Urszula Chmielewska

300167

EOPSY

Lab 3

Scheduling

INTRODUCTION

The aim of this task was to become familiarized with CPU scheduling. In our task processes were supposed to be run in an average of 2000 milliseconds with a standard deviation of zero. They were supposed to be blocked every 500 milliseconds. The whole simulation is run for 10000 milliseconds. I have tested the simulation for 2, 5 and for 10 processes.

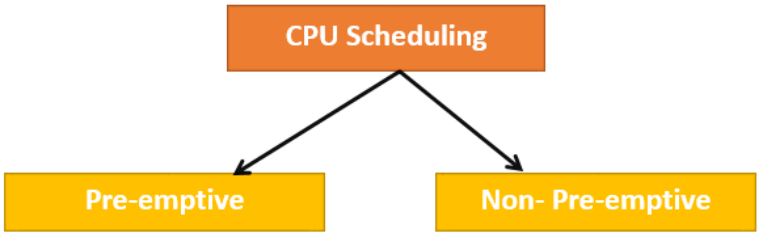
Scheduling type: Non-Preemptive

Scheduling name: First-Come First-Served

Theoretical part

CPU Scheduling is a process which determines which process will be executed while another process is blocked. Operating System choses at least one of the process available to be executed, while CPU remains idle.

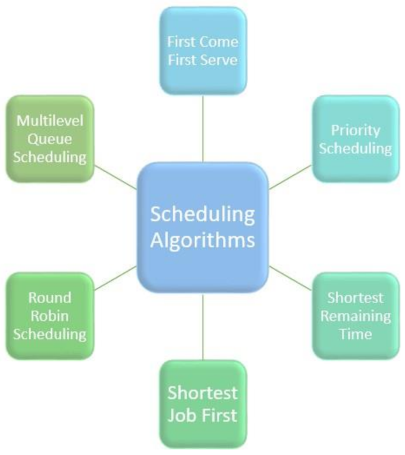
CPU Scheduling can be divided into two types:



Preemptive Scheduling – tasks are executed with their priorities. If there is a task with higher priority to be executed, while task with lower priority is still running, the lower priority task is hold for some time and resumes when the higher priority task has finished. Moreover, in preemptive scheduling process may change from running to ready state, or from waiting to ready state.

Non-Preemptive Scheduling – in this type of scheduling process, when CPU allocates resources to the given process, the process holds CPU till it get terminated or reach a waiting state. In this scheduling type, processes cannot be interrupted before its termination or before the time is up.

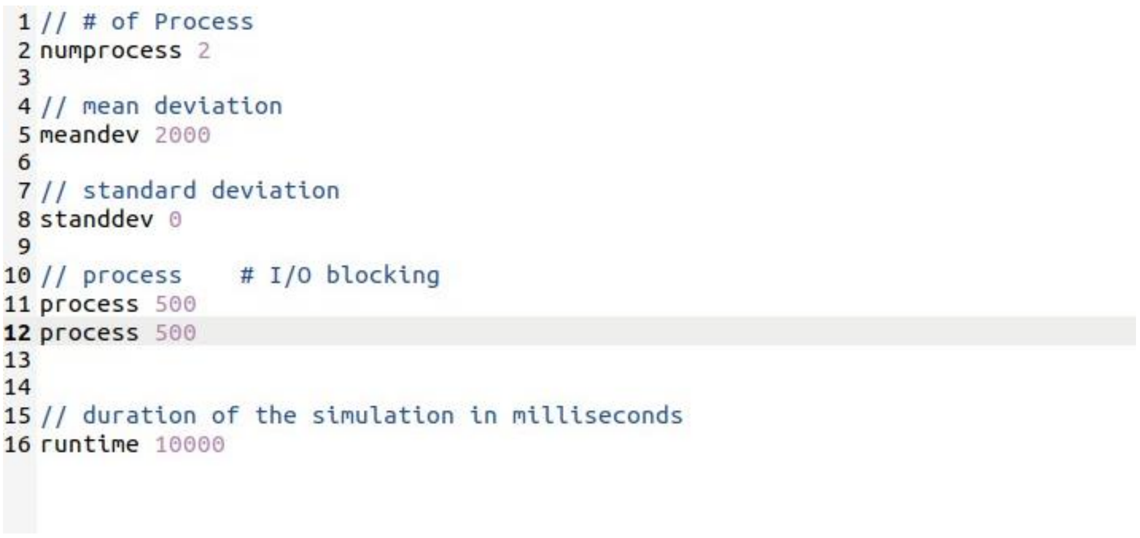
There are six typed of process scheduling algorithms:



In our task we use **First-Come First-Served algorithm**. It is not very efficient but it is the easiest CPU scheduling algorithm in use. It queues processes in the order that they arrive. Firstly it executes process which arrives first and next processes starts after its execution. The general wait time for this method is high.

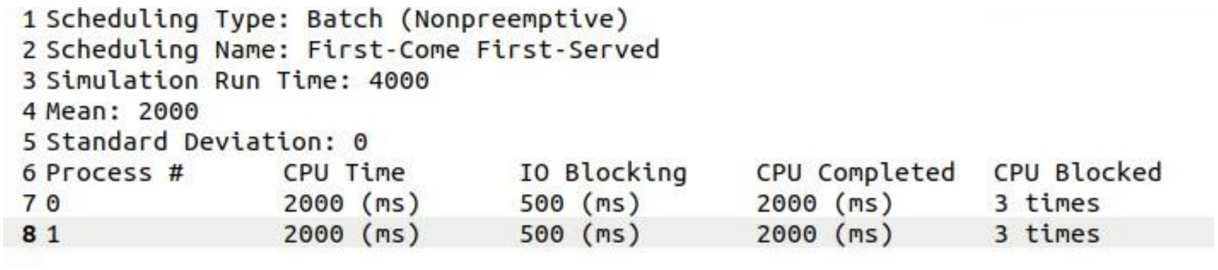
TWO PROCESSES

Configuration File



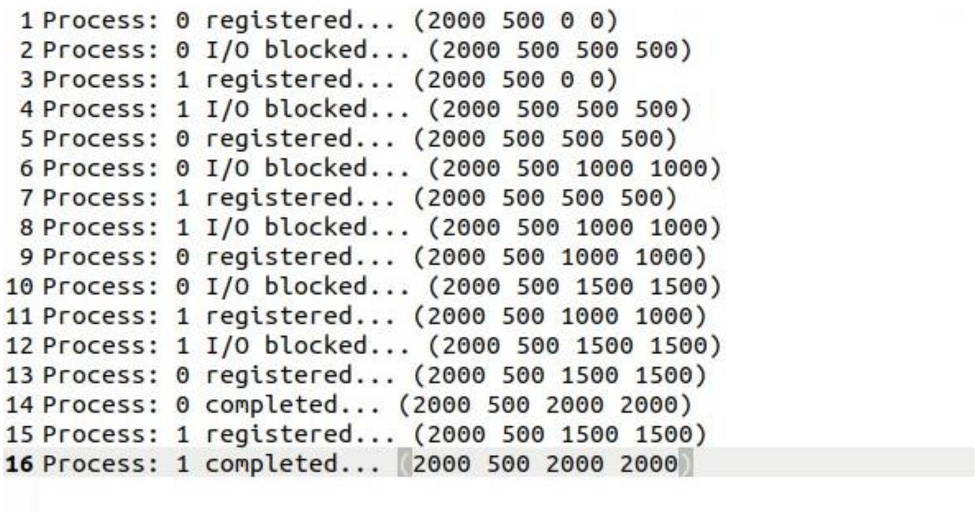
```
// # of Process
numprocess 2

// mean deviation
meandev 2000

// standard deviation
standdev 0

// process    # I/O blocking
process 500
process 500


// duration of the simulation in milliseconds
runtime 10000
```

Summary Result



```
Scheduling Type: Batch (Nonpreemptive)
Scheduling Name: First-Come First-Served
Simulation Run Time: 4000
Mean: 2000
Standard Deviation: 0
Process #         CPU Time        IO Blocking     CPU Completed    CPU Blocked
0                 2000 (ms)       500 (ms)        2000 (ms)        3 times
1                 2000 (ms)       500 (ms)        2000 (ms)        3 times
```

Summary Processes



```
Process: 0 registered... (2000 500 0 0)
Process: 0 I/O blocked... (2000 500 500 500)
Process: 1 registered... (2000 500 0 0)
Process: 1 I/O blocked... (2000 500 500 500)
Process: 0 registered... (2000 500 500 500)
Process: 0 I/O blocked... (2000 500 1000 1000)
Process: 1 registered... (2000 500 500 500)
Process: 1 I/O blocked... (2000 500 1000 1000)
Process: 0 registered... (2000 500 1000 1000)
Process: 0 I/O blocked... (2000 500 1500 1500)
Process: 1 registered... (2000 500 1000 1000)
Process: 1 I/O blocked... (2000 500 1500 1500)
Process: 0 registered... (2000 500 1500 1500)
Process: 0 completed... (2000 500 2000 2000)
Process: 1 registered... (2000 500 1500 1500)
Process: 1 completed... (2000 500 2000 2000)
```

Conclusions

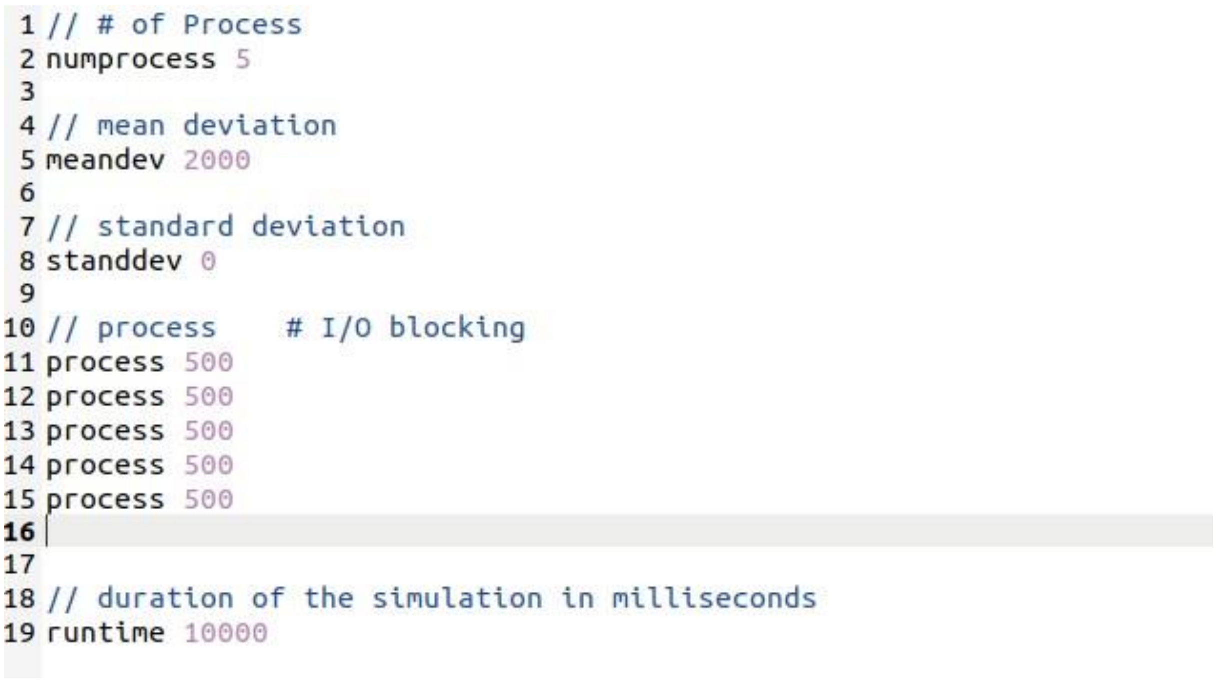
From the summary result file we can see that 2 processes were created as expected. Each process runtime was set for 2000 ms and took exactly 2000 ms. Each process was blocked 3 times after every 500 ms. Processes can only happen one at a time so when one is being executed, the other one is blocked. The lifetime of a single process looks like that:

1. process is registered and works for 500 ms
2. process is blocked by I/O
3. *another process operates*
4. come back to the first process and work for 500 ms
5. process is blocked by I/O again
6. *another process operates*
7. come back to the first process and work for 500 ms
8. process is blocked by I/O again
9. *another process operates*
10. come back to the first process and work for 500 ms and completing its operation

To sum up, in case with 2 processes, the entire runtime simulation took only 4000 ms, due to the fact that single process lasts for 2000 ms. The simulation did not reach, set to 10000 ms, duration of simulation time.

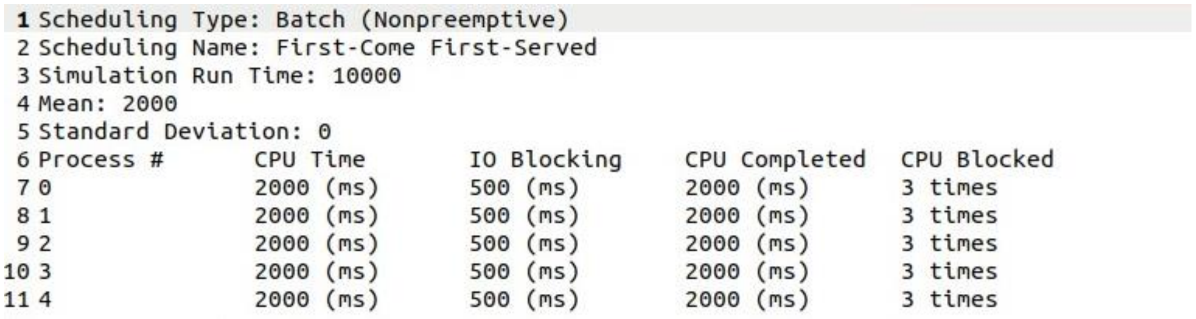
FIVE PROCESSES

Configuration File



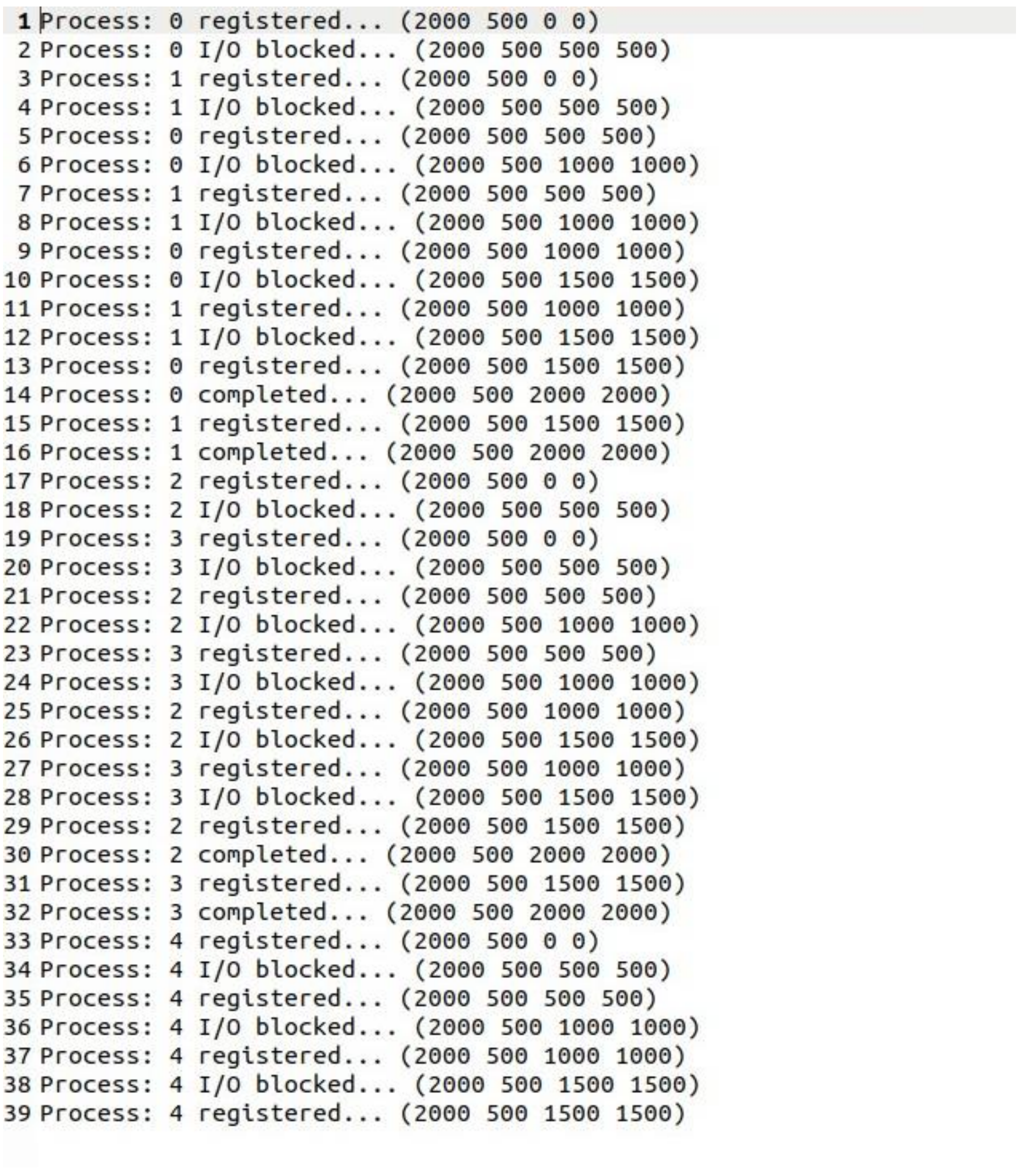
```
// # of Process
numprocess 5

// mean deviation
meandev 2000

// standard deviation
standdev 0

// process    # I/O blocking
process 500
process 500
process 500
process 500
process 500


// duration of the simulation in milliseconds
runtime 10000
```

Summary Result



```
Scheduling Type: Batch (Nonpreemptive)
Scheduling Name: First-Come First-Served
Simulation Run Time: 10000
Mean: 2000
Standard Deviation: 0
Process #       CPU Time        IO Blocking     CPU Completed   CPU Blocked
0               2000 (ms)       500 (ms)        2000 (ms)       3 times
1               2000 (ms)       500 (ms)        2000 (ms)       3 times
2               2000 (ms)       500 (ms)        2000 (ms)       3 times
3               2000 (ms)       500 (ms)        2000 (ms)       3 times
4               2000 (ms)       500 (ms)        2000 (ms)       3 times
```

Summary Processes



```
Process: 0 registered... (2000 500 0 0)
Process: 0 I/O blocked... (2000 500 500 500)
Process: 1 registered... (2000 500 0 0)
Process: 1 I/O blocked... (2000 500 500 500)
Process: 0 registered... (2000 500 500 500)
Process: 0 I/O blocked... (2000 500 1000 1000)
Process: 1 registered... (2000 500 500 500)
Process: 1 I/O blocked... (2000 500 1000 1000)
Process: 0 registered... (2000 500 1000 1000)
Process: 0 I/O blocked... (2000 500 1500 1500)
Process: 1 registered... (2000 500 1000 1000)
Process: 1 I/O blocked... (2000 500 1500 1500)
Process: 0 registered... (2000 500 1500 1500)
Process: 0 completed... (2000 500 2000 2000)
Process: 1 registered... (2000 500 1500 1500)
Process: 1 completed... (2000 500 2000 2000)
Process: 2 registered... (2000 500 0 0)
Process: 2 I/O blocked... (2000 500 500 500)
Process: 3 registered... (2000 500 0 0)
Process: 3 I/O blocked... (2000 500 500 500)
Process: 2 registered... (2000 500 500 500)
Process: 2 I/O blocked... (2000 500 1000 1000)
Process: 3 registered... (2000 500 500 500)
Process: 3 I/O blocked... (2000 500 1000 1000)
Process: 2 registered... (2000 500 1000 1000)
Process: 2 I/O blocked... (2000 500 1500 1500)
Process: 3 registered... (2000 500 1000 1000)
Process: 3 I/O blocked... (2000 500 1500 1500)
Process: 2 registered... (2000 500 1500 1500)
Process: 2 completed... (2000 500 2000 2000)
Process: 3 registered... (2000 500 1500 1500)
Process: 3 completed... (2000 500 2000 2000)
Process: 4 registered... (2000 500 0 0)
Process: 4 I/O blocked... (2000 500 500 500)
Process: 4 registered... (2000 500 500 500)
Process: 4 I/O blocked... (2000 500 1000 1000)
Process: 4 registered... (2000 500 1000 1000)
Process: 4 I/O blocked... (2000 500 1500 1500)
Process: 4 registered... (2000 500 1500 1500)
```

Conclusions

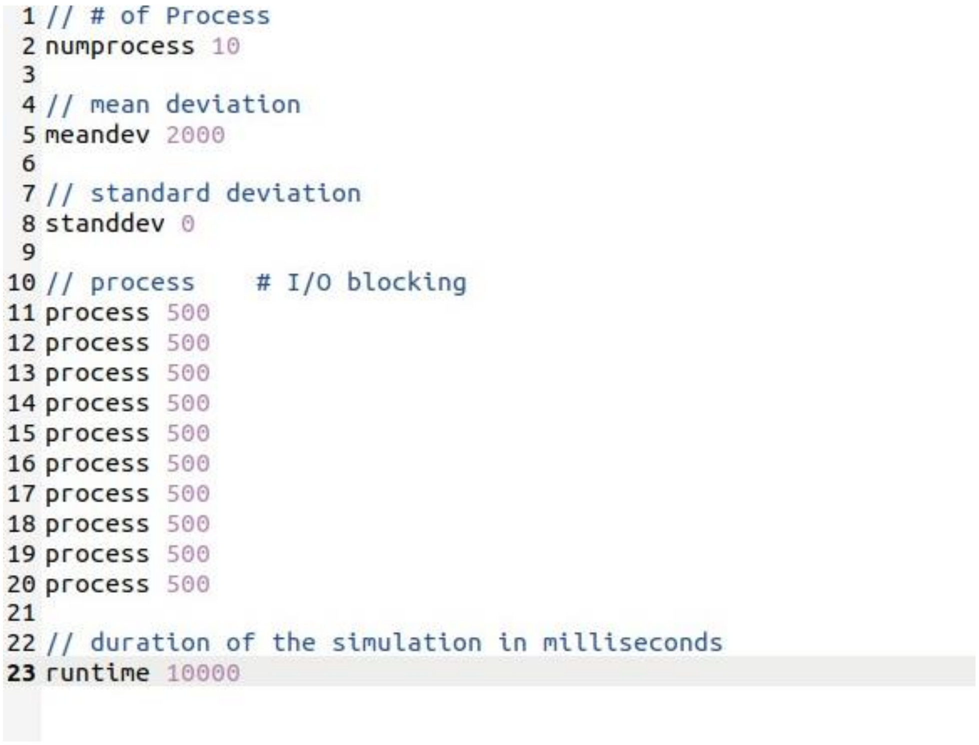
In this case the whole time of 5 processes is equal to exactly 10000 ms as the duration of simulation time was set. Each process took 2000 ms and it was blocked 3 times during simulation before executing.

Moreover in this case we can observe some characteristics resulting from used algorithm First-Come First-Served. This algorithm favors the first process that requested processor. After first I/O block processor turns its attention from process 0 to process 1. After block 1'st process it comes back to the process 0 instead of executing 2nd process. Second process is registered after 0'th and 1'st are completed.

Another interesting thing in this case is that despite of the fact that each process takes 2000 ms and 5 processes take 10000 ms time, on the summary process file we can see that program did not write message about completion process number 4. It was caused due to the fact that simulation reaches it runtime in the same time that the process 4 reaches the end.

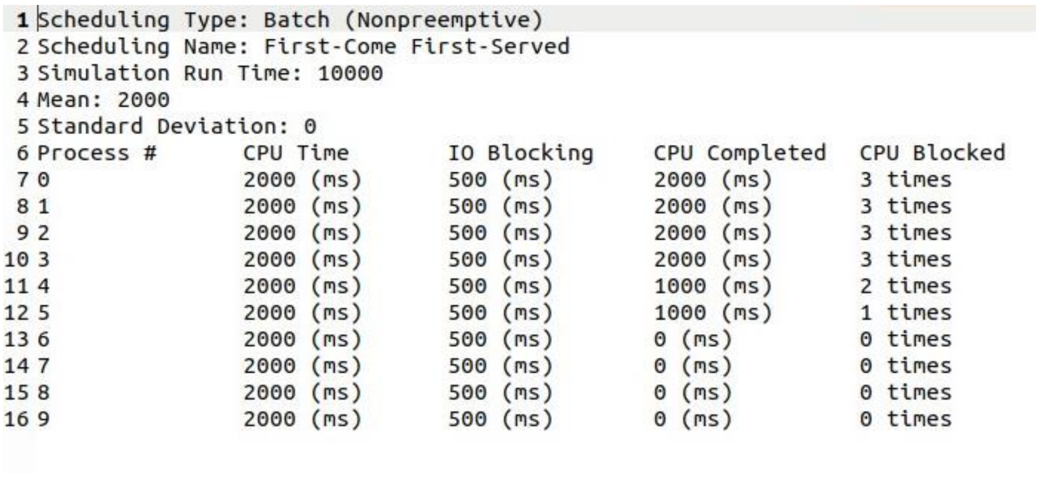
TEN PROCESSES

Configuration File



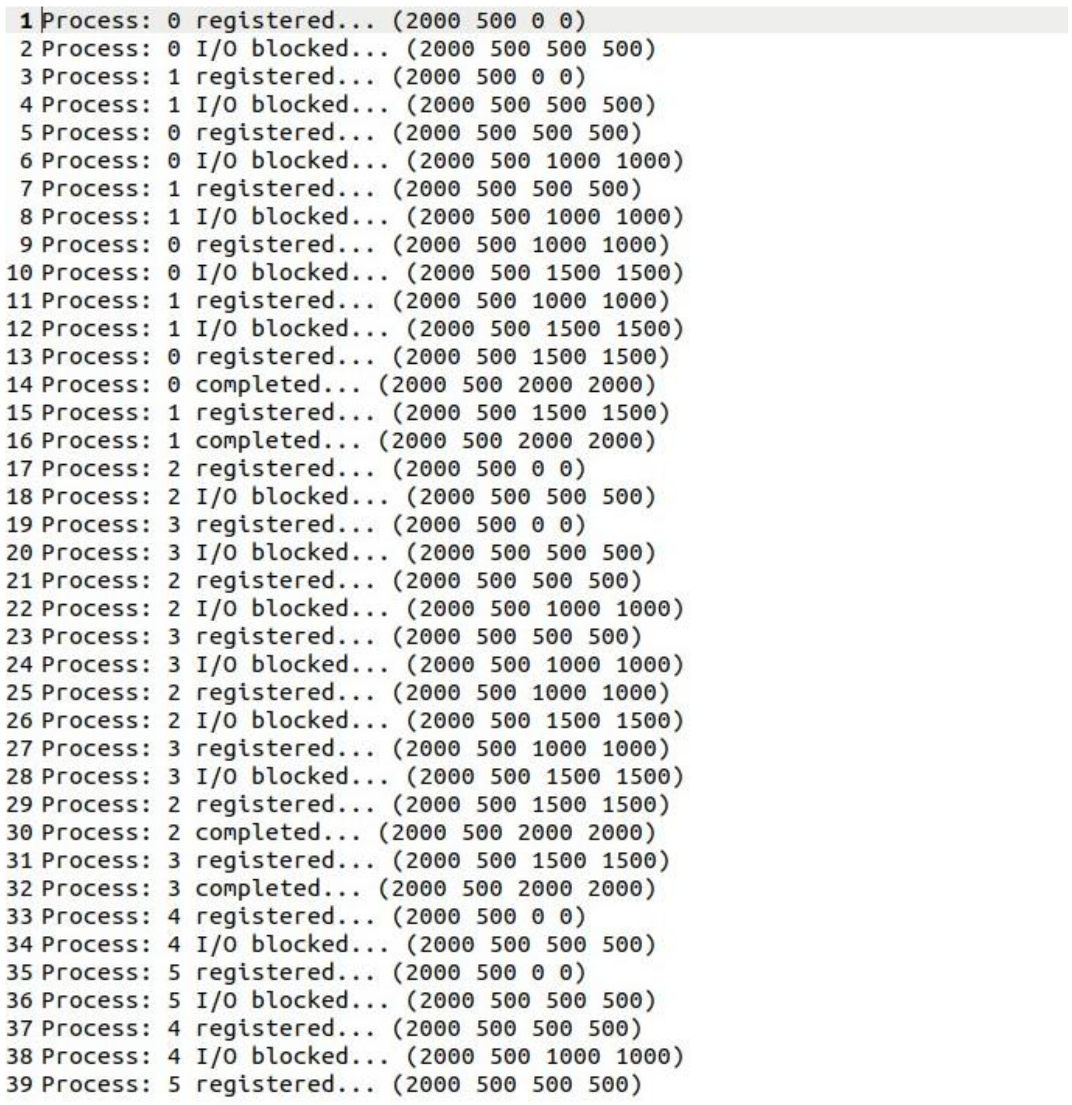
```
// # of Process
numprocess 10

// mean deviation
meandev 2000

// standard deviation
standdev 0

// process    # I/O blocking
process 500
process 500
process 500
process 500
process 500
process 500
process 500
process 500
process 500
process 500

// duration of the simulation in milliseconds
runtime 10000
```

Summary Result



```
Scheduling Type: Batch (Nonpreemptive)
Scheduling Name: First-Come First-Served
Simulation Run Time: 10000
Mean: 2000
Standard Deviation: 0
Process #       CPU Time        IO Blocking     CPU Completed   CPU Blocked
0               2000 (ms)       500 (ms)        2000 (ms)       3 times
1               2000 (ms)       500 (ms)        2000 (ms)       3 times
2               2000 (ms)       500 (ms)        2000 (ms)       3 times
3               2000 (ms)       500 (ms)        2000 (ms)       3 times
4               2000 (ms)       500 (ms)        1000 (ms)       2 times
5               2000 (ms)       500 (ms)        1000 (ms)       1 times
6               2000 (ms)       500 (ms)        0 (ms)          0 times
7               2000 (ms)       500 (ms)        0 (ms)          0 times
8               2000 (ms)       500 (ms)        0 (ms)          0 times
9               2000 (ms)       500 (ms)        0 (ms)          0 times
```

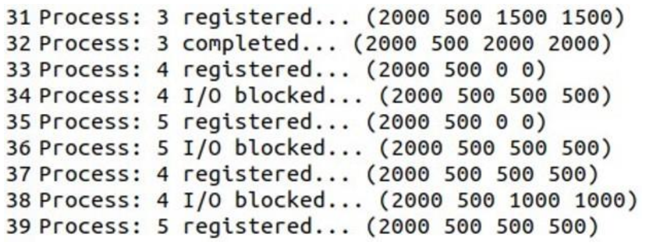
Summary Processes



```
Process: 0 registered... (2000 500 0 0)
Process: 0 I/O blocked... (2000 500 500 500)
Process: 1 registered... (2000 500 0 0)
Process: 1 I/O blocked... (2000 500 500 500)
Process: 0 registered... (2000 500 500 500)
Process: 0 I/O blocked... (2000 500 1000 1000)
Process: 1 registered... (2000 500 500 500)
Process: 1 I/O blocked... (2000 500 1000 1000)
Process: 0 registered... (2000 500 1000 1000)
Process: 0 I/O blocked... (2000 500 1500 1500)
Process: 1 registered... (2000 500 1000 1000)
Process: 1 I/O blocked... (2000 500 1500 1500)
Process: 0 registered... (2000 500 1500 1500)
Process: 0 completed... (2000 500 2000 2000)
Process: 1 registered... (2000 500 1500 1500)
Process: 1 completed... (2000 500 2000 2000)
Process: 2 registered... (2000 500 0 0)
Process: 2 I/O blocked... (2000 500 500 500)
Process: 3 registered... (2000 500 0 0)
Process: 3 I/O blocked... (2000 500 500 500)
Process: 2 registered... (2000 500 500 500)
Process: 2 I/O blocked... (2000 500 1000 1000)
Process: 3 registered... (2000 500 500 500)
Process: 3 I/O blocked... (2000 500 1000 1000)
Process: 2 registered... (2000 500 1000 1000)
Process: 2 I/O blocked... (2000 500 1500 1500)
Process: 3 registered... (2000 500 1000 1000)
Process: 3 I/O blocked... (2000 500 1500 1500)
Process: 2 registered... (2000 500 1500 1500)
Process: 2 completed... (2000 500 2000 2000)
Process: 3 registered... (2000 500 1500 1500)
Process: 3 completed... (2000 500 2000 2000)
Process: 4 registered... (2000 500 0 0)
Process: 4 I/O blocked... (2000 500 500 500)
Process: 5 registered... (2000 500 0 0)
Process: 5 I/O blocked... (2000 500 500 500)
Process: 4 registered... (2000 500 500 500)
Process: 4 I/O blocked... (2000 500 1000 1000)
Process: 5 registered... (2000 500 500 500)
```

Conclusions

In this case, in summary result file we can see that processes 4 and 5 were distributed for 1000 ms each. It was caused by the fact that after completing process number 3, processor registered 4th process and after its first block it registered 5th process. It is visible precisely on the picture presented below:



```
Process: 3 registered... (2000 500 1500 1500)
Process: 3 completed... (2000 500 2000 2000)
Process: 4 registered... (2000 500 0 0)
Process: 4 I/O blocked... (2000 500 500 500)
Process: 5 registered... (2000 500 0 0)
Process: 5 I/O blocked... (2000 500 500 500)
Process: 4 registered... (2000 500 500 500)
Process: 4 I/O blocked... (2000 500 1000 1000)
Process: 5 registered... (2000 500 500 500)
```

Despite of the fact that processes 4 and 5 were distributed for 1000 ms each, process 4 was blocked 2 times while 5th process only 1 time. The reason is that processor received one more I/O block for the 4th one, before running out of time.

Processes 6,7,8 and 9 were never reached. Due to this fact they have values 0 in columns CPU Completed and CPU Blocked.