



Programá
tu futuro



Municipalidad de
Tres de Febrero

www.tresdefebrero.gov.ar/tecno3f

#ProgramáTuFuturo



Programá
tu futuro



Municipalidad de
Tres de Febrero



Municipalidad de
Tres de Febrero



Programá
tu futuro

PYTHON INTERMEDIO

¡Les damos la bienvenida!

</>





Municipalidad de
Tres de Febrero



Programá
tu futuro

EXCEPCIONES

CLASE 2



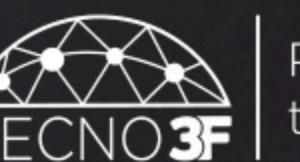
¿QUÉ ES?

En Python, una excepción es un evento que ocurre durante la ejecución de un programa que indica que algo ha salido mal. Las excepciones pueden ser causadas por una variedad de factores, como errores de sintaxis, errores de ejecución o errores de lógica.

Cuando ocurre una excepción, el flujo de ejecución del programa se detiene y se genera una excepción. La excepción puede ser manejada por el programa, o puede provocar que el programa se detenga.



TIPOS



Programá
tu futuro



Municipalidad de
Tres de Febrero

Principales

Si bien existen muchas más, les compartimos las más comunes

TypeError: Ocurre cuando se aplica una operación o función a un dato del tipo inapropiado.

IndexError: Ocurre cuando se intenta acceder a una secuencia con un índice que no existe.

KeyError: Ocurre cuando se intenta acceder a un diccionario con una clave que no existe.

ZeroDivisionError: Ocurre cuando se intenta dividir por cero.

<https://docs.python.org/3/library/exceptions.html>

Ejemplos

TypeError: Por ejemmplo, cuando intentamos sumar un string y un int

```
IONES > type.py > ...
def sumar(a, b):
    return a + b

resultado = sumar("hola", 5)
```

Ejemplos

IndexError: Este es bastante claro...

```
lista = [1, 2, 3]  
  
print(lista[5])
```

Ejemplos

KeyError: Al igual que el anterior.. este tambien es bastante claro.

```
diccionario = {"nombre": "Juan", "edad": 30}

print(diccionario["apellido"])
```

Ejemplos

ZeroDivisionError: Cuando queremos dividir por cero

```
resultado = 10 / 0
print(resultado)
```

TRY EXCEPT



Programá
tu futuro



Municipalidad de
Tres de Febrero

Try, Except

El bloque **try** es el bloque con las sentencias que quieras ejecutar. Sin embargo, podrían llegar a haber errores de ejecución y el bloque se dejará de ejecutarse.

El bloque **except** se ejecutará cuando el bloque try falle debido a un error. Este bloque contiene sentencias que generalmente nos dan un contexto de lo que salió mal en el bloque try

```
# try:  
#     bloque código 1  
# except excepción:  
#     bloque código 2
```

Siempre deberías de mencionar el tipo de error que se espera, como una excepción dentro del bloque

Podrías usar except sin especificar el **tipo de error**. Pero no es una práctica recomendable

Ejemplo

Así veríamos si lo usamos sin el tipo de error

```
8  try:
9      resultado = 10 / "asd"
10 except :
11     print(f"Ha ocurrido un error.")
12
13
```

PROBLEMAS

SALIDA

CONSOLA DE DEPURACIÓN

TERMINAL

PUERTOS

```
garo-pc@garopc-MS-7C52:~/Documentos/Tecno3F/Python/2024,
/Tecno3F/Python/2024/ls-2024/excepciones/try_except.py
Ha ocurrido un error.
```

Como podemos observar, no nos da información específica del tipo de error que está ocurriendo

Ejemplo

Asi veriamos si lo usamos con el tipo de error

```
8  try:
9      resultado = 10 / "asd"
10     print(resultado)
11 except TypeError:
12     print(f"Ha ocurrido un error.")
13
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
/bin/python3.11 /home/garo-pc/Documentos/Tecno3F/Python/2024/1s-2024/excepciones/try_except.py
• garo-pc@garopc-MS-7C52:~/Documentos/Tecno3F/Python/2024/1s-2024/excepciones/try_except.py
Ha ocurrido un error.
```

Como podemos ver, aca sabemos por que motivo esta fallando, esto nos da el pie a poder usar **multiples except**.

Ejemplo

Asi veriamos si lo capturamos el error y luego mostramos informacion de el

```
8  try:
9      resultado = 10 / "asd"
10     print(resultado)
11 except Exception as e:
12     print(f"Ha ocurrido un error. {e}")
13
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
garo-pc@garopc-MS-7C52:~/Documentos/Tecno3F/Python/2024/ls-2024$ /bin/python3
/Tecno3F/Python/2024/ls-2024/excepciones/try_except.py
Ha ocurrido un error. unsupported operand type(s) for /: 'int' and 'str'
garo-pc@garopc-MS-7C52:~/Documentos/Tecno3F/Pvthon/2024/ls-2024$
```

Esta es una de las formas mas usadas, ya que nos da un poco mas de informacion sobre los diversos error que podrian ocurrir.

ELSE



Programá
tu futuro



Municipalidad de
Tres de Febrero

Continuamos

El bloque else, solo se ejecutara si el bloque try se ejecuta sin errores

```
# try:  
#     bloque código 1  
# except excepción:  
#     bloque código 2  
# else:  
#     bloque código 3  
# Esto se ejecutara si el bloque try se ejecuta sin errores
```

El codigo utilizado en el bloque else, suele ser una continuacion del codigo del bloque try, por ej, si en el try intentamos conectarnos a una base de datos, ssi no hay errores, podemos en el else seguir con los comando a usar

Ejemplo

Ejemplo sencillo de su uso

```
11  try:
12      resultado = 10 / 5
13      print("Ejecutado sin errores \n")
14  except Exception as e:
15      print(f"Ha ocurrido un error. {e}")
16  else:
17      print(resultado)
```

PROBLEMAS

SALIDA

CONSOLA DE DEPURACIÓN

TERMINAL

PUERTOS

```
garo-pc@garopc-MS-7C52:~/Documentos/Tecno3F/Python/2
/Tecno3F/Python/2024/ls-2024/excepciones/else.py
Ejecutado sin errores
```

2.0

FINALLY



Programá
tu futuro



Municipalidad de
Tres de Febrero

Es el fin...

El bloque finallu se ejecuta siempre, ocurra o no error

```
# try:  
#     bloque código 1  
# except excepción:  
#     bloque código 2  
# else:  
#     bloque código 3  
#     Esto se ejecutara si el bloque try se ejecuta sin errores  
# finally:  
#     bloque código 4  
#     Esto se ejecutara siempre
```

Ejemplo

Aca podemos ver al bloque ejecutandose

```
12  try:
13      resultado = 10 / 5
14      print("Ejecutado sin errores \n")
15  except Exception as e:
16      print(f"Ha ocurrido un error. {e}")
17 else:
18     print(resultado)
19 finally:
20     print("\nMe ejecuto siempre")
21
```

PROBLEMAS	SALIDA	CONSOLA DE DEPURACIÓN	TERMINAL	P
• garo-pc@garopc-MS-7C52:~/Documentos/Tecno3F/Python/Tecno3F/Python/2024/ls-2024/excepciones/fin.py				
	Ejecutado sin errores			
	2.0			
	Me ejecuto siempre			

```
12  try:
13      resultado = 10 / 0
14      print("Ejecutado sin errores \n")
15  except Exception as e:
16      print(f"Ha ocurrido un error. {e}")
17 else:
18     print(resultado)
19 finally:
20     print("\nMe ejecuto siempre")
21
```

PROBLEMAS	SALIDA	CONSOLA DE DEPURACIÓN	TERMINAL	P
• garo-pc@garopc-MS-7C52:~/Documentos/Tecno3F/Python/Tecno3F/Python/2024/ls-2024/excepciones/fin.py				
		/bin/python3.11 /home/garo-pc/Documentos/Tecno3F/Python/Tecno3F/Python/2024/ls-2024/excepciones/fin.py		
		Ha ocurrido un error. division by zero		
	Me ejecuto siempre			

PARA PRACTICAR



Programá
tu futuro



Municipalidad de
Tres de Febrero

Practica.. Practica.. Practica

Escribe un programa que intente dividir dos números. Si el segundo número es cero, captura la excepción `ZeroDivisionError` y muestra un mensaje de error al usuario.

Escribe un programa que intente sumar un número y una cadena. Si se produce un error de tipo, captura la excepción `TypeError` y muestra un mensaje de error al usuario.

Escribe un programa que intente acceder a una clave que no existe en un diccionario. Si se produce una excepción `KeyError`, captura la excepción y muestra

Escribe un programa que intente abrir un archivo que no existe. Si se produce una excepción `FileNotFoundException`, captura la excepción y muestra un mensaje de error al usuario. Sin embargo, también intenta crear el archivo si no existe.

Escribe un programa que intente dividir dos números. Si el segundo número es cero, captura la excepción `ZeroDivisionError`. Si el primer número es un número no válido, captura la excepción `ValueError`. En cualquier caso, muestra un mensaje de error al usuario.

¡MUCHAS GRACIAS!



Programá
tu futuro



Municipalidad de
Tres de Febrero