

Manejo de errores

Try-Except-Else-Finally

- **try:**
 - Contiene el código que podría causar una excepción.
 - Si ocurre una excepción, el flujo del programa se transfiere al bloque **except**.
- **except:**
 - Contiene el código que se ejecuta si se produce una excepción en el bloque **try**.
 - Puede especificar tipos de excepción para manejar diferentes tipos de errores de manera distinta.
- **else:**
 - Contiene el código que se ejecuta si no ocurre ninguna excepción en el bloque **try**.
- **finally:**
 - Contiene el código que se ejecuta siempre, independientemente de si ocurrió una excepción o no.
 - Se usa típicamente para liberar recursos o realizar limpieza.

Ejemplo: manejo de errores mediante Try-Except-Else-Finally

1. Bloque **try**:

- Se intenta realizar una división entre **numerador** y **denominador**:
- Si se produce una excepción, el flujo de ejecución pasa al bloque **except**. Si no ocurre ninguna excepción, se calcula el resultado de la división.

2. Bloque **except**:

- Captura cualquier excepción que pueda ocurrir y se imprime un mensaje detallado:

- La variable `e` captura el mensaje de error de la excepción. En este caso, como `denominador` es 0, se producirá una excepción de división por cero, y se imprimirá el mensaje "Error: division by zero".

3. Bloque `else`:

- Se ejecuta solo si no se produjo ninguna excepción en el bloque `try`:
- Si la división se realiza sin errores, se imprimirá el resultado de la división. En este caso, debido a la excepción, este bloque no se ejecutará.

4. Bloque `finally`:

- Se ejecuta siempre, haya ocurrido o no una excepción:
- Este bloque asegura que el mensaje "El programa terminó exitosamente" se imprima sin importar si ocurrió una excepción o no.

```
numerador = 10
denominador = 0

try:
    division = numerador / denominador
except Exception as e:
    print(f"Error: {e}")
else:
    print(f"La división es: {division}")
finally:
    print(f"El programa terminó exitosamente")
```

```
Error: division by zero
El programa terminó exitosamente
```

También se puede realizar el manejo de errores, utilizando el tipo de error adecuado.

1. Bloque `try`:

- Se intenta realizar una división entre `numerador` y `denominador`:
- Si se produce una excepción, el flujo de ejecución pasa al bloque `except`. Si no ocurre ninguna excepción, se calcula el resultado de la división.

2. Bloque `except` con Tipo de Error Específico:

- Captura la excepción específica `ZeroDivisionError` y se imprime un mensaje detallado:

- La variable `e` captura el mensaje de error de la excepción. En este caso, dado que `denominador` es 0, se producirá una excepción de división por cero, y se imprimirá el mensaje "Error: division by zero. Introduzca un denominador distinto de cero".

3. Bloque `else`:

- Se ejecuta solo si no se produjo ninguna excepción en el bloque `try`:
- Si la división se realiza sin errores, se imprimirá el resultado de la división. En este caso, dado que se produce una excepción, este bloque no se ejecutará.

4. Bloque `finally`:

- Se ejecuta siempre, haya ocurrido o no una excepción:
- Este bloque asegura que el mensaje "El programa terminó exitosamente" se imprima sin importar si ocurrió una excepción o no.

```
numerador = 10
denominador = 0

try:
    division = numerador / denominador
except ZeroDivisionError as e:
    print(f"Error: {e}. Introduzca un denominador distinto de cero")
else:
    print(f"La división es: {division}")
finally:
    print(f"El programa terminó exitosamente")
```

```
Error: division by zero. Introduzca un denominador distinto de cero
El programa terminó exitosamente
```