**Task 1:**

**Try adding emojis for temperature and humidity to see if the OLED supports them.**

**Answer:**

The oled display does not support emoji directly but we can make emoji by using pixels

**Blow on the sensor and observe whether it detects minor changes in temperature and humidity.**

**Answer:**

Yes it observe minor change in temperature and humidity when I blow on the sensor   
  
**Task 2:**

**Run the same functionality of the lab work without interrupt & log your observations. Also state, what happened before and after interrupt?**

**Answer:**

I ran the same code by removing the interrupt and I was not able to use button. The code only displays temperature and humidity readings, and the button press is ignored.

**Task 3:**

**What is a debounce issue and why we get rid of it?**

**Answer:**

Debounce is an issue that occurs when a mechanical switch (like a button) is pressed or released. Instead of registering a single clean signal, the button may send multiple rapid signals due to mechanical vibrations (bouncing).

We get rid of debounce to

Prevents unwanted multiple inputs.

Ensures reliable and accurate signal detection.

Improves system stability and user experience.

**In which applications/domains, debounce issue can be threatening if not resolved in the system?**

**Answer:**

The debounce issue can cause problems in many critical systems, including:

Embedded Systems & IoT

Smart home devices (e.g., turning lights on/off multiple times).

Wearable devices (e.g., detecting multiple taps on a smartwatch).

Industrial Automation

Machines may start/stop unexpectedly, causing accidents.

Faulty button presses could damage expensive equipment.

Medical Devices

Equipment like ventilators or infusion pumps must register precise inputs.

False readings could lead to dangerous consequences.

**Why debounce occurs? Is it a compiler error, logical error or micro-controller is cheap**

**Answer:**

Debounce happens because of the mechanical nature of switches. When a button is pressed or released:

The contacts inside the switch physically bounce before settling.

These rapid ON/OFF signals create multiple unwanted signals.

It is because of mechanical switch limitation

**Task 4:**

**Why do we use interrupt?**

An interrupt is a signal that pauses the main program to execute a special task (Interrupt Service Routine - ISR) and then resumes normal execution. Interrupts allow the microcontroller to react to external events immediately without constantly checking for changes.

**How does interrupt lower the processing cost of the micro-controller?**

**Without Interrupts**

The microcontroller keeps checking the input continuously.

Consumes CPU cycles even when there's no change in input.

Leads to higher power consumption and wasted processing time.

**With Interrupts**

The CPU executes other tasks until an event occurs.

When an interrupt is triggered, only the necessary code (ISR) runs.

After handling the event, the CPU resumes its previous task without wasting time.