Lab 5: Embedded System
Part 1
Due at the end of the lab (before 11:00am)

Write and test the embedded code to run an electronic device with the following features:

The device will read and display the following:

1. **Current** humidity and temperature levels
2. The **maximum** and **minimum** readings of both humidity and temperature since it was last reset by the user
3. The current **trend** of both humidity and temperature.
    a. The trend of humidity will compare the last two humidity readings and determine if humidity is increasing, decreasing, or stable.
    b. The trend of temperature will compare the last two temperature readings and determine if temperature is increasing, decreasing, or stable.
4. Humidity **Check**: The device will display a humidity check status:
    a. OK if the relative humidity is between 30% and 55%
    b. High: if relative humidity is above 55%
    c. Low: if relative humidity is below 30%
5. The device has a **reset** button to reset the maximum and the minimum values of both humidity and temperature to current readings

Input:

1- The device will read the **current relative humidity** from an input humidity sensor
    Allowed input range: 0% – 100% relative humidity
2- The device will read the **current temperature** from an input temperature sensor
    Allowed input range: 0 – 125 degrees Fahrenheit

Input sensitivity of temperature sensor is one degree Fahrenheit
Input sensitivity of humidity sensor is one degree Fahrenheit

Output:

Your embedded code will display the following values:

For relative humidity:

1. The current relative humidity
2. The maximum relative humidity reading since it was last reset
3. The minimum relative humidity reading since it was last reset
4. The relative humidity trend (up, down, or stable)

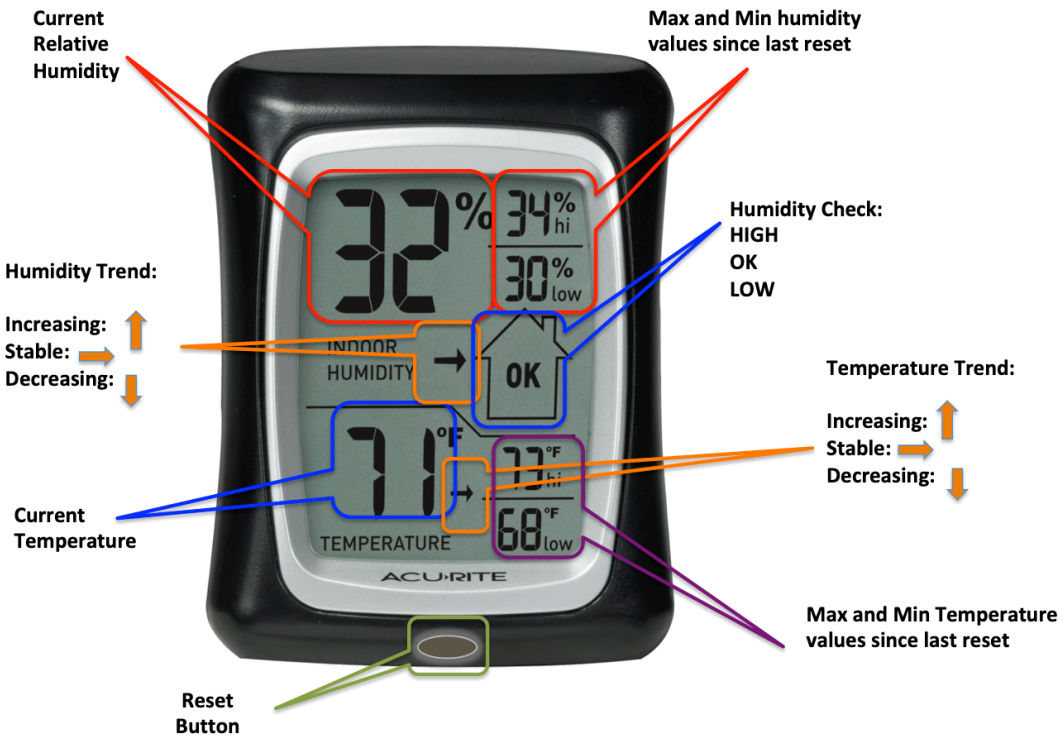
For temperature:

5. The current temperature
6. The maximum temperature reading since it was last reset
7. The minimum temperature reading since it was last reset
8. The temperature trend (up, down, or stable)

For Humidity Check:

9. Humidity status (Hi, OK, or Low)

See next page for demonstration →



Deliverables:

1- Your source code to run this embedded system. The **output** can be simple alphanumeric values (for example, for humidity:

Current Humidity: 33%
Maximum Humidity: 52%
Minimum Humidity: 31%
Humidity Trend: Increasing

Do the same for temperature and the humidity check

Make sure your code compiles and runs correctly. You do not need to take screenshot of any specific input values or design any test cases yet.

Tips:

- Make sure you initialize your values **correctly** after each reset.
- Think of the Reset as a function or a class,
- Java has multiple time libraries that you could use
- Select your sensor reading (and hence your reading refresh rate wisely). Reading too frequently (e.g. every second) will run the battery down fast. Reading every few minutes will provide low-quality feedback (with high latency.)

Lab 5: Embedded System
Part 2
Due Tuesday 4/12/2022

a) Run your code using the following sequence of values:

Input the following **relative humidity** sequence (in %) after a reset: **53, 51, 48, 49, 54, 56**
Input the following **temperature sequence** after a reset: **66, 68, 69, 67, 63, 59, 53**

Take a screenshot of the 9 output values after these sequences are read and include it in the deliverable below

b) Design at least **two** test cases to test each of the output values.
Current humidity and temperature
Max humidity and temperature
Min humidity and temperature
Trends: Write separate test cases for each trend (up, stable, and down)
Each test will include code with input values **or input sequences** as needed for each test
Run your tests and take a screenshot of each

c) Refactoring:
You will notice that the algorithms used for humidity and temperature are similar (for calculating the **max**, the **min**, and the **trend**). Refactor your code to use one algorithm that could be used in both humidity and temperature calculations.

Deliverable:

i) Include one screenshot of the 9 output values after the sequences in part a are read by the system

ii) Write a test report showing each of your test cases (with narrative description of what you're testing in each one) and take a screenshot of the result of each test

iii) Submit your refactored code (with comments)

iv) As the same person who developed, refactored, and tested the code, does your refactored code make it easier or harder to test it, explain and provide examples

v) If you received the refactored code (written by another developer) to just test it, would it be easier or harder than case iv) above.

vi) Would you prefer to test the original code or the refactored code, if both were written by another developer