**Task 1**

**The OLED display does not natively support emojis, but custom emoji-like symbols can be created using pixel patterns. Blowing on the sensor caused minor fluctuations in both temperature and humidity, which the sensor successfully detected.**

**Task 2**

**After removing the interrupt, the button stopped functioning. The program continued displaying temperature and humidity readings, but pressing the button had no effect. Before removing the interrupt, the button was responsive, allowing user interaction while displaying sensor data. After removal, the system ignored button inputs and only displayed sensor values.**

**Task 3**

**Debounce is a mechanical issue that occurs when a switch is pressed or released. Instead of registering a single clean signal, the button generates multiple rapid signals due to internal contact bouncing. This can cause unintended multiple inputs, leading to unreliable operation. Eliminating debounce ensures accurate input detection, improves system stability, and enhances user experience.**

**Debounce issues can affect embedded systems, IoT devices, industrial automation, and medical equipment. In smart devices, multiple unintended inputs can cause erratic behavior. In industrial automation, false triggers may lead to unexpected machine operations, posing safety risks. Medical devices, such as ventilators or infusion pumps, require precise input registration, and debounce issues could lead to inaccurate readings or dangerous malfunctions.**

**Debounce is not a programming or compiler error, nor is it caused by cheap microcontrollers. It is a hardware limitation due to the physical nature of mechanical switches. When a button is pressed, the internal contacts momentarily oscillate before settling, generating multiple signals instead of a single clean input.**

**Task 4**

**Interrupts allow a microcontroller to respond to external events immediately without constantly monitoring inputs. When an interrupt occurs, the system temporarily pauses its main task, executes a specific function (Interrupt Service Routine - ISR), and then resumes normal operation.**

**Without interrupts, the microcontroller continuously checks for input changes, wasting processing power and increasing energy consumption. With interrupts, the CPU can focus on other tasks and only responds when an event occurs, improving efficiency and reducing power usage.**

**Top of Form**

**Bottom of Form**