

## FUNDAMENTOS DE PROGRAMACIÓN EN PYTHON

[CC-BY-NC-SA] Leonardo Marco

### FPP-PE-P8. Control de excepciones

ALUMNO: Alejandro Bermúdez Gorozabel

1) Crea un script que pida un número al usuario y lo convierta a `int`.

- Si se obtiene un error de conversión capturará la excepción, informará del error y volverá a pedir el número.'
- Si no se encontraba dentro del rango, lanzará un excepción `ValueError`, la capturará, informará del error y volverá a pedir el número.
- Así hasta obtener un número correcto.

```
def pedir_numero(rango_minimo = 0, rango_maximo = 10):  
  
    while True:  
  
        try:  
  
            num = input(f"Introduce un número entre {rango_minimo} y {rango_maximo}: ")  
  
            numero = int(num)  
  
            if not (rango_minimo <= numero <= rango_maximo):  
  
                raise ValueError(f"El número debe estar entre {rango_minimo} y {rango_maximo}.")  
  
        return numero
```

```

except ValueError as e:

    print(f"Error: {e}. Inténtalo de nuevo.\n")

numero_usuario = pedir_numero()

print(f"Número correcto introducido: {numero_usuario}")

```

2) Modifica el script anterior para definir un nuevo tipo de excepción `OutOfRangeError` que herede de `ValueError` y lanzarla en caso de introducir un número fuera de rango.

```

class OutOfRangeError(ValueError):

    pass

def pedir_numero(rango_minimo=0, rango_maximo=10):

    while True:

        try:

            num = input(f"Introduce un número entre {rango_minimo} y {rango_maximo}: ")

            numero = int(num)

            if not (rango_minimo <= numero <= rango_maximo):

                raise OutOfRangeError(f"El número debe estar entre {rango_minimo} y {rango_maximo}.")

```

```

        return numero

    except OutOfRangeError as e:

        print(f"Error de rango: {e}. Inténtalo de nuevo.\n")

    except ValueError:

        print("Error: Debes introducir un número entero válido.
Inténtalo de nuevo.\n")

numero_usuario = pedir_numero()

print(f"Número correcto introducido: {numero_usuario}")

```

3) Modifica el script de la práctica anterior en el que había que adivinar un número para que cuando se introduzca un valor no numérico se capture el error y se informe del error.

```

import random

numero = int(input("Introduce un número mayor que 0: "))

numeroSecreto = int(random.randint(0, numero))

while True:

```

```
intento = input("Adivina el número (o pulsa 'q' para salir): ")

if intento.lower() == 'q':

    print(f"Has salido del juego. El número secreto era {numeroSecreto}.")

    break

try:

    intento = int(intento)

except ValueError:

    print("Error: Debes introducir un número entero válido.\n")

    continue

if intento < numeroSecreto:

    print("El número es mayor.")

elif intento > numeroSecreto:

    print("El número es menor.")

else:

    print(f"Has acertado el número {numeroSecreto}.")

    break
```

4) Ejecuta el script del ejemplo y razona cuál de los `print` comentados del 1 al 5 se ejecuta en cada caso:

1. Introduciendo los valores 10 y 5.
2. Introduciendo como primer valor `hola` (salta la excepción `ValueError`)
3. Introduciendo los valores 10 y 0 (salta la excepción `ZeroDivisionError`)

```
try:
    num1 = float(input("Introduzca num1: "))
    num2 = float(input("Introduzca num2: "))
    print("1. Números introducidos correctamente")          # PRINT 1
    num3 = num1/num2

except ValueError as e:
    print(f"2. Error capturado: {e}")                      # PRINT 2

else:
    print(f"3. Resultado: {num3}")                         # PRINT 3

finally:
    print("4. Reseteando números")                        # PRINT 4
    num1 = 0
    num2 = 0

print("5. Continuando script...")                         # PRINT 5
```

#### Caso 1:

Los dos números se convierten bien, así que se muestra el print 1.

La división 10/5 también funciona, así que entra en el else y se muestra el print 3.

Después se ejecuta siempre el finally, entonces aparece el print 4.

El programa continúa y se muestra el print 5.

#### Caso 2:

Al escribir "hola", falla la conversión a número y salta un `ValueError`.

El error lo captura el except, así que se muestra el print 2.

Después se ejecuta el finally, que muestra el print 4.

El programa sigue y aparece el print 5.

### **Caso 3:**

Los dos números se convierten bien, aparece el print 1.

El programa divide  $10/0$  y salta un `ZeroDivisionError`, pero no se captura porque solo el programa solo controla `ValueError`.

Se ejecuta el finally, mostrando el print 4.