Phase 1

• Discussion of the data structures used in the project for each scheduling algorithm

We used an array for the process table of size MAX_PROC_TABLE_SIZE.
It is the circular array and whenever it reaches the end it loops back from the start, when a process is terminated or leaves the system, its block in the process table is flagged to be free so other incoming processes can reside on it.

When the current processes in the system reaches MAX_PROC_TABLE_SIZE , the new coming processes get discarded.

Your algorithm explanation and results.

1. First Come First Serve (FCFS)

By imagining that process table as a circular array, the pcb_ind is the index of the last entered process in the process table and the pcb_curr is the index to the currently executing process, it is always the case that pcb_curr is behind or the same as pcb_ind, that assures that the processes will execute by the order of arrival.

2. Shortest Job First (SJF)

As this is a batch system, at first we start by picking the least running time process and after finishing we loop again and pick the least running time process and so on.

3. Preemptive Highest Priority First (HPF)

In this algorithm, we need to check for the highest process in terms of priority and execute it, whether there was a process executing before it or not, that is because this algorithm is preemptive so a process might be stopped to run another.

4. Shortest Remaining Time Next (SRTN)

Same as HPF but the condition over preemption is for the remaining time of the process which is kept track of by the process control block of this very process.

5. Round Robin (RR)

We use the quantum entered by the user to determine how much a process will run when it is picked by the scheduler, when no quantum time is entered by the user it will default to ‘1’ clock cycle (1s).

The technique used is that when a process is running, any processes entering the system will reside at the end of the virtual queue data structure and when the quantum of the running process ends it will get back at the back of the queue.

TL;DR, when a process is stopped, it goes to the far back of the queue, back so that any incoming process at the same clock of the running process being stopped will reside before the stopped process in turn in the virtual queue.

***Note: the virtual queue mentioned is the same array described above, but it needs some functions to manipulate it to act like a queue for round robin to work as expected.**

• Your assumptions.

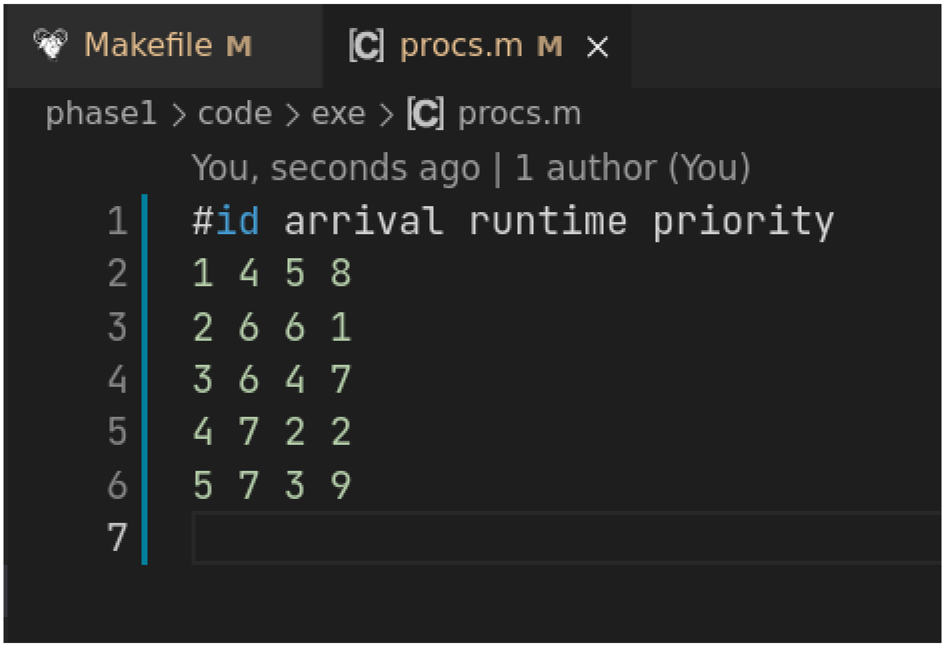
1- The maximum size of processes in the process table is 1024.
2- A process that takes zero time to execute instantaneously.
#3-memory per process is at least 1 block,

• Workload distribution.

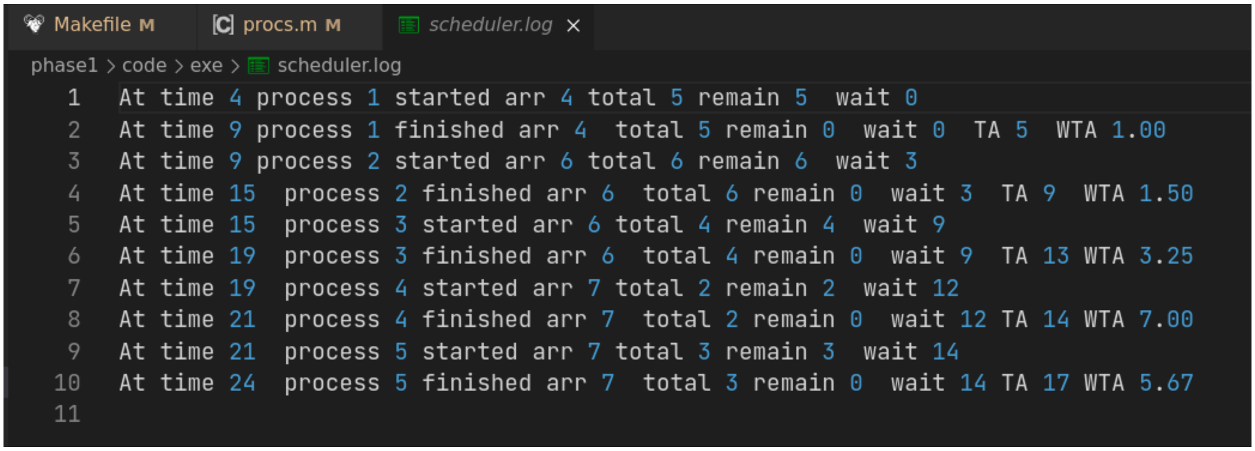
1- Mostafa Sobhy: FCFS & SJF
2- Mostafa Magdy: HPF & STRN
3- Zyad Hasan: Communication between Process generator and Scheduler.
4- Omer Yacine: RR

Test Cases

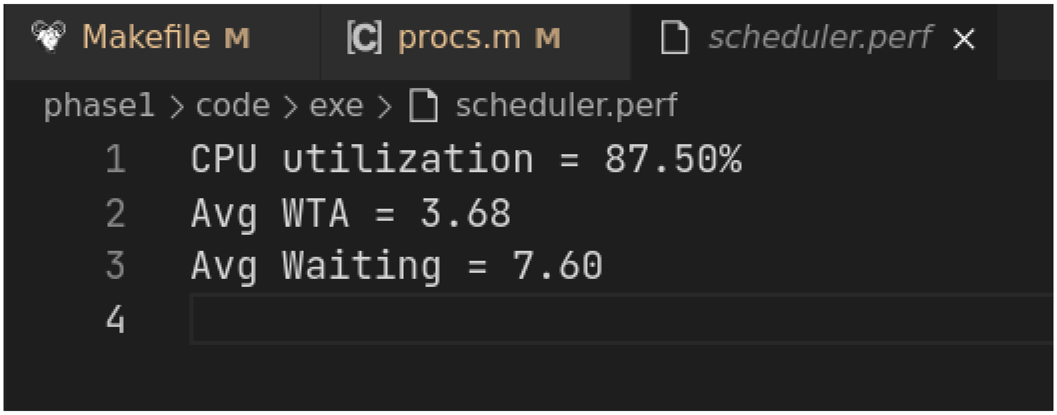
1. First Come First Serve (FCFS)


Makefile M
procs.m M
phase1 > code > exe > procs.m
You, seconds ago | 1 author (You)

```
#id arrival runtime priority
1 4 5 8
2 6 6 1
3 6 4 7
4 7 2 2
5 7 3 9

```

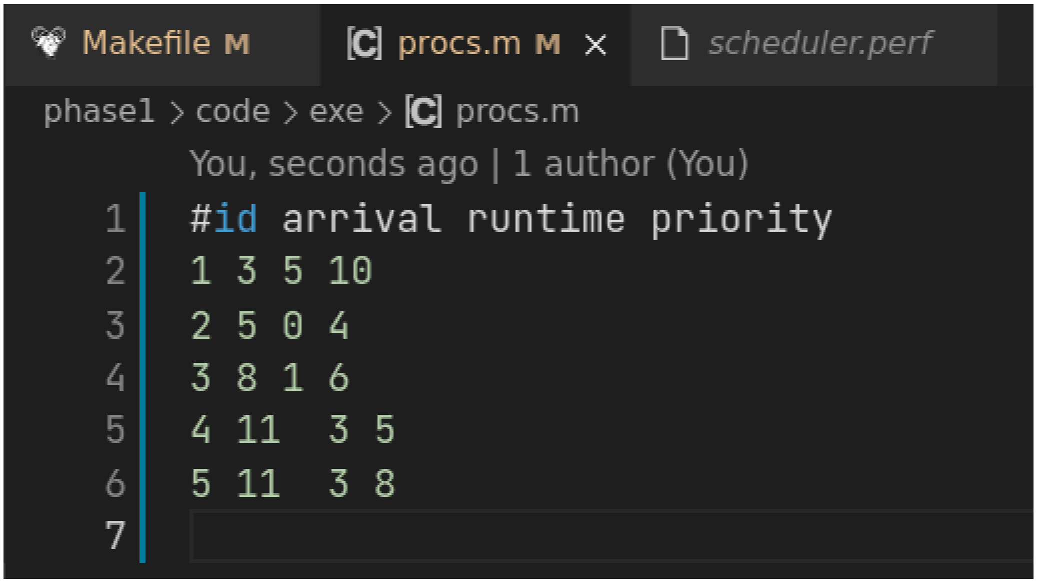


```
At time 4 process 1 started arr 4 total 5 remain 5  wait 0
At time 9 process 1 finished arr 4  total 5 remain 0  wait 0  TA 5  WTA 1.00
At time 9 process 2 started arr 6 total 6 remain 6  wait 3
At time 15  process 2 finished arr 6  total 6 remain 0  wait 3  TA 9  WTA 1.50
At time 15  process 3 started arr 6 total 4 remain 4  wait 9
At time 19  process 3 finished arr 6  total 4 remain 0  wait 9  TA 13 WTA 3.25
At time 19  process 4 started arr 7 total 2 remain 2  wait 12
At time 21  process 4 finished arr 7  total 2 remain 0  wait 12 TA 14 WTA 7.00
At time 21  process 5 started arr 7 total 3 remain 3  wait 14
At time 24  process 5 finished arr 7  total 3 remain 0  wait 14 TA 17 WTA 5.67

```

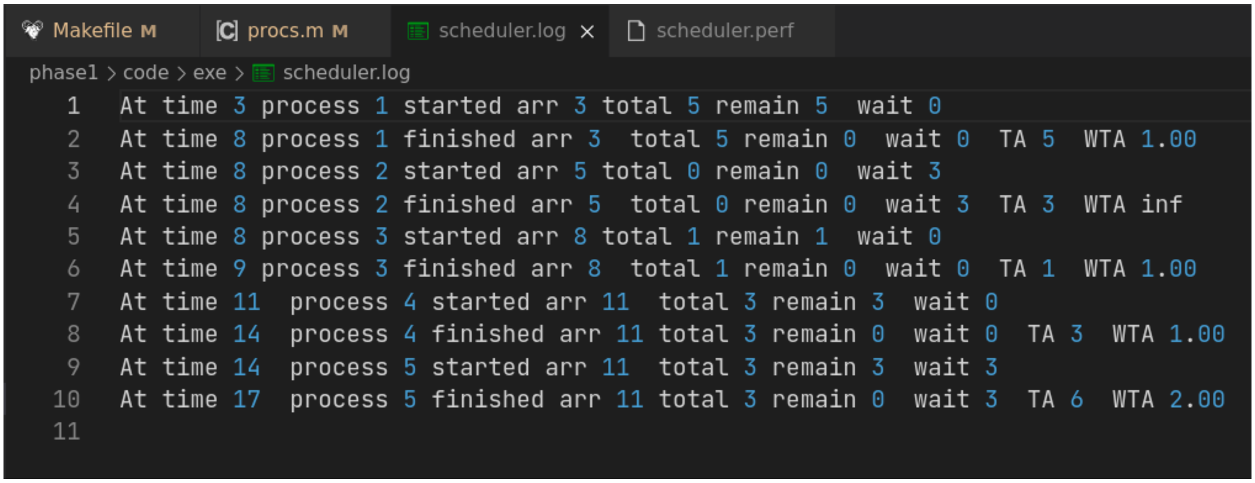


```
CPU utilization = 87.50%
Avg WTA = 3.68
Avg Waiting = 7.60

```

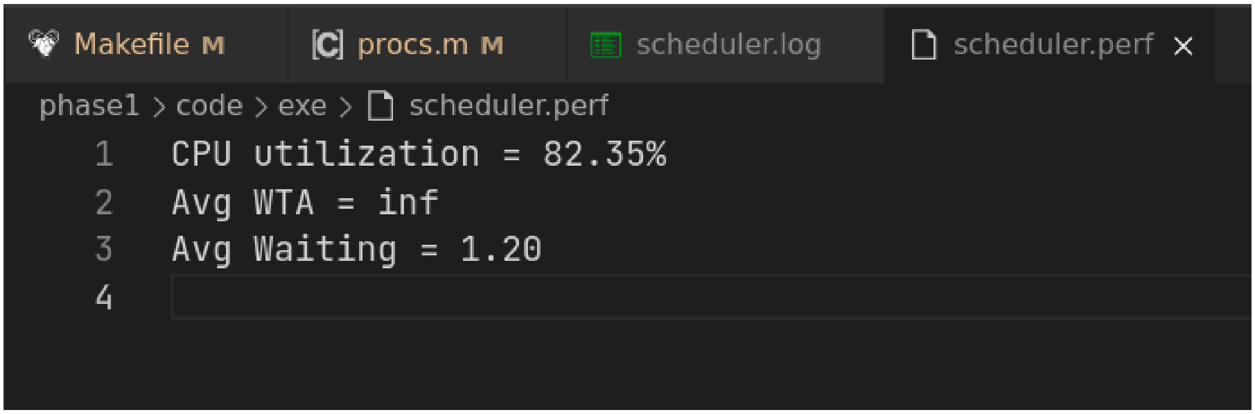
2. Shortest Job First (SJF)


Makefile M
procs.m M
scheduler.perf
phase1 > code > exe > procs.m
You, seconds ago | 1 author (You)

```
#id arrival runtime priority
1 3 5 10
2 5 0 4
3 8 1 6
4 11  3 5
5 11  3 8
```

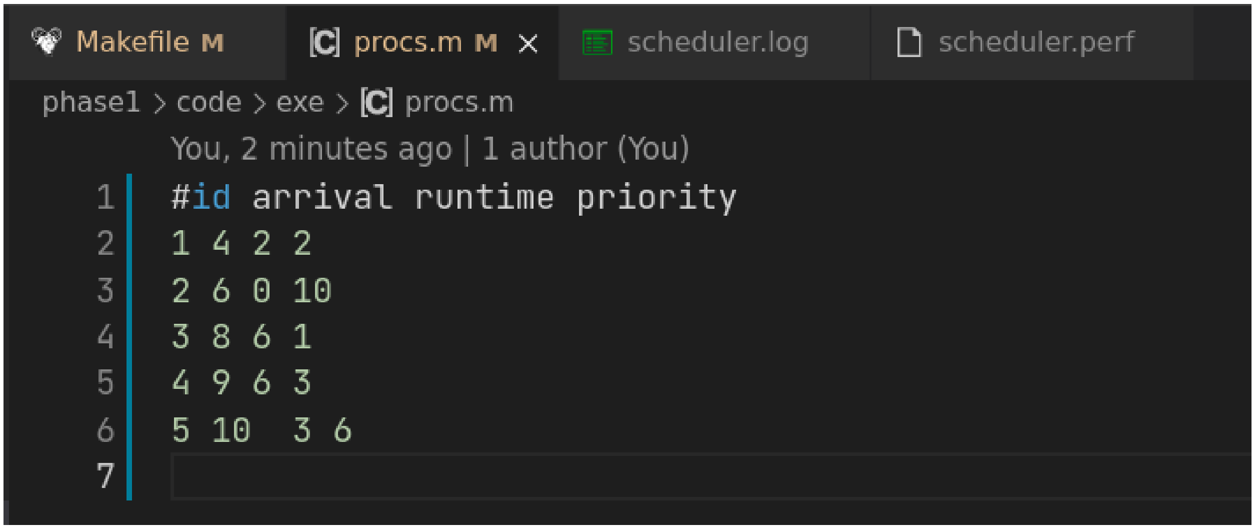


```
At time 3 process 1 started arr 3 total 5 remain 5  wait 0
At time 8 process 1 finished arr 3  total 5 remain 0  wait 0  TA 5  WTA 1.00
At time 8 process 2 started arr 5 total 0 remain 0  wait 3
At time 8 process 2 finished arr 5  total 0 remain 0  wait 3  TA 3  WTA inf
At time 8 process 3 started arr 8 total 1 remain 1  wait 0
At time 9 process 3 finished arr 8  total 1 remain 0  wait 0  TA 1  WTA 1.00
At time 11  process 4 started arr 11  total 3 remain 3  wait 0
At time 14  process 4 finished arr 11 total 3 remain 0  wait 0  TA 3  WTA 1.00
At time 14  process 5 started arr 11  total 3 remain 3  wait 3
At time 17  process 5 finished arr 11 total 3 remain 0  wait 3  TA 6  WTA 2.00
```



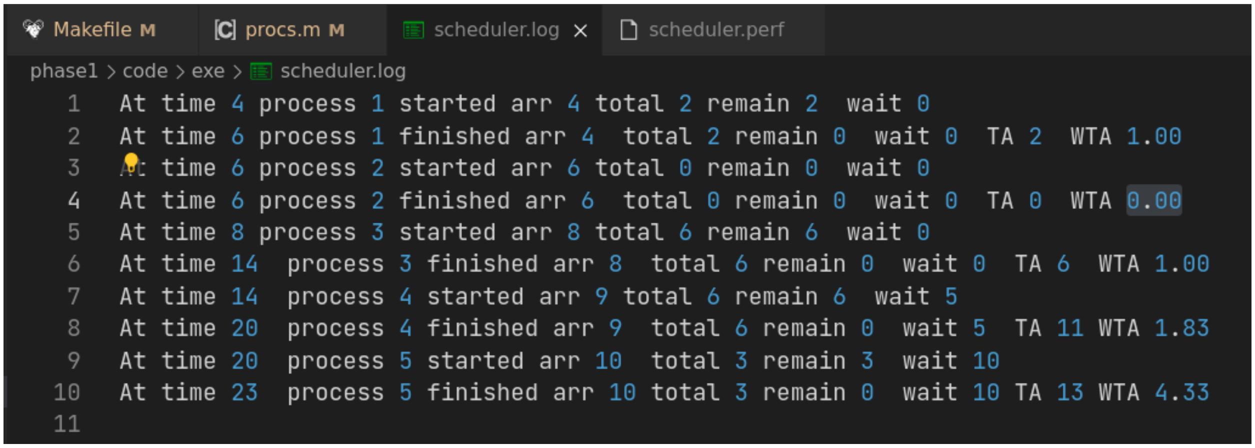
```
CPU utilization = 82.35%
Avg WTA = inf
Avg Waiting = 1.20
```

3. Preemptive Highest Priority First (HPF)

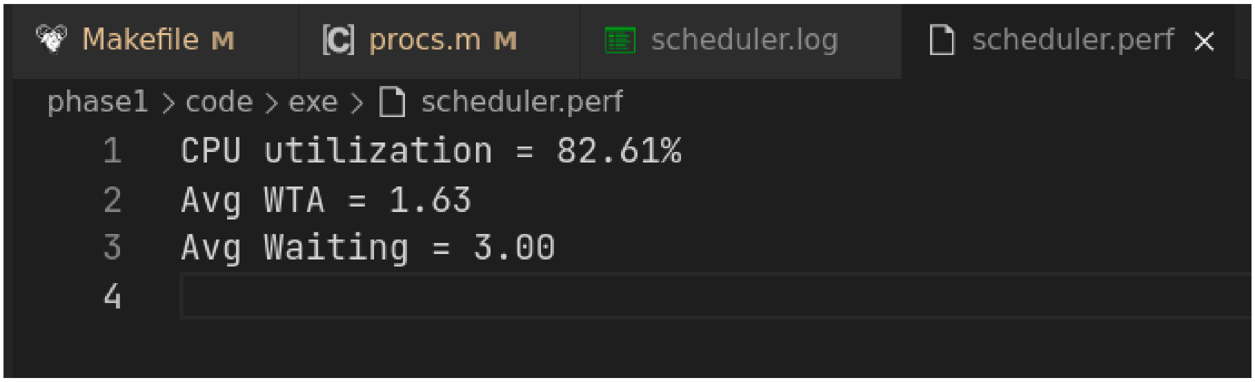

Makefile M
procs.m M
scheduler.log
scheduler.perf
phase1 > code > exe > procs.m
You, 2 minutes ago | 1 author (You)

```
#id arrival runtime priority
1 4 2 2
2 6 0 10
3 8 6 1
4 9 6 3
5 10  3 6

```

Note that there is a process with runtime=0 this time, it executes instantaneously but when calculating the WTA, it is considered as if its runtime was 1 clock cycle.

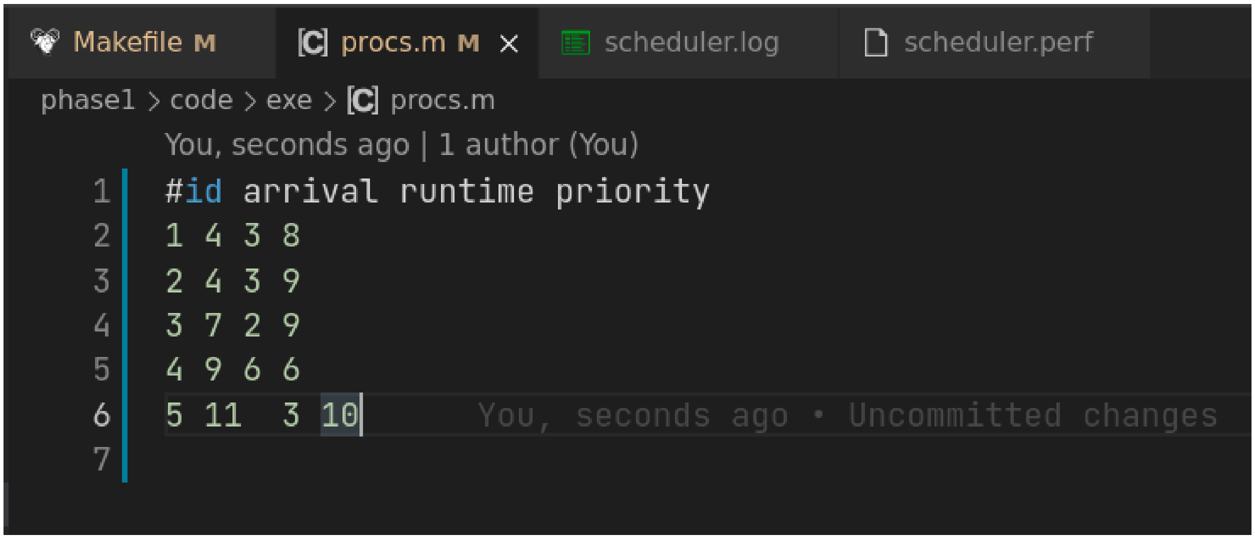


```
At time 4 process 1 started arr 4 total 2 remain 2  wait 0
At time 6 process 1 finished arr 4  total 2 remain 0  wait 0  TA 2  WTA 1.00
At time 6 process 2 started arr 6 total 0 remain 0  wait 0
At time 6 process 2 finished arr 6  total 0 remain 0  wait 0  TA 0  WTA 0.00
At time 8 process 3 started arr 8 total 6 remain 6  wait 0
At time 14  process 3 finished arr 8  total 6 remain 0  wait 0  TA 6  WTA 1.00
At time 14  process 4 started arr 9 total 6 remain 6  wait 5
At time 20  process 4 finished arr 9  total 6 remain 0  wait 5  TA 11 WTA 1.83
At time 20  process 5 started arr 10  total 3 remain 3  wait 10
At time 23  process 5 finished arr 10 total 3 remain 0  wait 10 TA 13 WTA 4.33

```

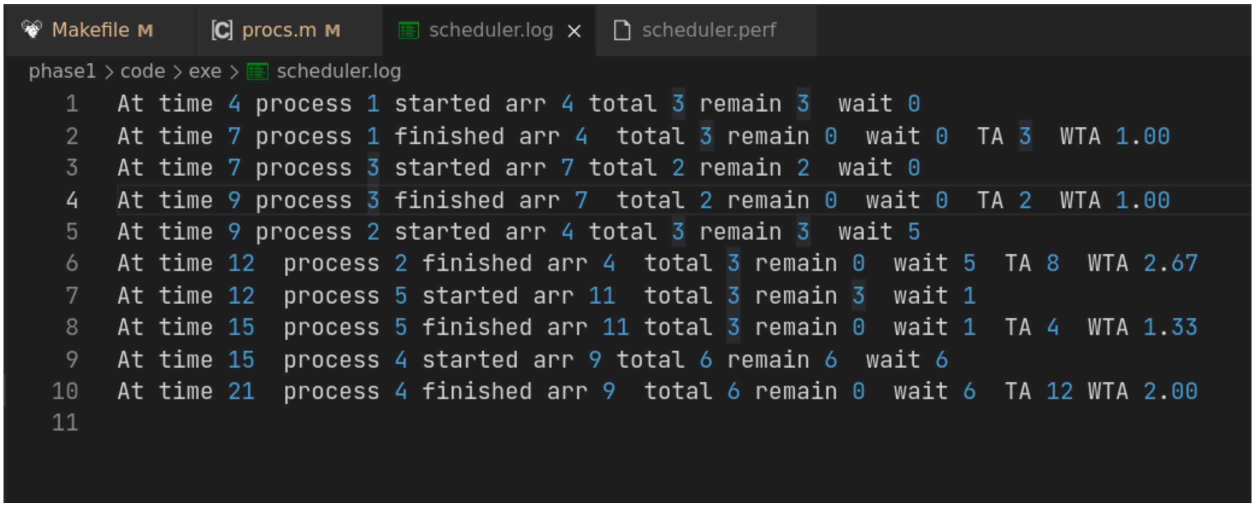


```
CPU utilization = 82.61%
Avg WTA = 1.63
Avg Waiting = 3.00

```

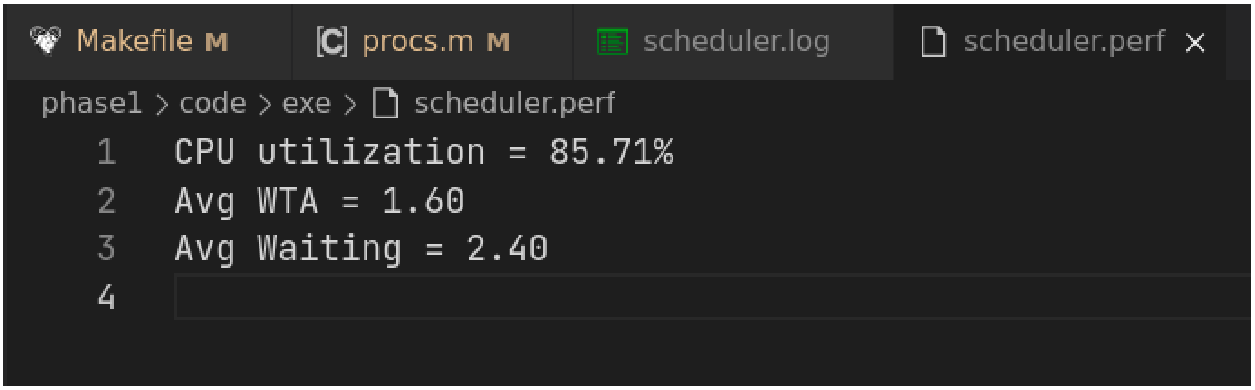
4. Shortest Remaining Time Next (SRTN)


Makefile M
procs.m M
scheduler.log
scheduler.perf
phase1 > code > exe > procs.m
You, seconds ago | 1 author (You)
You, seconds ago • Uncommitted changes

```
#id arrival runtime priority
1 4 3 8
2 4 3 9
3 7 2 9
4 9 6 6
5 11  3 10
```

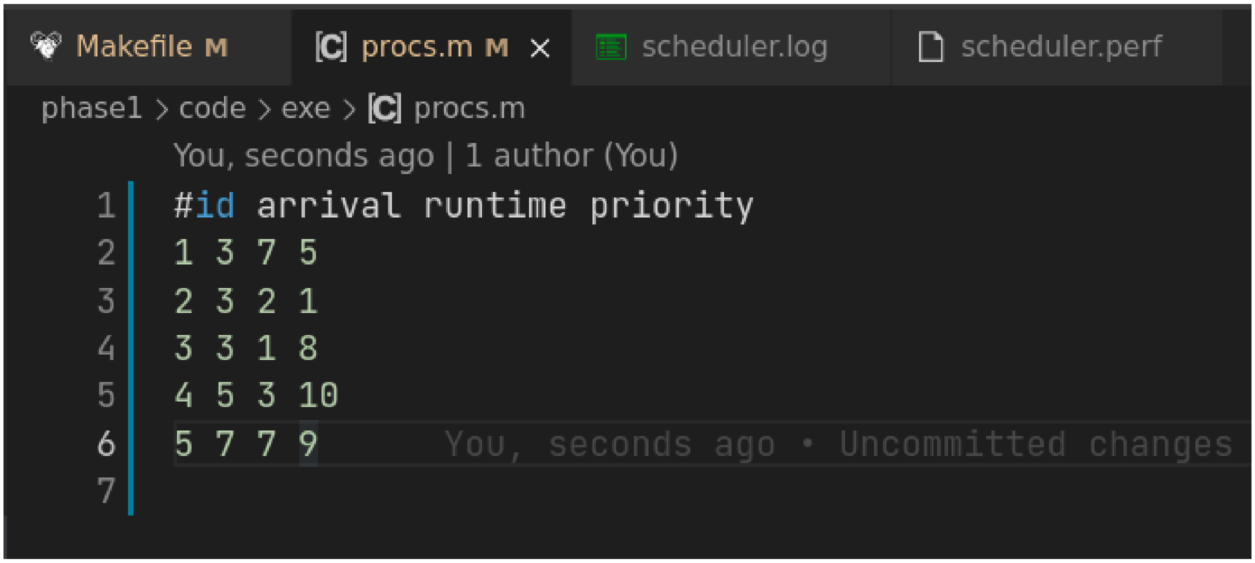


```
At time 4 process 1 started arr 4 total 3 remain 3  wait 0
At time 7 process 1 finished arr 4  total 3 remain 0  wait 0  TA 3  WTA 1.00
At time 7 process 3 started arr 7 total 2 remain 2  wait 0
At time 9 process 3 finished arr 7  total 2 remain 0  wait 0  TA 2  WTA 1.00
At time 9 process 2 started arr 4 total 3 remain 3  wait 5
At time 12  process 2 finished arr 4  total 3 remain 0  wait 5  TA 8  WTA 2.67
At time 12  process 5 started arr 11  total 3 remain 3  wait 1
At time 15  process 5 finished arr 11 total 3 remain 0  wait 1  TA 4  WTA 1.33
At time 15  process 4 started arr 9 total 6 remain 6  wait 6
At time 21  process 4 finished arr 9  total 6 remain 0  wait 6  TA 12 WTA 2.00
```

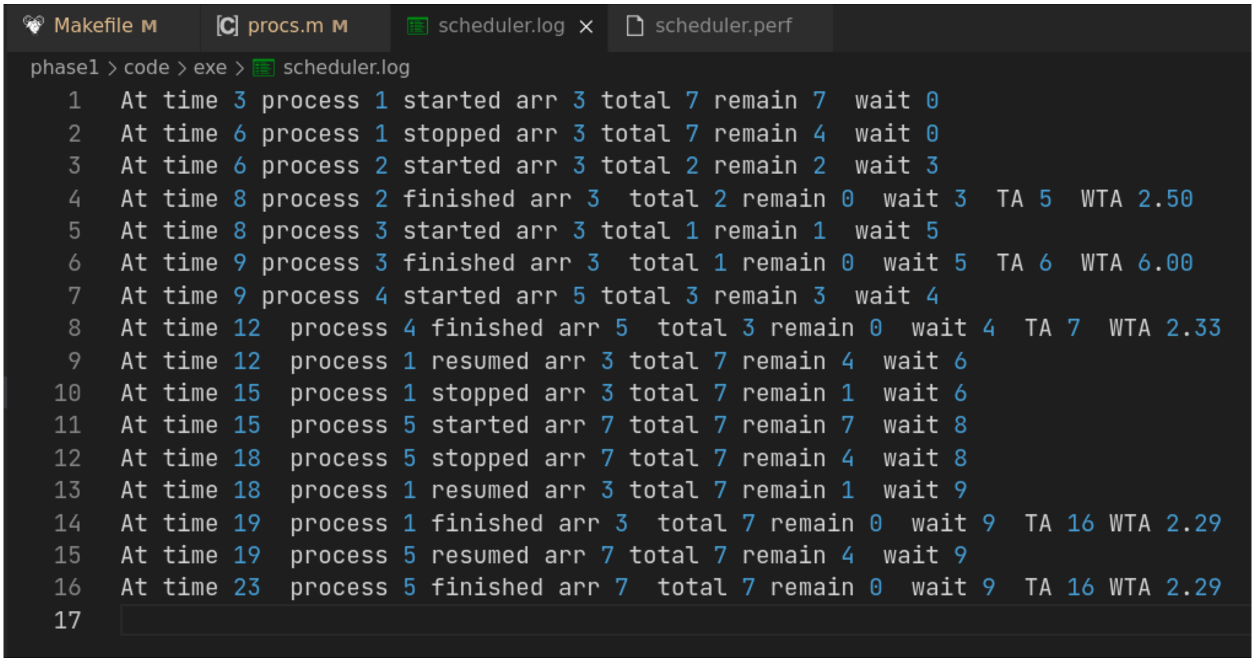


```
CPU utilization = 85.71%
Avg WTA = 1.60
Avg Waiting = 2.40
```

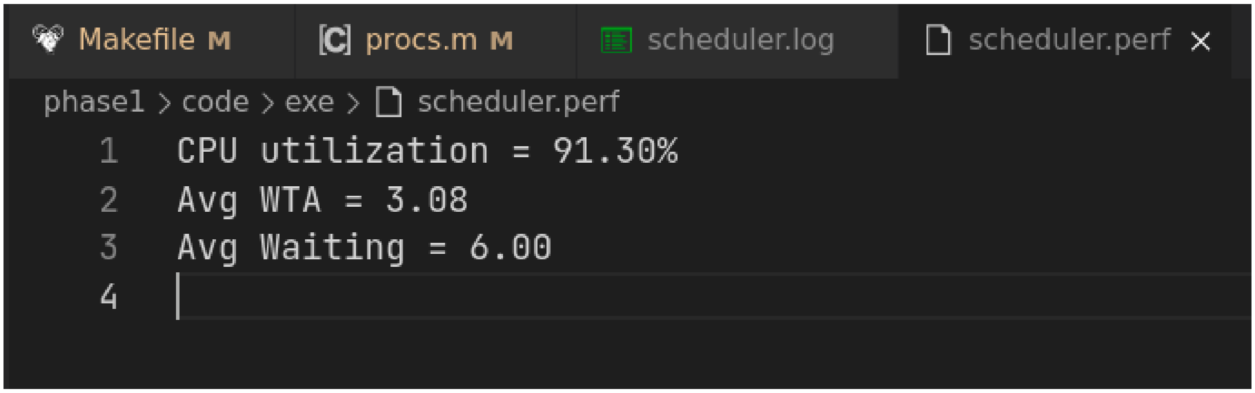
5. Round Robin (RR)


Makefile M
procs.m M ×
scheduler.log
scheduler.perf
phase1 > code > exe > procs.m
You, seconds ago | 1 author (You)
You, seconds ago • Uncommitted changes

```
#id arrival runtime priority
1 3 7 5
2 3 2 1
3 3 1 8
4 5 3 10
5 7 7 9
```



```
At time 3 process 1 started arr 3 total 7 remain 7  wait 0
At time 6 process 1 stopped arr 3 total 7 remain 4  wait 0
At time 6 process 2 started arr 3 total 2 remain 2  wait 3
At time 8 process 2 finished arr 3  total 2 remain 0  wait 3  TA 5  WTA 2.50
At time 8 process 3 started arr 3 total 1 remain 1  wait 5
At time 9 process 3 finished arr 3  total 1 remain 0  wait 5  TA 6  WTA 6.00
At time 9 process 4 started arr 5 total 3 remain 3  wait 4
At time 12  process 4 finished arr 5  total 3 remain 0  wait 4  TA 7  WTA 2.33
At time 12  process 1 resumed arr 3 total 7 remain 4  wait 6
At time 15  process 1 stopped arr 3 total 7 remain 1  wait 6
At time 15  process 5 started arr 7 total 7 remain 7  wait 8
At time 18  process 5 stopped arr 7 total 7 remain 4  wait 8
At time 18  process 1 resumed arr 3 total 7 remain 1  wait 9
At time 19  process 1 finished arr 3  total 7 remain 0  wait 9  TA 16 WTA 2.29
At time 19  process 5 resumed arr 7 total 7 remain 4  wait 9
At time 23  process 5 finished arr 7  total 7 remain 0  wait 9  TA 16 WTA 2.29
```



```
CPU utilization = 91.30%
Avg WTA = 3.08
Avg Waiting = 6.00
```