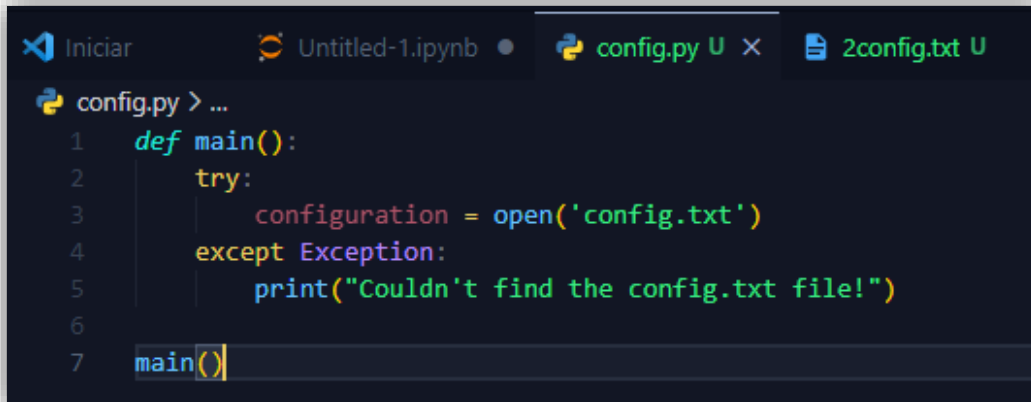


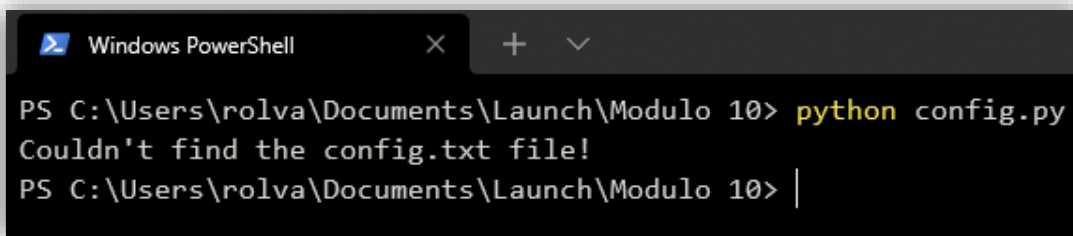
Controlando las excepciones

Una manera poco útil de controlar este error sería detectar todas las excepciones posibles para evitar un traceback. Para comprender por qué detectar todas las excepciones es problemático, probaremos actualizando la función main():



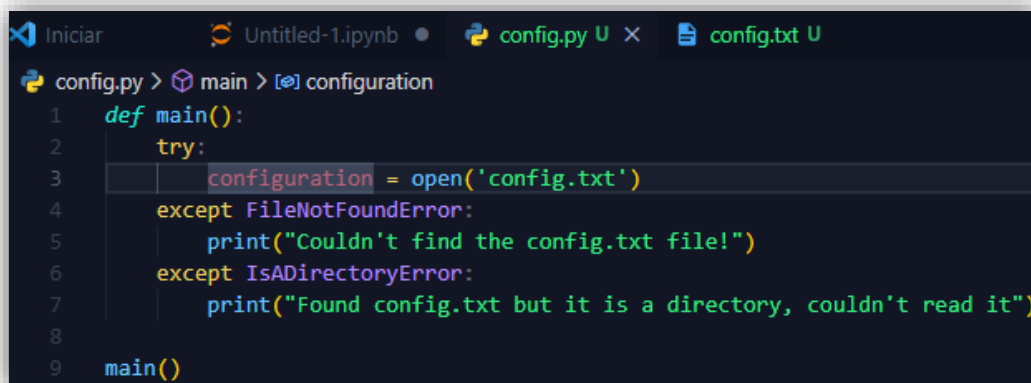
```
config.py > ...
1 def main():
2     try:
3         configuration = open('config.txt')
4     except Exception:
5         print("Couldn't find the config.txt file!")
6
7 main()
```

Ahora volvemos a ejecutar el código en el mismo lugar donde existe el archivo config.txt con permisos incorrectos:



```
Windows PowerShell
PS C:\Users\rolva\Documents\Launch\Modulo 10> python config.py
Couldn't find the config.txt file!
PS C:\Users\rolva\Documents\Launch\Modulo 10> |
```

Vamos a corregir este fragmento de código para abordar todas estas frustraciones



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

Cambiamos un poco más la configuración para cubrir más errores.

```
config.py > main
1 def main():
2     try:
3         configuration = open('config.txt')
4     except FileNotFoundError:
5         print("Couldn't find the config.txt file!")
6     except IsADirectoryError:
7         print("Found config.txt but it is a directory, couldn't read it")
8     except (BlockingIOError, TimeoutError):
9         print("Filesystem under heavy load, can't complete reading configuration file")
10
11 main()
```

Generación de excepciones

Los astronautas limitan su uso de agua a unos 11 litros al día. 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:

```
generacionDeExepciones.py > ...
1 def water_left(astronauts, water_left, days_left):
2     daily_usage = astronauts * 11
3     total_usage = daily_usage * days_left
4     total_water_left = water_left - total_usage
5     return f"Total water left after {days_left} days is: {total_water_left} liters"
6
7 print(water_left(5, 100, 2))
```

Probemos con cinco astronautas, 100 litros de agua sobrante y dos días:

```
Windows PowerShell
PS C:\Users\rolva\Documents\Launch\Modulo 10> python generacionDeExepciones.py
Total water left after 2 days is: -10 liters
PS C:\Users\rolva\Documents\Launch\Modulo 10> |
```

Si eres un ingeniero(a) que programa el sistema de navegación, podrías generar una excepción en la función `water_left()` para alertar de la condición de error:

```
Inicio  Untitled-1.ipynb  config.py  generacionDeExepciones.py  config.txt
generacionDeExepciones.py > water_left
1 def water_left(astronauts, water_left, days_left):
2     daily_usage = astronauts * 11
3     total_usage = daily_usage * days_left
4     total_water_left = water_left - total_usage
5     if total_water_left < 0:
6         raise RuntimeError(f"There is not enough water for {astronauts} astronauts after {days_left} days!")
7     return f"Total water left after {days_left} days is: {total_water_left} liters"
8
9 print(water_left(5, 100, 2))
```

Ahora volvemos a ejecutarlo

```
Windows PowerShell
PS C:\Users\rolva\Documents\Launch\Modulo 10> python generacionDeExepciones.py
Traceback (most recent call last):
  File "C:\Users\rolva\Documents\Launch\Modulo 10\generacionDeExepciones.py", line 9, in <module>
    print(water_left(5, 100, 2))
  File "C:\Users\rolva\Documents\Launch\Modulo 10\generacionDeExepciones.py", line 6, in water_left
    raise RuntimeError(f"There is not enough water for {astronauts} astronauts after {days_left} days!")
RuntimeError: There is not enough water for 5 astronauts after 2 days!
PS C:\Users\rolva\Documents\Launch\Modulo 10> |
```

Cambiamos los valores de entrada para le método

```
water_left("3", "200", None)
```

Y se obtiene el siguiente error.

```
Windows PowerShell
PS C:\Users\rolva\Documents\Launch\Modulo 10> python generacionDeExepciones.py
Traceback (most recent call last):
  File "C:\Users\rolva\Documents\Launch\Modulo 10\generacionDeExepciones.py", line 10, in <module>
    water_left("3", "200", None)
  File "C:\Users\rolva\Documents\Launch\Modulo 10\generacionDeExepciones.py", line 3, in water_left
    total_usage = daily_usage * days_left
TypeError: can't multiply sequence by non-int of type 'NoneType'
PS C:\Users\rolva\Documents\Launch\Modulo 10> |
```

Mejoramos un poco el código para que el error sea un poco más descriptivo.

```
generacionDeExepciones.py U X
generacionDeExepciones.py > ...
1 def water_left(astronauts, water_left, days_left):
2     for argument in [astronauts, water_left, days_left]:
3         try:
4             # If argument is an int, the following operation will work
5             argument / 10
6         except TypeError:
7             # TypeError will be raised only if it isn't the right type
8             # Raise the same exception but with a better error message
9             raise TypeError(f"All arguments must be of type int, but received: '{argument}'")
10    daily_usage = astronauts * 11
11    total_usage = daily_usage * days_left
12    total_water_left = water_left - total_usage
13    if total_water_left < 0:
14        raise RuntimeError(f"There is not enough water for {astronauts} astronauts after {days_left} days!")
15    return f"Total water left after {days_left} days is: {total_water_left} liters"
16
17
18 water_left("3", "200", None)
```

Y ahora obtenemos un error más entendible.

```
Windows PowerShell
PS C:\Users\rolva\Documents\Launch\Modulo 10> python generacionDeExepciones.py
Traceback (most recent call last):
  File "C:\Users\rolva\Documents\Launch\Modulo 10\generacionDeExepciones.py", line 5, in water_left
    argument / 10
TypeError: unsupported operand type(s) for /: 'str' and 'int'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\rolva\Documents\Launch\Modulo 10\generacionDeExepciones.py", line 18, in <module>
    water_left("3", "200", None)
  File "C:\Users\rolva\Documents\Launch\Modulo 10\generacionDeExepciones.py", line 9, in water_left
    raise TypeError(f"All arguments must be of type int, but received: '{argument}'")
TypeError: All arguments must be of type int, but received: '3'
PS C:\Users\rolva\Documents\Launch\Modulo 10> |
```