CPSC 503 / CPEG 308 **Summer 2024**
Assignment 1 **University of Bridgeport**

In this assignment you will implement a Multilevel Queue for CPU scheduling. You are free to code this in an object-oriented manner in a language of your choice. In addition to the example covered in the lecture, you can find another well-defined example at:
Multilevel Queue (MLQ) CPU Scheduling - GeeksforGeeks
(https://www.geeksforgeeks.org/multilevel-queue-mlq-cpu-scheduling/)

PART ONE
The following is the description of a four-level queue. Depending on the group size, you can scale down the problem. If you are working as a single person group, choose any two scheduling algorithms. Implement a multi-level feedback Q scheduler satisfying the following requirements:

- The scheduler consists of 4 linear Qs
- The first Q is FIFO, second Q is priority-based, third Q is SJF, and the fourth (highest Priority) is round robin
- Feedback occurs through aging, aging parameters differ, i.e., each Q has a different aging threshold (time) before a process can migrate to a higher priority Q (should be a variable parameter for each Q)
- The time slot for the Round Robin Q is a variable parameter
- Implement a switch (variable parameter) to allow choosing pre-emptive and non-preemptive scheduling in the SJF, and priority-based Qs
- Jobs cannot execute in a Q unless there are no jobs in *all* higher priority Qs
- The jobs are to be created with the following fields in their PCB: Job number, arrival time, execution/burst time, priority, Queue number (process type 1-4). The creation is done randomly, choose your own ranges for the different fields in the PCB definition
- Output should indicate a *time line*, i.e., every time step, indicate which processes are created (if any), which ones are completed (if any), processes which aged in different Qs, etc.
- You should allow a variable context switching time to be specified by the user

PART TWO
After completing the implementation and doing a few sample runs, start thinking of this problem as a *multivariable optimization problem*. From an algorithmic design point of view, you can think of this problem as a problem that involves the following parameters:

- Context switching time
- Three Aging Parameters
- Time slot
- Two switches for Preemption/Non-preemption

The eventual goal could be to optimize two or more performance measures (criteria) such as:

- Average waiting time
- Average turnaround time
- CPU utilization

Out: Tue May 21, 2024
Due: Mon May27, 2024 with Demo to Instructor on Tuesday May 28 during assigned slots

- Maximum turnaround time
- Maximum wait time
- CPU throughput

Perform several test runs and write a summation indicating how sensitive are some of the performance measures to some of the above parameters (or a combination of the above parameters). Include some of the results (time lines and/or graphs) with your observations and conclusions regarding the effect of changing the values of different parameters on the performance measures.

Bonus Opportunities Available (*if interested* 😊)

Out: Tue May 21, 2024
Due: Mon May27, 2024 with Demo to Instructor on Tuesday May 28 during assigned slots