GMIT

BEng(H) Software & Electronic Engineering

C++ Programming Assessment

Lecturer: Michelle Lynch

Assignment

Design, code, and test C++ software to represent an **Accelerometer**.

The Accelerometer is a triple-axis accelerometer, with x, y and z-axis data, however you are ONLY required to write member function source code and test code for the **Z-AXIS** functionality.

Your code is required as part of a larger software engineering project that requires an Accelerometer.

Requirements

- It is required to store n samples of accelerometer **z-axis data** (only); n can be any value but limit n to 10 for testing. Data will be in the range +/-2 g. An example set of data is: [0.5, 1.4, 0.0, -2.0, -0.8, …].
- Each Accelerometer should also have a unique identifier or name.
- Develop your own code. You may re-use/adapt code you developed during the C++ Programming module if desired. Include reference to any online third-party code/examples referenced in the comment header at the top of your relevant source file(s).
- Use object-oriented programming for the Accelerometer design code. Use procedural programming for test code.
- Use modern C++.
- Provide a basic interface for the user of your design to enable them to perform basic common tasks, such as create new Accelerometer(s), read and/or write to an Accelerometer.
- Include a copy assignment operator in your design.
- Create & run unit tests for your code.
- Add functionality to your Accelerometer design to enable the use of random data for testing.
- Add functionality to your Accelerometer design to enable the Accelerometer's data to be adjusted by a scalar addition.
- Add functionality to your Accelerometer to add a new data sample to the data, while keeping the total number of samples stored the same.
- Add functionality to your Accelerometer to calculate a 2-tap moving average of the z-axis data, i.e., average every two adjacent data samples, and to use the moving average results to determine the direction of change of the accelerometer.
- Add any additional design and/or test feature(s) to your code, of your own choosing, that you think would be useful and demonstrate your C++ skills.
- Make any assumptions or coding decisions necessary to deliver what you consider to be your best quality design and test code.
- Create 'clean code': well designed, efficient, demonstrating software engineering principles with good coding style.