HW 2

- **HW#2 Implement the controller from HW #1(Due 11/08/'18)**
- The following tasks shall be implemented:
  - **OperatorConsole**: Output pressure and temperature log at 2 Hz. on the console.
  - **PressureController**: read pressure and apply the following input at pump at 10 Hz.

    if p < 0.9 then u = A - B p

    else u = 0

    where p is the measured pressure, A and B are configurable values.
  - **TemperatureController**: At 3 Hz., read temperature and turn on heater if the temperature is below C , and turn off heater if above D

    where C and D are configurable values
- Each of these tasks shall be embedded in separate classes (*OperatorConsole, PressureController and TemperatureController classes having their own threads*)

HW 2

- **HW#2 Implement the controller from HW #1(Due 11/08/'18)**
- Implementation Rules:
  - Use C++ with Microsoft Visual Studio on Windows (Community Edition Available for download)
  - Write a report containing the software requirements and test cases of your controller
  - Use C++11 threading primitives for multithreading: **std::thread**
  - Use C++11 chrono API for timing requirements
    - i.e., **std::this_thread::sleep_for(std::chrono::milliseconds(…))**
    - i.e., **std::chrono::high_resolution_clock::now()**
  - Configurable values: A, B, C and D can be defined as *const double* and initialized to default in the headers.

HW 2

- **HW#2 Implement the controller from HW #1(Due 11/08/'18)**
- Interface: A simulator implementing the following interface will be used during evaluation tests.

```
class ISimulator {
public:
        void triggerADCPressure() = 0;
        void triggerADCTemperature() = 0;
        double readADCPressure() = 0;
        double readADCTemperature() = 0;
        void writeDACPump(const double value) = 0;
        void switchHeater(bool isOn) = 0;
}
```

- Before reading the ADC's, they should be started by calling the triggerADC functions. That is, every readADC call should be preceded by its corresponding triggerADC call.
- readADC functions are blocking calls. After triggering the ADC, readADC call will be blocked until the ADC result is returned. The time it takes for the ADC to return the result is not known.
- Hint: You will need to implement this interface for testing your own code before I do it with mine.