**OBJETIVOS CLAROS:**

1. Desarrollar algoritmos de control para mantener el sistema entorno a 25 grados de temperatura y humedad al 80%.
2. Aplicar los estándares de seguridad y calidad a dichas actuaciones mediante acondicionamiento y procesamiento de los algoritmos de control y actuación sobre las magnitudes manipulables del proceso aplicando los conocimientos aprendidos en el temas 7.
3. Display LCD amigable:
   1. Presentar temperatura anterior y temperatura actual
   2. Complejo (opcional) presentar el gasto energético diario.

**ALGORITMO DE CONTROL:**

1. Determinar cuánto se aleja la variable medida del sensor con la variable deseada:

*Error = TempDeseada – TempMedida*

1. Utilizar algoritmo de control para calcular la acción de control. Actuador + Elemento final.

**CONTROL DE LA TEMPERATURA:** sistema de calentamiento y/o enfriamiento de las baterías

* Relación proporcional de 0.5 entre la acción del elemento final y el proceso y un retardo de 500ms (es decir que incrementa o decrementa medio grado cada 500ms)

**ALGORITMO DISCONTINUO DE TRES POSICIONES SIN HISTÉRESIS PERO CON ZONA MUERTA**

*Límite Máximo de Control = TempDeseada + 3*

Límite Mínimo de Control = TempDeseada – 3

- Si la temperatura medida es mayor del límite máximo debe enfriarse el sistema (activar el dispositivo que abre la electroválvula de enfriamiento)

- Si la temperatura medida está en el rango entre los dos límites se desactivan ambas acciones.

- Si la temperatura está por debajo del límite mínimo se debe calentar el sistema y se activa el dispositivo que abre la electroválvula del fluido caliente.

Solución: combinar el algoritmo discontinuo con acciones continuas proporcionales al error de la medición con respecto al valor deseado.

**Estructura lógica**

**1. Inicialización (setup)**

* Inicializar comunicación serie (Serial.begin)
* Inicializar el sensor DHT22
* Inicializar los motores paso a paso (frío y calor)
* Inicializar los LEDs indicadores

**2. Bucle principal (loop)**

**2.1 Leer temperatura ambiente desde DHT22 cada 1 segundo**

* Tm = dht.readTemperature()
* ¿Lectura válida?
  + **Sí**: usar el valor de Tm
  + **No**: usar la temperatura interna Ts como fallback (evita errores de lectura)

**2.2 Controlador de 3 posiciones con zona muerta**

* Definir límites:
  + limiteMin = T\_setpoint - Δ
  + limiteMax = T\_setpoint + Δ
* Evaluar condiciones:

Si Tm > limiteMax:

PWM\_frio = Kp × (Tm - limiteMax)

PWM\_frio = constrain(PWM\_frio, 0, 255)

PWM\_calor = 0

Sino si Tm < limiteMin:

PWM\_calor = Kp × (limiteMin - Tm)

PWM\_calor = constrain(PWM\_calor, 0, 255)

PWM\_frio = 0

Sino:

PWM\_frio = 0

PWM\_calor = 0

**2.3 Convertir PWM a posición de válvula**

* Para cada motor:

Ángulo objetivo (θ) = (PWM / 255.0) × 359.0 grados

Mover motor paso a paso hacia θ

**2.4 Simulación del sistema térmico interno (Ts)**

Cada 500ms:

Ts += 0.5 × (PWM\_calor / 255.0) // Calor

Ts -= 0.5 × (PWM\_frio / 255.0) // Frío

// Influencia del ambiente

diferencia = Tm - Ts

Ts += diferencia × α // α ≈ 0.005

**2.5 Actualizar estado de los motores**

* Ejecutar .run() para cada motor si queda movimiento pendiente

**2.6 Encender LEDs según actividad**

Si PWM\_frio > 0:

LED\_AZUL = ON

Sino:

LED\_AZUL = OFF

Si PWM\_calor > 0:

LED\_ROJO = ON

Sino:

LED\_ROJO = OFF