AMERICAN UNIVERSITY OF BEIRUT
Faculty of Arts and Sciences
Department of Computer Science
CMPS 240 – Operating Systems
Fall 2022 – 2023

Assignment #4

M. J. Khabbaz

Due Date: November 28, 2022 at 23:59 P.M.

A. Objectives:

The purpose of this assignment is to design and implement two types of threads running as part of a process, namely: *a*) `SpecialReaders`, and, *b*) `SpecialWriters`. A `reader` (respectively `writer`) thread always executes a `read()` (respectively `write()`) function. The following three constraints must be maintained:

1. A `SpecialReader` and a `SpecialWriter` cannot execute their main function (`read()`/`write()`) simultaneously.
2. Readers can run their main function concurrently. However, there should never be more than three special readers (respectively special writers) executing their main functions.
3. Special readers and special writers should not starve.

Special readers and writers are modelled as threads. The designed program should be parametrized with: *i*) `nr` being the number of special readers, *ii*) `nw` being the number of special writers, and, *iii*) an integer `n` indicating that the program must stop running when both readers and writers execute their main functions `n` times. As output, the program must report the state of the readers and the writers (enter/execute/exit).

B. Benchmarks:

In order to study fairness of the program implementation, there is need to count the number of times readers and writers have executed. Run the program for five readers and five writers and for `n` varying in the range `10`, `20`, `50`, `100`, `500` and `1000`.

Create a file `output.txt` that contains the following information in a tabular form as follows (*i.e.* fill in the table below with the actual numbers):

`n`	Number of Readers	Number of Writers
`10`		
`20`		
`50`		
`100`		
`500`		
`1000`		

It is then required to draw a graph consisting of two curves plotted concurrently on the same graph with a legend. The first (respectively the second) curve represents the number of readers (respectively the writers) that entered the room as a function of `n`. Consequently, according to the obtained graph (saved in a file `bench.pdf`), draw your conclusions (saved in a file `answer.txt`) discussing the fairness of your implementation.

C. Submission:

Submit via Moodle a compressed file called `cmps-240-asst3.USER-ID.tar.bz2` (or `.zip`) containing your source code, benchmarks and answers.