# EOPSY LAB3

## Scheduling

*Michał Skarzyński 293054*

## TABLE OF CONTENTS

Task ..... 2

Package ..... 2

Output explained ..... 3

- The Summary-Results File ..... 3
- The Summary-Processes File ..... 4

Obtained results ..... 4

- Two processes ..... 4
  - Simulation parameters ..... 4
  - The Summary-Results File ..... 5
  - The Summary-Processes File ..... 5
- Five processes ..... 6
  - Simulation parameters ..... 6
  - The Summary-Results File ..... 6
  - The Summary-Processes File ..... 7
- Ten processes ..... 8
  - Simulation parameters ..... 8
  - The Summary-Results File ..... 8
  - The Summary-Processes File ..... 9

Results analysys ..... 10

- Simulation conditions ..... 10
- First case ..... 10
- Second case ..... 11
- Third case ..... 13

# TASK

Create a configuration file in which all processes run an average of 2000 milliseconds with a standard deviation of zero, and which are blocked for input or output every 500 milliseconds.

Run the simulation for 10000 milliseconds with 2 processes.

Examine the two output files.

Try again for 5 processes.

Try again for 10 processes.

Prepare a document with explanation what's happening and why.

# PACKAGE

The scheduling simulator illustrates the behavior of scheduling algorithms against a simulated mix of process loads. The user can specify the number of processes, the mean and standard deviation for compute time and I/O blocking time for each process, and the duration of the simulation. At the end of the simulation a statistical summary is presented.

# OUTPUT EXPLAINED

## The Summary-Results File

| Field | Description |
|---|---|
| Scheduling Type: | The type of the scheduling algorithm used. The value displayed is "hard coded" in the SchedulingAlgorithm.java file. |
| Scheduling Name: | The name of the scheduling algorithm used. The value displayed is "hard coded" in the SchedulingAlgorithm.java file. |
| Simulation Run Time: | The number of milliseconds that the simulation ran. This may be less than or equal to the total amount of time specified by the "runtime" configuration parameter. |
| Mean: | The average amount of runtime for the processes as specified by the "meandev" configuration parameter. |
| Standard Deviation: | The standard deviation from the average amount of runtime for the processes as specified by the "standdev" configuration parameter. |
| Process # | The process number assigned to the process by the simulator. The process number is between 0 and n-1, where n is the number specified by the "numprocess" configuration parameter. |
| CPU Time | The randomly generated total runtime for the process in milliseconds. This is determined by the "meandev" and "standdev" parameters in the configuration file. |
| IO Blocking | The amount of time the process runs before it blocks for input or output. This is specified for each process by a "process" directive in the configuration file. |
| CPU Completed | The amount of runtime in milliseconds completed for the process. Note that this may be less than the CPU Time for the process if the simulator runs out of time as specified by the "runtime" configuration parameter. |
| CPU Blocked | The number of times the process blocked for input or output during the simulation. |

## The Summary-Processes File

| Field | Description |
|---|---|
| *process-number* | The process number assigned to the process by the simulator. This is a number between 0 and n-1, where n is the value specified for the "numprocess" configuration parameter. |
| *process-status* | The status of the process at this point in time. If "registered" then the process is under consideration by the scheduling algorithm. If "I/O blocked", then the scheduling algorithm has noticed that the process is blocked for input or output. If "completed", then the scheduling algorithm has noticed that the process has met or exceeded its allocated execution time. |
| *cpu-time* | The total amount of run time allowed for this process. This number is randomly generated for the process based on the "meandev" and "standdev" values specified in the configuration file. |
| *block-time* | The amount of time in milliseconds to execute before blocking process. This number is specified for the process by the "process" directive in the configuration file. |
| *accumulated-time* | The total amount of time process has executed in milliseconds. (This number appears twice in the log file; one should be removed). |

# OBTAINED RESULTS

## Two processes

### SIMULATION PARAMETERS

```
// # of Process
numprocess 2

// mean deivation
meandev 2000

// standard deviation
standdev 0

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

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

## THE SUMMARY-RESULTS FILE

```
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
```

## THE SUMMARY-PROCESSES FILE

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

# Five processes

## SIMULATION PARAMETERS

```
// # of Process
numprocess 5

// mean deivation
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
```

## THE SUMMARY-RESULTS FILE

```
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
```

## THE SUMMARY-PROCESSES FILE

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

## Ten processes

### SIMULATION PARAMETERS

```
// # of Process
numprocess 10

// mean deivation
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
```

### THE SUMMARY-RESULTS FILE

```
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
```

## THE SUMMARY-PROCESSES FILE

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

# RESULTS ANALYSYS

In computing, scheduling is the method by which work is assigned to resources that complete the work. In our task this was the case of so called process scheduler. Which is a part of the operating system that decides which process runs at a certain point in time. It usually has the ability to pause a running process, move it to the back of the running queue and start a new process; such a scheduler is known as preemptive scheduler, otherwise it is a non-preemptive scheduler.

## Simulation conditions

- We simulated this behavior using MOSS Scheduling Simulator.
- All of the simulations was conducted using First-Come First-Served algorithm, in Batch (Non - preemptive) mode.
- First-Come First-Served algorithm is based on FIFO queue, which implies that jobs are executed on first come; first serve basis.
- Since it's working in Non – preemptive mode then once a process enters the running state, it cannot be preempted until it completes its allotted time.
- The simulation was bounded in the total length of 10000ms.
- All of the processes have the same 2000 ms of runtime with no deviation.

## First case

Let us look firstly on case which were involved only 2 processes.

In this case we created process 0 and 1. After looking into java file we can see that they are stored in Vector sorted by its id, therefore process 0 starts first. It executes for 500 ms, when I/O block happens, then process is switched to next one, in this case process 1, which then executes for 500ms and its blocked etc. Until all processes reach 2000ms runtime. Since it is basically sequential operation whole executions takes 4000ms.

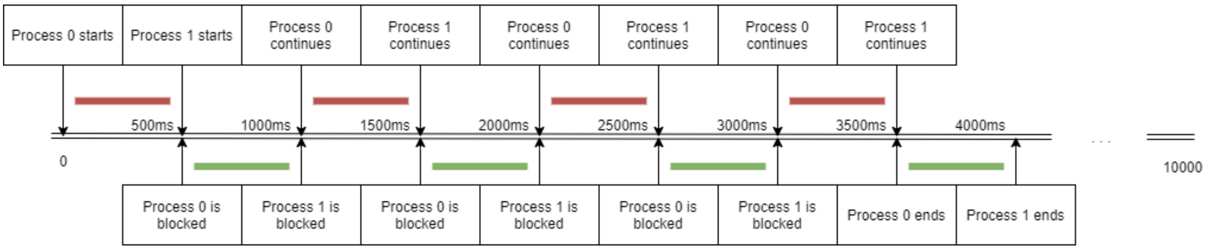

*Diagram showing execution of 2 processes.*

Red line is the time in which Process 0 is active.

Green line is the time in which Process 1 is active.

## Second case

This is very similar to first case but here the mechanism of changing processes must be taken into account.

As I said earlier this is a FIFO queue. And in code this is represented as this loop:

```
for (i = size - 1; i >= 0; i--) {
      process = (sProcess) processVector.elementAt(i);
      if (process.cpudone < process.cputime) {
        currentProcess = i;
      }
  }
```

Which basically means, that we will always take available process with the least position in processVector.

Therefore in this case when process 0 is blocked it takes the lowest possible process ID as its ancestor, which is process 1.

Then when process 1 is blocked, it switches execution to the lowest possible available process which is process 0.

And after process 0 is completed, then process 1, when its blocked must switch to process 2, then process 1 is completed, and this situation is repeated until process 2 and 3 are finished. Then control is given to process 4.

*One odd remark : In this simulator this 10000ms is time of **runtime**, so it does not count time when all processes are blocked.*

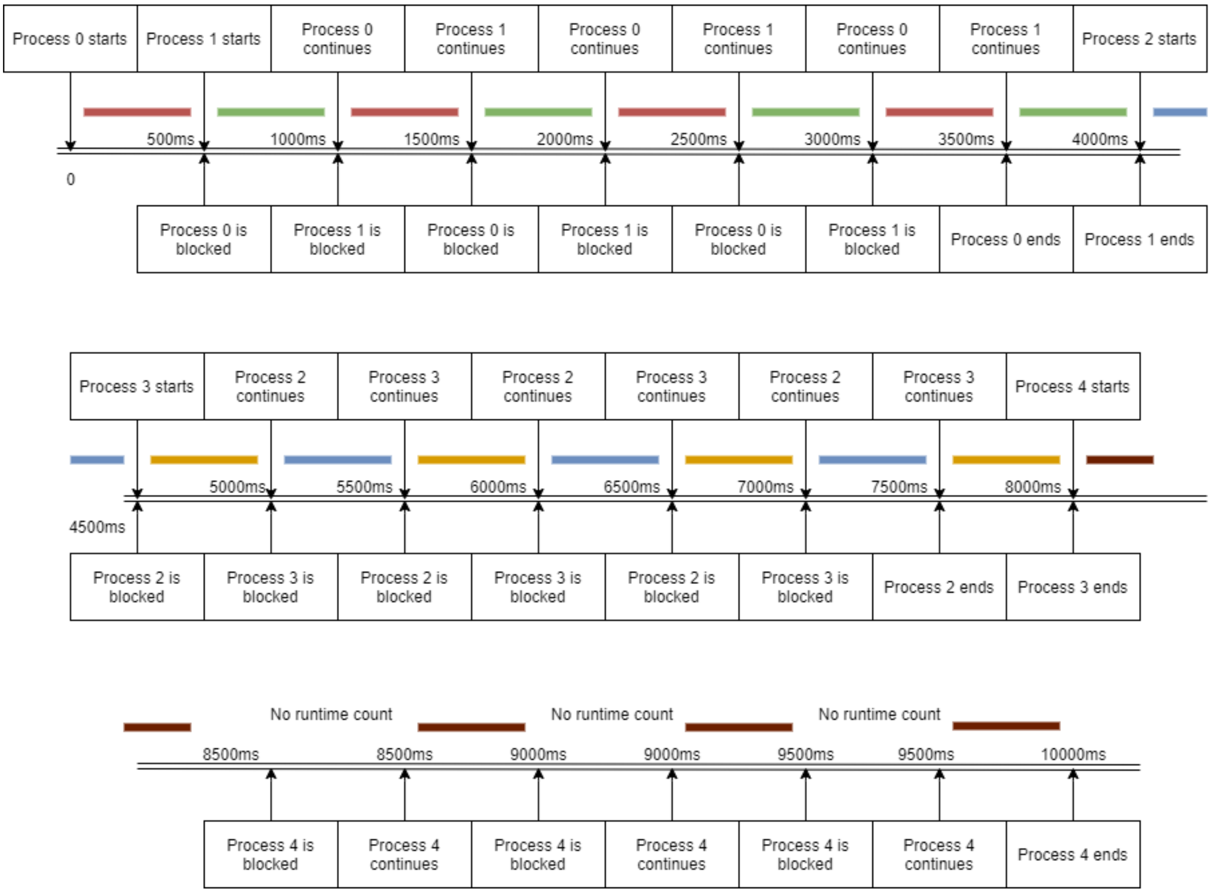

*Diagram showing execution of 5 processes.*

We can see, that when possible process which has lowest ID is chosen at each step.

This configuration uses whole available runtime, and all of the processes finish.

# Third case

In this case we will have the same situation as in case 2 up to 8000ms of runtime.

In second case we had those “blank” spots, now we can put some processes in it, namely process number 5. Unfortunately now we count execution of process 5 into global 10000ms simulation bond, therefore either of processes have enough resources to finish.

Process 4 and 5 are not completed, and processes through 6 up to 9 don’t even start.

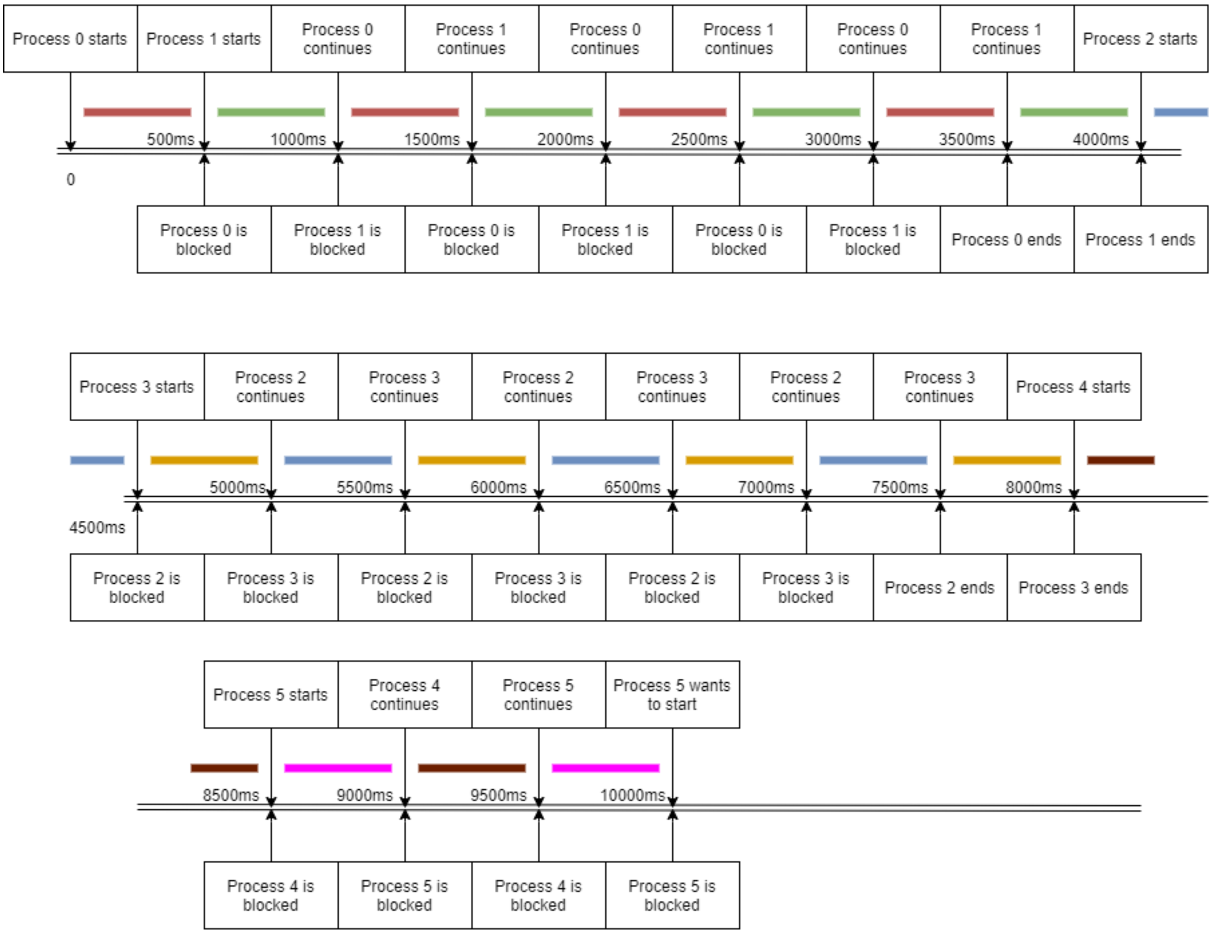

*Diagram showing execution of 10 processes.*