
📌 Definición de Función en Python

> Una **función** es un bloque de código reutilizable que realiza una tarea específica. Se puede definir una vez y usar muchas veces, pasándole datos (parámetros) y obteniendo un resultado (valor de retorno).

🧪 **Ejemplo:**

```
def calcular_igv(monto):
    return monto * 0.18

igv = calcular_igv(1000)
print(igv)  # Resultado: 180.0
```

📌 Definición de Método en Python

> Un **método** es una función **asociada a un objeto específico**. Cada tipo de objeto (como texto, listas, diccionarios) tiene sus propios métodos incorporados que permiten manipularlo.

🧪 **Ejemplo:**

```
texto = "abogado"
print(texto.upper())  # Resultado: "ABOGADO"
```

> Aquí `upper()` es un método del objeto `str`.

🧠 Comparación sencilla:

Función	Método
Se usa sola: `función()`	Se usa con un punto: `objeto.método()`
`print()`, `len()`, `range()`	`"texto".lower()`, `[1,2,3].append(4)`

✅ `ejercicio01.py` – Imprimir saludos con distintos formatos

Objetivo: Transformar en una función que reciba un nombre.

```
# Versión con función

def saludar(nombre):

    print("Hola, mi nombre es", nombre)

    print("Hola,", nombre)

    print("Hola " + nombre)

    print(f"Hola {nombre}")

    return nombre



saludar("Test")

print("bienvenido ", saludar("madrid"))
```

➡️ **Aquí** usas una **función** (`saludar`) que encapsula toda la lógica.
➡️ También introduces el **método** `str.format()` como alternativa a `f-strings`.

✅ `ejercicio02.py` – Suma de números y hora actual

Objetivo: Separar operaciones en funciones reutilizables.

```
from datetime import datetime

def sumar(a, b):
    return int(a) + int(b)

def mostrar_fecha_hora():
    ahora = datetime.now()
    print("Fecha y hora actual:", ahora)
    print("Solo la hora:", ahora.strftime("%H:%M"))   # método strftime()
```

```
    print("Solo la fecha:", ahora.strftime("%d/%m/%Y"))

a = input("Ingresa el primer número: ")
b = input("Ingresa el segundo número: ")
print(f"La suma es: {sumar(a, b)}")

mostrar_fecha_hora()
```

➡️ `strftime()` es un **método** del objeto `datetime`.
➡️ `sumar()` y `mostrar_fecha_hora()` son **funciones** que encapsulan lógica.

✅ `ejercicio03.py` – Comparación de valores

```
def comparar_valores(x, y):
    if x < y:
        print("x es menor que y")

comparar_valores(10, 20)
```

➡️ La función `comparar_valores()` permite reusar esa lógica con cualquier número.

✅ `ejercicio04.py` – Edad y validación de usuario

```
def verificar_edad(edad):
    if edad >= 18:
        print("Eres mayor de edad")
    else:
        print("Eres menor de edad")

def validar_login(usuario, clave):
    if usuario == "admin" and clave == "1234":
        print("Acceso permitido")
    else:
        print("Acceso denegado")

verificar_edad(int(input("¿Cuál es tu edad? ")))
validar_login("admin", "1234")
```

➡️ Mismo código pero más limpio y **modular**.

✅ `ejercicio05.py` – Listas, tuplas y diccionarios

Objetivo: Demostrar métodos (`append`) y funciones que los usen.

```
def mostrar_frutas(frutas):
    print(frutas[2])
    frutas.append("plátano")  # método
    print(frutas)

def mostrar_coordenadas(coordenadas):
    print(f"Latitud: {coordenadas[0]}, Longitud: {coordenadas[1]}")

def mostrar_persona(persona):
    print(persona["nombre"])
    persona["profesion"] = "Ingeniero"
    print(persona)

frutas = ["manzana", "pera", "naranja"]
mostrar_frutas(frutas)

coordenadas = (10.5, 20.3)
mostrar_coordenadas(coordenadas)

persona = {
    "nombre": "Marco",
    "edad": 55,
    "ciudad": "Lima"
}
mostrar_persona(persona)
```
