COSC 3360-Operating System Fundamentals
Assignment #1: Process Scheduling

➔ Due Friday, February 16 2018 at 11:59:59 pm ➔

1. Objective

This assignment will introduce you to CORE scheduling.

2. Specifications

You are to simulate the execution of processes by a tablet with a large memory, one display, a ***multi-core*** processing unit, and ***one*** solid-state drive. Each process will be described by its start time followed by a sequence of resource requests.

These resources requests will include core requests (**CORE**), SSD requests (**SSD**) and input requests (**INPUT**). Your input will be a sequence of pairs as in:

```
NCORES 2     // number of cores
NEW   12000 // new process
CORE   100   // request CORE for 100 ms
INPUT 5000  // request INPUT for 5000 ms
CORE   80    // request CORE for 80 ms
SSD   1     // request SSD for 1 ms
CORE   30    // request CORE for 30 ms
SSD   1     // request SSD for 1 ms
CORE   20    // request CORE for 20 ms
NEW  12040  // new process
...
```

All times will be expressed in milliseconds. All arrival times will be ***monotonically increasing***.

Processor Management: Your program should have a ***single ready queue***. That queue should be a FIFO queue and keep all processes ordered according to their queue arrival time in strict ***first-come first-served*** order.

SSD Management: SSD access times are much shorter than disk access times with write requests taking less than a millisecond and read requests taking much less than that. As a result, write request timings will be rounded up to one millisecond and read requests timing will be rounded down to zero ms. SSD scheduling will be strictly ***first-come first-served*** (FCFS).

To simplify your life, we will also assume that:

1. There is no contention for main memory,
2. Context switch times can be neglected, and
3. User think times and other delays, like overlapping windows, are included in the input times.

Program organization: Your program should read its input file name though input redirection as in:

```
./a.out < input.txt
```

Your program should have one process table with one entry per process containing a process sequence number, the process class, its process arrival time and its current state (RUNNING, READY or BLOCKED).

Since you are to focus on the scheduling actions taken by the system you are simulating, your program will only have to intervene whenever

1. A process is loaded into memory,
2. A process completes a computational step.

All times should be simulated.

Each time a process ***starts or terminates*** your program should print:

1. The current simulated time in milliseconds,
2. The sequence number of the process that terminates, and the states of all other active processes

When all the processes in your input stream have completed, your simulator should print a summary report listing:

1. The number of processes that have completed,
2. The total number of SSD accesses,
3. The average duration of a SSD access (including the waiting time in the SSD queue),
4. The processing unit utilization, that is, the average number of busy cores (between zero and NCORES),
5. The SSD utilization, that is, the fraction of time that device was busy (between zero and one).

3. Important

To get full credit, your program should start by a block of comments containing your name, the course number, the due date and a very short description of the assignment. Each class, method or function should start by a very brief description of the task it performs.

This document was created on ***Tuesday, January* 23, 2018** and updated last on ***Wednesday, January* 24, 2018**.