DSAI 3202 – Parallel and distributed computing

Lab – 4: Temperature Monitoring System

# Objectives:

* Develop a Python program that simulates temperature readings from multiple sensors, calculates average temperatures, and displays the information in real-time in the console.

# Tools and Concepts:

* Python: Programming language.
* Threading: For concurrent execution.
* Queue: For thread-safe data transfer.
* Locks and Conditions: For thread synchronization and communication.

# Tasks

## Implement Sensor Simulation.

* Write a function called simulate\_sensor that simulates temperature readings from a sensor.
* Use random.randint(15, 40) to generate random temperatures.
* Make simulate\_sensor update a global dictionary latest\_temperatures with its readings every second.

## Implement Data Processing

* Write a function called process\_temperatures that continuously calculates the average temperature from readings placed in a queue.
* Make process\_temperatures update a global dictionary temperature\_averages with the calculated averages.

## Integrate Threading

* Create threads for each call simulate\_sensor and the process\_temperatures function.
* Understand how to use daemon=True to manage thread lifecycle with the main program.

## Implement Display Logic

* Write a function initialize\_display to print the initial layout for displaying temperatures. The print should look like this.

Current temperatures:

Latest Temperatures: Sensor 0: --°C Sensor 1: --°C Sensor 2: --°C

Sensor 1 Average: --°C

Sensor 2 Average: --°C

Sensor 3 Average: --°C

In the following steps of your programs, you will write a program that only replaces the “--”, without erasing the console.

* Develop update\_display to refresh the latest temperatures and averages in place on the console without erasing the console.

## Synchronize Data Access

* Use RLock and Condition from the threading module to synchronize access to shared data structures and control the timing of updates.

*What should you use for which task?*

## Finish building the Main Program and organize your files.

* Put the functions in a separate file.
* Create a file for the maim program.
* Initialize a queue and share data structures.
* Start the sensors and data processors threads.
* Initialize the console display and start the display update thread. Make the display updated every 5s.
* Ensure the main thread keeps running to allow the daemon threads to operate.

**Midterm Bonus 5%: Make the latest temperature updated every 1s, and the average temperatures update every 5s.**

# Questions:

1. Why did the professor not ask you to compute metrics?