Explicación Detallada del Código Final (Monitor Ambiental)

Este código está diseñado para ser eficiente, evitando el uso de la función `delay()` para que el Arduino pueda realizar múltiples tareas (leer sensor, actualizar display, alternar datos) de forma casi simultánea.

1. Configuración y Declaraciones Iniciales

Esta sección define las librerías necesarias y establece las conexiones de hardware y las variables de estado.

Código	Explicación
`#include <Wire.h>`	**Librería I2C:** Esencial para la comunicación de dos hilos (SDA y SCL) con el módulo LCD.
`#include <LiquidCrystal_I2C.h>`	**Librería LCD I2C:** Contiene las funciones para controlar la pantalla.
`#include <DHT.h>`	**Librería DHT:** Proporciona las funciones para leer los datos del sensor DHT11.
`LiquidCrystal_I2C lcd(0x3F, 16, 2);`	**Instancia LCD:** Crea el objeto de la pantalla. Se define la dirección I2C (`0x3F`) y sus dimensiones (16 columnas, 2 filas). **Nota:** Si la pantalla no funciona, cambia `0x3F` por `0x27`.
`const int PIN_DHT = 2;`	**Pin de Datos DHT:** Define que el pin de datos del sensor DHT11 está conectado al pin digital 2 del Arduino.
`DHT sensorDHT(PIN_DHT, TIPO_DHT);`	**Instancia Sensor:** Inicializa el objeto del sensor, pasándole el pin y el tipo (`DHT11`).
`float temperatura = 0.0;`	**Variables de Datos:** Variables globales para almacenar los valores de temperatura y humedad leídos.

`bool mostrarHumedad = false;`	**Bandera de Estado:** Variable que controla qué dato se muestra en la fila superior del LCD (`false` = Temperatura; `true` = Humedad).
`unsigned long tiempoUltimaLecturaDHT = 0;`	**Control de Tiempo (DHT):** Guarda el tiempo en milisegundos de la última lectura.
`const unsigned long intervaloLectura = 1500;`	**Intervalo DHT:** Define que el sensor debe ser leído cada 1500 milisegundos (1.5 segundos).
`const unsigned long intervaloAlternancia = 5000;`	**Intervalo LCD:** Define que la pantalla debe alternar la visualización cada 5000 milisegundos (5 segundos).

2. Funciones de Control (Sin Bloqueos)

Estas funciones utilizan el tiempo actual (`millis()`) para ejecutar acciones periódicamente sin detener el programa principal.

`void leer_sensor_si_toca()`

```
// Lógica interna
unsigned long tiempoActual = millis();
if (tiempoActual - tiempoUltimaLecturaDHT >= intervaloLectura) {
   humedad = sensorDHT.readHumidity();
   temperatura = sensorDHT.readTemperature();
   // ... manejo de errores ...
   tiempoUltimaLecturaDHT = tiempoActual;
}
```

- **Propósito:** Lee el sensor solo si ha transcurrido el tiempo (`1.5 segundos`).
- **Mecánica:** Compara el `tiempoActual` con el `tiempoUltimaLecturaDHT`. Si la diferencia es mayor al `intervaloLectura` (1500 ms), realiza la lectura y luego **reinicia** `tiempoUltimaLecturaDHT` con el valor de `tiempoActual`.

`void manejar_alternancia_auto()`

```
// Lógica interna
unsigned long tiempoActual = millis();
if (tiempoActual - tiempoUltimaAlternancia >= intervaloAlternancia) {
   mostrarHumedad = !mostrarHumedad;
   tiempoUltimaAlternancia = tiempoActual;
}
```

- **Propósito:** Cambia el estado de visualización del LCD (`Temperatura` a `Humedad` o viceversa).
- **Mecánica:** Compara el tiempo transcurrido con el `intervaloAlternancia` (5 segundos). Si se cumple, `mostrarHumedad = !mostrarHumedad;` invierte el valor de la bandera, y el tiempo de la última alternancia se reinicia.

`void actualizar_display(...)`

- **Propósito:** Renderiza los datos en la pantalla.
- **Mecánica (Fila 0):** Usa `if (mostrar_hum)` para decidir qué imprimir. Si es verdadero, imprime el valor de la humedad; si es falso, imprime la temperatura, utilizando `lcd.print((char)223)` para el símbolo de grado (°).
- **Mecánica (Fila 1):** Siempre imprime el contador de tiempo (`segundosActuales`).

3. Funciones Principales (`setup` y `loop`)

`void setup()`

Código	Explicación
`Serial.begin(9600);`	Inicializa el puerto de comunicación serial.
`lcd.init(); lcd.backlight();`	Inicializa la pantalla LCD y enciende la luz de fondo.
`lcd.print("Iniciando..."); delay(1000); lcd.clear();`	Muestra un mensaje de bienvenida durante 1 segundo antes de comenzar la monitorización.
`sensorDHT.begin();`	Inicia la comunicación con el sensor DHT.
`tiempoUltimaLecturaDHT = millis();`	Inicializa la variable de tiempo para la primera lectura.

`void loop()`

El corazón del programa. Se repite constantemente.

Código	Explicación
`manejar_alternancia_auto();`	Llama a la función para ver si es momento de alternar la pantalla.
`leer_sensor_si_toca();`	Llama a la función para ver si es momento de leer el sensor.
`unsigned long segundosActuales = ...`	**Cálculo del Tiempo de Ejecución:** Determina cuántos segundos han pasado desde el inicio, aplicando el retraso inicial (`retardoInicioContador`) para un conteo limpio.
`actualizar_display(temperatura, humedad, ...)`	Pasa los datos (los valores del sensor y el contador) a la función para que se muestren en el LCD.
`delay(100);`	**Pausa Mínima:** Una pausa muy corta (100 ms) para evitar que el microcontrolador gaste el 100% de su capacidad, pero lo suficientemente pequeña para no bloquear las funciones controladas por `millis()`.