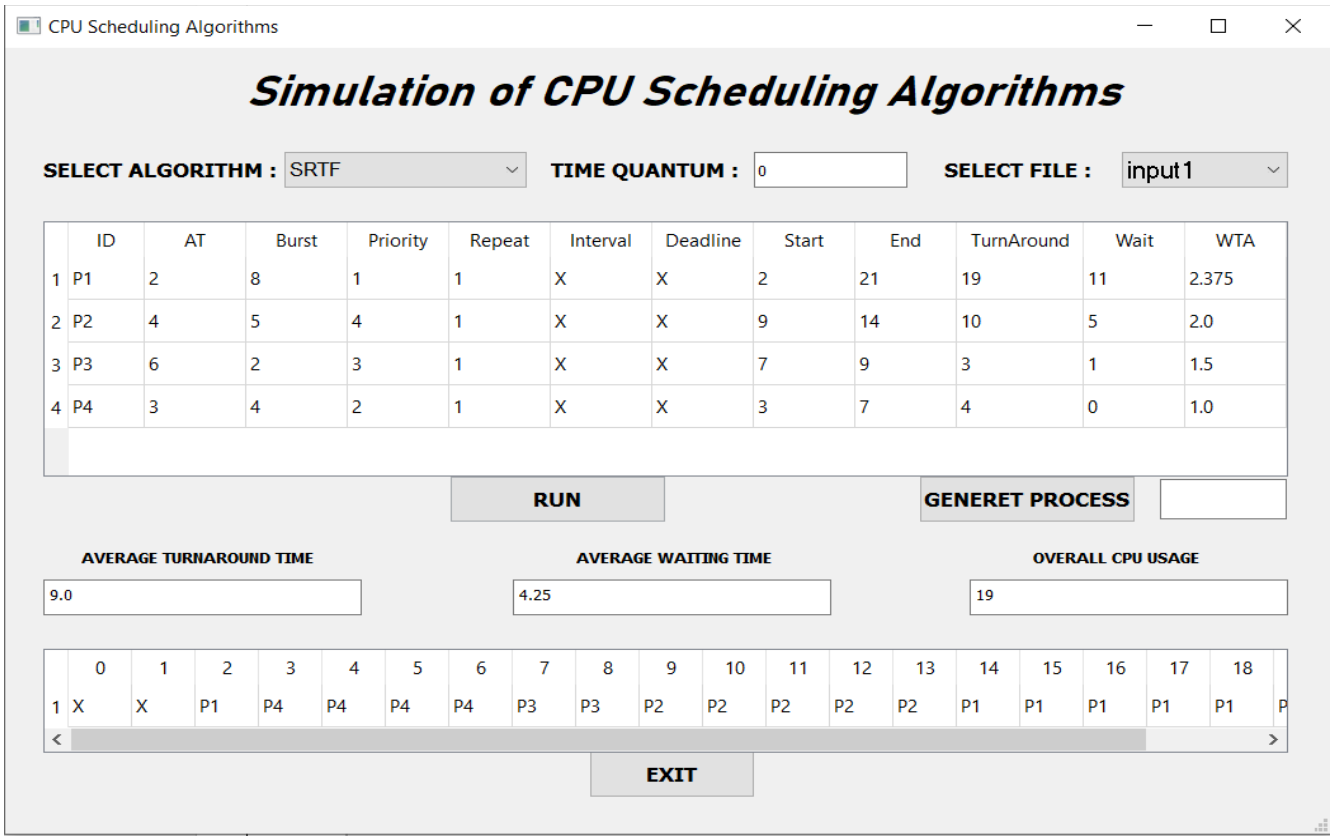
3) FCFS:


CPU Scheduling Algorithms
Simulation of CPU Scheduling Algorithms
SELECT ALGORITHM : FCFS
TIME QUANTUM : 0
SELECT FILE : input1
ID AT Burst Priority Repeat Interval Deadline Start End TurnAround Wait WTA
1 P1 2 8 1 1 X X 2 10 8 0 1.0
2 P2 4 5 4 1 X X 14 19 15 10 3.0
3 P3 6 2 3 1 X X 19 21 15 13 7.5
4 P4 3 4 2 1 X X 10 14 11 7 2.75
RUN
GENERET PROCESS
AVERAGE TURNAROUND TIME
12.25
AVERAGE WAITING TIME
7.5
OVERALL CPU USAGE
19
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 X X P1 P1 P1 P1 P1 P1 P1 P1 P4 P4 P4 P4 P2 P2 P2 P2 P2
EXIT

4) SJF:


CPU Scheduling Algorithms
Simulation of CPU Scheduling Algorithms
SELECT ALGORITHM : SJF
TIME QUANTUM : 0
SELECT FILE : input1
ID AT Burst Priority Repeat Interval Deadline Start End TurnAround Wait WTA
1 P1 2 8 1 1 X X 2 10 8 0 1.0
2 P2 4 5 4 1 X X 16 21 17 12 3.4
3 P3 6 2 3 1 X X 10 12 6 4 3.0
4 P4 3 4 2 1 X X 12 16 13 9 3.25
RUN
GENERET PROCESS
AVERAGE TURNAROUND TIME
11.0
AVERAGE WAITING TIME
6.25
OVERALL CPU USAGE
19
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 X X P1 P1 P1 P1 P1 P1 P1 P1 P3 P3 P4 P4 P4 P4 P2 P2 P2
EXIT

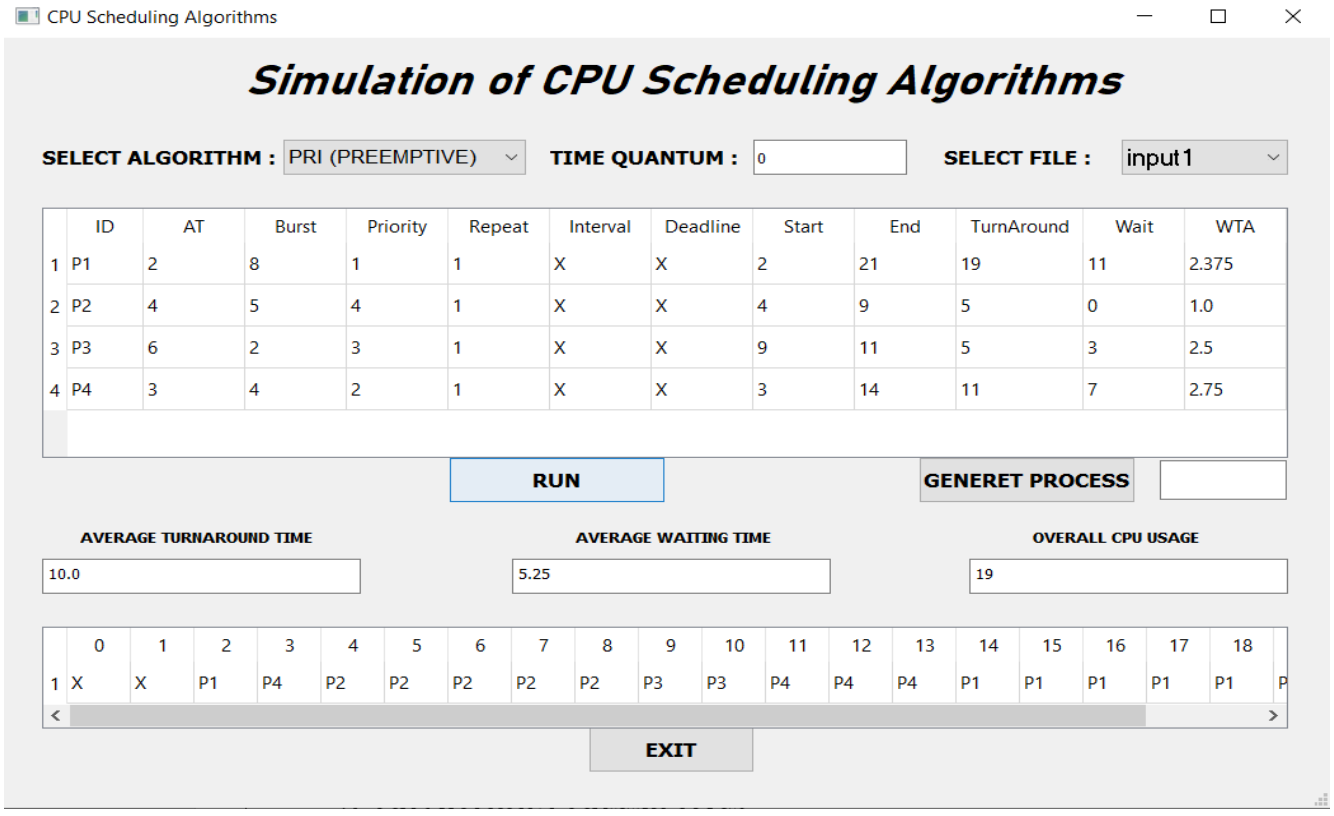
5) SRTF :



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	2	8	1	1	X	X	2	21	19	11	2.375
2	P2	4	5	4	1	X	X	9	14	10	5	2.0
3	P3	6	2	3	1	X	X	7	9	3	1	1.5
4	P4	3	4	2	1	X	X	3	7	4	0	1.0

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	X	X	P1	P4	P4	P4	P4	P3	P3	P2	P2	P2	P2	P2	P1	P1	P1	P1	P1

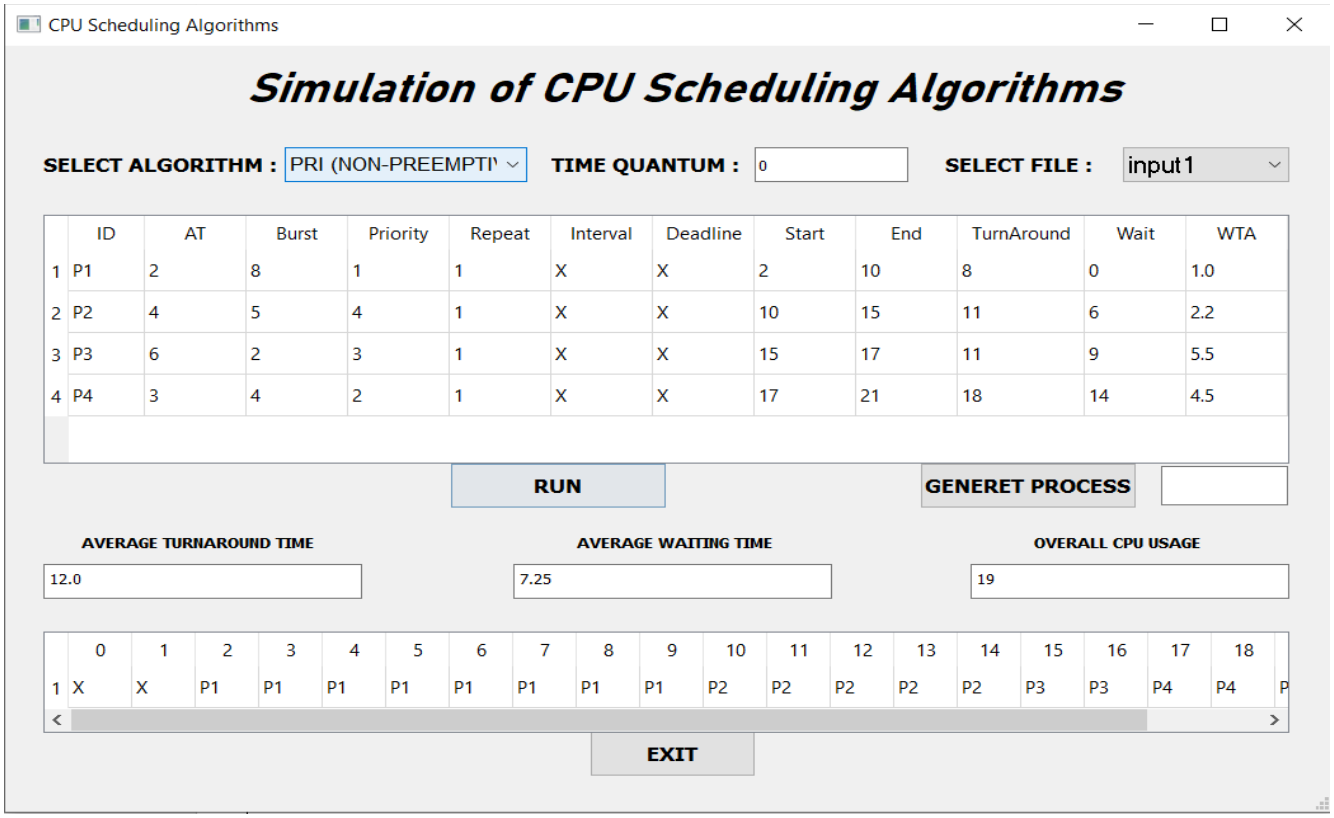
6) PRI (PREEMPTIVE) :



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	2	8	1	1	X	X	2	21	19	11	2.375
2	P2	4	5	4	1	X	X	4	9	5	0	1.0
3	P3	6	2	3	1	X	X	9	11	5	3	2.5
4	P4	3	4	2	1	X	X	3	14	11	7	2.75

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	X	X	P1	P4	P2	P2	P2	P2	P2	P3	P3	P4	P4	P4	P1	P1	P1	P1	P1

7) PRI (WITHOUT PREEMPTIVE) :

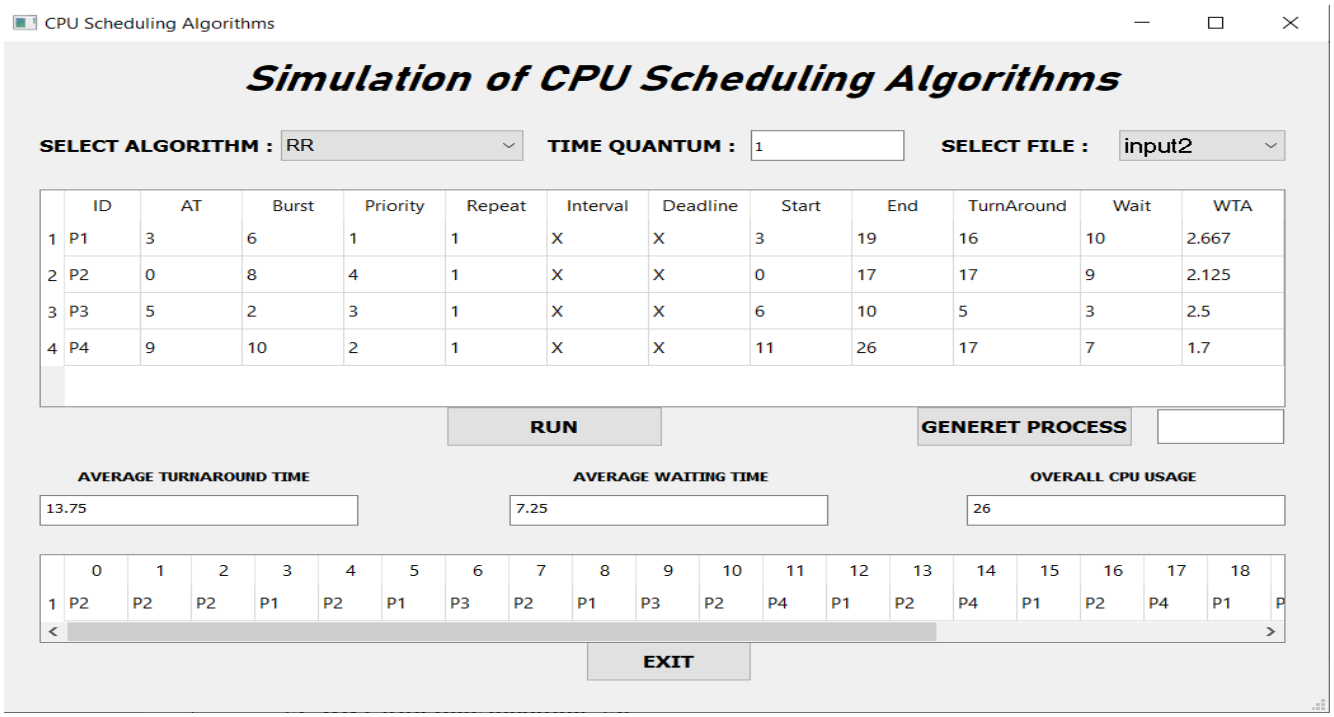


	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	2	8	1	1	X	X	2	10	8	0	1.0
2	P2	4	5	4	1	X	X	10	15	11	6	2.2
3	P3	6	2	3	1	X	X	15	17	11	9	5.5
4	P4	3	4	2	1	X	X	17	21	18	14	4.5

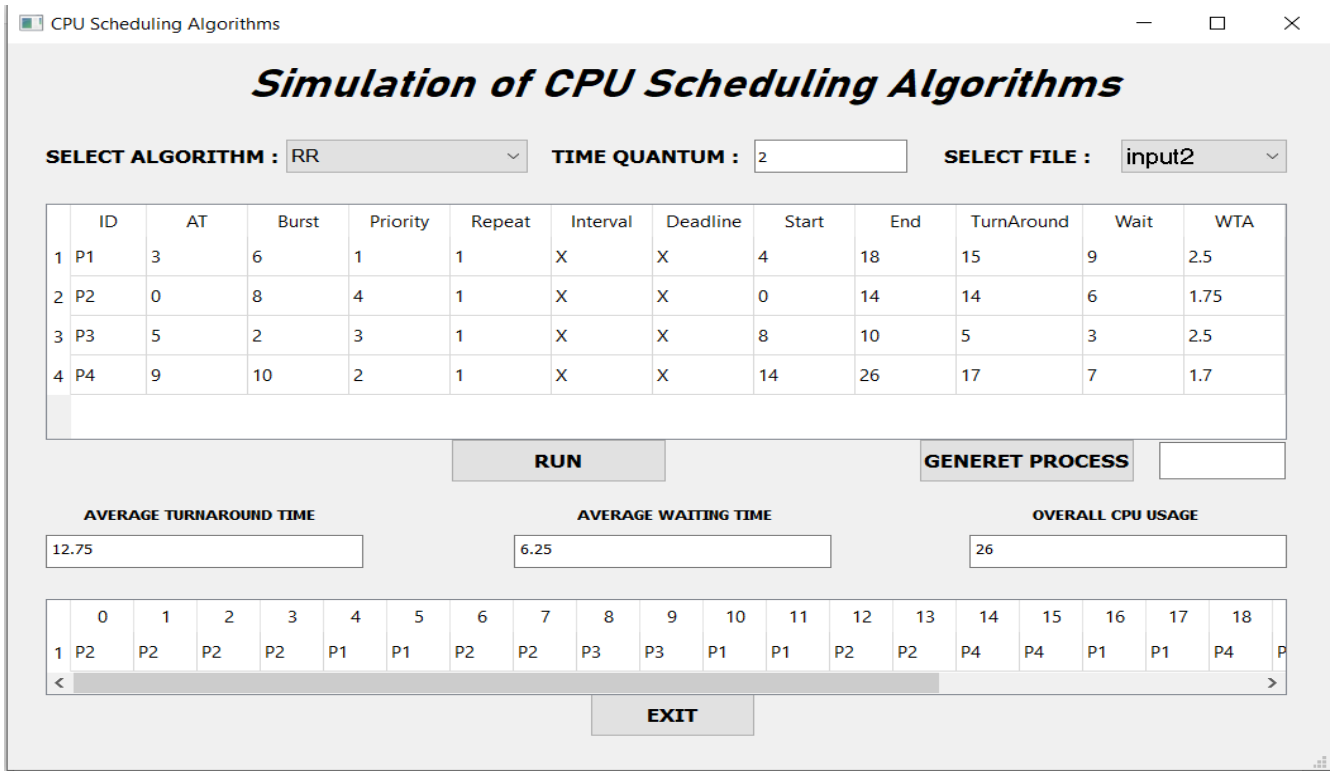
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	X	X	P1	P1	P1	P1	P1	P1	P1	P1	P2	P2	P2	P2	P2	P3	P3	P4	P4

- **Input file 2**:

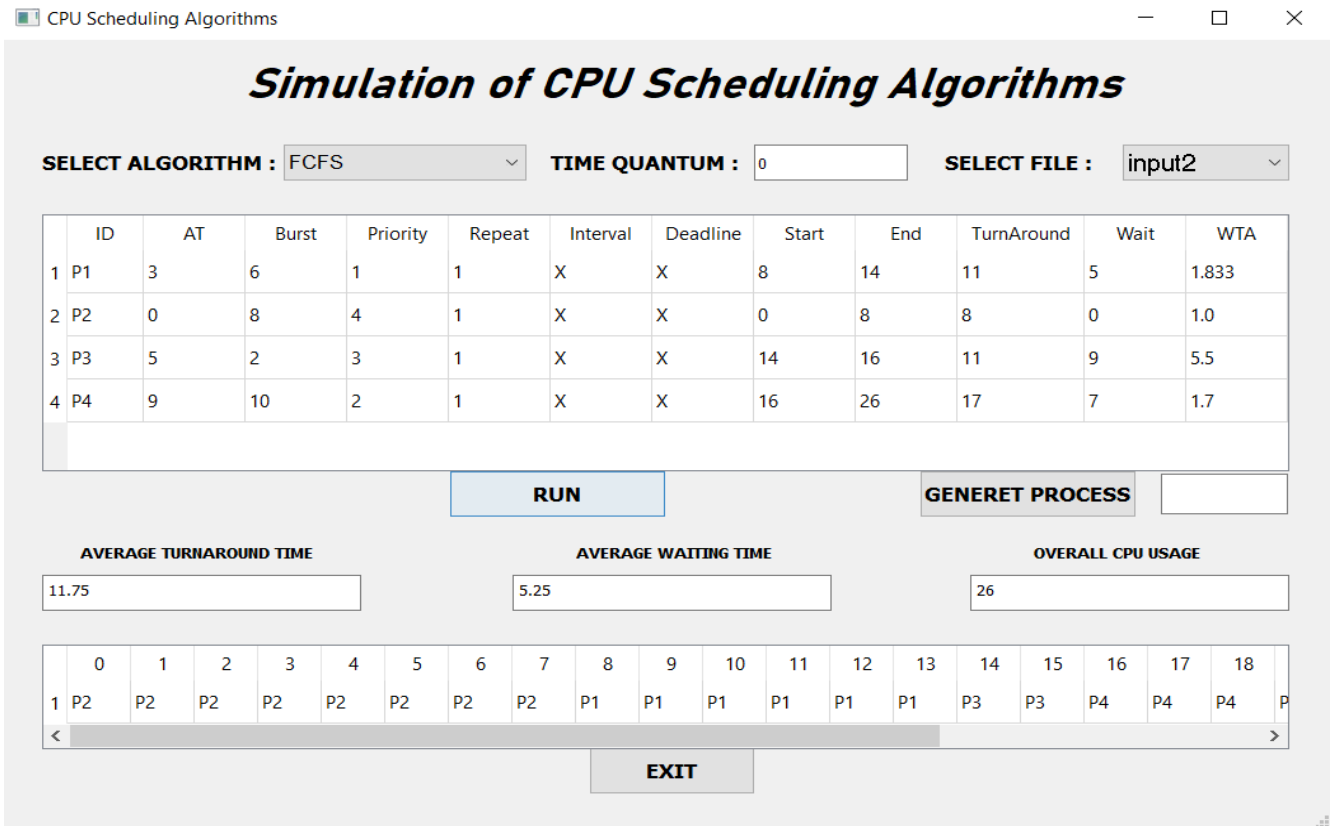
1) RR (with time quantum=1):



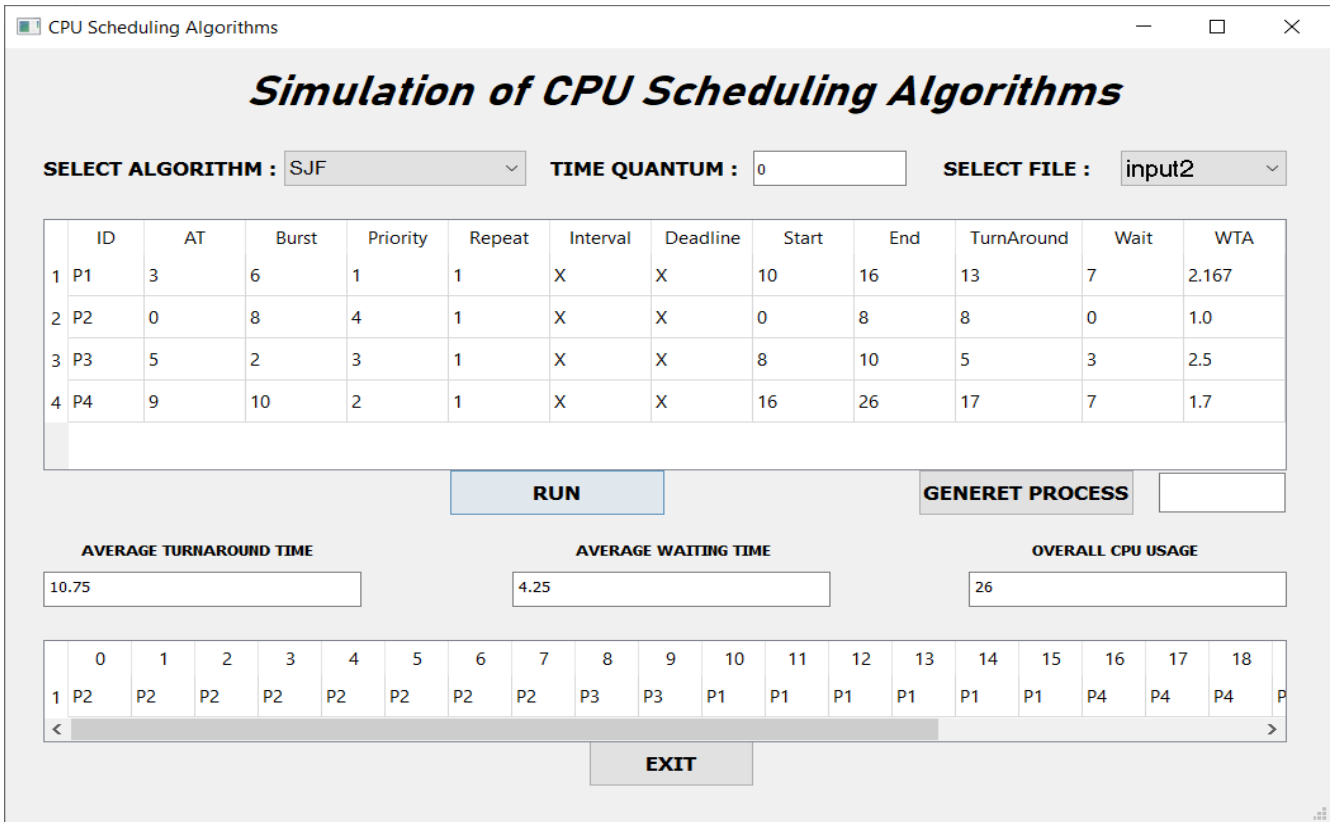
2) RR (with time quantum=2):



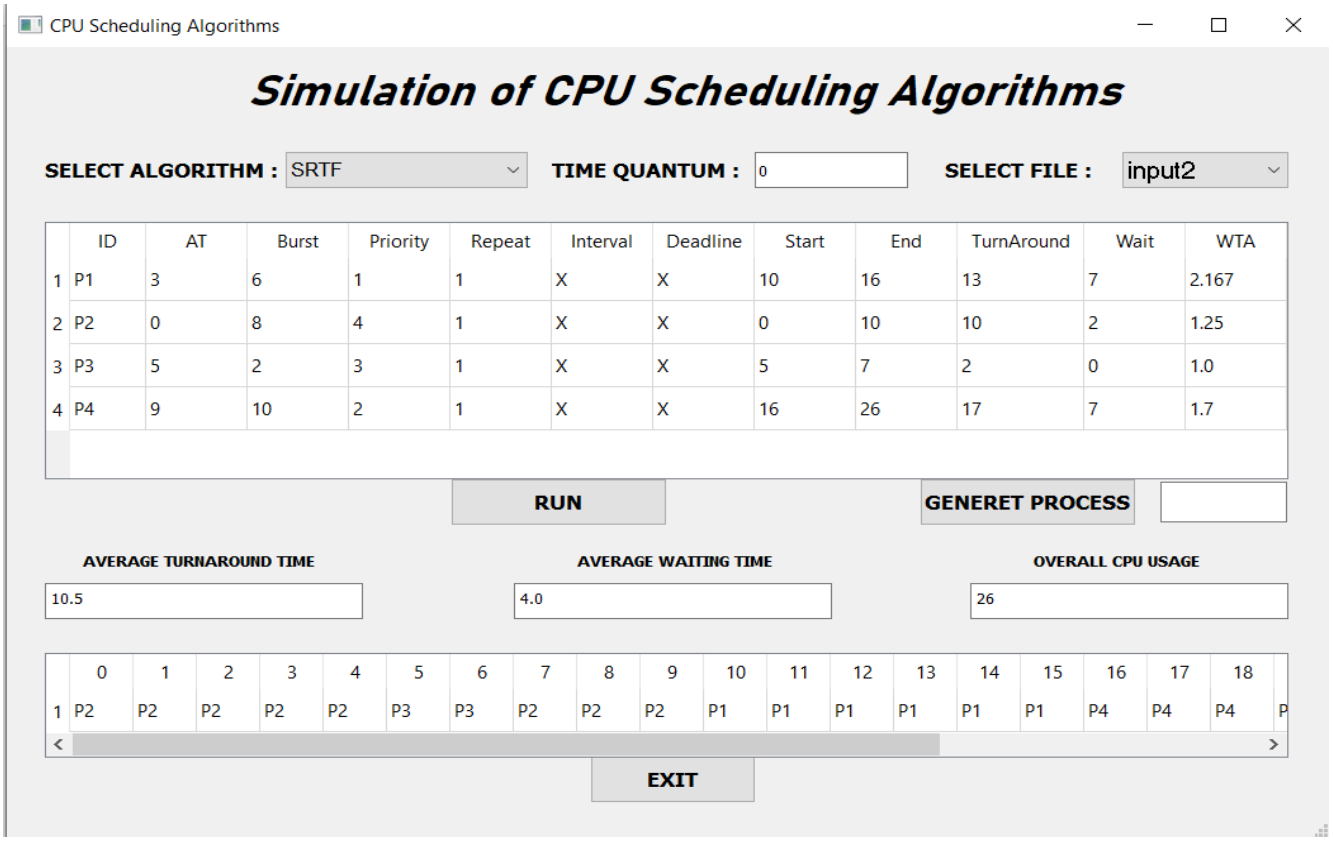
3) FCFS:


CPU Scheduling Algorithms
Simulation of CPU Scheduling Algorithms
SELECT ALGORITHM : FCFS
TIME QUANTUM : 0
SELECT FILE : input2
ID AT Burst Priority Repeat Interval Deadline Start End TurnAround Wait WTA
1 P1 3 6 1 1 X X 8 14 11 5 1.833
2 P2 0 8 4 1 X X 0 8 8 0 1.0
3 P3 5 2 3 1 X X 14 16 11 9 5.5
4 P4 9 10 2 1 X X 16 26 17 7 1.7
RUN
GENERET PROCESS
AVERAGE TURNAROUND TIME
AVERAGE WAITING TIME
OVERALL CPU USAGE
11.75
5.25
26
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 P2 P2 P2 P2 P2 P2 P2 P2 P1 P1 P1 P1 P1 P1 P3 P3 P4 P4 P4
EXIT

4) SJF:


CPU Scheduling Algorithms
Simulation of CPU Scheduling Algorithms
SELECT ALGORITHM : SJF
TIME QUANTUM : 0
SELECT FILE : input2
ID AT Burst Priority Repeat Interval Deadline Start End TurnAround Wait WTA
1 P1 3 6 1 1 X X 10 16 13 7 2.167
2 P2 0 8 4 1 X X 0 8 8 0 1.0
3 P3 5 2 3 1 X X 8 10 5 3 2.5
4 P4 9 10 2 1 X X 16 26 17 7 1.7
RUN
GENERET PROCESS
AVERAGE TURNAROUND TIME
AVERAGE WAITING TIME
OVERALL CPU USAGE
10.75
4.25
26
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 P2 P2 P2 P2 P2 P2 P2 P2 P3 P3 P1 P1 P1 P1 P1 P1 P4 P4 P4
EXIT

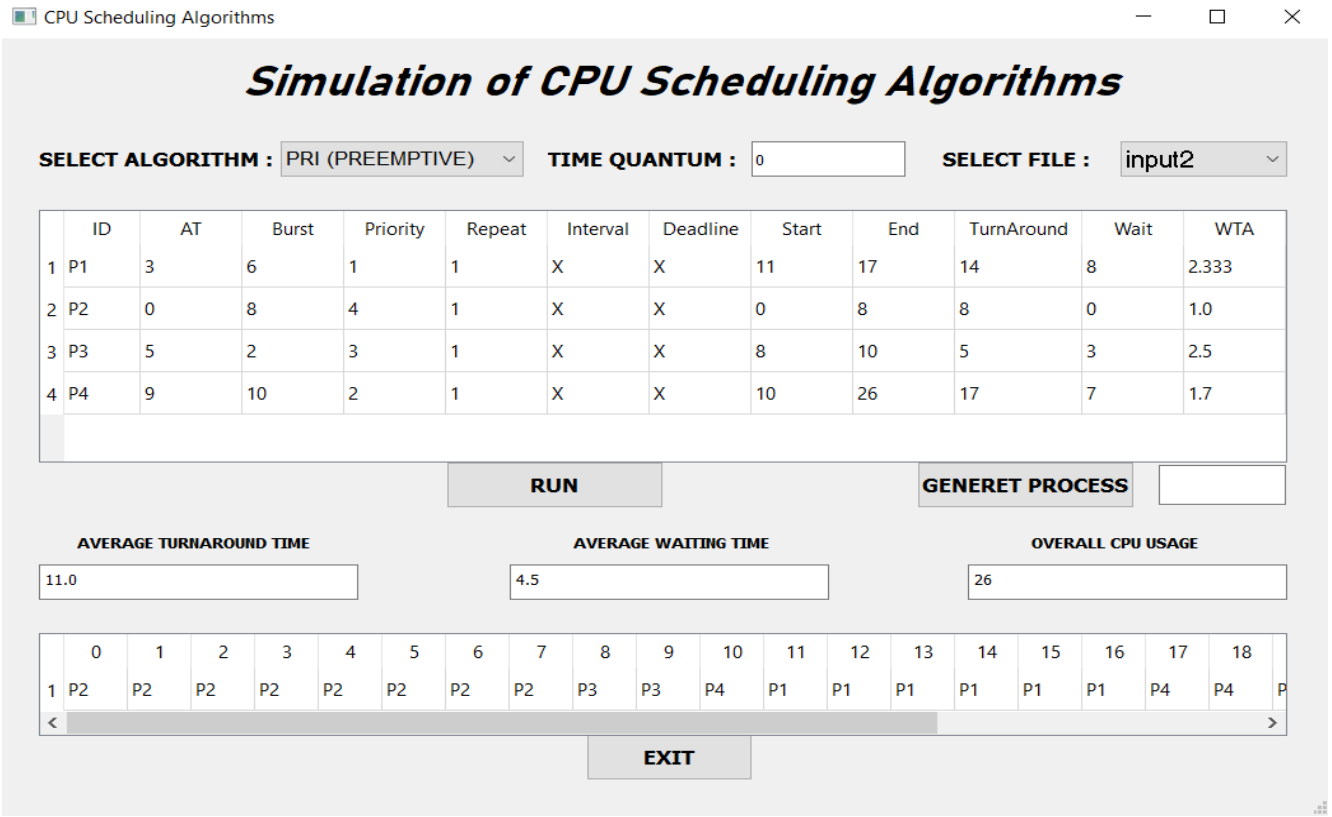
5) SRTF :



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	3	6	1	1	X	X	10	16	13	7	2.167
2	P2	0	8	4	1	X	X	0	10	10	2	1.25
3	P3	5	2	3	1	X	X	5	7	2	0	1.0
4	P4	9	10	2	1	X	X	16	26	17	7	1.7

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	P2	P2	P2	P2	P2	P3	P3	P2	P2	P2	P1	P1	P1	P1	P1	P1	P4	P4	P4	P

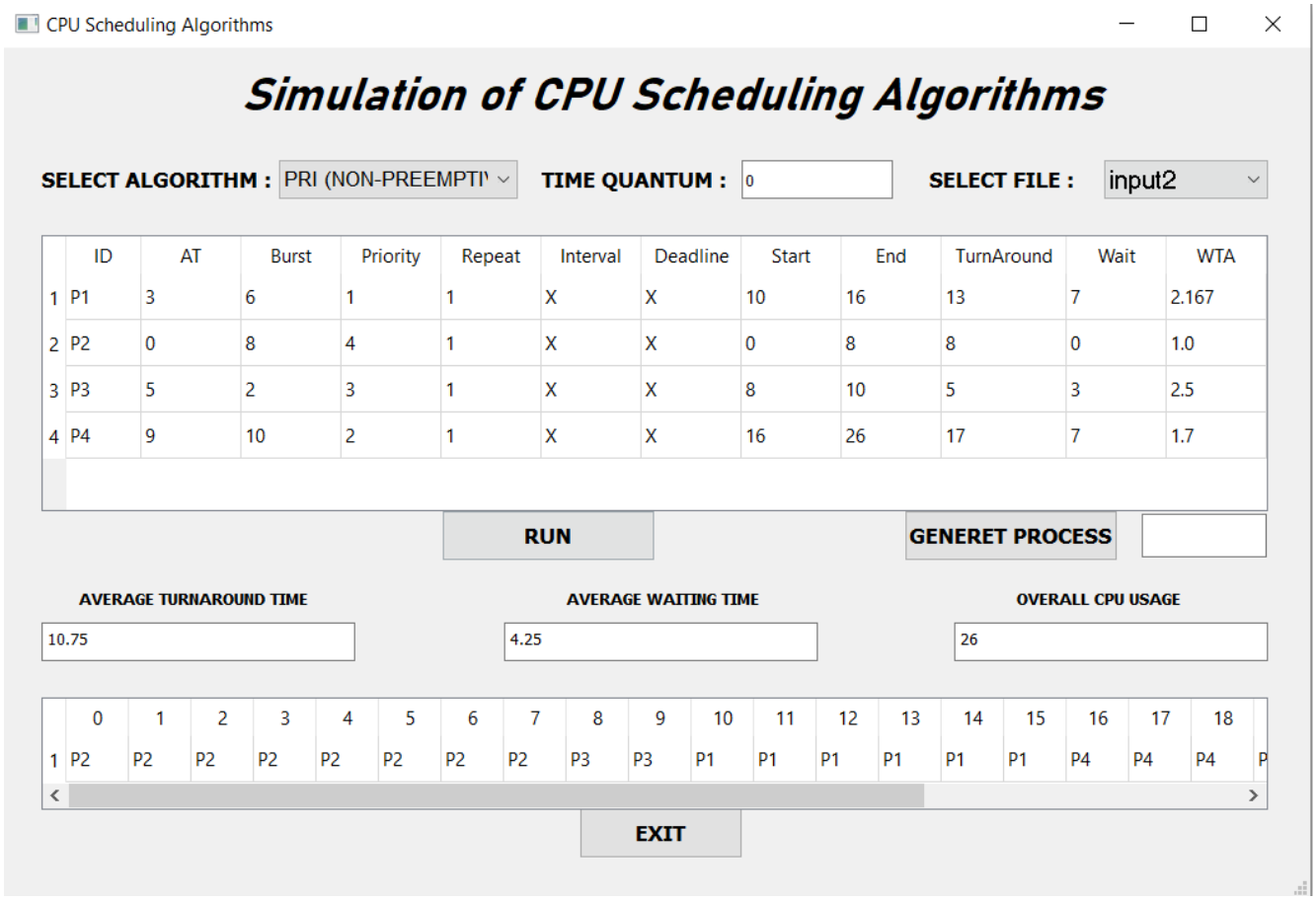
6) PRI (PREEMPTIVE) :



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	3	6	1	1	X	X	11	17	14	8	2.333
2	P2	0	8	4	1	X	X	0	8	8	0	1.0
3	P3	5	2	3	1	X	X	8	10	5	3	2.5
4	P4	9	10	2	1	X	X	10	26	17	7	1.7

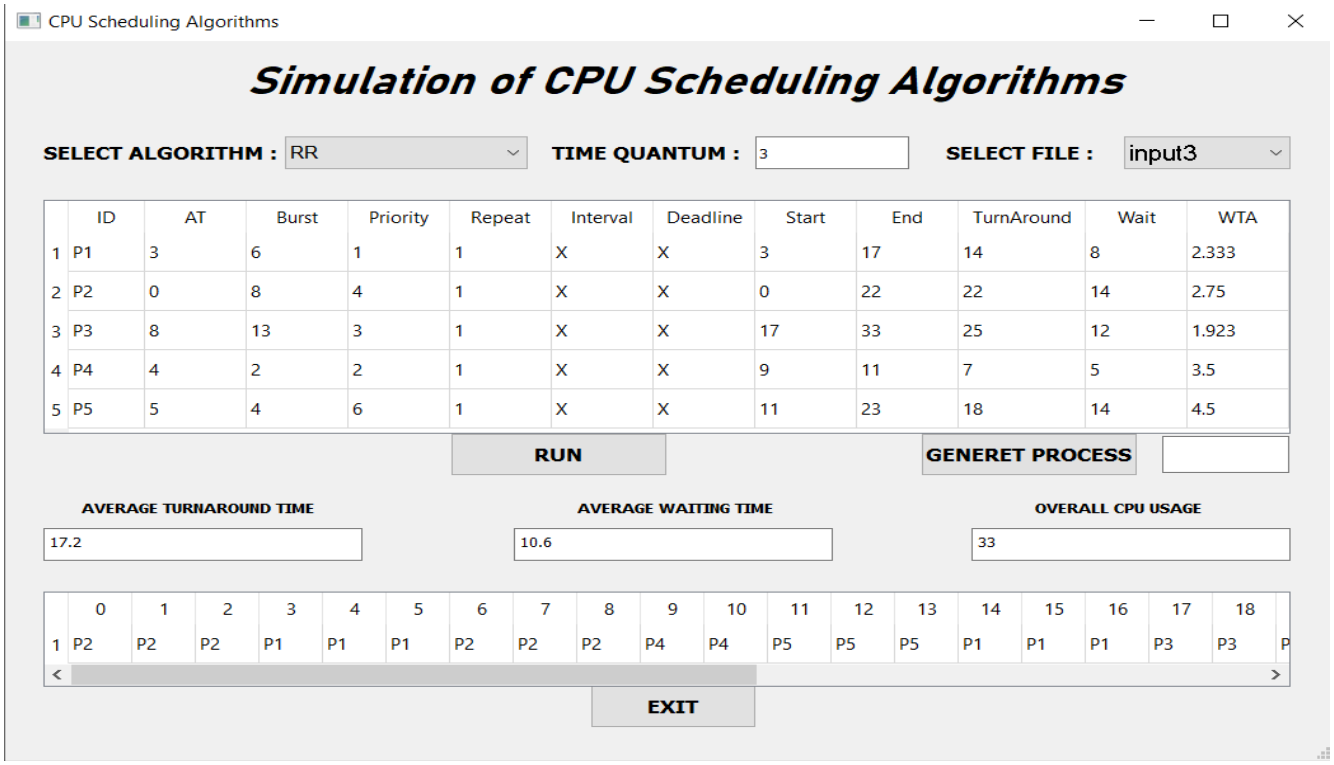
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	P2	P2	P2	P2	P2	P2	P2	P2	P3	P3	P4	P1	P1	P1	P1	P1	P1	P4	P4	P

7) PRI (WITHOUT PREEMPTIVE) :



- **Input file 3:**

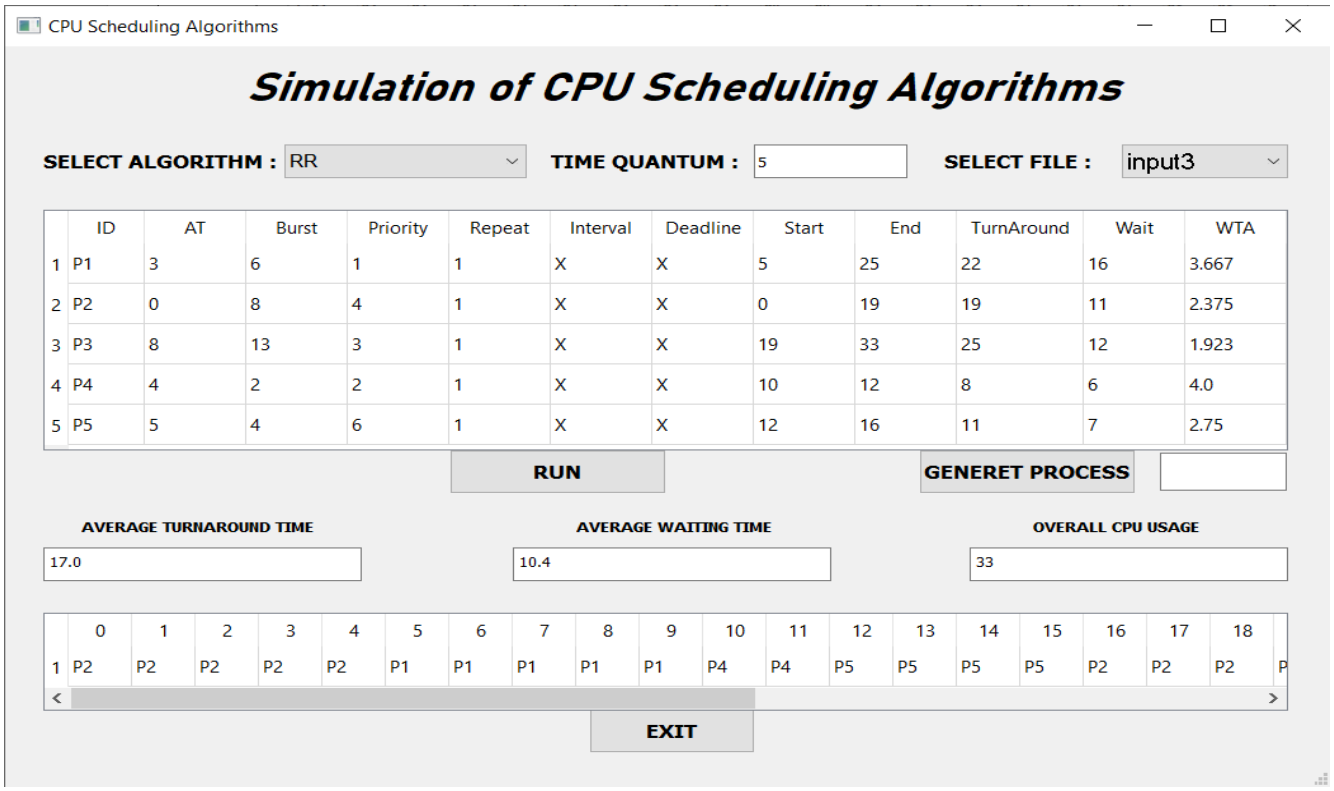
1) RR (with time quantum=3):



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	3	6	1	1	X	X	3	17	14	8	2.333
2	P2	0	8	4	1	X	X	0	22	22	14	2.75
3	P3	8	13	3	1	X	X	17	33	25	12	1.923
4	P4	4	2	2	1	X	X	9	11	7	5	3.5
5	P5	5	4	6	1	X	X	11	23	18	14	4.5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	P2	P2	P2	P1	P1	P1	P2	P2	P2	P4	P4	P5	P5	P5	P1	P1	P1	P3	P3

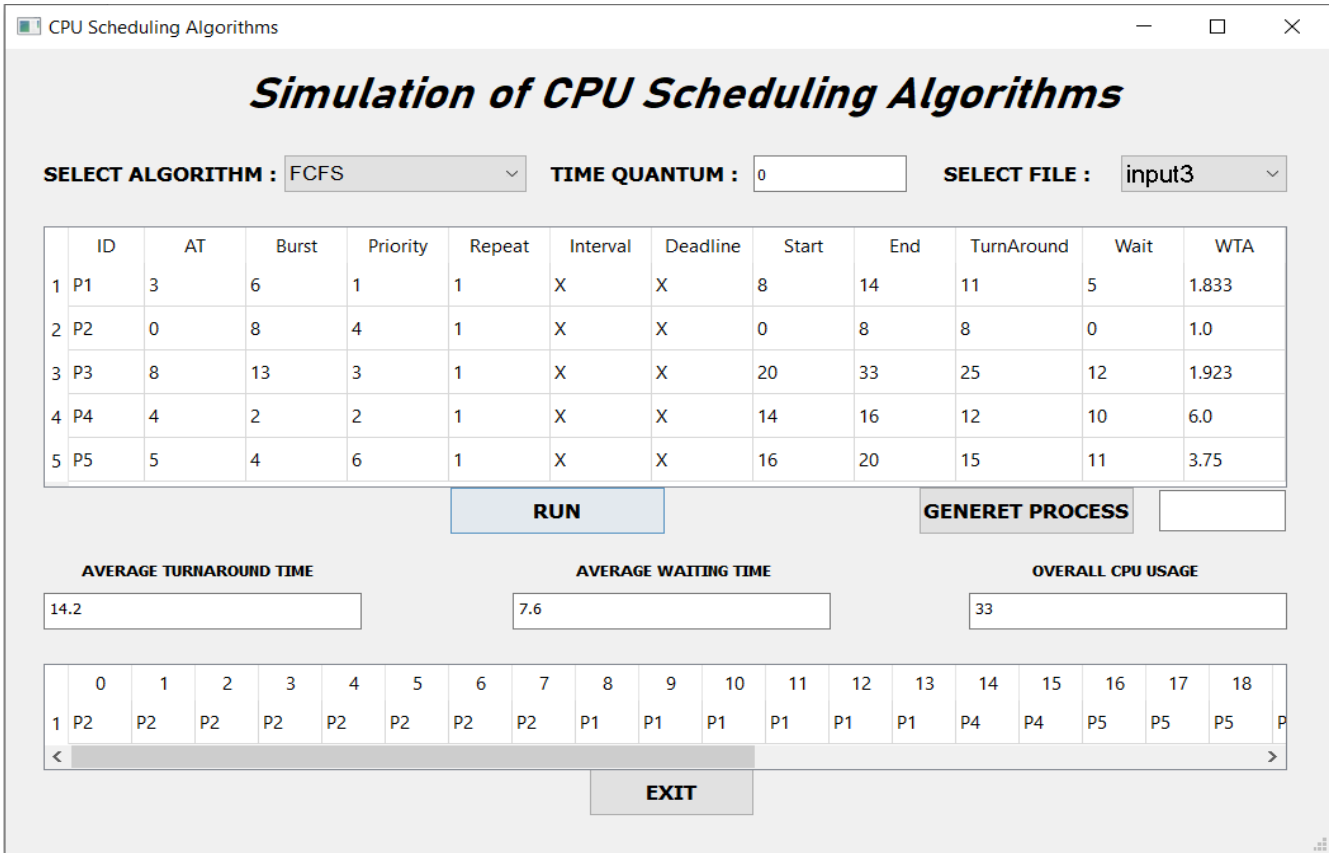
2) RR (with time quantum=5):



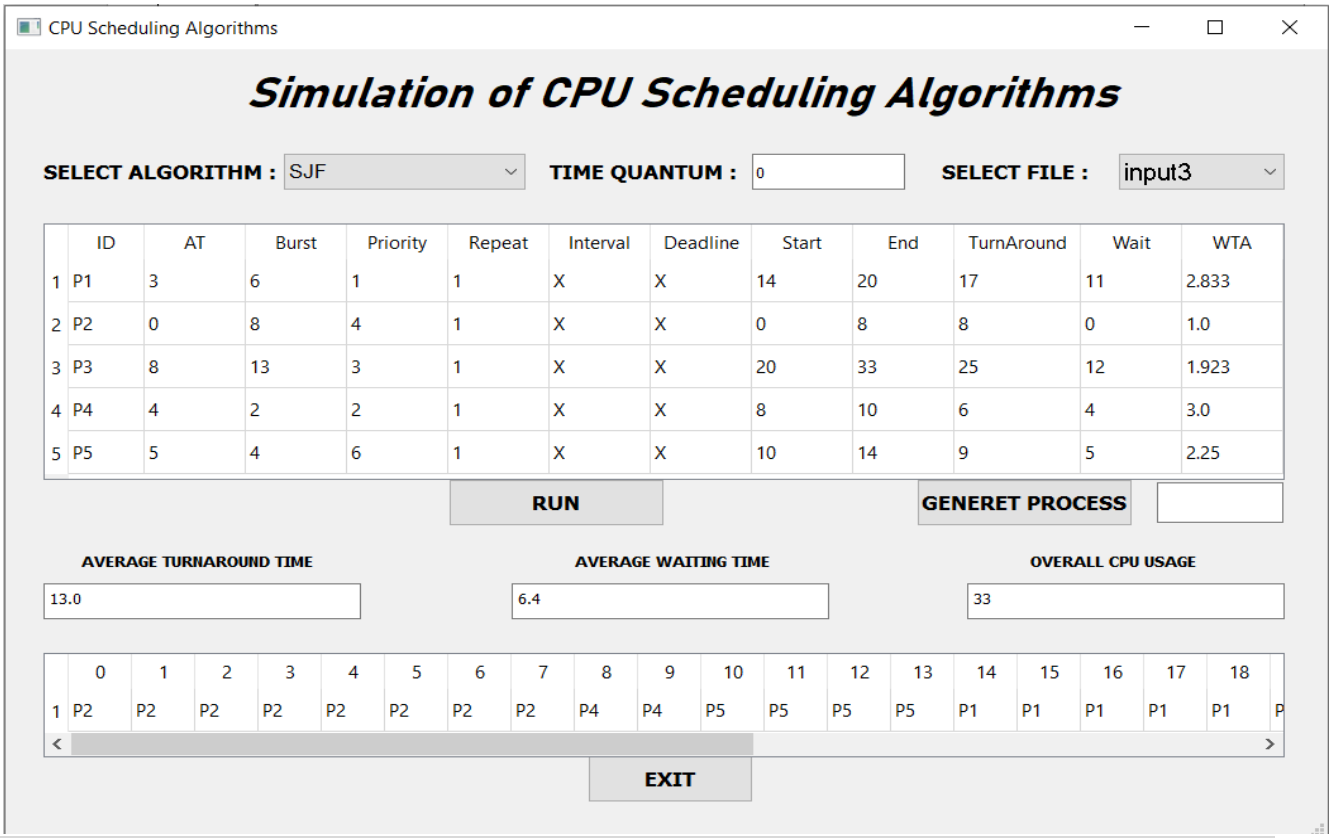
	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	3	6	1	1	X	X	5	25	22	16	3.667
2	P2	0	8	4	1	X	X	0	19	19	11	2.375
3	P3	8	13	3	1	X	X	19	33	25	12	1.923
4	P4	4	2	2	1	X	X	10	12	8	6	4.0
5	P5	5	4	6	1	X	X	12	16	11	7	2.75

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	P2	P2	P2	P2	P2	P1	P1	P1	P1	P1	P4	P4	P5	P5	P5	P5	P2	P2	P2

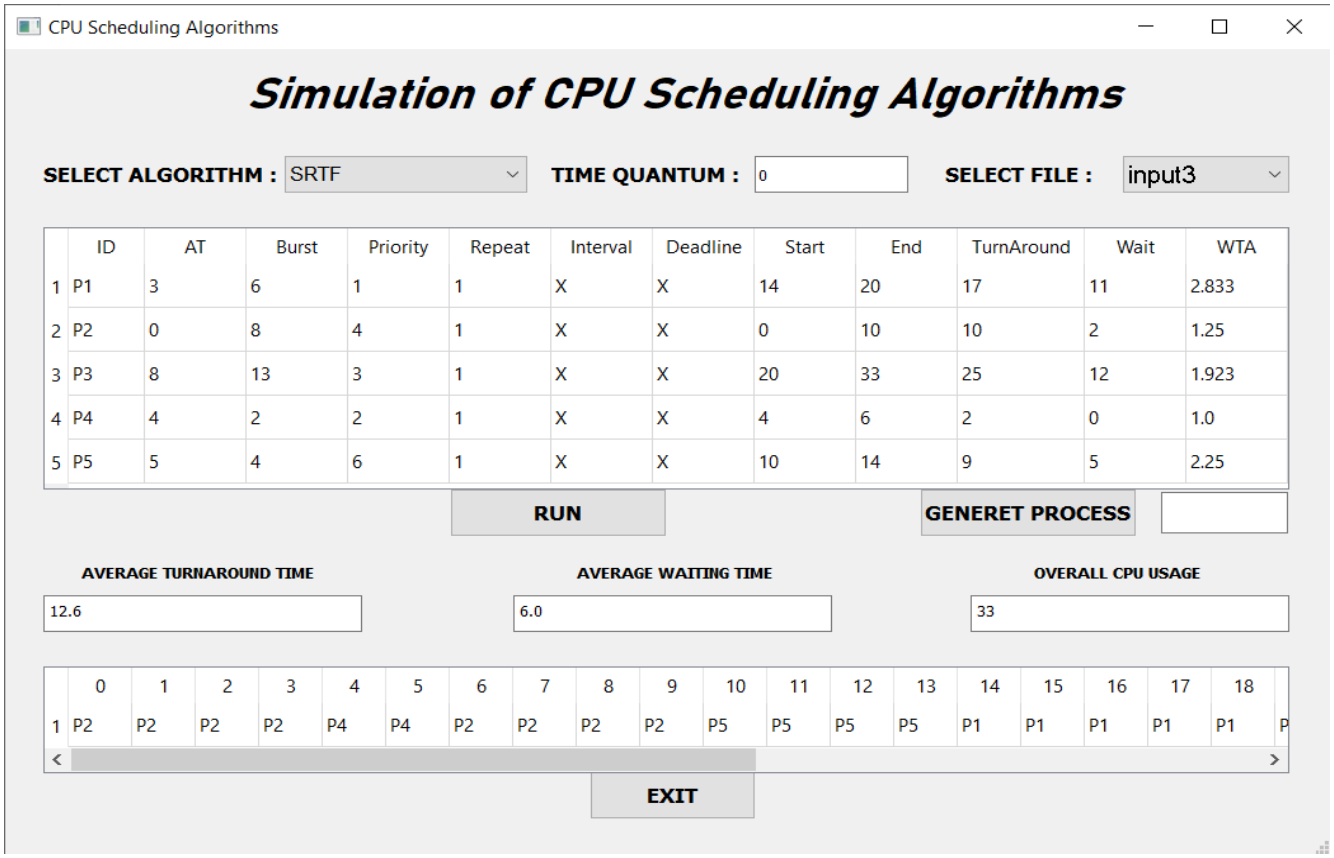
3) FCFS:


CPU Scheduling Algorithms
Simulation of CPU Scheduling Algorithms
SELECT ALGORITHM : FCFS
TIME QUANTUM : 0
SELECT FILE : input3
ID AT Burst Priority Repeat Interval Deadline Start End TurnAround Wait WTA
1 P1 3 6 1 1 X X 8 14 11 5 1.833
2 P2 0 8 4 1 X X 0 8 8 0 1.0
3 P3 8 13 3 1 X X 20 33 25 12 1.923
4 P4 4 2 2 1 X X 14 16 12 10 6.0
5 P5 5 4 6 1 X X 16 20 15 11 3.75
RUN
GENERET PROCESS
AVERAGE TURNAROUND TIME
AVERAGE WAITING TIME
OVERALL CPU USAGE
14.2
7.6
33
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 P2 P2 P2 P2 P2 P2 P2 P2 P1 P1 P1 P1 P1 P1 P4 P4 P5 P5 P5
EXIT

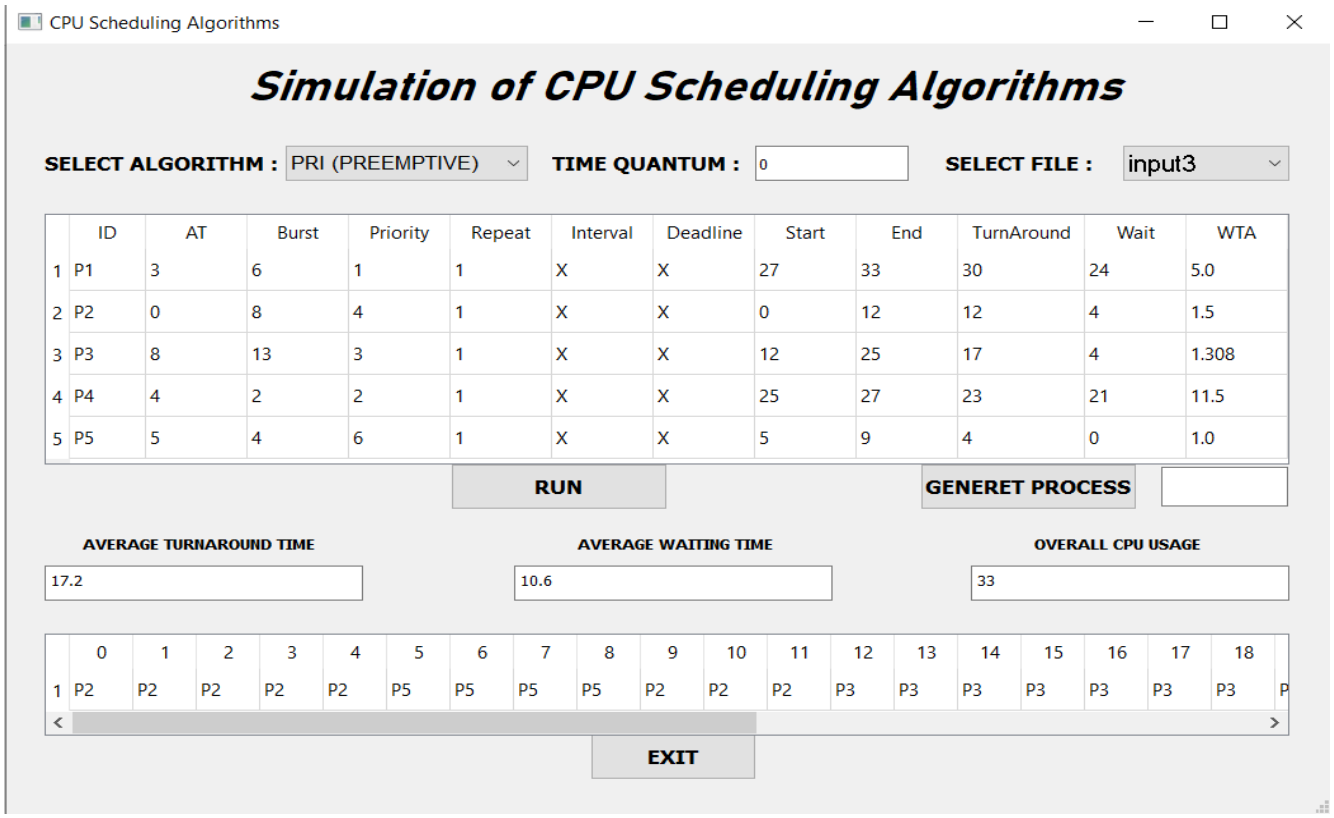
4) SJF:


CPU Scheduling Algorithms
Simulation of CPU Scheduling Algorithms
SELECT ALGORITHM : SJF
TIME QUANTUM : 0
SELECT FILE : input3
ID AT Burst Priority Repeat Interval Deadline Start End TurnAround Wait WTA
1 P1 3 6 1 1 X X 14 20 17 11 2.833
2 P2 0 8 4 1 X X 0 8 8 0 1.0
3 P3 8 13 3 1 X X 20 33 25 12 1.923
4 P4 4 2 2 1 X X 8 10 6 4 3.0
5 P5 5 4 6 1 X X 10 14 9 5 2.25
RUN
GENERET PROCESS
AVERAGE TURNAROUND TIME
AVERAGE WAITING TIME
OVERALL CPU USAGE
13.0
6.4
33
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 P2 P2 P2 P2 P2 P2 P2 P2 P4 P4 P5 P5 P5 P5 P1 P1 P1 P1 P1
EXIT

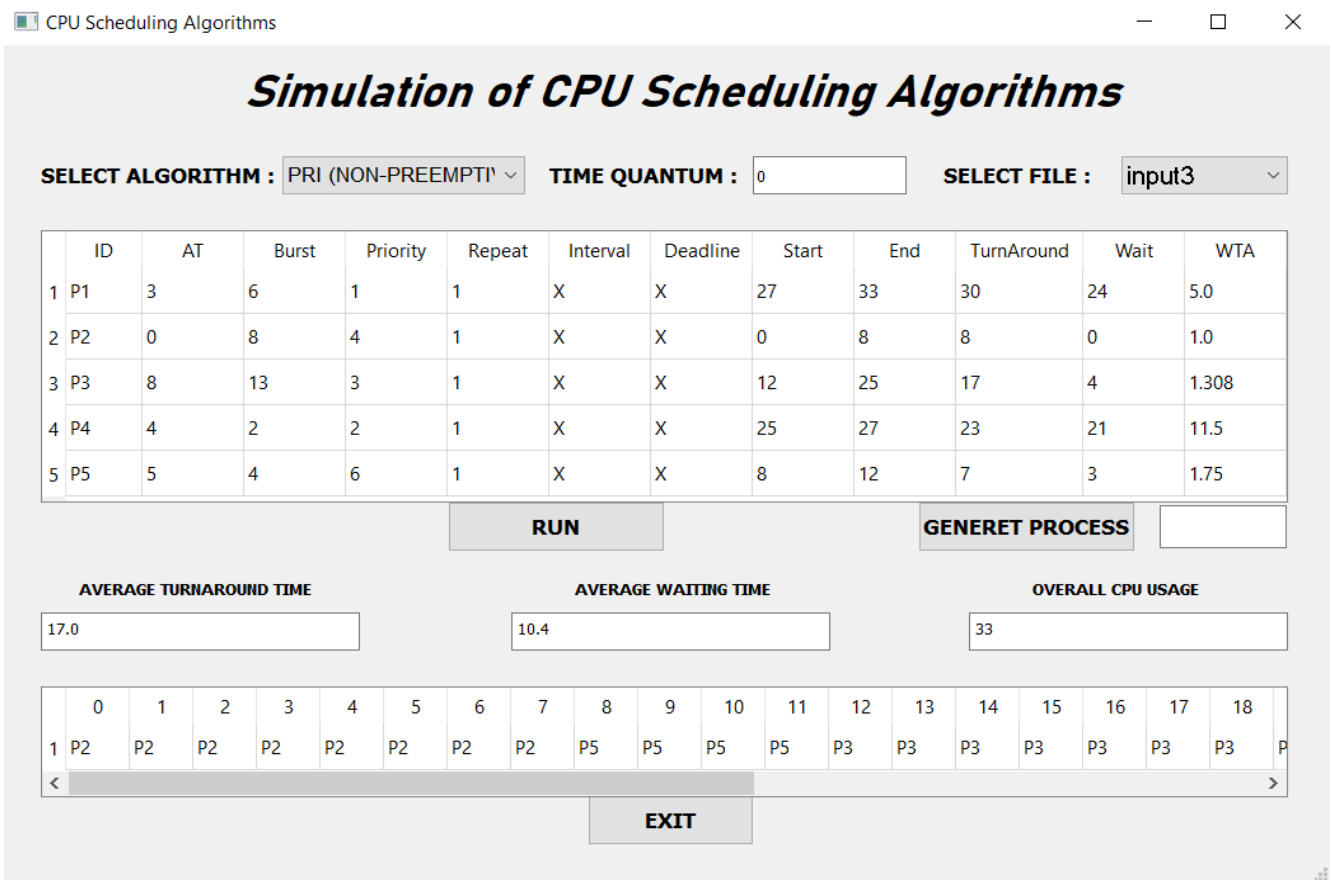
5) SRTF :



6) PRI (PREEMPTIVE) :

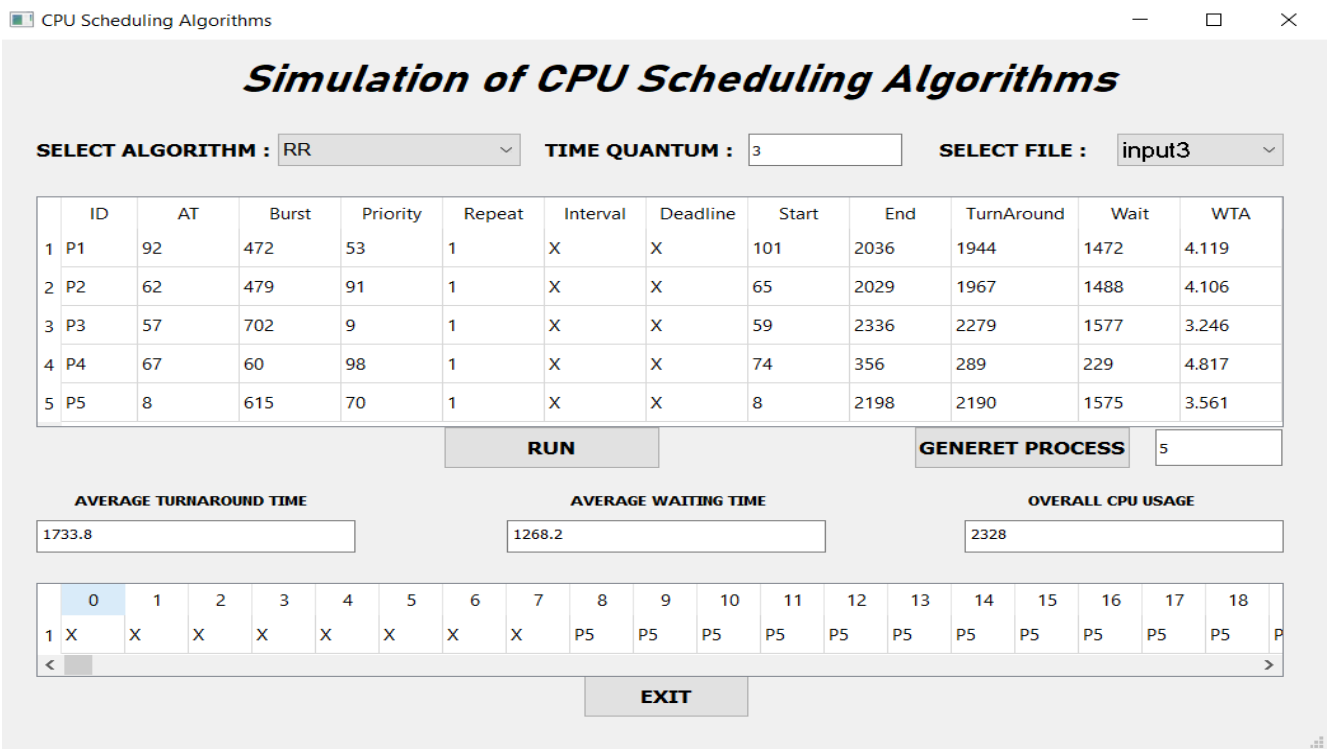


7) PRI (WITHOUT PREEMPTIVE) :

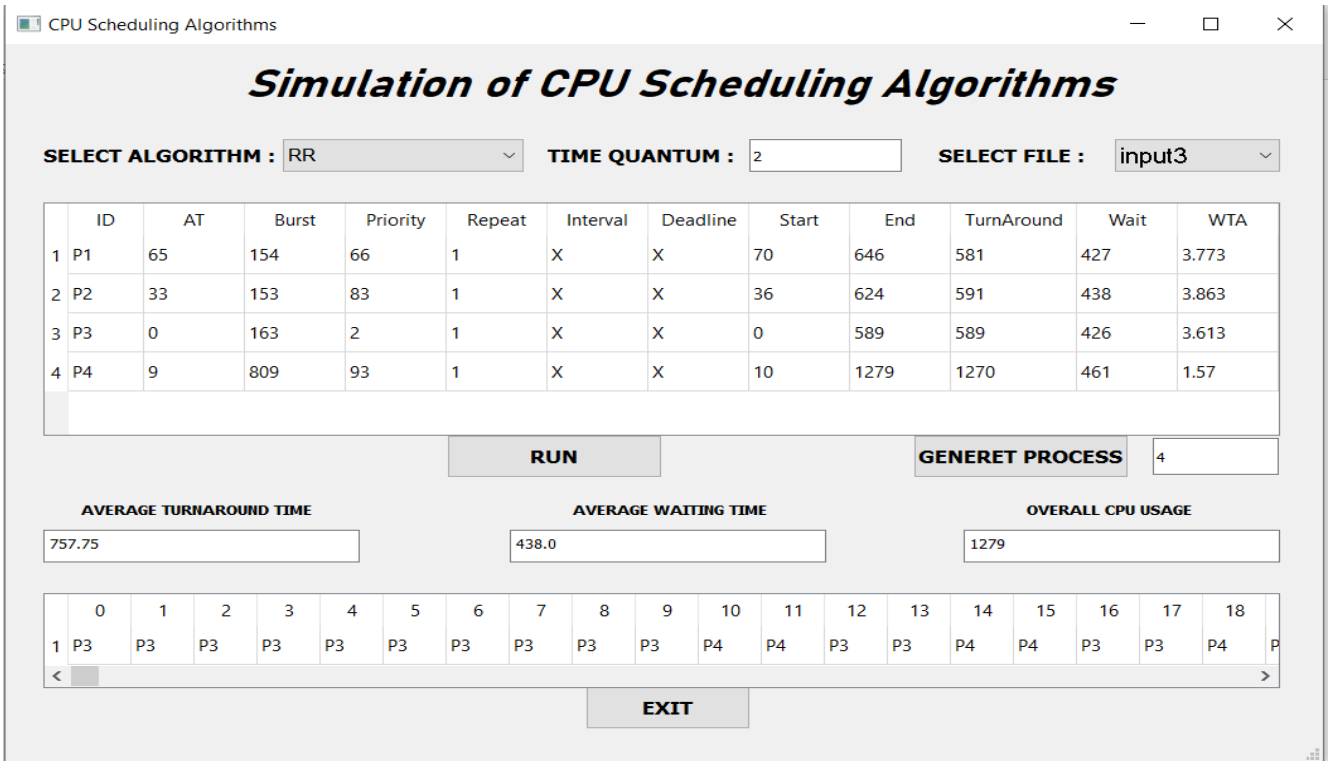


- **Generate process :**

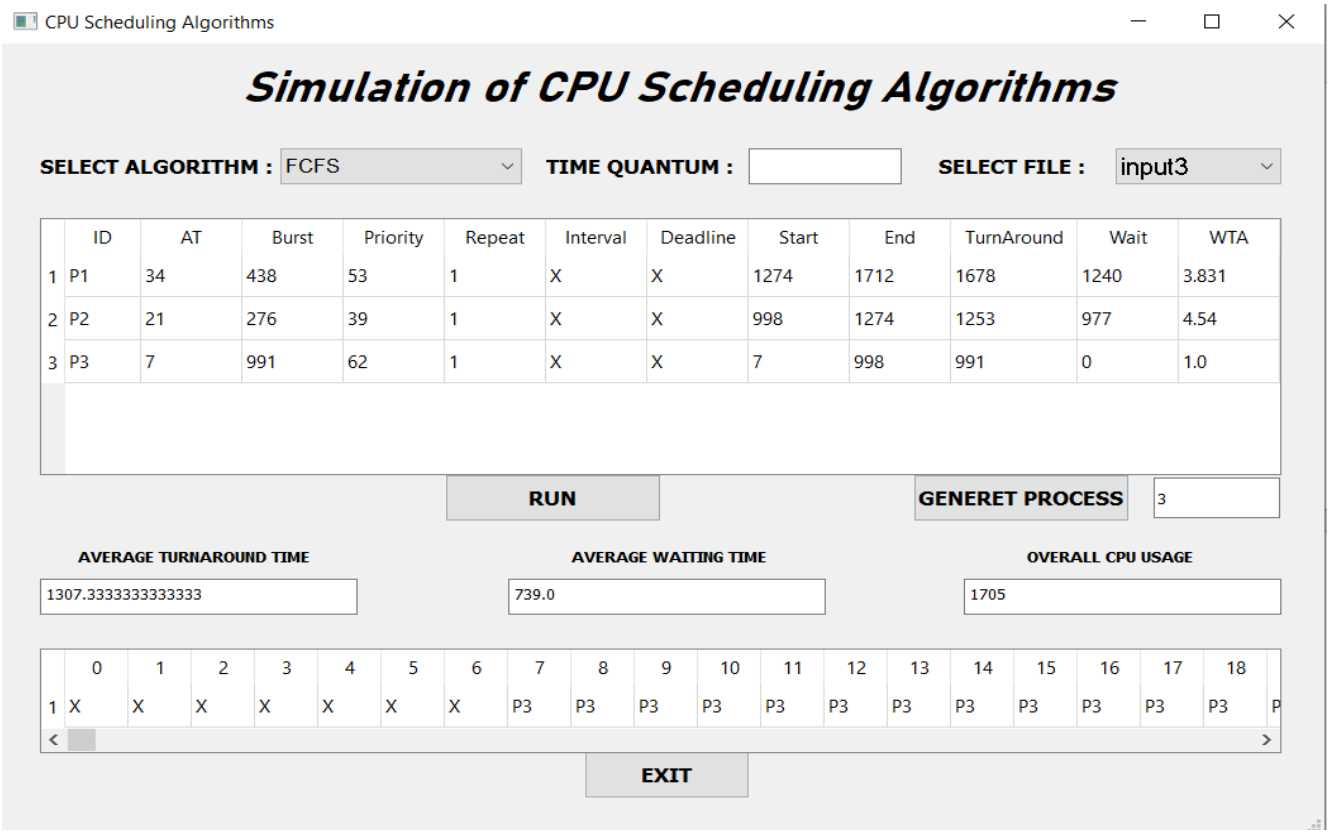
1) RR (with time quantum=3 & generate process = 5):



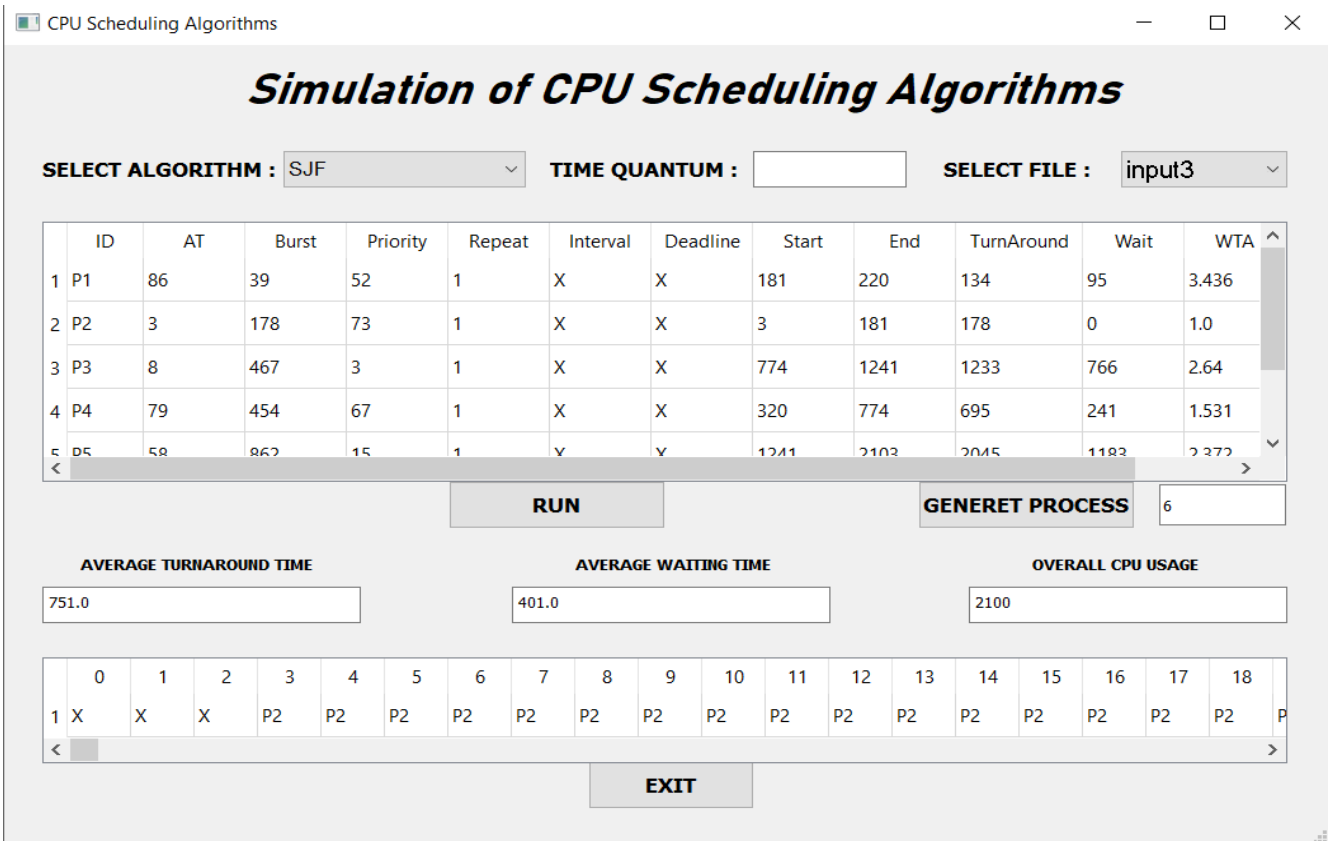
2) RR (with time quantum=2 & generate process =4):



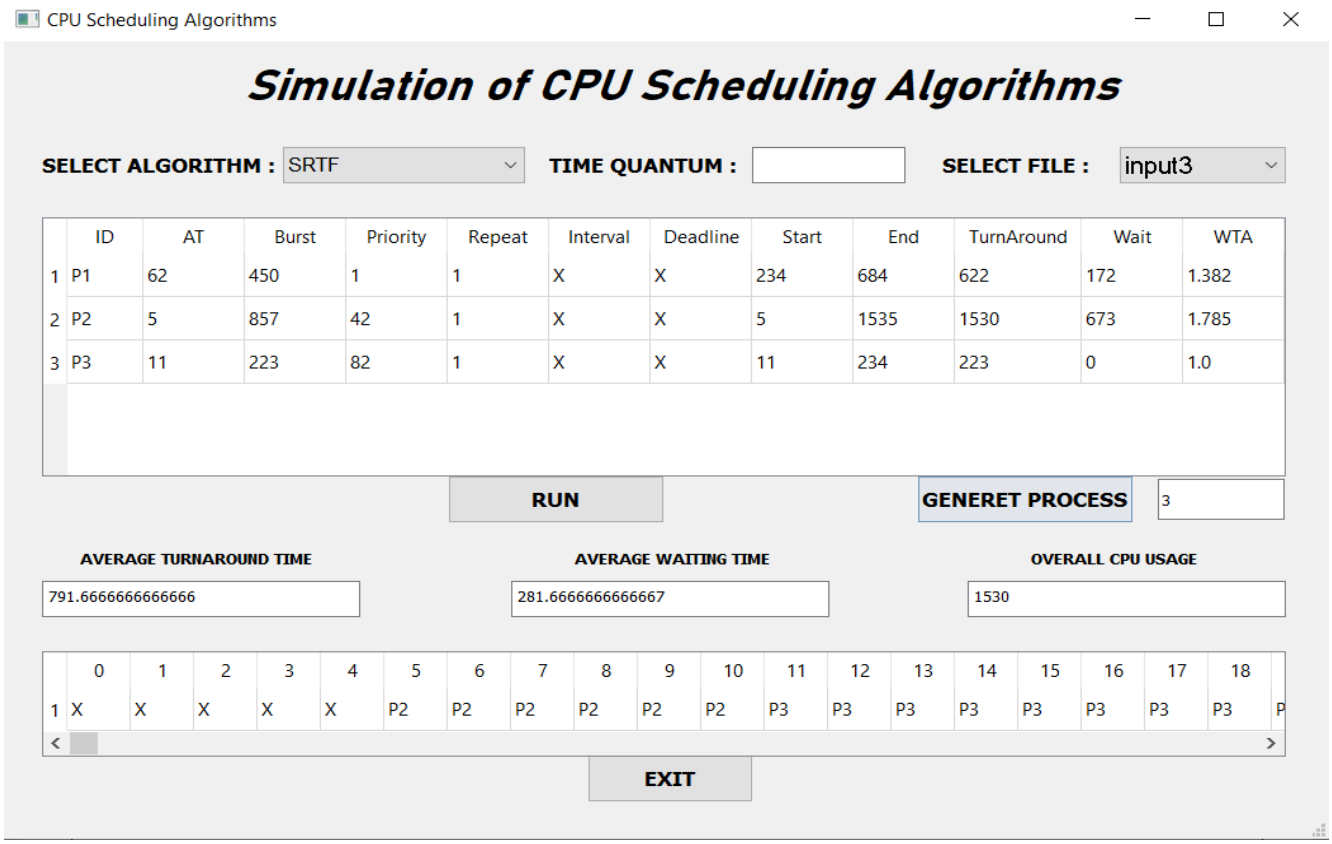
3) FCFS (with generate process = 3):



4) SJF (with generate process =6):



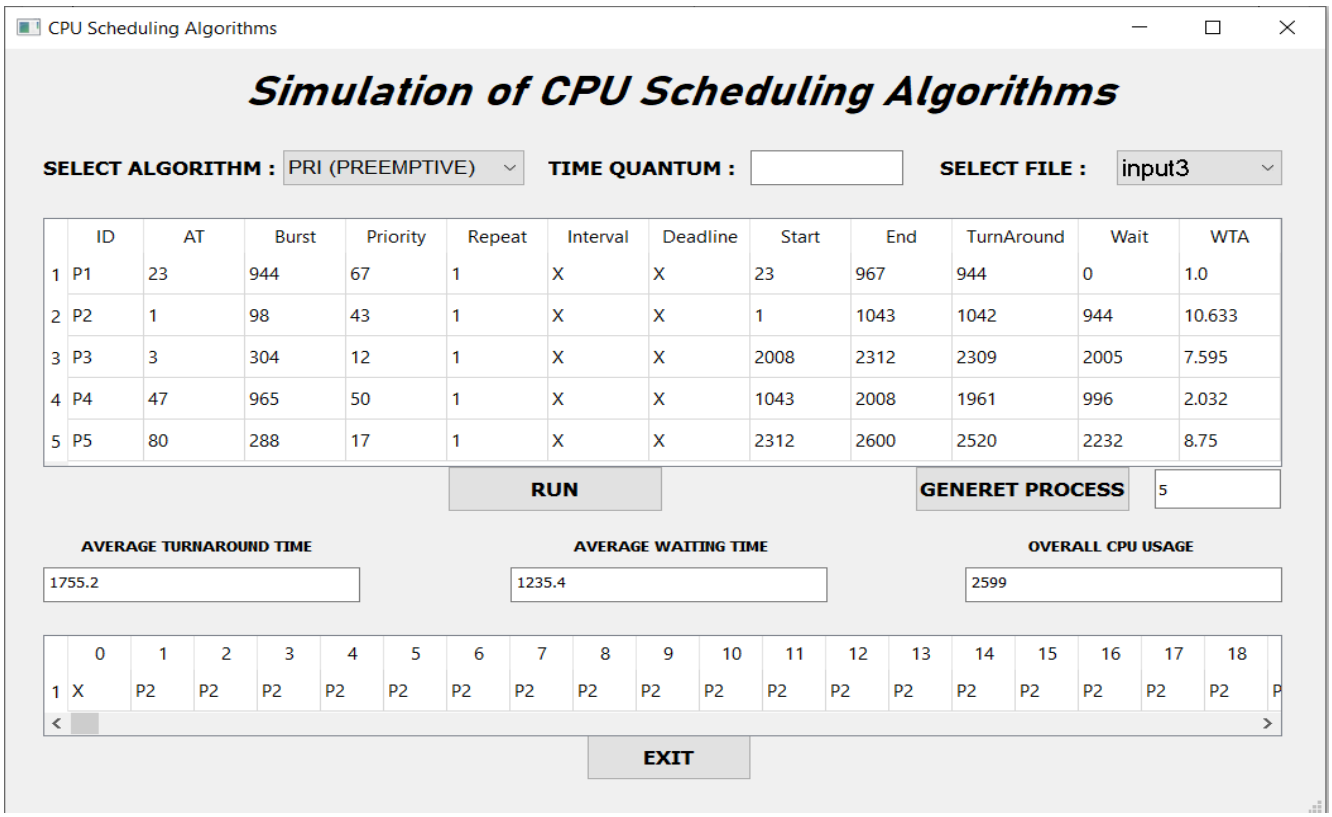
5) SRTF (with generate process =3):



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	62	450	1	1	X	X	234	684	622	172	1.382
2	P2	5	857	42	1	X	X	5	1535	1530	673	1.785
3	P3	11	223	82	1	X	X	11	234	223	0	1.0

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	X	X	X	X	X	P2	P2	P2	P2	P2	P2	P3	P3	P3	P3	P3	P3	P3	P3

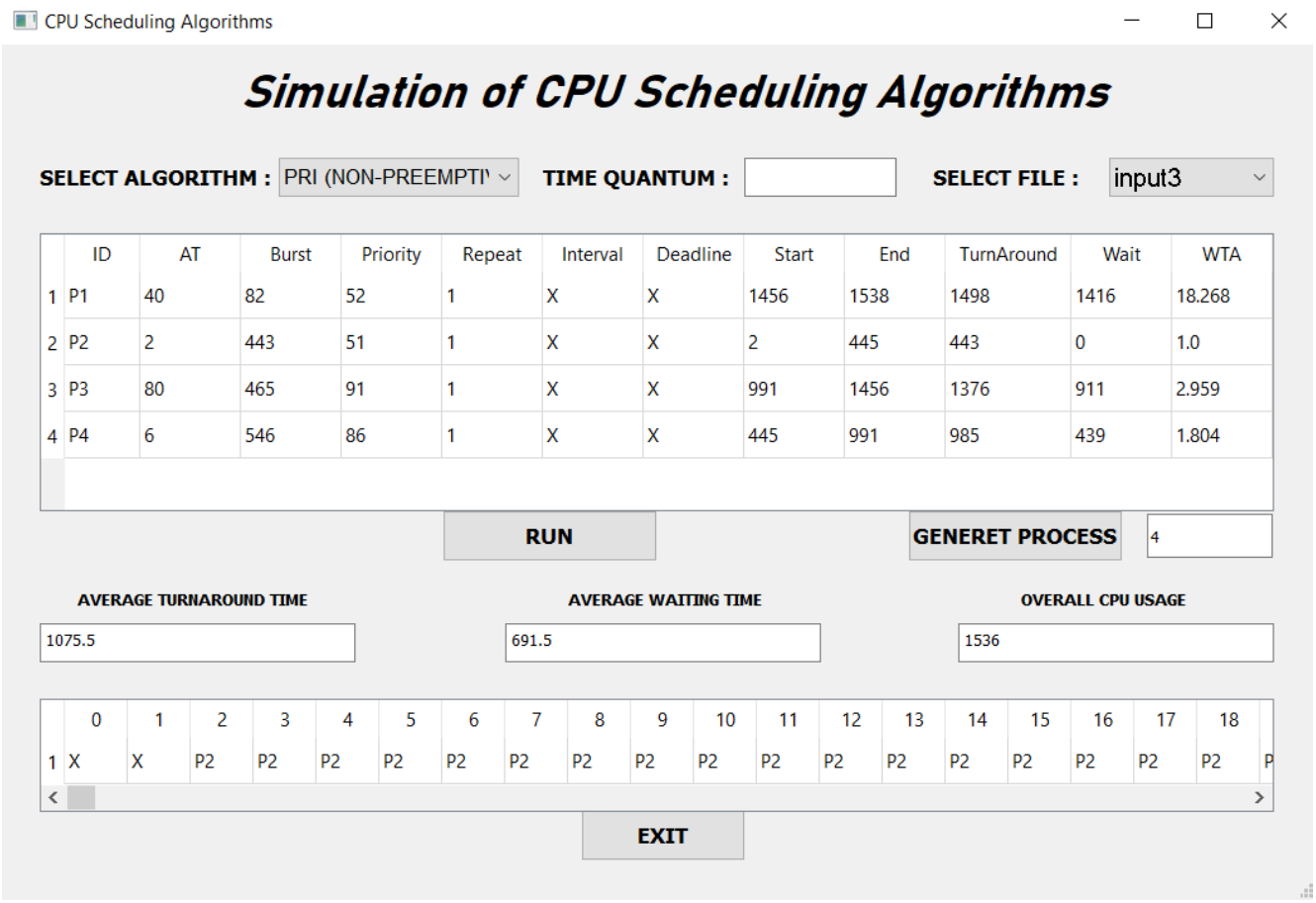
6) PRI(PREEMPTIVE) (with generate process =5):



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	23	944	67	1	X	X	23	967	944	0	1.0
2	P2	1	98	43	1	X	X	1	1043	1042	944	10.633
3	P3	3	304	12	1	X	X	2008	2312	2309	2005	7.595
4	P4	47	965	50	1	X	X	1043	2008	1961	996	2.032
5	P5	80	288	17	1	X	X	2312	2600	2520	2232	8.75

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	X	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2

7) PRI(NON PREEMPTIVE) (with generate process =4):



	ID	AT	Burst	Priority	Repeat	Interval	Deadline	Start	End	TurnAround	Wait	WTA
1	P1	40	82	52	1	X	X	1456	1538	1498	1416	18.268
2	P2	2	443	51	1	X	X	2	445	443	0	1.0
3	P3	80	465	91	1	X	X	991	1456	1376	911	2.959
4	P4	6	546	86	1	X	X	445	991	985	439	1.804

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	X	X	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2	P2

❖References

- https://cutt.ly/6tqQF7 Wednesday 18-11-2020 @ 5:00 pm

- https://www.geeksforgeeks.org/difference-between-preemptive-priority-based-and-non-preemptive-priority-based-cpu-scheduling-algorithms/ Wednesday 18-11-2020 @ 5:30 pm

- A SIMULATION PROJECT FOR AN OPERATING SYSTEMS COURSE(John K. Estell, The University of Toledo)