LAB #3 Report

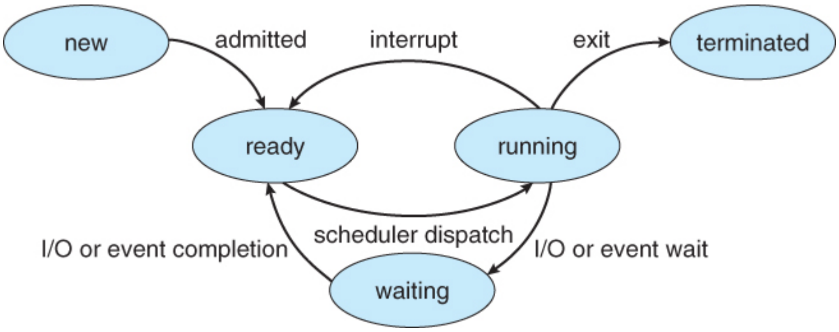
Dominik Kurasbediani

Theoretical Introduction:

To determine the order of execution and to execute a process itself, process states have been introduced:

1. ***New*** - This is the state when the process has just been created. It is the initial state in the process life cycle
2. ***Ready*** - In the ready state, the process is waiting to be assigned the processor by the short term scheduler, so it can run. This state is immediately after the new state for the process.
3. ***Running*** - The process is said to be in running state when the process instructions are being executed by the processor. This is done once the process is assigned to the processor using the short-term scheduler
4. ***Blocked -*** The process is in blocked state if it is waiting for some event to occur. This event may be I/O as the I/O events are executed in the main memory and don't require the processor. After the event is complete, the process again goes to ready state.
5. ***Terminated -*** The process is terminated once it finishes its execution. In the terminated state, the process is removed from main memory and its process control block is also deleted.

They are presented in this the following process lifecycle:



Process lifecycle, credit

Processes' execution is managed by scheduling. There are multiples algorithms but in this task we focus on a non-preemptive algorithm called First Come First Serve. Nonpreemptive means that once a process enters its running state, it cannot be pre-empted until it completes its allotted time. In the FCFS algorithm, as it implies in the name, jobs are prioritised by their respective position in the queue. Even though this algorithm is easy to understand and implement, its performance is far from ideal since the average wait time is high.

Tasks

1. Simulation for 2 processes:

Configuration:

```
// # 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
```

Summary results:

```
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)
```

Conclusion:
While one process is being executed, the other process is blocked. They switch every 500 ms: the process is the middle of execution is blocked, while the one that was previously blocked runs. The simulation was run for 4000 ms because each process was run for 2000 ms. They were managed according to nonpreemptive scheduling (process B is only run after I/O Blockage of process A) The blockage in this case lasts 500 ms.

2. Simulation for 5 processes:

Configuration:

```
// # 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
```

Summary results:

```
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)
```

Conclusion:

This time the execution took exactly the same amount of time the simulation was allocated (2000 * 5 = 10000 ms). The processes were separated in pairs and handled the same way they were handled in the previous task. Namely, parts of the processes were executed sequentially for 500 ms, followed by a I/O blockage and active process switch, until they were completed. Every process was blocked 3 times in total. The 5th process does not have a pair, so it was executing and blocking itself, however that did not affect the execution time. The scheduling type is also batch nonpreemptive.

3. Simulation for 10 processes:

Configuration:

```
// # 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
```

Summary results:

```
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)
```

Conclusion:

10000 ms is not a sufficient amount of time to execute all 10 processes (their execution time is twice the amount of allocated simulation time). A way to fix it is either increase the simulation time or decrease the execution time of the processes. Up until the 5th process, the behavior was coincidental with the previous tasks. However, the simulation time ended half way through the execution of 5th and 6th process pair. They were still in progress, while the next processes did not even start.

Bibliography

1. https://www.tutorialspoint.com/operating_system/os_process_scheduling_algorithms.htm
2. https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/3_Processes.html
3. http://smurf.mimuw.edu.pl/node/877