

# Sesión n.º 2022

**Miércoles 23 de julio**

AE 2

# Módulo 4: Programación Avanzada en Python

# Introducción

## Programación Orientada a Objetos

### Aprendizaje Esperado N° 2

Objetivos de hoy:

- Presentar el Aprendizaje Esperado N° 2 y sus temas
- Explorar en detalle dos pilares: Clases, Constructor y Atributos
- Realizar ejemplos prácticos en Visual Studio Code
- Proponer una actividad final y dinámica de refuerzo



# Aprendizaje Esperado N° 2

## Fundamentos de la Programación Orientada a Objetos

**Descripción breve:** Este bloque cubre los siguientes temas:

1

Definición y uso de clases

3

Encapsulación

Constructor y atributos

Herencia

5

Polimorfismo

Abstracción

7

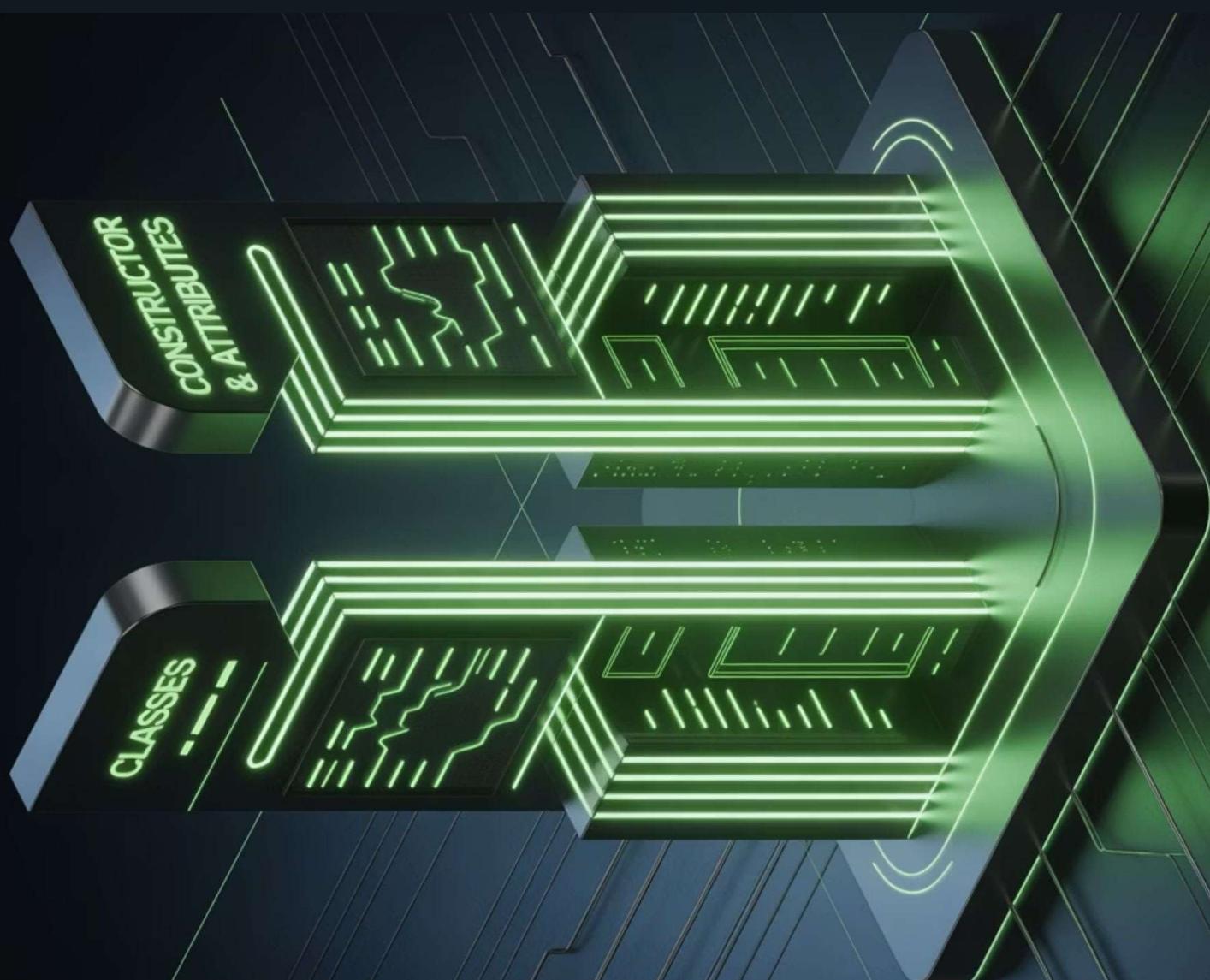
Métodos mágicos

Métodos estáticos y de clase

# Temas del Dí profundidad

3.1 ¿Qué es una clase

3.2 ¿Qué es un constructor y los atributos?



# 3.1 ¿Qué es una clase?

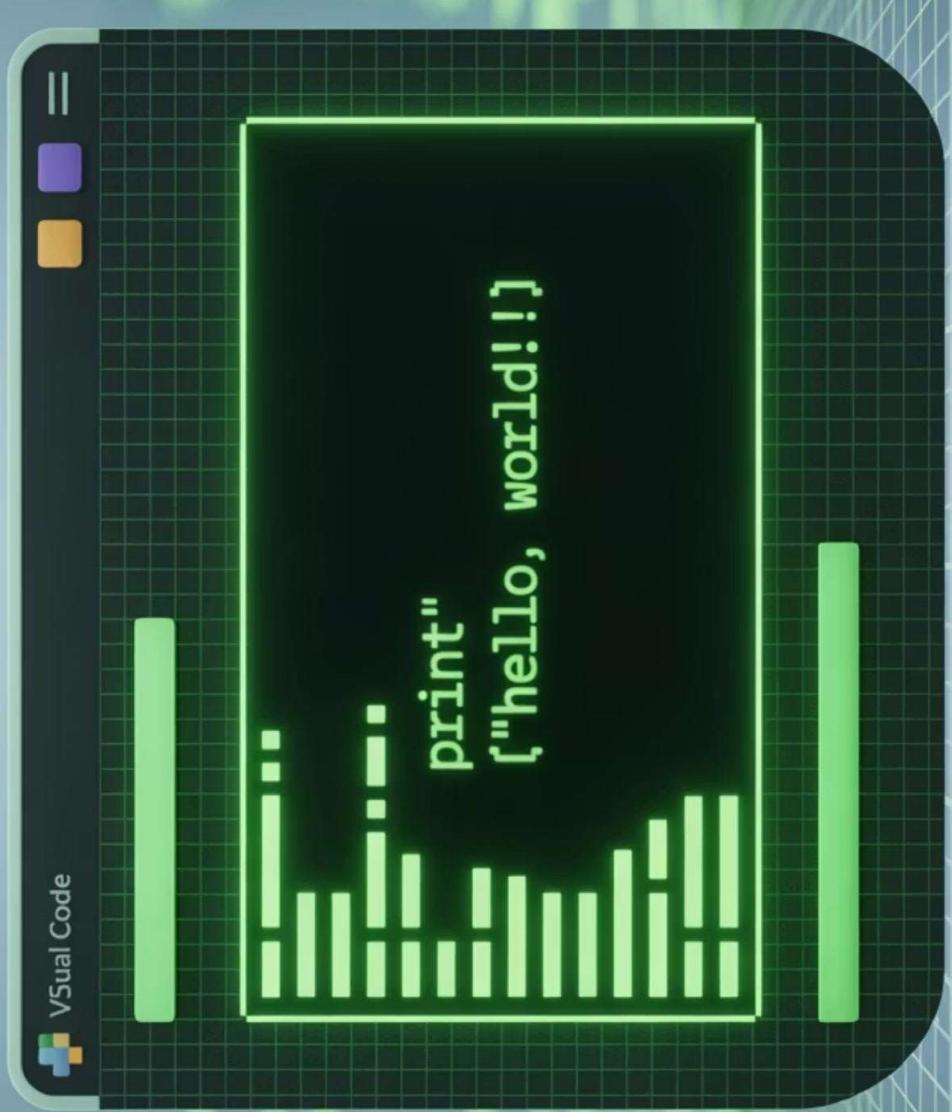
Una clase es un "molde" o plantilla para crear objetos que agrupan datos y funciones.

**Analogía:** una receta de cocina. La receta (clase) describe ingredientes y pasos; cada plato cocinado (objeto) es una instancia.



# Ejemplo en V

```
# archivo: persona.py
class Persona: """Define una persona.
edad."""
pass # aún no tiene atributos n
```



# Exploración paso a paso (clase)

```
class Persona: - declara la clase.  
    """... - docstring que describe su propósito.  
    pass - espacio reservado;  
  
más adelante agregaremos detalles.
```



## 3.2 ¿Qué es un constructor y qué son los atributos?

### Constructor (`__init__`)

Método que se ejecuta al crear una instancia, inicializando sus atributos.

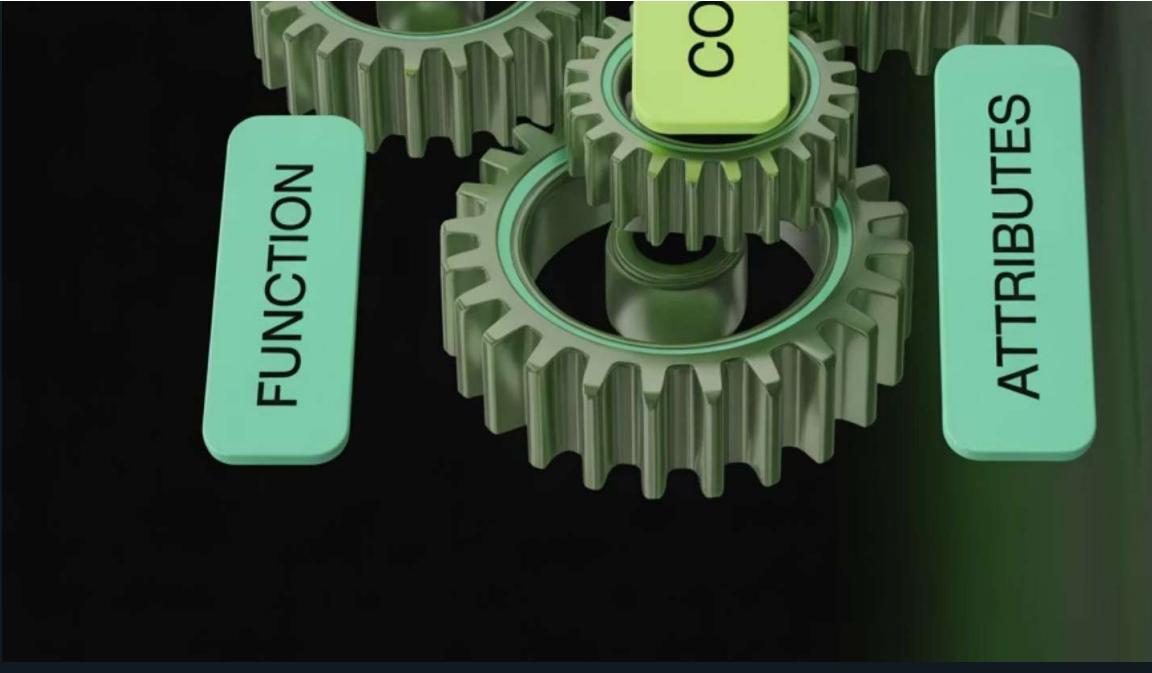
### Atributos

Variables que describen las propiedades de cada objeto.

FUNCTION

CO

ATTRIBUTES



# Ejemplo completo en persona.py

```
# archivo: persona.py

class Persona: """Define una persona con nombre y
edad."""

def __init__(self, nombre, edad): # constructor:
    receive nombre y edad

    self.nombre = nombre # atributo nombre

    self.edad = edad # atributo edad

def saludar(self): # método que imprime un saludo
    print(f"¡Hola! Soy {self.nombre} y tengo
{self.edad} años.")
```

# Crear instancias (main.py)

```
# Crear instancias en otro archivo, p.e. main.py
if __name__ == "__main__":
    p1 = Persona("Ana", 30)
    p1.saludar() # ¡Hola! Soy Ana y tengo 30 años.
```

# Explicación paso a paso (constructor)

</>

```
def __init__(self, nombre, edad):
```

Define el constructor

self.nombre = nombre

Crea el atributo nombre

f(x)

```
self.edad = edad
```

Crea el atributo edad

saludar()

Método que utiliza los atributos |

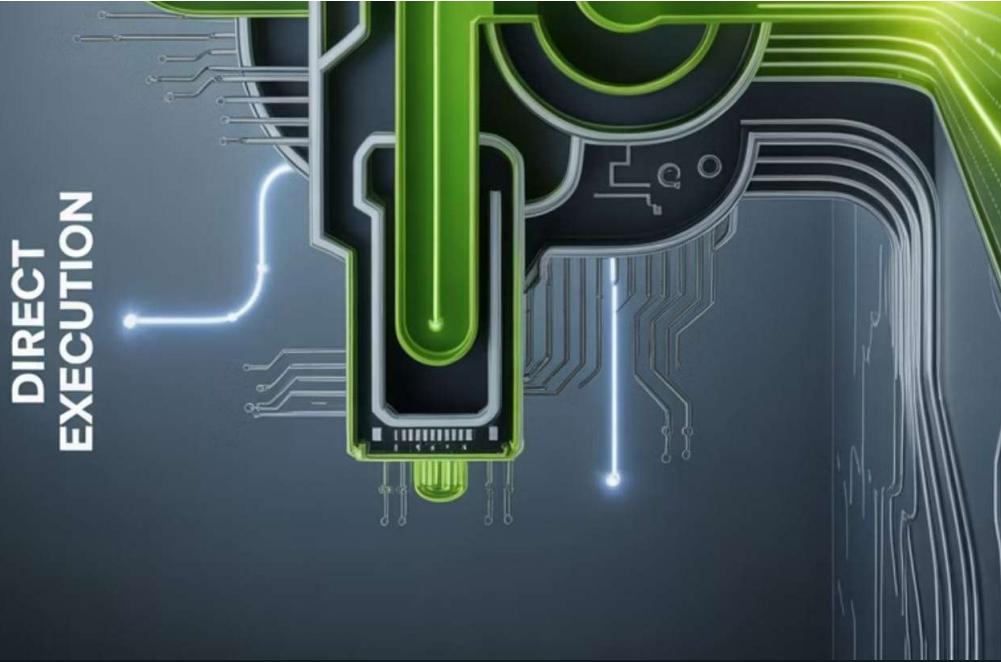
# Guardián if `__name__ == "main"`

## 1. ¿Qué es `__name__`?

- Módulo recibe `__name__ = "main"` si se ejecuta directamente.
- Si se importa, `__name__` es el nombre del módulo.

## 2. ¿Para qué sirve?

- Distinguir ejecución directa de importación.
- Permite pruebas rápidas y organización.



# Crear instancias de una clase

Importar Persona desde persona.py:

```
from persona import Persona
```

Crear objeto:

```
p1 = Persona("Ana", 30)
```

Llamar método:

```
p1.saludar()
```



# ¿Qué sucede en cada línea?

p1 = Persona("Ana", 30): Llama al constructor,  
crea objeto, lo referencia en p1.

p1.saludar(): invoca el método, imprime saludo.

1010  
CTOR  
010011  
01010010  
010101  
CONST  
01101010  
OBJECT

# ¿Por qué usar otro archivo?

**Separación de responsabilidades**

Definición vs ejecución.

**Reutilización**

Importar la clase sin código de ejemplo.

**Man**

Facilita  
del pro

# Recapitulación Paso a Paso

1

Crear el módulo de la clase  
(persona.py)

2

Importar la clase (f  
import Persona)

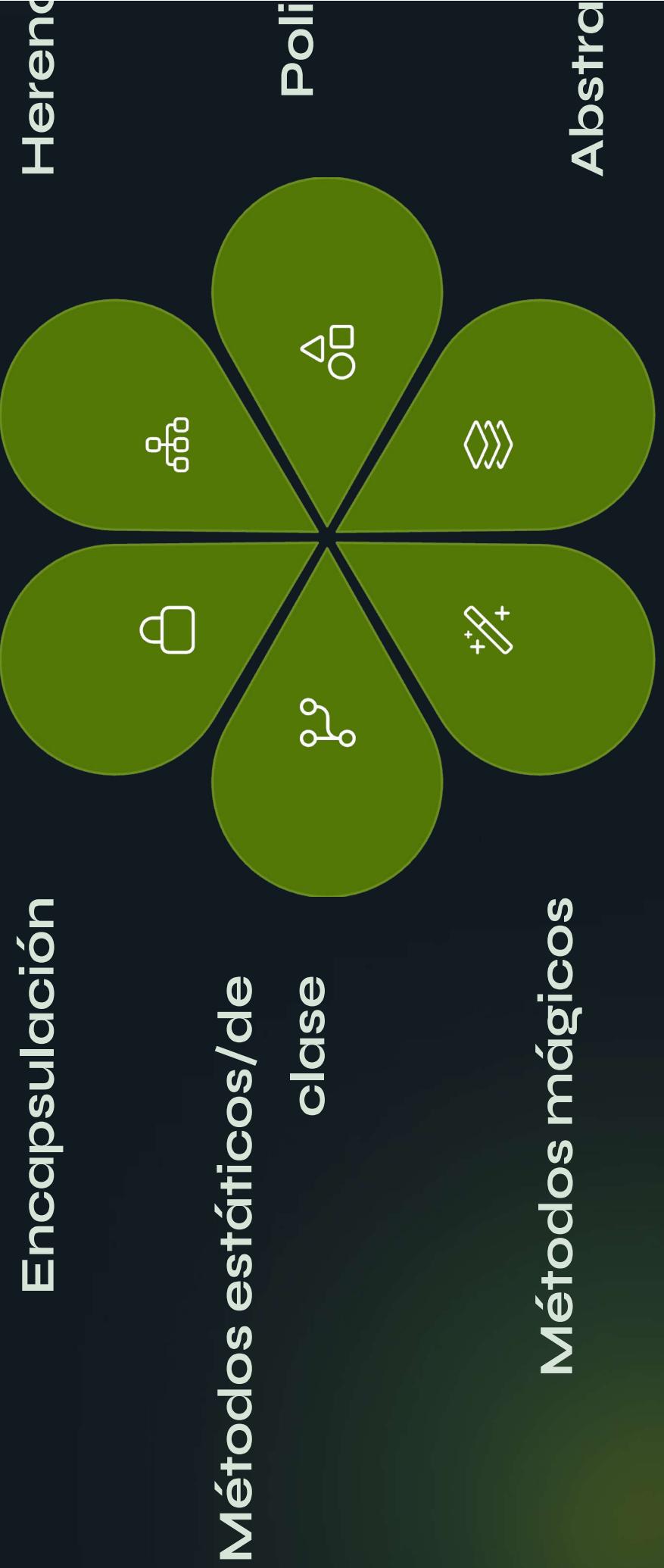
3

Proteger el flujo con if \_\_name\_\_  
== "\_\_main\_\_":

4

Instanciar y usar (p  
l1.saludar())

# Mención Superficial de Otros



# Metodologías Activas & Mini Desafíos



## 1. Reflexión (2 min):

- ¿Diferencias entre función y método?
- 2. Mini desafío (5 min):
  - En parejas, clase Teléfono con atributos (4) y métodos (2).
- 3. Feedback rápido:
  - Cada pareja comparte y explica su constructor.

# Actividad Práctica Final

## Tarea (30 min):

1. Crea clase Libro(título, autor, año\_publicación, género, ISBN)
2. Método presentar() que imprima los atributos del objeto.
3. Probar con 5 instancias distintas
4. Ahora haz lo mismo pero con la clase Auto, con 4 atributos y al menos los métodos, encender(), acelerar(), frenar() y detenerse(). Al ejecutar cada método se debe imprimir el



# Cierre y Conclusión

Hoy vimos en profundidad **Clases, Constructor y**

**Atributos**

Practicamos creación de plantillas y objetos

**Mañana:** Encapsulación y Herencia

**Viernes:** Polimorfismo y Abstracción

| "La práctica constante hace al maestro: isigan codificando!"

