INFORMATION AND COMMUNICATION TECHNOLOGY

OPERATING SYSTEM

TEAM – 50

MEMBERS – Vishwa Parmar(18BIT117)

Shrey Shah(18BIT104)

Khelan Shah(18BIT055)

Kenil Patel(18BIT053)

Herat Devisha(18BIT026)

TOPIC – SCHEDULING ALGORITHM APP

(Android Studio – JAVA)

SCHEDULING ALGORITHMS

A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms.

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.

The Purpose of a Scheduling algorithm

1. Maximum CPU utilization
2. Fare allocation of CPU
3. Maximum throughput
4. Minimum turnaround time
5. Minimum waiting time
6. Minimum response time

Following are some of the popular scheduling algorithms with their description and working:-

FCFS – First Come First Serve

First come first serve (FCFS) scheduling algorithm simply schedules the jobs according to their arrival time. The job which comes first in the ready queue will get the CPU first. The lesser the arrival time of the job, the sooner will the job get the CPU. FCFS scheduling may cause the problem of starvation if the burst time of the first process is the longest among all the jobs.

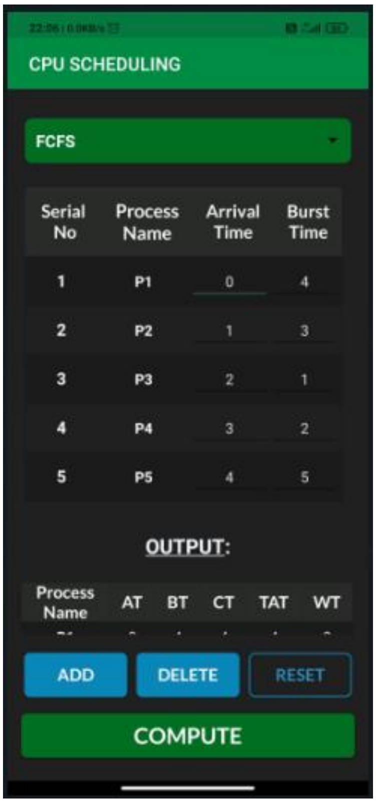
Advantages of FCFS

1. Simple
2. Easy
3. First come, First serve

Disadvantages of FCFS

1. The scheduling method is non preemptive, the process will run to the completion.
2. Due to the non-preemptive nature of the algorithm, the problem of starvation may occur.
3. Although it is easy to implement, but it is poor in performance since the average waiting time is higher as compare to other scheduling algorithms.

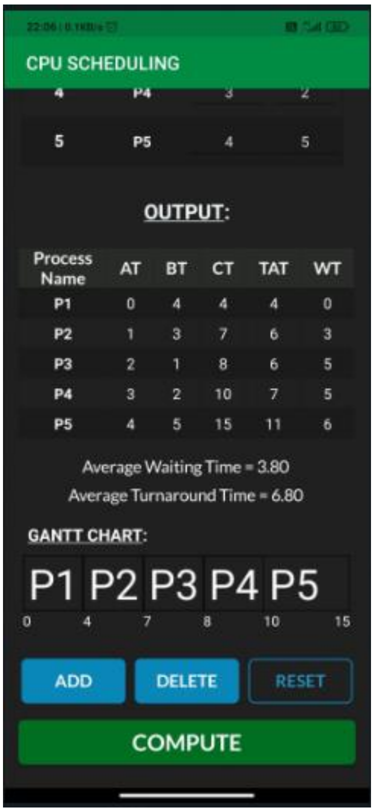
EXAMPLE - 1



Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	4
2	P2	1	3
3	P3	2	1
4	P4	3	2
5	P5	4	5

OUTPUT:

Process Name	AT	BT	CT	TAT	WT



4	P4	3	2
5	P5	4	5

OUTPUT:

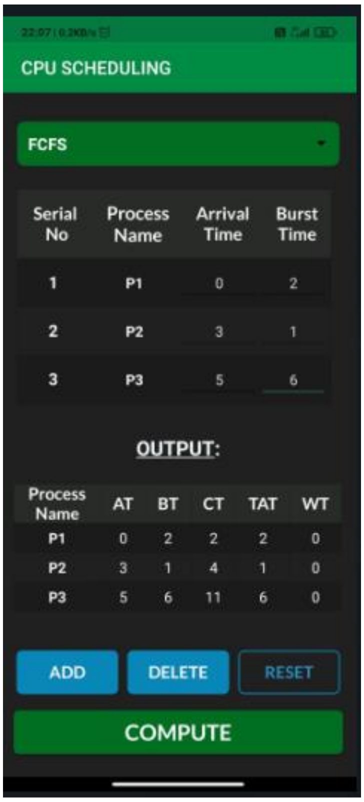
Process Name	AT	BT	CT	TAT	WT
P1	0	4	4	4	0
P2	1	3	7	6	3
P3	2	1	8	6	5
P4	3	2	10	7	5
P5	4	5	15	11	6

Average Waiting Time = 3.80

Average Turnaround Time = 6.80

GANTT CHART:

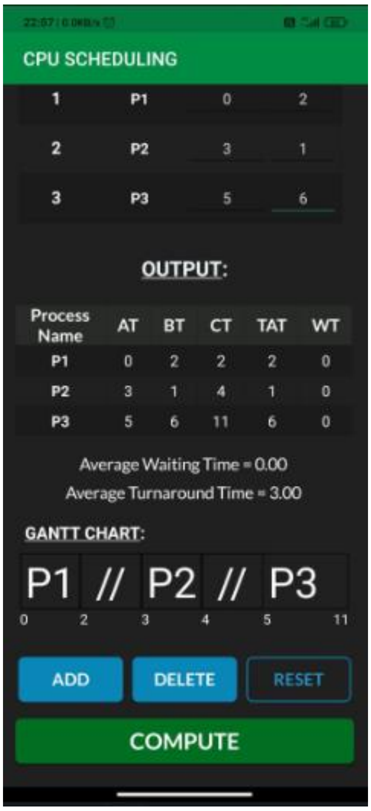
EXAMPLE - 2



Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	2
2	P2	3	1
3	P3	5	6

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	2	2	2	0
P2	3	1	4	1	0
P3	5	6	11	6	0



1	P1	0	2
2	P2	3	1
3	P3	5	6

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	2	2	2	0
P2	3	1	4	1	0
P3	5	6	11	6	0

Average Waiting Time = 0.00

Average Turnaround Time = 3.00

GANTT CHART:

SJF – Shortest Job First

Till now, we were scheduling the processes according to their arrival time (in FCFS scheduling). However, SJF scheduling algorithm, schedules the processes according to their burst time.

In SJF scheduling, the process with the lowest burst time, among the list of available processes in the ready queue, is going to be scheduled next.

However, it is very difficult to predict the burst time needed for a process hence this algorithm is very difficult to implement in the system.

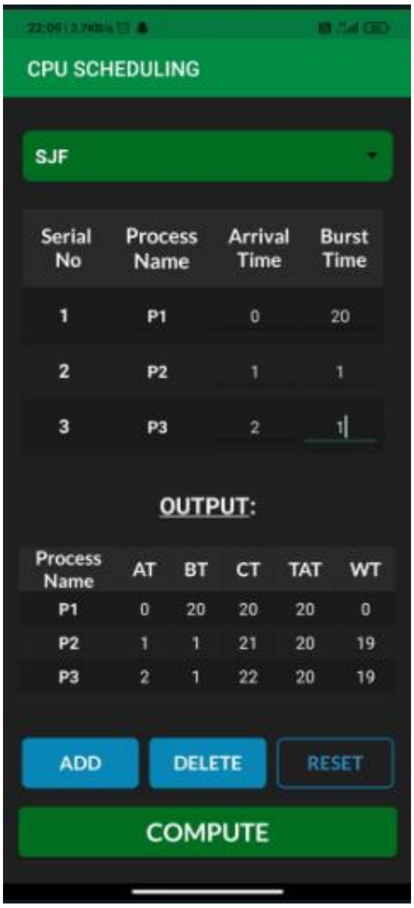
Advantages of SJF

1. Maximum throughput
2. Minimum average waiting and turnaround time

Disadvantages of SJF

1. May suffer with the problem of starvation
2. It is not implementable because the exact Burst time for a process can't be known in advance.

EXAMPLE - 1



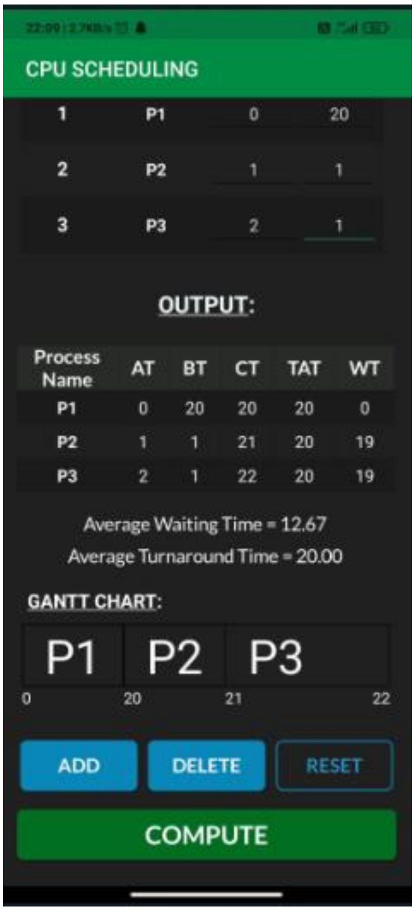
CPU SCHEDULING

SJF

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	20
2	P2	1	1
3	P3	2	1

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	20	20	20	0
P2	1	1	21	20	19
P3	2	1	22	20	19



CPU SCHEDULING

1	P1	0	20
2	P2	1	1
3	P3	2	1

OUTPUT:

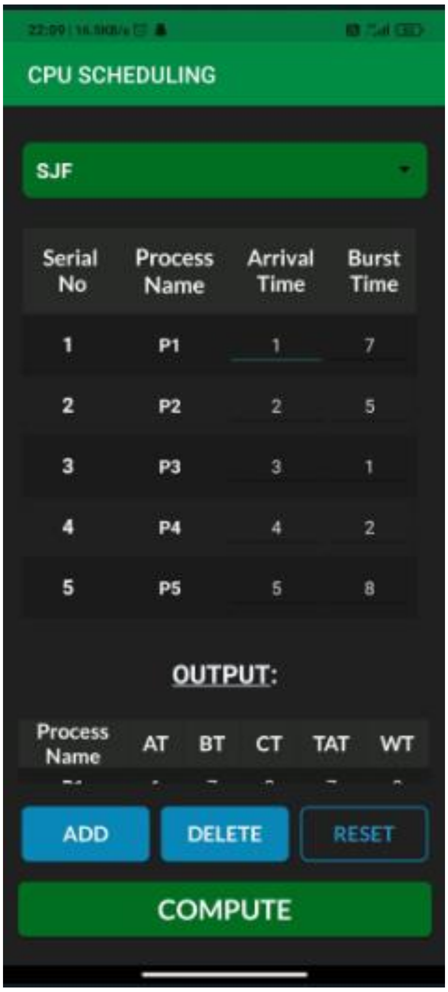
Process Name	AT	BT	CT	TAT	WT
P1	0	20	20	20	0
P2	1	1	21	20	19
P3	2	1	22	20	19

Average Waiting Time = 12.67

Average Turnaround Time = 20.00

GANTT CHART:

EXAMPLE - 2



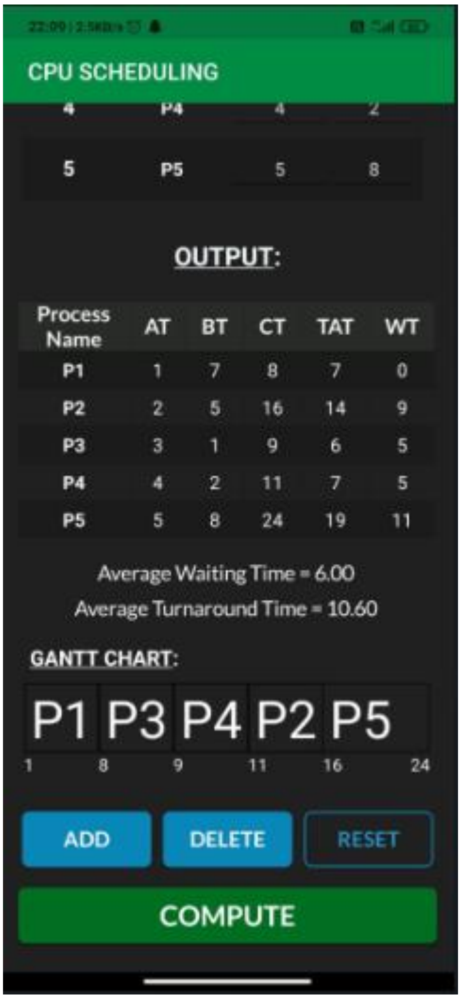
CPU SCHEDULING

SJF

Serial No	Process Name	Arrival Time	Burst Time
1	P1	1	7
2	P2	2	5
3	P3	3	1
4	P4	4	2
5	P5	5	8

OUTPUT:

Process Name	AT	BT	CT	TAT	WT



CPU SCHEDULING

4	P4	4	2
5	P5	5	8

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	1	7	8	7	0
P2	2	5	16	14	9
P3	3	1	9	6	5
P4	4	2	11	7	5
P5	5	8	24	19	11

Average Waiting Time = 6.00

Average Turnaround Time = 10.60

GANTT CHART:

P1	P3	P4	P2	P5

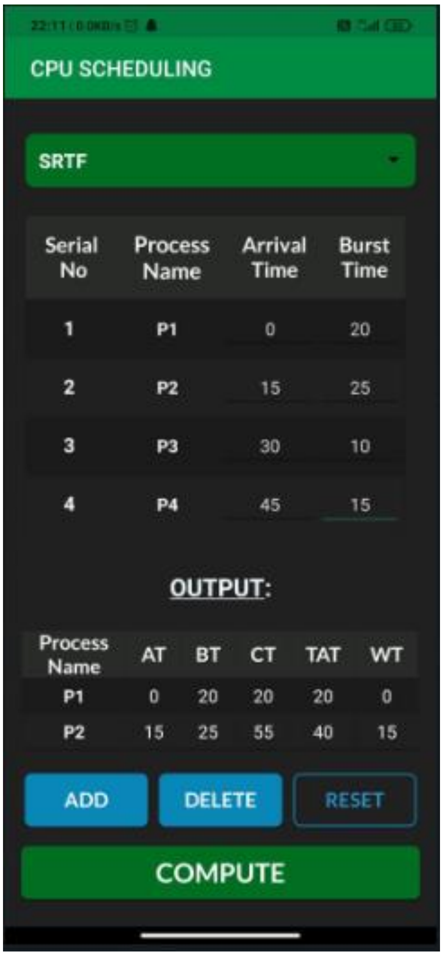
1 8 9 11 16 24

SRTF – Shortest Remaining Time First

This Algorithm is the **preemptive version** of **SJF scheduling**. In SRTF, the execution of the process can be stopped after certain amount of time. At the arrival of every process, the short term scheduler schedules the process with the least remaining burst time among the list of available processes and the running process.

Once all the processes are available in the **ready queue**, No preemption will be done and the algorithm will work as **SJF scheduling**. The context of the process is saved in the **Process Control Block** when the process is removed from the execution and the next process is scheduled. This PCB is accessed on the **next execution** of this process.

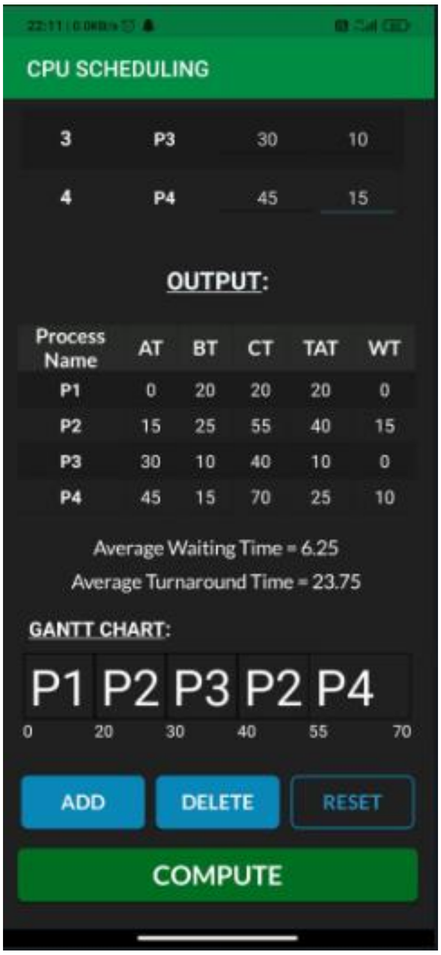
EXAMPE - 1



Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	20
2	P2	15	25
3	P3	30	10
4	P4	45	15

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	20	20	20	0
P2	15	25	55	40	15



3	P3	30	10
4	P4	45	15

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	20	20	20	0
P2	15	25	55	40	15
P3	30	10	40	10	0
P4	45	15	70	25	10

Average Waiting Time = 6.25

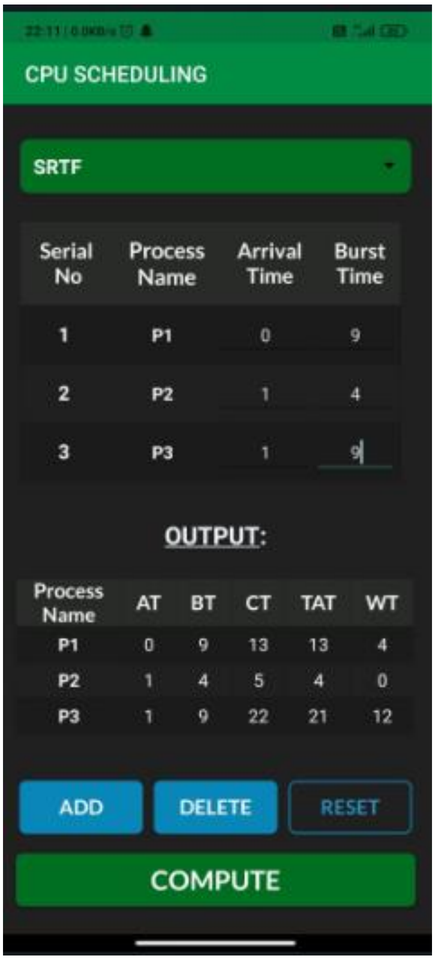
Average Turnaround Time = 23.75

GANTT CHART:

P1	P2	P3	P2	P4

0 20 30 40 55 70

EXAMPLE - 2



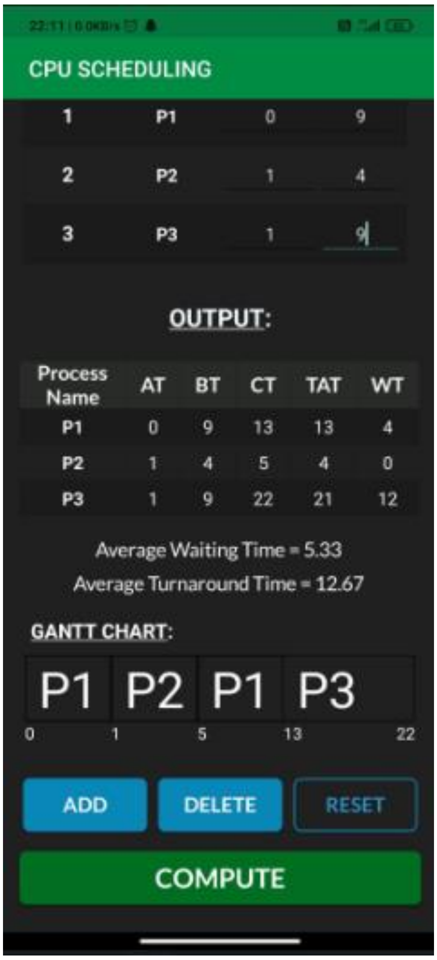
CPU SCHEDULING

SRTF

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	9
2	P2	1	4
3	P3	1	9

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	9	13	13	4
P2	1	4	5	4	0
P3	1	9	22	21	12



CPU SCHEDULING

1	P1	0	9
2	P2	1	4
3	P3	1	9

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	9	13	13	4
P2	1	4	5	4	0
P3	1	9	22	21	12

Average Waiting Time = 5.33

Average Turnaround Time = 12.67

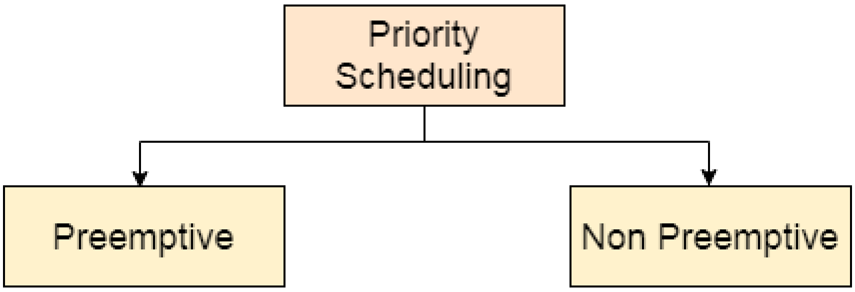
GANTT CHART:

P1	P2	P1	P3

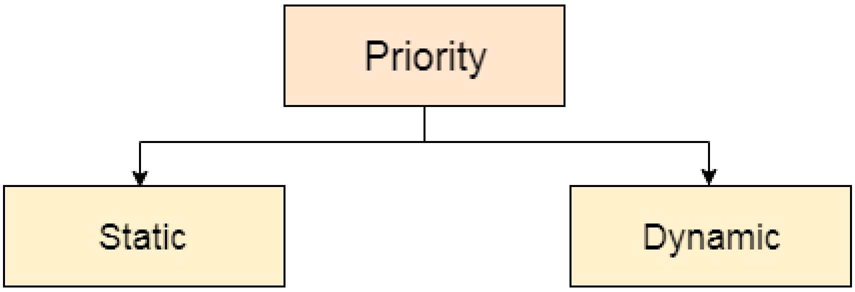
0 1 5 13 22

PRIORITY

In Priority scheduling, there is a priority number assigned to each process. In some systems, the lower the number, the higher the priority. While, in the others, the higher the number, the higher will be the priority. The Process with the higher priority among the available processes is given the CPU. There are two types of priority scheduling algorithm exists. One is **Preemptive** priority scheduling while the other is **Non Preemptive** Priority scheduling.



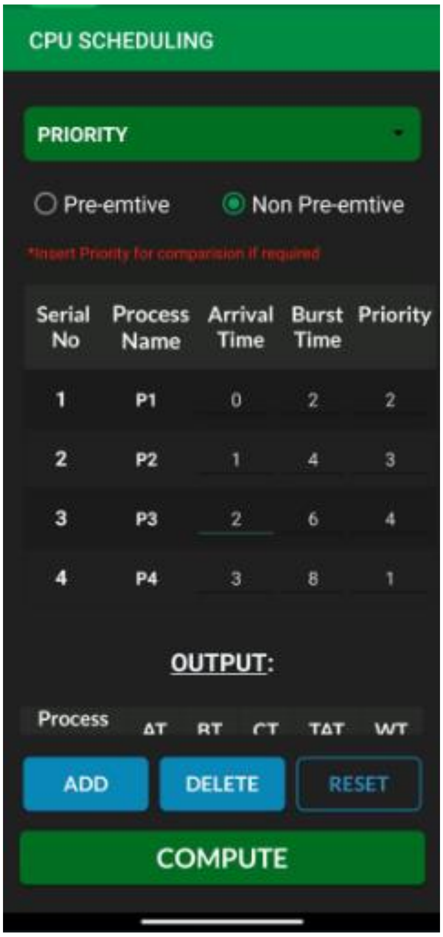
The priority number assigned to each of the process may or may not vary. If the priority number doesn't change itself throughout the process, it is called **static priority**, while if it keeps changing itself at the regular intervals, it is called **dynamic priority**.



NON-PREEMPTIVE PRIORITY

In the Non Preemptive Priority scheduling, The Processes are scheduled according to the priority number assigned to them. Once the process gets scheduled, it will run till the completion. Generally, the lower the priority number, the higher is the priority of the process.

EXAMPLE



CPU SCHEDULING

PRIORITY

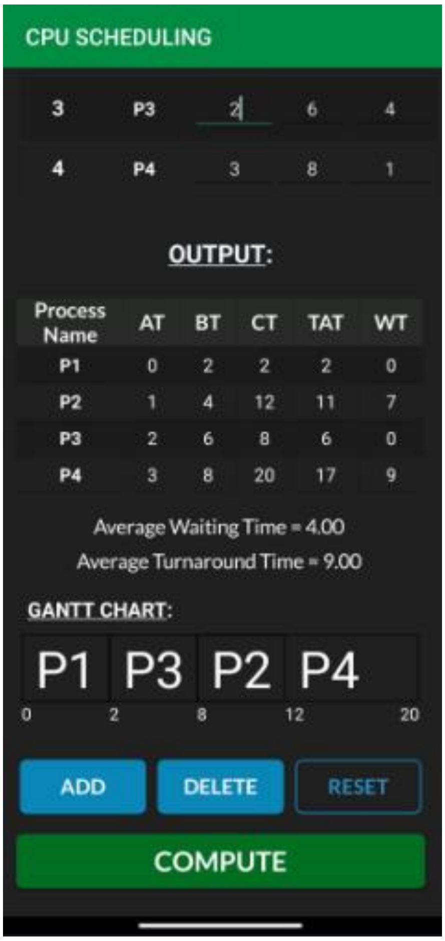
○ Pre-emtive ● Non Pre-emtive

*Insert Priority for comparision if required

Serial No	Process Name	Arrival Time	Burst Time	Priority
1	P1	0	2	2
2	P2	1	4	3
3	P3	2	6	4
4	P4	3	8	1

OUTPUT:

Process AT BT CT TAT WT



CPU SCHEDULING

3	P3	2	6	4
4	P4	3	8	1

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	2	2	2	0
P2	1	4	12	11	7
P3	2	6	8	6	0
P4	3	8	20	17	9

Average Waiting Time = 4.00

Average Turnaround Time = 9.00

GANTT CHART:

P1 P3 P2 P4

0 2 8 12 20

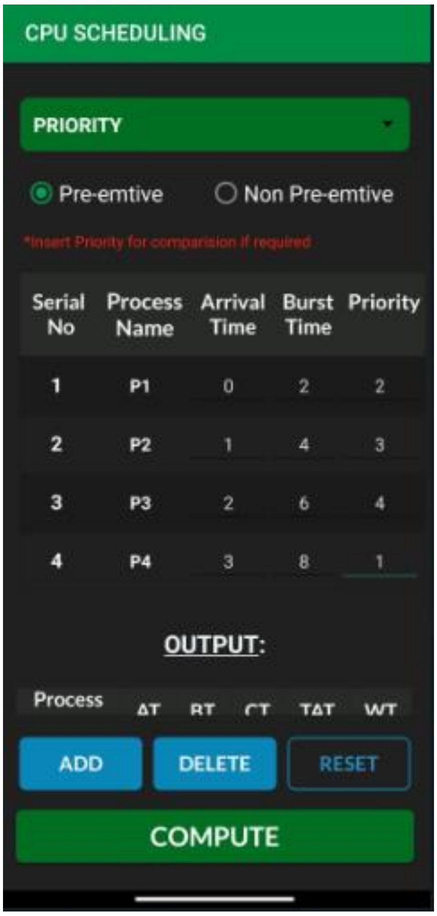
PREEMPTIVE PRIORITY

In Preemptive Priority Scheduling, at the time of arrival of a process in the ready queue, its Priority is compared with the priority of the other processes present in the ready queue as well as with the one which is being executed by the CPU at that point of time. The One with the highest priority among all the available processes will be given the CPU next.

The difference between preemptive priority scheduling and non preemptive priority scheduling is that, in the preemptive priority scheduling, the job which is being executed can be stopped at the arrival of a higher priority job.

Once all the jobs get available in the ready queue, the algorithm will behave as non-preemptive priority scheduling, which means the job scheduled will run till the completion and no preemption will be done.

EXAMPLE



PRIORITY

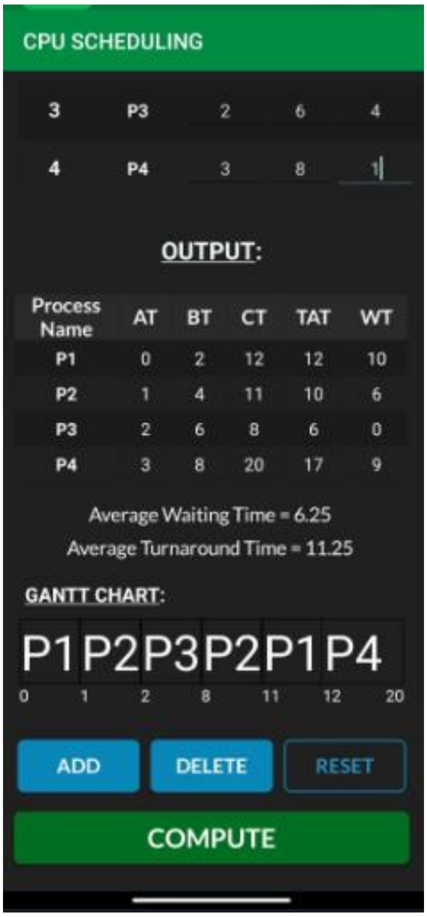
● Pre-emtive ○ Non Pre-emtive

*Insert Priority for comparision if required

Serial No	Process Name	Arrival Time	Burst Time	Priority
1	P1	0	2	2
2	P2	1	4	3
3	P3	2	6	4
4	P4	3	8	1

OUTPUT:

Process AT BT CT TAT WT



3	P3	2	6	4
4	P4	3	8	1

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	2	12	12	10
P2	1	4	11	10	6
P3	2	6	8	6	0
P4	3	8	20	17	9

Average Waiting Time = 6.25

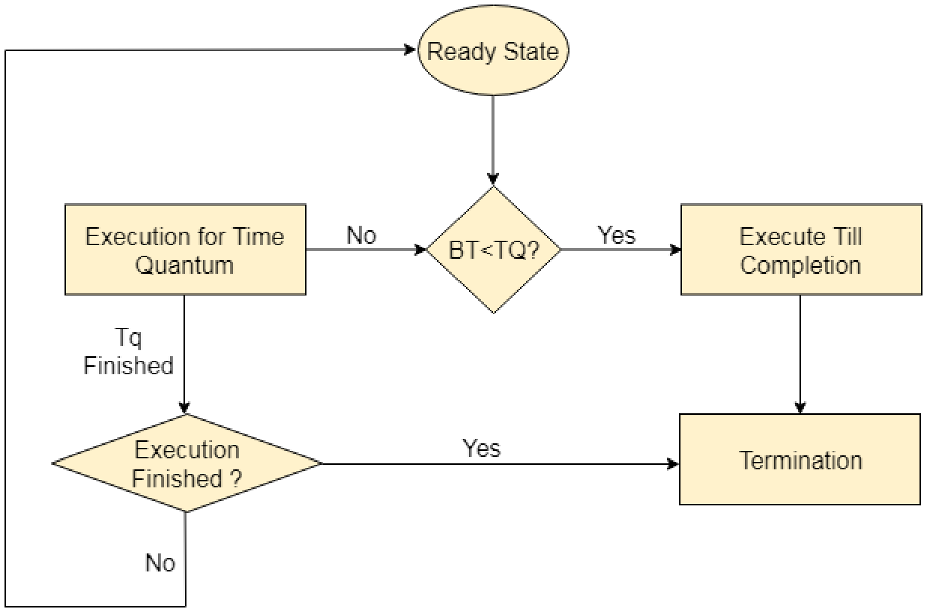
Average Turnaround Time = 11.25

GANTT CHART:

P1	P2	P3	P2	P1	P4	
0	1	2	8	11	12	20

ROUND ROBIN

Round Robin scheduling algorithm is one of the most popular scheduling algorithm which can actually be implemented in most of the operating systems. This is the **preemptive version** of first come first serve scheduling. The Algorithm focuses on Time Sharing. In this algorithm, every process gets executed in a **cyclic way**. A certain time slice is defined in the system which is called time **quantum**. Each process present in the ready queue is assigned the CPU for that time quantum, if the execution of the process is completed during that time then the process will **terminate** else the process will go back to the **ready queue** and waits for the next turn to complete the execution.



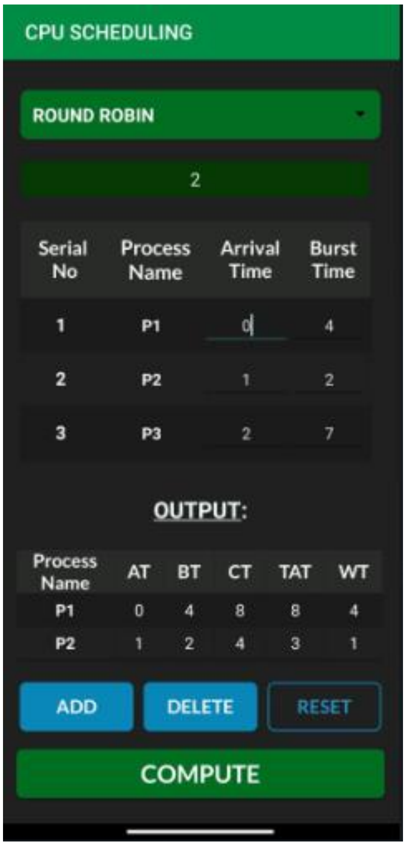
Advantages

1. It can be actually implementable in the system because it is not depending on the burst time.
2. It doesn't suffer from the problem of starvation or convoy effect.
3. All the jobs get a fare allocation of CPU.

Disadvantages

1. The higher the time quantum, the higher the response time in the system.
2. The lower the time quantum, the higher the context switching overhead in the system.
3. Deciding a perfect time quantum is really a very difficult task in the system.

EXAMPLE - 1



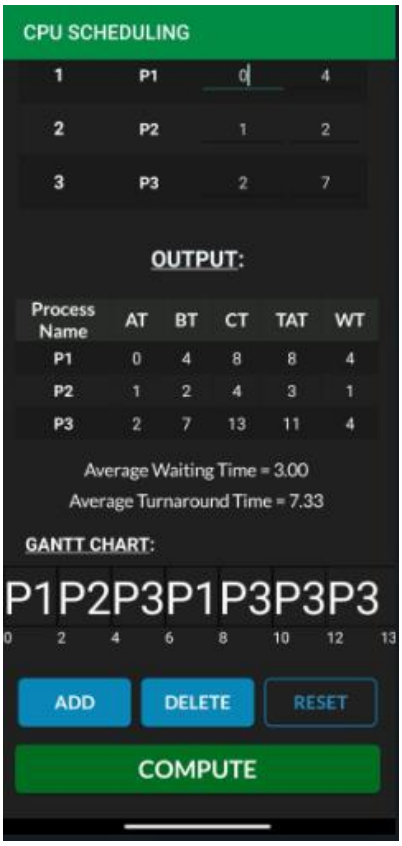
ROUND ROBIN

2

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	4
2	P2	1	2
3	P3	2	7

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	4	8	8	4
P2	1	2	4	3	1



1	P1	0	4
2	P2	1	2
3	P3	2	7

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	4	8	8	4
P2	1	2	4	3	1
P3	2	7	13	11	4

Average Waiting Time = 3.00

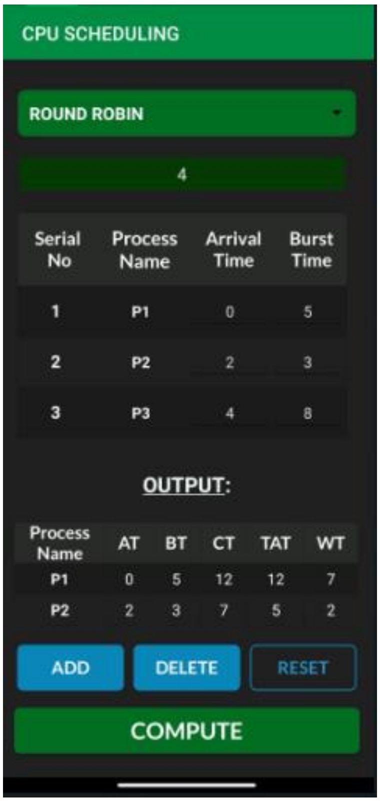
Average Turnaround Time = 7.33

GANTT CHART:

P1	P2	P3	P1	P3	P3	P3

0 2 4 6 8 10 12 13

EXAMPLE - 2



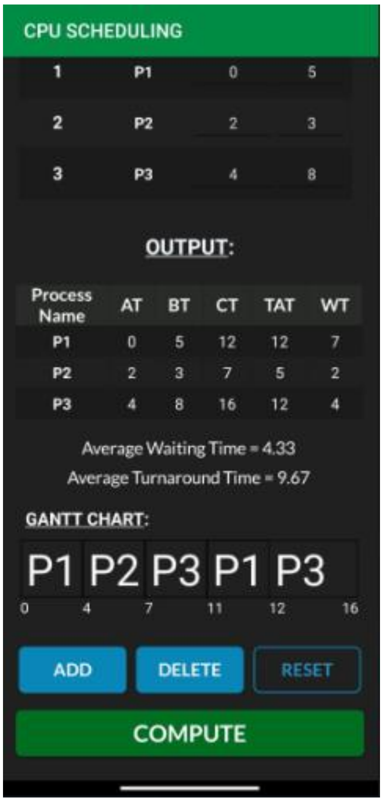
ROUND ROBIN

4

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	5
2	P2	2	3
3	P3	4	8

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	5	12	12	7
P2	2	3	7	5	2



1	P1	0	5
2	P2	2	3
3	P3	4	8

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	5	12	12	7
P2	2	3	7	5	2
P3	4	8	16	12	4

Average Waiting Time = 4.33

Average Turnaround Time = 9.67

GANTT CHART:

P1	P2	P3	P1	P3

0 4 7 11 12 16

LJF – Longest Job First

In LJF Scheduling,

- Out of all the available processes, CPU is assigned to the process having largest burst time.
- In case of a tie, it is broken by FCFS Scheduling.
- LJF Scheduling can be used in both preemptive and non-preemptive mode.
- Preemptive mode of Longest Job First is called as Longest Remaining Time First (LRTF).

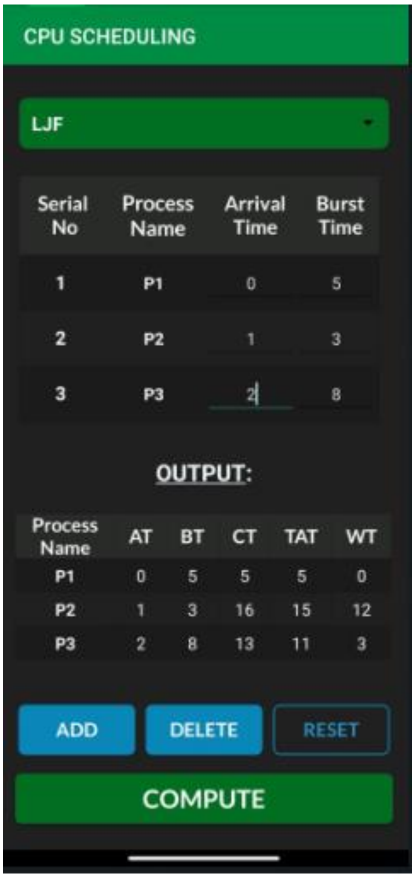
Advantages

1. No process can complete until the longest job also reaches its completion.
2. All the processes approximately finishes at the same time.

Disadvantages

1.The waiting time is high.
2.Processes with smaller burst time may starve for CPU.

EXAMPLE - 1



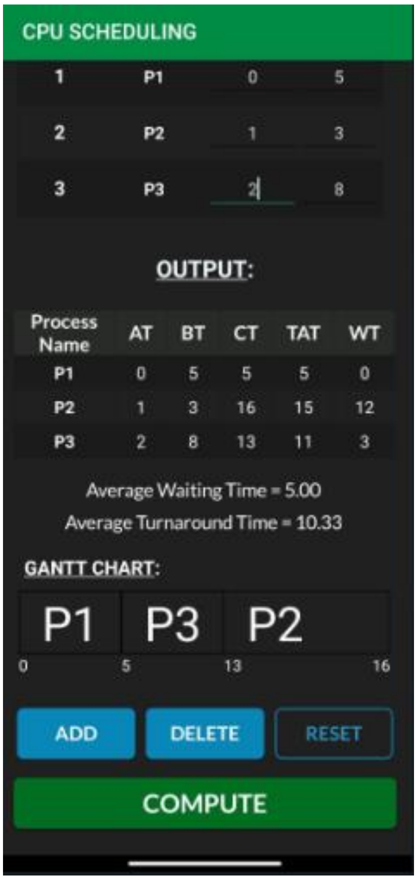
CPU SCHEDULING

LJF

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	5
2	P2	1	3
3	P3	2	8

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	5	5	5	0
P2	1	3	16	15	12
P3	2	8	13	11	3



CPU SCHEDULING

1	P1	0	5
2	P2	1	3
3	P3	2	8

OUTPUT:

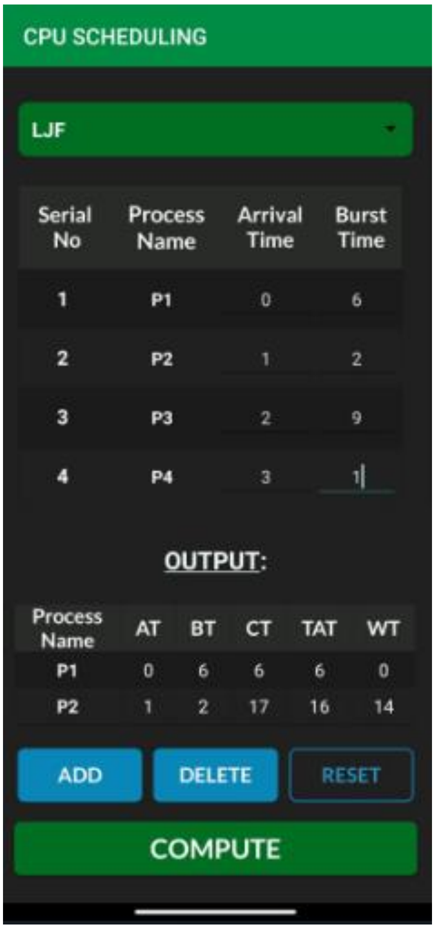
Process Name	AT	BT	CT	TAT	WT
P1	0	5	5	5	0
P2	1	3	16	15	12
P3	2	8	13	11	3

Average Waiting Time = 5.00

Average Turnaround Time = 10.33

GANTT CHART:

EXAMPLE – 2

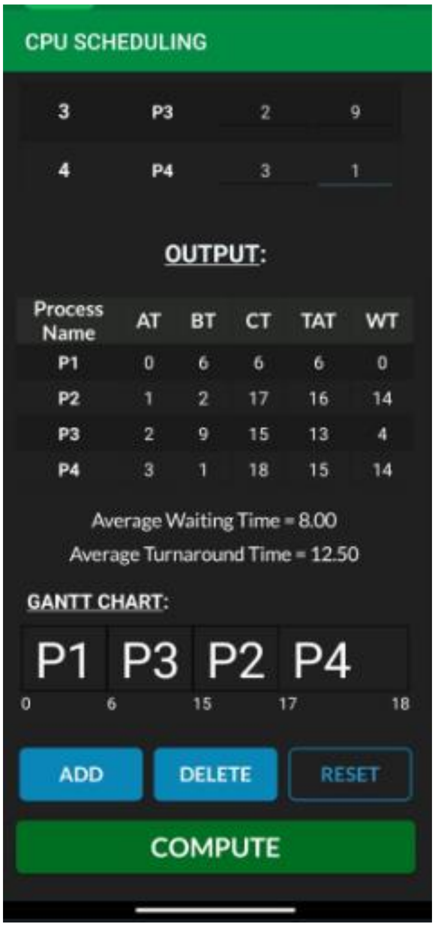

CPU SCHEDULING

LJF

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	6
2	P2	1	2
3	P3	2	9
4	P4	3	1

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	6	6	6	0
P2	1	2	17	16	14


CPU SCHEDULING

3	P3	2	9
4	P4	3	1

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	6	6	6	0
P2	1	2	17	16	14
P3	2	9	15	13	4
P4	3	1	18	15	14

Average Waiting Time = 8.00

Average Turnaround Time = 12.50

GANTT CHART:

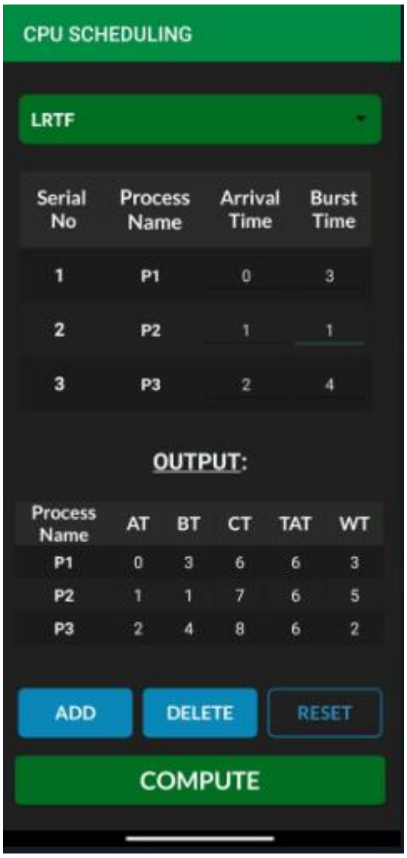
P1	P3	P2	P4

0 6 15 17 18

LRTF – Longest Remaining Time First

Longest Remaining Time First is a scheduling Algorithm used by the operating system to schedule the incoming processes so that they can be executed in a systematic way. This algorithm schedules those processes first which have the longest processing time remaining for completion. This algorithm can also be called as the preemptive version of the LJF scheduling algorithm.

EXAMPLE - 1



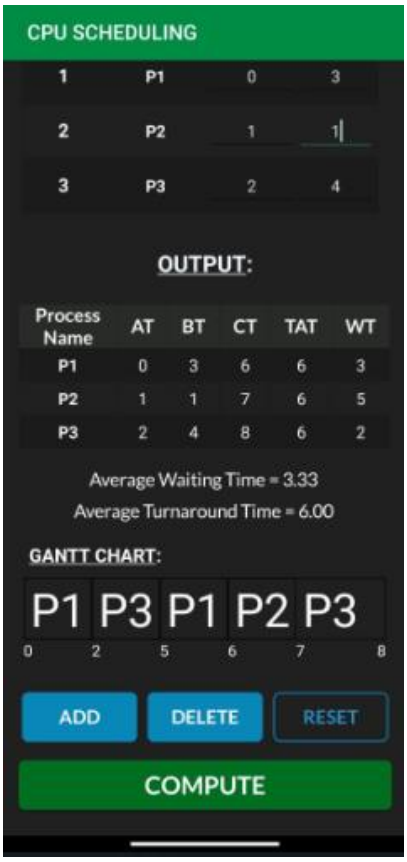
CPU SCHEDULING

LRTF

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	3
2	P2	1	1
3	P3	2	4

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	3	6	6	3
P2	1	1	7	6	5
P3	2	4	8	6	2



CPU SCHEDULING

1	P1	0	3
2	P2	1	1
3	P3	2	4

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	3	6	6	3
P2	1	1	7	6	5
P3	2	4	8	6	2

Average Waiting Time = 3.33

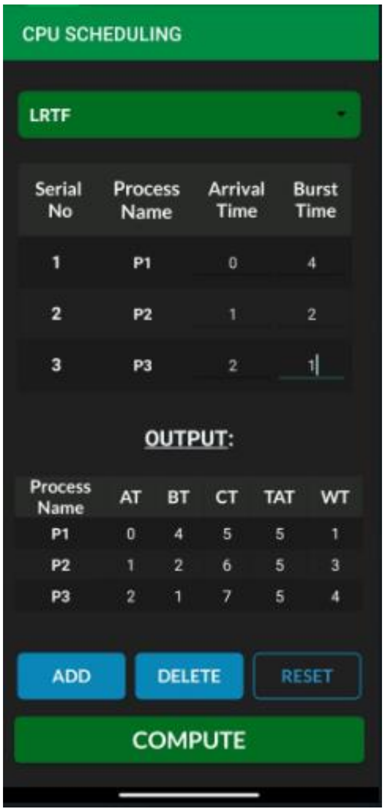
Average Turnaround Time = 6.00

GANTT CHART:

P1	P3	P1	P2	P3

0 2 5 6 7 8

EXAMPLE - 2



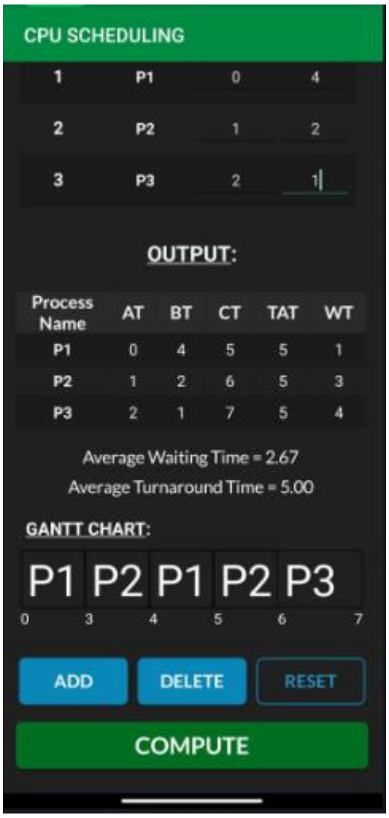
CPU SCHEDULING

LRTF

Serial No	Process Name	Arrival Time	Burst Time
1	P1	0	4
2	P2	1	2
3	P3	2	1

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	4	5	5	1
P2	1	2	6	5	3
P3	2	1	7	5	4



CPU SCHEDULING

1	P1	0	4
2	P2	1	2
3	P3	2	1

OUTPUT:

Process Name	AT	BT	CT	TAT	WT
P1	0	4	5	5	1
P2	1	2	6	5	3
P3	2	1	7	5	4

Average Waiting Time = 2.67

Average Turnaround Time = 5.00

GANTT CHART:

P1	P2	P1	P2	P3

0 3 4 5 6 7