En MicroPython —igual que en Python estándar— **ambas estructuras funcionan**, pero **no son equivalentes**. La elección depende de *qué* quieres capturar con el try-except y *cuándo* quieres que el programa se recupere del error.

A partir de ahí, hay una opción que suele ser más práctica en la mayoría de proyectos con microcontroladores.

🧠 Dos patrones posibles

1. while True: dentro de try

```
try:
  while True:
    hacer_algo()
except Exception as e:
  print("Error:", e)
```

✔️ Cuándo usarlo

- Cuando **quieres que cualquier error dentro del bucle salga al except** y *rompa* el bucle.
- Cuando un fallo debe detener el programa o reiniciarlo.
- Cuando el error no debe repetirse en bucle.

➕ Ventajas

- Código más limpio.
- El except captura errores globales del bucle.

➖ Inconvenientes

- Si ocurre un error, **el bucle termina** y no se reanuda automáticamente.

---

2. try: dentro del while True

```
while True:
  try:
    hacer_algo()
  except Exception as e:
    print("Error:", e)
```

✔️ **Cuándo usarlo**

- Cuando quieres que **el bucle siga funcionando aunque haya errores puntuales**.
- Ideal para sensores, comunicaciones, lectura de pines, etc., donde puede haber fallos transitorios.

➕ **Ventajas**

- El programa **no se detiene** por un error.
- Muy útil en sistemas embebidos que deben seguir vivos.

➖ **Inconvenientes**

- Si el error es persistente, puedes entrar en un **bucle infinito de excepciones**.
- Conviene añadir sleep() o lógica de recuperación.

⭐ **Recomendación práctica para MicroPython**

En la mayoría de proyectos de microcontroladores (ESP32, Raspberry Pi Pico, etc.) **es más útil poner el try-except dentro del while True**, porque:

- Los fallos suelen ser intermitentes (lecturas, WiFi, UART...).
- No quieres que el programa se detenga.
- Permite reintentar sin reiniciar el dispositivo.

Es decir, **opción 2** suele ser la más robusta para sistemas que deben seguir funcionando.

🧩 **Ejemplo típico robusto**

```
while True:
    try:
        leer_sensor()
        enviar_datos()
    except Exception as e:
        print("Fallo:", e)
        time.sleep(0.5)   # Evita bucles de error rápidos
```