CSC3002F 2023
OPERATING SYSTEMS PART 2
LECTURER: MICHELLE KUTTEL
ASSIGNMENT:

THE MASTER SCHEDULE: ANDRE STRIKES AGAIN

Version 1, April 2024

INTRODUCTION

This assignment aims to increase your understanding of **CPU process scheduling,** particularly the different **metrics** by which schedulers are compared, and to give you some experience with testing and exploring scheduling algorithms using a **simulation**. Computer simulation is used to model the behaviour of a complex system. The theory is that the simplified model has predictive properties and can tell us something about a real world system.

You are given a Java package for simulating a version of process scheduling. This is simple simulation (the output is just text) involves a scenario of a busy bar where patrons arrive at random times throughout the evening and place orders for 1-5 drinks (the drink orders are the "jobs" to be processed). The drinks orders are filled by Andre the Barman (who here is the CPU and scheduler in one).

In the code you are given, Andre the Barman operates on **First Come First Served** (FCFS) schedule. In this assignment, you will need to implement an alternative **Shortest Job First (SJF)** schedule and, using a range of metrics, decide which schedule is the best.

You can also implement and test a **Round Robin Scheduler** for extra credit.

ASSIGNMENT SPECIFICATIONS

You are provided with a simple and incomplete Java package. You will need to inspect it carefully, run it with different inputs and figure out what it does. Note carefully the **synchronisation mechanisms** used in the code to keep it thread safe (we discussed thread safety last year and will do so again in this OS2 block.)

Your task is to extend the code supplied so that you can compare the performance of an FCFS with an SJF scheduler for this scenario (which you will implement) across the following metrics:

- **throughput** (the number of patrons served per time unit);
- **turnaround time** (the time it takes from when a patron places their order to when it is completed);
- **waiting time** (how long each patron waits while their order is not being worked on) and
- **response time** (how long before a patron receives their *first* drink).

You will need to add timing code to measure these metrics, and to write their values to files.

You must then plot the distribution of these metrics for both algorithms as histograms.

Note that you will have to decide how best to test the algorithms, bearing in mind that experiments need to be run multiple times for reliable data.

You may want to write Python scripts to extract and visualise the data, or just use Excel. But your graphs must be clear and sensible.

Then must submit a **short report** containing all the graphs, **explaining** what you see in the graphs and **drawing conclusions**. You must report on and discuss the average value of each of the metrics and **what this means**, the **predictability of each algorithm** (this is linked to the variance of each metric), the fairness and the possibility of starvation.

Finally you need conclude by **making a recommendation** for the best schedule for Andre the Barman, and justify your conclusions.

CONSTRAINTS

- You have to extend the original code – you can't start from scratch.
- You cannot change the command-line arguments for the code.
- The list of drinks is fixed – you cannot change it.
- You can't change the arrival times for the patrons.
- Your algorithms cannot use pre-emption.

SUBMISSION

Submit the following as separate files

- A zipped archive of your extended Java program.
- A separate document in **PDF format** containing your report. It should be about 2-5 pages in length.

The code archive must include:

- **All classes** needed to run your program.
- A **Makefile** for compilation.
- A **Readme file** explaining how to run the program (input parameters cannot be changed).
- A **GIT usage log** in .txt format (use git log –all to display all commits and save).