QA Software Coding Challenge: Fan Control

# Problem

A robot has many subsystems that each generate their own individual temperature measurement. There are also multiple fans onboard the robot that are used to cool the electronics to prevent overheating. Because the fans are very loud when running at 100% duty cycle, and because the robot operates alongside humans, the fan speeds are set so that the noise level is minimized without endangering the electronics.

# Fan Control Requirements

* The temperature of each subsystem is provided as a 32-bit floating point number in °C via IPC.
* The speed of each fan is set by writing a 32-bit unsigned integer to a specific hardware register that is different for each fan. This integer is in PWM counts and is proportional to fan duty cycle.
* The PWM counts corresponding to 100% duty cycle may be different for different fans. You may assume that 0 PWM counts always represents 0% duty cycle
* The most recent temperature measurements from each subsystem should be collected, and the fan duty cycle should be computed from **the** **maximum** **of the most recent temperatures of all subsystems**.
* All fans should be set to the same duty cycle.
* If the temperature is 25° C or below, the fans should run at 20% duty cycle.
* If the temperature is 75° C or above, the fans should run at 100% duty cycle.
* If the maximum measured subsystem temperature is in between 25° C and 75° C, the fans should run at a duty cycle linearly interpolated between 20% and 100% duty cycle.

# Task

Your task is to validate the above requirements for Fan Control

* Describe the test plan and test cases you want to use to test the Fan Control functionality on the robot.
* List the set of APIs on the robot you expect to call in order to implement the test cases.
* Implement the test cases. This can be done in whatever language you prefer (Python or C++).
* Write a sample program to run the test cases. You can stub out or mock the robot APIs and other functions as needed to simplify your work.

# Guidelines

The submission should adhere to the following guidelines:

* Code must compile and run.
* You may assume that the platform running your application has an IEEE-754 compliant floating-point unit.
* You may use additional open-source libraries for your submission but be prepared to discuss how the libraries work and why you chose to use them.
* Please include the source of any third-party libraries with your submission if you choose to use them.
* Please include build infrastructure (Makefiles, CMake files, etc.) necessary to recompile your submission.
* In addition, please write a one-page report explaining your assumptions, thought process, challenges and future enhancement.

# Hints

Please spend as much time as you have available. The challenge is intentionally open ended, so **focus on the parts of the challenge that you feel are most valuable**. Organize your effort so that what you believe to be the most critical pieces are completed to a level that you would consider production ready.