Excepciones

Al ejecutar un programa, puede producirse un error por múltiples razones.

Por ejemplo, a veces no se puede controlar el contexto de ejecución: ¿Qué ocurre si queremos abrir un fichero que no existe?

In [1]:
```
f = open('no_existe.txt','r')
```

```
---------------------------------------------------------------------
----
FileNotFoundError                         Traceback (most recent call l
ast)
Cell In[1], line 1
----> 1 f = open('no_existe.txt','r')

File C:\ProgramData\anaconda3\Lib\site-packages\IPython\core\interactiv
eshell.py:286, in _modified_open(file, *args, **kwargs)
    279 if file in {0, 1, 2}:
    280     raise ValueError(
    281         f"IPython won't let you open fd={file} by default "
    282         "as it is likely to crash IPython. If you know what you
are doing, "
    283         "you can use builtins' open."
    284     )
--> 286 return io_open(file, *args, **kwargs)

FileNotFoundError: [Errno 2] No such file or directory: 'no_existe.txt'
```

El nombre técnico es "excepción", se produce una excepción y el programa termina. Hay muchas otras excepciones con las que ya nos hemos topado.

In [2]:
```
pares = range(0, 11, 2)    # pares = (0, 2, 4, 6, 8, 10)

print(pares[0])
print(pares[1])
print(pares[5])
print(pares[6])
```

```
0
2
10

---------------------------------------------------------------------------
IndexError                                Traceback (most recent call l
ast)
Cell In[2], line 5
      3 print(pares[1])
      4 print(pares[5])
----> 5 print(pares[6])

IndexError: range object index out of range
```

In [3]:
```
1/0
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call l
ast)
Cell In[3], line 1
----> 1 1/0

ZeroDivisionError: division by zero
```

Podemos hacer que nuestros programas controlen estos errores (estas excepciones) y puedan tomar medidas la respecto.

In [4]:
```
try:
    1/0
except:
    print('Esa división no tiene futuro...')
```

```
Esa división no tiene futuro...
```

In [5]:
```
pares = range(0, 11, 2)    # pares = (0, 2, 4, 6, 8, 10)

try:
    for i in range(10):
        print('accedo a {0}: {1}'.format(i,pares[i]))
except:
    print('pero no puedo seguir!!')
```

```
accedo a 0: 0
accedo a 1: 2
accedo a 2: 4
accedo a 3: 6
accedo a 4: 8
accedo a 5: 10
pero no puedo seguir!!
```

In [6]:
```
try:
    f = open('no_existe.txt', 'r')
except:
    print('El archivo no está presente... ¡pero el programa no se interr
```

```
El archivo no está presente... ¡pero el programa no se interrumpe!
```

He aquí una aplicación típica: tratamiento de un dato missing.

Tenemos una línea de un archivo `csv`, con datos numéricos separados por comas:

"2,4,36,-1,345,,23"

Deseamos convertirla en la lista de los números correspondientes. Pero hay algún dato missing o incluso alguna información no convertible en un entero:

In [17]:
```
linea = "2,4,NaN,10,345,,23"
datos = [int(cad) for cad in linea.split(",")]

print(datos)
```

```
---------------------------------------------------------------------
----
ValueError                                Traceback (most recent call l
ast)
Cell In[17], line 2
      1 linea = "2,4,NaN,10,345,,23"
----> 2 datos = [int(cad) for cad in linea.split(",")]
      4 print(datos)

Cell In[17], line 2, in <listcomp>(.0)
      1 linea = "2,4,NaN,10,345,,23"
----> 2 datos = [int(cad) for cad in linea.split(",")]
      4 print(datos)

ValueError: invalid literal for int() with base 10: 'NaN'
```

¿Qué podemos hacer?

In [16]:
```
def to_int(cad):
    try:
        return int(cad)
    except:
        return -1


linea = "2,4,NaN,10,345,,23"
datos = [to_int(n) for n in linea.split(",")]

print(datos)
```

```
[2, 4, -1, 10, 345, -1, 23]
```

El mecanismo de las excepciones también sirve para que nuestros programas indiquen que se ha producido un error, por ejemplo cuando los datos de entrada de una función no son los adecuados. En estos casos usamos `raise` . De esta forma, otros programas que estén usando nuestro código pueden responder a ese error.

In [7]:
```
import math

def circle(radius):
    """
    Function that computes the surface of a circle
    """
    if radius >= 0:
        sur = math.pi * radius ** 2
    else:
        raise Exception('Radio negativo')
    return sur
```

Como puedes ver, en `raise Exception('Radio negativo')` hemos creado el error que deseamos lanzar.

In [8]:
```
circle(4)
```

Out[8]:
```
50.26548245743669
```

In [9]:
```
circle(-4)
```

```
---------------------------------------------------------------------
----
Exception                                 Traceback (most recent call l
ast)
Cell In[9], line 1
----> 1 circle(-4)

Cell In[7], line 10, in circle(radius)
      8     sur = math.pi * radius ** 2
      9 else:
---> 10     raise Exception('Radio negativo')
     11 return sur

Exception: Radio negativo
```

In [10]:
```
try:
    for x in range(5, -5, -1):
        print('radio {0} area {1}'.format(x, circle(x)))
except:
    print('error')
```

```
radio 5 area 78.53981633974483
radio 4 area 50.26548245743669
radio 3 area 28.274333882308138
radio 2 area 12.566370614359172
radio 1 area 3.141592653589793
radio 0 area 0.0
error
```

In [11]:
```
try:
    for x in range(5,-5,-1):
        print('radio {0} area {1}'.format(x, circle(x)))
except Exception as e:
        print(e)
```

```
radio 5 area 78.53981633974483
radio 4 area 50.26548245743669
radio 3 area 28.274333882308138
radio 2 area 12.566370614359172
radio 1 area 3.141592653589793
radio 0 area 0.0
Radio negativo
```

También podemos usar esta versión ligeramente más sofisticada de `except` con las excepciones predefinidas

In [12]:
```
try:
    1/0
except Exception as e:
    print('Esa división no tiene futuro...')
    raise e
```

```
Esa división no tiene futuro...

---------------------------------------------------------------------
----
ZeroDivisionError                         Traceback (most recent call l
ast)
Cell In[12], line 5
      3 except Exception as e:
      4     print('Esa división no tiene futuro...')
----> 5     raise e

Cell In[12], line 2
      1 try:
----> 2     1/0
      3 except Exception as e:
      4     print('Esa división no tiene futuro...')

ZeroDivisionError: division by zero
```

In [13]:
```
l = range(20,25)
try:
    for i in range(10):
        print 'accedo a l[',i,']', l[i]
except Exception as e:
    print 'pero no puedo seguir!!'
    print(e)
```

```
  Cell In[13], line 4
    print 'accedo a l[',i,']', l[i]
    ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print
(...)?
```

In [14]:
```
try:
    f = open('no_existe.txt','r')
except Exception as e:
    print 'Se ha producido un error, pero puedo continuar ejecutando!!'
    print(e)
```

```
  Cell In[14], line 4
    print 'Se ha producido un error, pero puedo continuar ejecutando!!'
    ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print
(...)?
```

Un ejemplo adicional

Tratamiento de datos raros o missing

In [15]:
```
numeros_y_errores = [1, 2, 3, 4, "Error", 5, 6, 7, 8, None]

def media(numeros):
    suma = 0.0
    for x in numeros:
        try:
            suma = suma + x
        except:
            suma = suma + 0
    return suma / len(numeros)

print(media(numeros_y_errores))
```

```
3.6
```