

Laboratory report

EOPSY LAB 3

Paweł Borsukiewicz

301106

Gitlab: <https://gitlab-stud.elka.pw.edu.pl/pborsuki/eopsy-lab3.git>

Table of contents

Task 1.....	2
Part 1. 2 processes.....	3
Part 2. 5 processes.....	4
Part 3. 10 processes	7

Task 1.

For the purpose of the exercise, following parameters were to be set:

```
// # 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
```

Figure 1. Initial configuration

It can be observed that the whole simulation is supposed to end after 10000 milliseconds, mean time of program execution is 2000 milliseconds with standard deviation equal to 0 which means that all processes should end exactly after 2000 milliseconds of execution.

Part 1. 2 processes

For this part of the exercise initial setup was used. Number of processes was set to 2 and they are blocked for input or output after 500 milliseconds of execution.

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	

Figure 2. Summary-Results output file

```
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)
```

Figure 3. Summary-Process output file

Unsurprisingly, as it can be observed on figure 2, program finished execution after 4000 millisecond (2 processes * 2000 milliseconds per process) and I/O was blocked 3 times per process. It makes sense as processes were stopped after 500, 1000 and 1500 milliseconds of their execution. There was no need to block them at 2000 milliseconds as it was the very time when they were completed. Analyzing figure 3, one may confirm those findings, but also

observe the pattern in which processes were executed. Firstly, process 0 was registered and it was executed for 500ms, then it was blocked and process 1 was registered and was allowed execution for 500ms. The routine of registering and blocking was looped 3 times. After 500ms after 4th registering of process 0, it was declared as completed (as 2000ms has passed) and process 1 was registered without blocking I/O from process 0 (it has already terminated). After 500ms process 1 was completed and the whole simulation was terminated as all processes have terminated.

Part 2. 5 processes

```
// # 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
```

Figure 4. Configuration for 5 processes

In this part of the task, number of processes was to be increased to 5.

```

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

Figure 5. Summary-Results output file for 5 processes

As it can be observed on figure 5, full simulation time was needed to execute all 5 processes ($5 \times 2000\text{ms} = 10000\text{ms}$). Similarly to the results of the previous subtask, each process was blocked 3 times, after each 500ms of execution.

Nonetheless, important patterns can be observed only on figure 6 presented below. From that figure we can learn that after 500ms I/O is blocked for process 0 and process 1 is executed. However, after another 500ms CPU time is not given to process 2, but to process 0 as we operate in scheduling system “First-Come First-Served”, which intends to finish the oldest process as fast as possible giving it CPU time whenever possible. For the same reason, after I/O block of process 0 it is process 1 that is given CPU time. Therefore, process 2 starts its execution after 4000ms of simulation start as until that time CPU time was used only by processes 0 and 1 in a form of the loop described in part 1 of this report. Then between 4000ms and 8000ms after program start the same loop is repeated, but for processes 2 and 3. At the end, when there is only one process left to execute it is provided all CPU time (after each I/O block process 4 regains its CPU time) until 10000ms timestamp when it should finish its execution, however the figure 6 does not show information that it was completed even though according to figure 5 it was provided 2000ms of CPU time. It is possible that the simulation end happens before the process end when they are terminated at the same time.

```

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)

```

Figure 6. Summary-Processes output file for 5 processes

Part 3. 10 processes

In this part of the task, number of processes was to be increased to 10.

```
// # 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
```

Figure 7. Configuration for 10 processes

Scheduling Type: Batch (Nonpreemptive)
Scheduling Name: First-Come First-Served
Simulation Run Time: 10000
Mean: 2000
Standard Deviation: 0

Process #	CPU Time	I/O 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

Figure 8. Summary-Results output file for 10 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)

```

Figure 9. Summary-Processes output file for 10 processes

Not surprisingly, as it can be observed on figure 8, not all processes were executed. 10s is not enough to executed 10 process, 2s each. What is interesting, is that there were 2 processes (4 and 5) that were only partially executed. It was due to the fact that our processes were executed in pairs in the very same way as in previous subtasks due to “First-Come First-Served” rule. Pairs were process 0 and process 1, process 2 and process 3, and so on. Each pair requires 4000ms to be fully executed with 3 I/O blocks per process. Therefore process 0-3 were fully executed after 8000ms. The remaining 2000ms of CPU time was divided between processes 4 and 5. Process 4 started at 8000ms timestamp and was blocked at 8500ms and then at 9500ms. Process 5 was only blocked at 9000ms as the simulation terminated after 10000ms. If simulation had more time, we would observe I/O blocked for process 5 at 10000ms timestamp. Following pair of processes – 6 and 7 would start operation at 12000ms timestamp with execution of process 6, however, it was never meant to happen as simulation ends after 10000ms. Similarly, due to the same reasons processes 8 and 9 were not even started.