

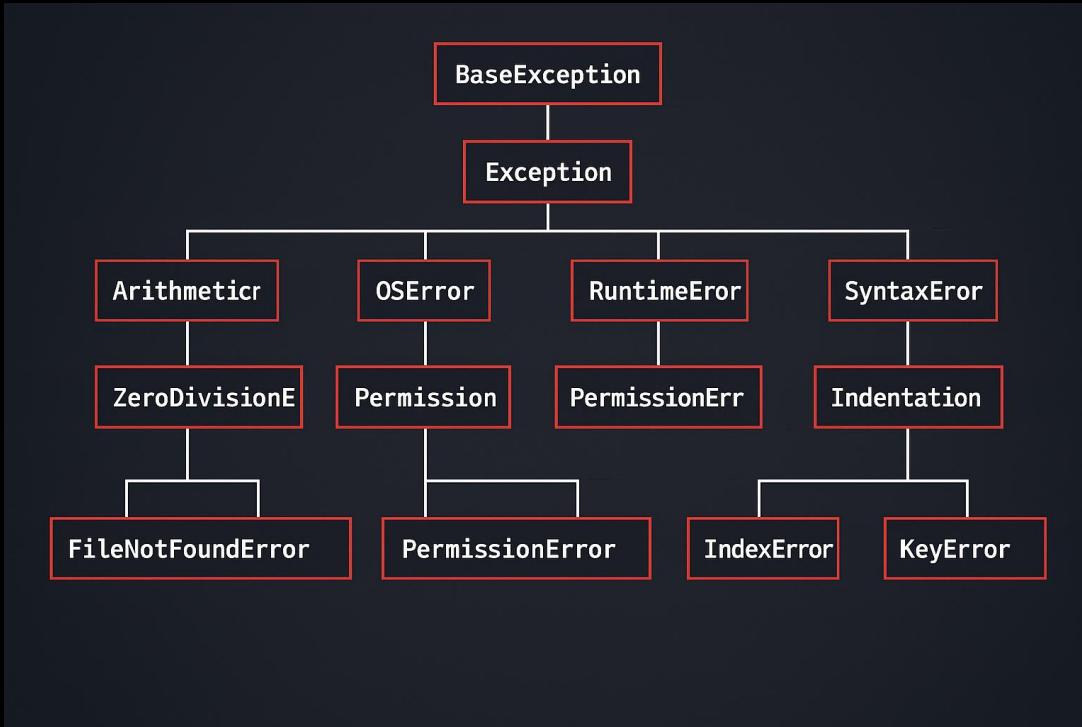
# MANEJO DE EXCEPCIONES EN PYTHON



## 🎯 Captura de Excepciones Específicas en Python

### Introducción

En esta lección profundizaremos en el manejo de errores en Python, aprendiendo a capturar excepciones específicas como `ZeroDivisionError` y `TypeError`, y cómo organizarlas correctamente para evitar que se omitan. También aprenderás por qué es importante el orden en que se escriben los bloques `except`. ✨



## 📌 Paso 1: Usamos el archivo base del proyecto

📄 Ruta y nombre del archivo:

`manejo-excepciones.py`

## 🚨 Paso 2: Capturar múltiples excepciones específicas

🔍 Descripción:

Manejamos los errores `ZeroDivisionError` y `TypeError` con bloques separados, imprimiendo quién los capturó y el tipo de error.

📄 Archivo: `manejo-excepciones.py`

```

resultado = None
a = '10'
b = 0

try:
    resultado = a / b
except ZeroDivisionError as e:
  
```

```
print(f'Ocurrió un error (ZeroDivisionError): {e}, tipo: {type(e)}')
except TypeError as e:
    print(f'Ocurrió un error (TypeError): {e}, tipo: {type(e)}')
```

### 💡 Explicación:

El orden importa: se revisan primero las clases hijas (`ZeroDivisionError`, `TypeError`). Así se captura el error correspondiente sin que se lo “robe” un bloque genérico.

---

## 🧩 Paso 3: Agregar una excepción genérica al final

### 🔍 Descripción:

Agregamos un bloque `except Exception` al final como respaldo para capturar cualquier otro error no específico.

### 📄 Archivo: `manejo-excepciones.py`

```
resultado = None
a = '10'
b = 0

try:
    resultado = a / b
except ZeroDivisionError as e:
    print(f'Ocurrió un error (ZeroDivisionError): {e}, tipo: {type(e)}')
except TypeError as e:
    print(f'Ocurrió un error (TypeError): {e}, tipo: {type(e)}')
except Exception as e:
    print(f'Ocurrió un error (Exception): {e}, tipo: {type(e)}')

print(f'Resultado: {resultado}')
print('Continuamos...')
```

### 💡 Explicación:

Colocar `Exception` al final asegura que solo se use si las otras excepciones no aplican. Así mantenemos un manejo más fino y estructurado de errores.

---

## ⌚ Paso 4: Probar distintos valores y verificar la excepción que se lanza

## 💡 Descripción:

Modificamos las variables `a` y `b` para generar diferentes errores y observar quién los captura.

### 📄 Archivo: `manejo-excepciones.py`

```
# Casos de prueba:  
a = 10  
b = 0 # Lanza ZeroDivisionError  
  
# o prueba con:  
# a = '10'  
# b = 2 # Lanza TypeError  
  
# o prueba con:  
# a = 10  
# b = 2 # No lanza excepción
```

## 📘 Explicación:

Según los valores que coloques, se lanzará una excepción distinta o ninguna. Así puedes comprobar cómo cada bloque `except` responde según el caso.

## ✅ Conclusión

En esta lección aprendiste cómo manejar múltiples excepciones específicas y una genérica. Comprendiste la importancia del orden de los bloques `except` y cómo cada uno puede capturar un tipo de error diferente.

Esto te permitirá escribir programas más seguros, controlados y fáciles de depurar. 🤓

¡Sigue practicando! Estás cada vez más cerca de dominar el manejo de errores en Python. 🚀 💡

Sigue adelante con tu aprendizaje 🚀, ¡el esfuerzo vale la pena!

¡Saludos! 🙌

Ing. Marcela Gamiño e Ing. Ubaldo Acosta

Fundadores de [GlobalMentoring.com.mx](#)