Excepciones múltiples

En una misma pieza de código pueden ocurrir muchos errores distintos y quizá nos interese actuar de forma diferente en cada caso.

Para esas situaciones algo que podemos hacer es asignar una excepción a una variable.

De esta forma es posible analizar el tipo de error que sucede gracias a su identificador:

```
try:
    n = input("Introduce un número: ")  # no transformamos a número
    5/n
except Exception as e:  # guardamos la excepción como una variable e
    print("Ha ocurrido un error =>", type(e).__name__)
```

```
Introduce un número: 10
Ha ocurrido un error =>  TypeError
```

Cada error tiene un identificador único que curiosamente se corresponde con su tipo de dato. Aprovechándonos de eso podemos mostrar la clase del error utilizando la sintaxis:

```
print( type(e) )
```

```
<class 'TypeError'>
```

Es similar a conseguir el tipo (o clase) de cualquier otra variable o valor literal:

```
print(type(1))
print(type(3.14))
print(type([]))
print(type(()))
print(type({}))
```

```
<class 'int'>
<class 'float'>
<class 'list'>
<class 'tuple'>
<class 'dict'>
```

Como vemos siempre nos indica eso de "class" delante. Eso es porque en Python todo son clases, pero hablaremos de este concepto más adelante. Lo importante ahora es que podemos mostrar solo el nombre del tipo de dato (la clase) consultando su propiedas especial name de la siguiente forma:

```
print( type(e).__name__)
print(type(1).__name__)
print(type(3.14).__name__)
print(type([]).__name__)
```

```
print(type(()).__name__)
print(type({}).__name__)
```

```
TypeError
int
float
list
tuple
dict
```

Gracias a los identificadores de errores podemos crear múltiples comprobaciones, siempre que dejemos en último lugar la excepción por defecto *Excepcion* que engloba cualquier tipo de error (si la pusiéramos al principio las demas excepciones nunca se ejecutarían):

```
try:
    n = float(input("Introduce un número divisor: "))
    5/n
except TypeError:
    print("No se puede dividir el número entre una cadena")
except ValueError:
    print("Debes introducir una cadena que sea un número")
except ZeroDivisionError:
    print("No se puede dividir por cero, prueba otro número")
except Exception as e:
    print("Ha ocurrido un error no previsto", type(e).__name__ )
```

```
Introduce un número divisor: 0
No se puede dividir por cero, prueba otro número
```

Última edición: 26 de Septiembre de 2018