4 – Funciones

Horas de Libre Configuración

Curso 2023 - 2024

¿Qué son?

- Secuencias de instrucciones identificadas por un nombre y que pueden incluir parámetros.
- Aporta las siguientes ventajas:
  - Modularidad.
  - Reutilización de código.
  - Reducción de errores.

Funciones predefinidas

- Proporcionadas por Python, se utilizan sin necesidad de definirlas.
- Ejemplos:
  - max, min, len, type.
  - int(), float(), str().

Módulos de funciones

- Bibliotecas de funciones que deben ser importarlas antes de su uso.

```
import <nombre_módulo>
```

- import crea un objeto que permite acceder a las funciones incluidas en el módulo.
- Listado de módulos aquí.

Ejemplos: math y random

import math	import random
math.sqrt(n) math.factorial(n) math.isnan(valor)	random.random() random.randint(a, b) random.choice(sec)

Definición de funciones

```
def <nombre>(<parámetros>):
    <instrucciones>
```

- Los parámetros son opcionales.
- Todas las funciones devuelven un valor.
  - El valor a devolver se indica con return.
  - Si no se indica nada, devuelve None.

Ejemplo

```
def suma(a, b):
    return a+b
```

- Se puede añadir más de una instrucción return (no es buena práctica. Es preferible un único punto de salida)

¿Y si necesito devolver más de un valor?

- Las tuplas permiten devolver más de un valor.

```
def ejemplo():
    return 1,2


x,y = ejemplo()
```

Paso de parámetros

- El paso de parámetros se puede hacer por posición o por nombre

```
def division(D, d):
    return D / d
resultado = division(8, 2)
resultado = division(d=2, D=8)
```

Valores por defecto

- Se puede definir un valor por defecto a un parámetro, convirtiéndolo en opcional.
- Un parámetro sin valor por defecto es obligatorio.

```
def p(base, exponente=2):
    return base ** exponente
resultado = p(2, 3) → 8
resultado = p(2)    → 4
```

Número variable de parámetros

- Se pone * delante del parámetro para indicar un número variable de valores de entrada.
- Los múltiples valores se guardan en una tupla.

```
def ejemplo(*valores):
```

Número variable de parámetros

```
def misuma(*numeros):
    s = 0
    for n in numeros:
        r += n
    return s
misuma(1, 3, 5, 2)
```

Recursividad

- Es la llamada de una función desde el propio cuerpo de la función.

DIRECTA	INDIRECTA

`def f():`
`  f()`
	`def f():`
	`  g()`
	`def g():`
	`  f()`