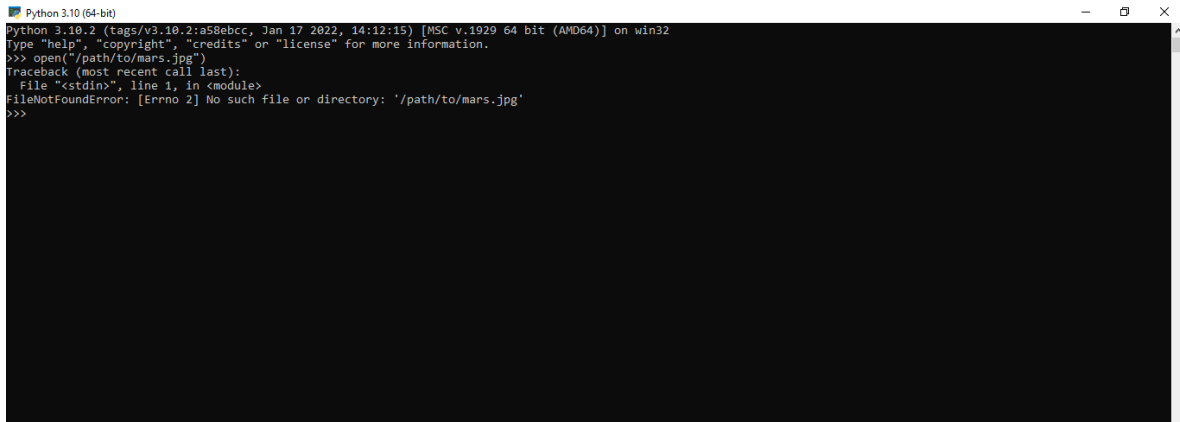


Kata módulo 10 “Manejo de errores”

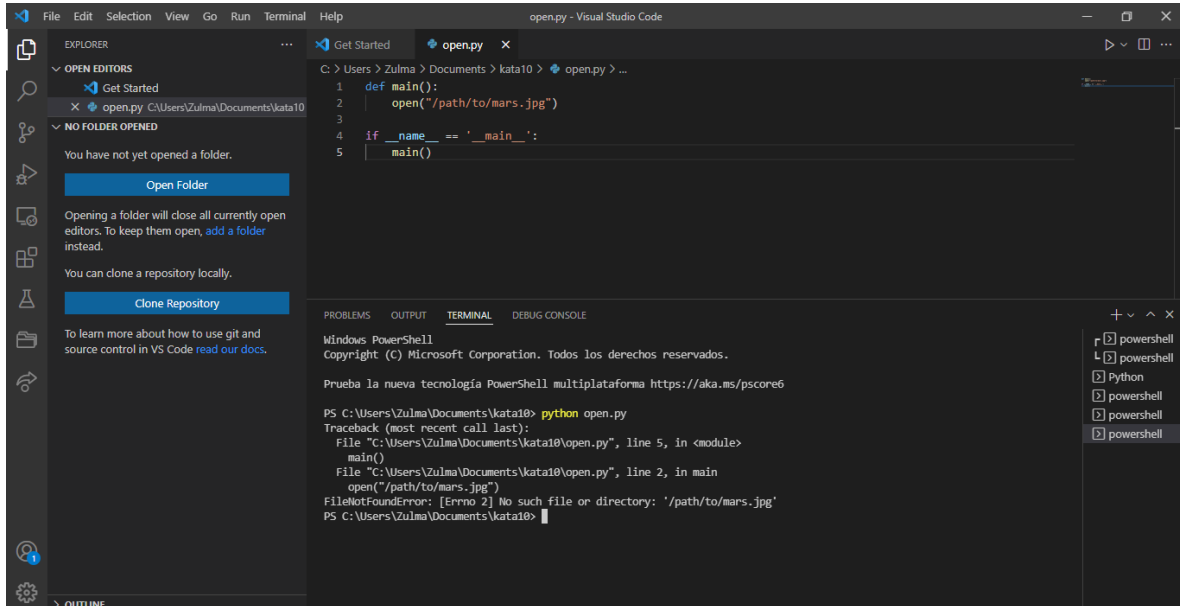
Tracebacks

- Si intentamos en un notebook, abrir un archivo inexistente sucede lo siguiente:



```
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> open("/path/to/mars.jpg")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: '/path/to/mars.jpg'
>>>
```

- Creamos archivo, Se trata de una sola función main() que abre el archivo inexistente.



```
File Edit Selection View Go Run Terminal Help
open.py - Visual Studio Code

EXPLORER
  OPEN EDITORS
    Get Started
    open.py C:\Users\Zulma\Documents\kata10
  NO FOLDER OPENED
    You have not yet opened a folder.
    Open Folder
    Opening a folder will close all currently open editors. To keep them open, add a folder instead.
    You can clone a repository locally.
    Clone Repository
    To learn more about how to use git and source control in VS Code read our docs.

  open.py
1 def main():
2     open("/path/to/mars.jpg")
3
4 if __name__ == '__main__':
5     main()

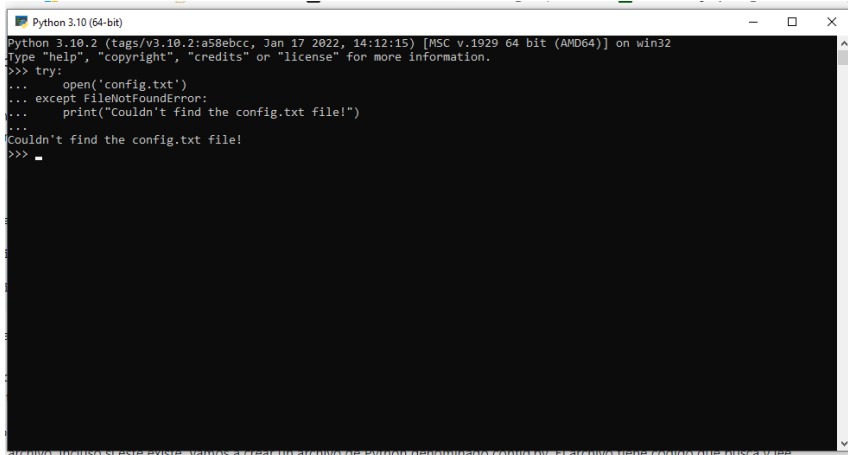
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\Zulma\Documents\kata10> python open.py
Traceback (most recent call last):
  File "C:\Users\Zulma\Documents\kata10\open.py", line 5, in <module>
    main()
  File "C:\Users\Zulma\Documents\kata10\open.py", line 2, in main
    open("/path/to/mars.jpg")
FileNotFoundError: [Errno 2] No such file or directory: '/path/to/mars.jpg'
PS C:\Users\Zulma\Documents\kata10>
```

Controlando las excepciones

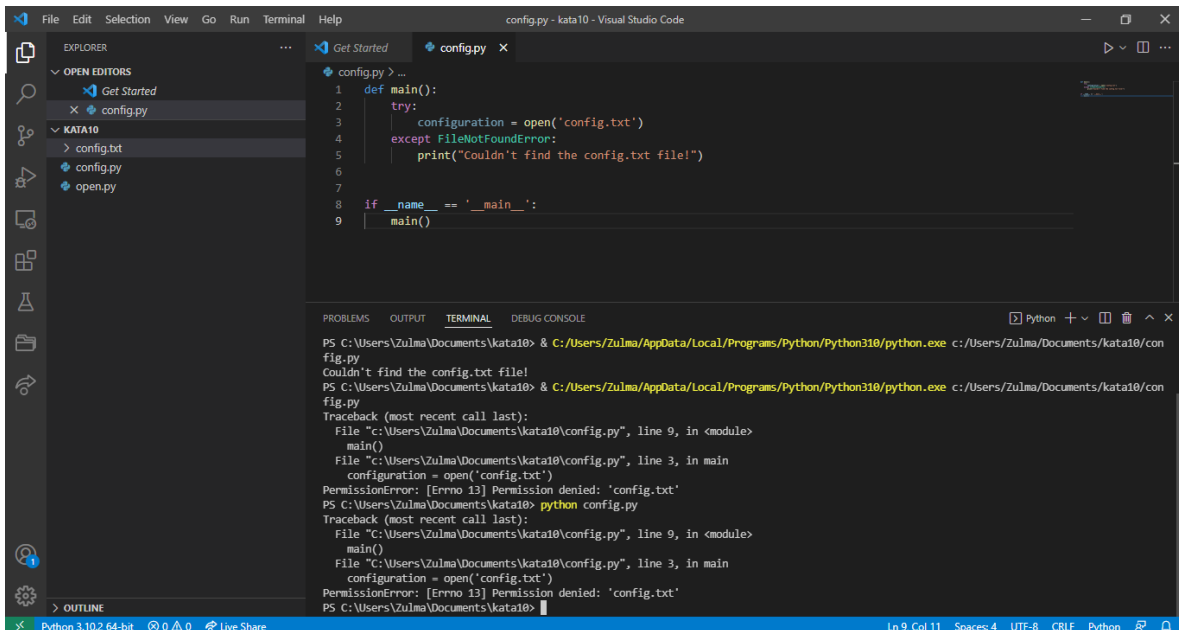
- Controlar excepción con bloque de try y except.



```
Python 3.10 (64-bit)
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> try:
...     open('config.txt')
... except FileNotFoundError:
...     print("Couldn't find the config.txt file!")
...
Couldn't find the config.txt file!
>>> _
```

archivo, incluso si este existe, vamos a crear un archivo de Python denominado config.py. El archivo tiene código que busca y lee

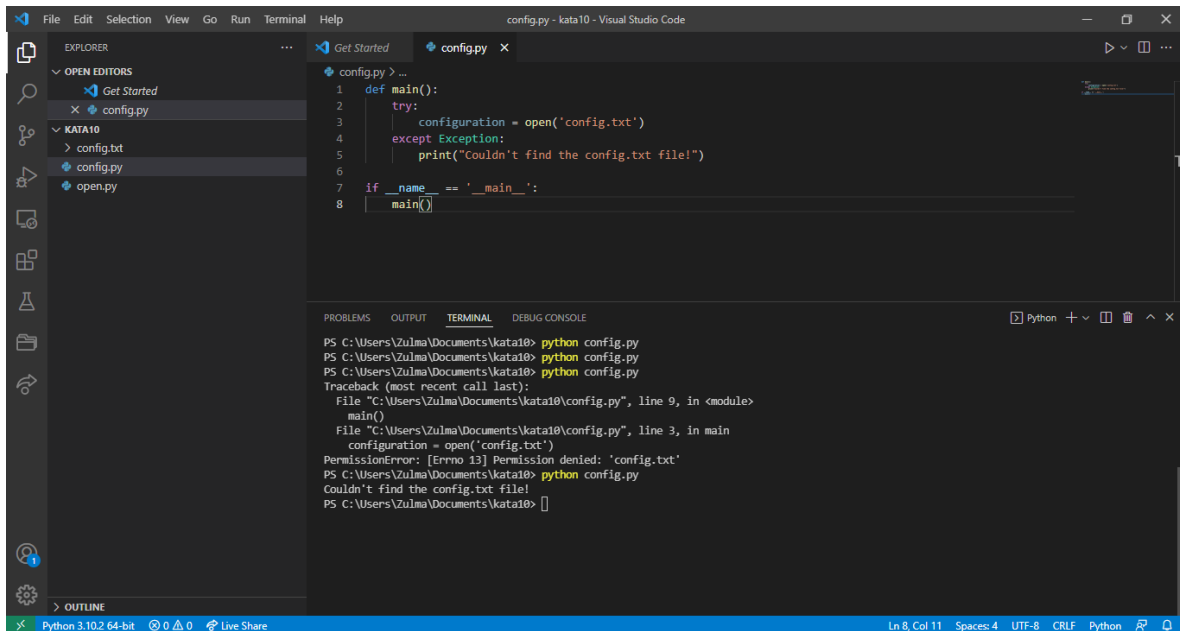
- crear un archivo de Python denominado config.py. El archivo tiene código que busca y lee el archivo de configuración del sistema de navegación, creamos un directorio denominado config.txt. Intentaremos llamar al archivo config.py para ver un error nuevo



```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except FileNotFoundError:
5         print("Couldn't find the config.txt file!")
6
7
8 if __name__ == '__main__':
9     main()

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
Python
PS C:\Users\Zulma\Documents\kata10> & C:\Users\Zulma\AppData\Local\Programs\Python\Python310\python.exe c:/Users/Zulma/Documents/kata10/config.py
Couldn't find the config.txt file!
PS C:\Users\Zulma\Documents\kata10> & C:\Users\Zulma\AppData\Local\Programs\Python\Python310\python.exe c:/Users/Zulma/Documents/kata10/config.py
Traceback (most recent call last):
  File "c:\Users\Zulma\Documents\kata10\config.py", line 9, in <module>
    main()
  File "c:\Users\Zulma\Documents\kata10\config.py", line 3, in main
    configuration = open('config.txt')
PermissionError: [Errno 13] Permission denied: 'config.txt'
PS C:\Users\Zulma\Documents\kata10> python config.py
Traceback (most recent call last):
  File "C:\Users\Zulma\Documents\kata10\config.py", line 9, in <module>
    main()
  File "C:\Users\Zulma\Documents\kata10\config.py", line 3, in main
    configuration = open('config.txt')
PermissionError: [Errno 13] Permission denied: 'config.txt'
PS C:\Users\Zulma\Documents\kata10>
```

- Controlar este error sería detectar todas las excepciones posibles para evitar un traceback, ejecutamos el código en el mismo lugar donde existe el archivo config.txt con permisos incorrectos



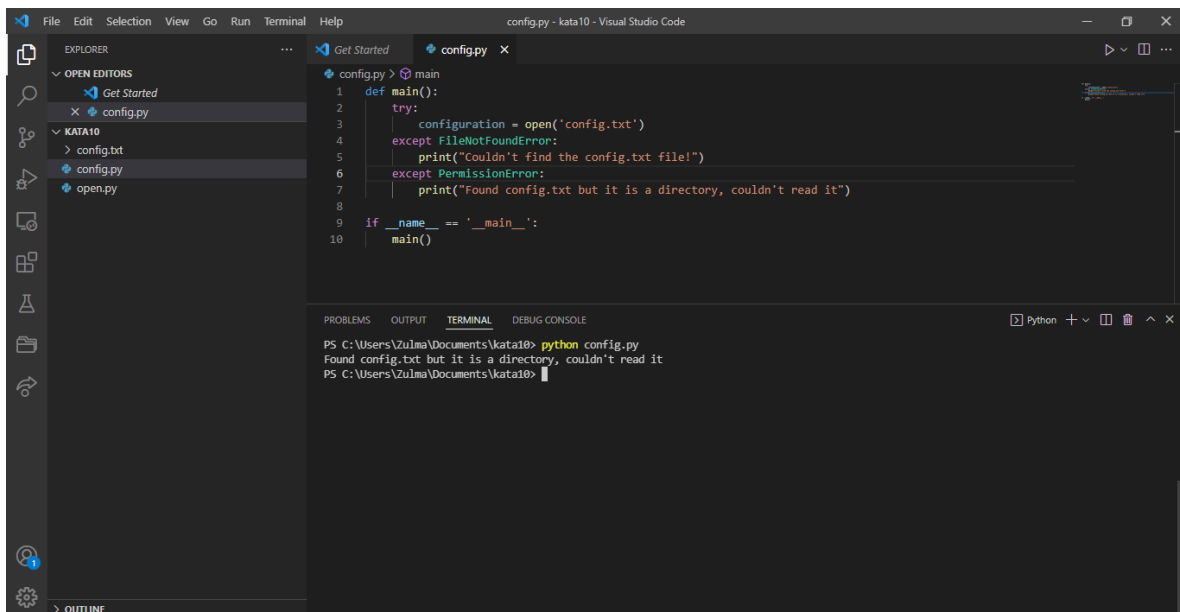
The screenshot shows the Visual Studio Code interface with a file explorer on the left and a code editor in the center. The code editor displays a Python script named `config.py` with the following content:

```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except Exception:
5         print("Couldn't find the config.txt file!")
6
7 if __name__ == '__main__':
8     main()
```

The terminal at the bottom shows the output of running the script:

```
PS C:\Users\Zulma\Documents\kata10> python config.py
PS C:\Users\Zulma\Documents\kata10> python config.py
PS C:\Users\Zulma\Documents\kata10> python config.py
Traceback (most recent call last):
  File "C:\Users\Zulma\Documents\kata10\config.py", line 9, in <module>
    main()
  File "C:\Users\Zulma\Documents\kata10\config.py", line 3, in main
    configuration = open('config.txt')
PermissionError: [Errno 13] Permission denied: 'config.txt'
PS C:\Users\Zulma\Documents\kata10> python config.py
Couldn't find the config.txt file!
PS C:\Users\Zulma\Documents\kata10>
```

- Vamos a corregir este fragmento de código para abordar todas estas frustraciones. Revertiremos la detección de `FileNotFoundError` y luego agregamos otro bloque `except` para detectar `PermissionError`



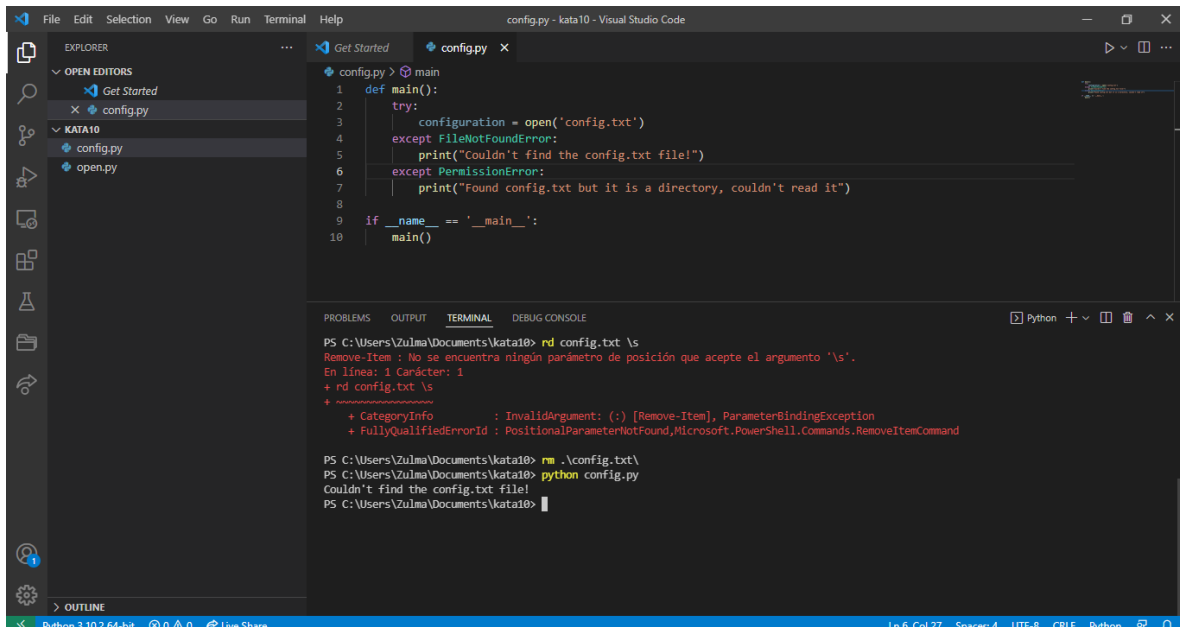
The screenshot shows the Visual Studio Code interface with the same file explorer and code editor. The code editor now displays the updated Python script:

```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except FileNotFoundError:
5         print("Couldn't find the config.txt file!")
6     except PermissionError:
7         print("Found config.txt but it is a directory, couldn't read it")
8
9 if __name__ == '__main__':
10     main()
```

The terminal at the bottom shows the output of running the script:

```
PS C:\Users\Zulma\Documents\kata10> python config.py
Found config.txt but it is a directory, couldn't read it
PS C:\Users\Zulma\Documents\kata10>
```

- Eliminamos el archivo config.txt para asegurarnos de que se alcanza el primer bloque except en su lugar



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows the file structure with 'config.py' selected. The Editor pane shows the code for 'config.py':

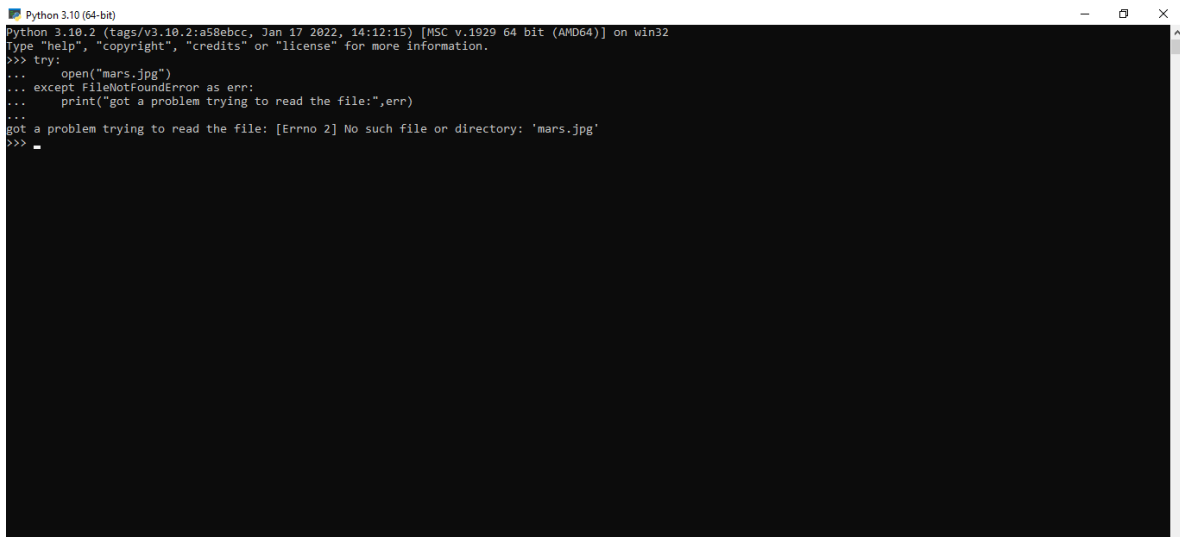
```
1 def main():
2     try:
3         configuration = open('config.txt')
4     except FileNotFoundError:
5         print("Couldn't find the config.txt file!")
6     except PermissionError:
7         print("Found config.txt but it is a directory, couldn't read it")
8
9 if __name__ == '__main__':
10     main()
```

The Terminal pane at the bottom shows the execution of the script in a PowerShell prompt:

```
PS C:\Users\Zulma\Documents\kata10> rd config.txt \s
Remove-Item : No se encuentra ningún parámetro de posición que acepte el argumento '\s'.
En línea: 1 Carácter: 1
+ rd config.txt \s
~
+ CategoryInfo          : InvalidArgument: (:) [Remove-Item], ParameterBindingException
+ FullyQualifiedErrorId : PositionalParameterNotFound,Microsoft.PowerShell.Commands.RemoveItemCommand

PS C:\Users\Zulma\Documents\kata10> rm .\config.txt\
PS C:\Users\Zulma\Documents\kata10> python config.py
Couldn't find the config.txt file!
PS C:\Users\Zulma\Documents\kata10>
```

- Si necesitas acceder al error asociado a la excepción, debes actualizar la línea except para incluir la palabra clave as. Esta técnica es práctica si una excepción es demasiado genérica y el mensaje de error puede ser útil



The screenshot shows a Python 3.10.2 (64-bit) terminal window. The code being executed is:

```
>>> try:
...     open("mars.jpg")
... except FileNotFoundError as err:
...     print("got a problem trying to read the file:",err)
...
got a problem trying to read the file: [Errno 2] No such file or directory: 'mars.jpg'
>>>
```

- as err significa que err se convierte en una variable con el objeto de excepción como valor.

```
Python 3.10 (64-bit)
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> try:
...     open("config.txt")
... except OSError as err:
...     if err.errno==2:
...         print("Couldn't find the config.txt file!")
...     elif err.errno==13:
...         print("Found config.txt but couldn't read it")
...
Couldn't find the config.txt file!
>>>
```

Generación de excepciones

- Vamos a crear una función que, con base al número de astronautas, pueda calcular la cantidad de agua quedará después de un día o más

```
Python 3.10 (64-bit)
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> def water_left(astronauts, water_left, days_left):
...     daily_usage = astronauts * 11
...     total_usage = daily_usage * days_left
...     total_water_left = water_left - total_usage
...     return f"Total water left after {days_left} days is: {total_water_left} liters"
...
>>> water_left(5, 100, 2)
'Total water left after 2 days is: -10 liters'
>>>
```

- Generar una excepción en la función `water_left()` para alertar de la condición de error:

```
Python 3.10 (64-bit)
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> def water_left(astronauts, water_left, days_left):
...     daily_usage = astronauts * 11
...     total_usage = daily_usage * days_left
...     total_water_left = water_left - total_usage
...     if total_water_left < 0:
...         raise RuntimeError(f"There is not enough water for {astronauts} astronauts after {days_left} days!")
...     return f"Total water left after {days_left} days is: {total_water_left} liters"
...
>>> water_left(5, 100, 2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 6, in water_left
RuntimeError: There is not enough water for 5 astronauts after 2 days!
>>> _
```

- pasamos argumentos que no sean enteros para comprobar la salida de error

```
Python 3.10 (64-bit)
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> def water_left(astronauts, water_left, days_left):
...     daily_usage = astronauts * 11
...     total_usage = daily_usage * days_left
...     total_water_left = water_left - total_usage
...     if total_water_left < 0:
...         raise RuntimeError(f"There is not enough water for {astronauts} astronauts after {days_left} days!")
...     return f"Total water left after {days_left} days is: {total_water_left} liters"
...
>>> try:
...     water_left(5, 100, 2)
... except RuntimeError as err:
...     print(err)
...
There is not enough water for 5 astronauts after 2 days!
>>> water_left("3", "200", None)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 3, in water_left
TypeError: can't multiply sequence by non-int of type 'NoneType'
>>> _
```

- Actualizaremos la función para que use TypeError, pero con un mensaje mejor

```
Python 3.10 (64-bit)
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> def water_left(astronauts, water_left, days_left):
...     for argument in [astronauts, water_left, days_left]:
...         try:
...             # If argument is an int, the following operation will work
...             argument / 10
...         except TypeError:
...             # TypeError will be raised only if it isn't the right type
...             # Raise the same exception but with a better error message
...             raise TypeError(f"All arguments must be of type int, but received: '{argument}'")
...     daily_usage = astronauts * 11
...     total_usage = daily_usage * days_left
...     total_water_left = water_left - total_usage
...     if total_water_left < 0:
...         raise RuntimeError(f"There is not enough water for {astronauts} astronauts after {days_left} days!")
...     return f"Total water left after {days_left} days is: {total_water_left} liters"
...
>>> water_left("3", "200", None)
Traceback (most recent call last):
  File "<stdin>", line 5, in water_left
TypeError: unsupported operand type(s) for /: 'str' and 'int'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 9, in water_left
TypeError: All arguments must be of type int, but received: '3'
>>>
```