Day 33

Introduction to OOPs in Python:

What is OOP?

OOP (Object-Oriented Programming) is a way of writing programs by grouping data and behaviour together.

Instead of thinking in terms of *functions*, we think in terms of objects.

Real-World Example

- Car
    - Data → color, model, speed
    - Behavior → drive(), brake(), stop()

Key OOP Concepts:

Python OOP is based on 4 pillars:

1. Class
2. Object
3. Encapsulation
4. Inheritance
5. Polymorphism
6. Abstraction

Classes and Objects in Python:

What is a Class?

A class is a blueprint or template to create objects.

```
class Car:
    pass
```

This defines a class named Car

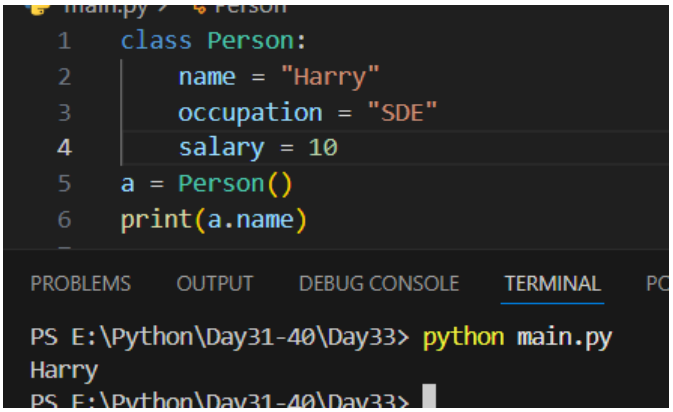
What is an Object?

An object is an instance of a class.

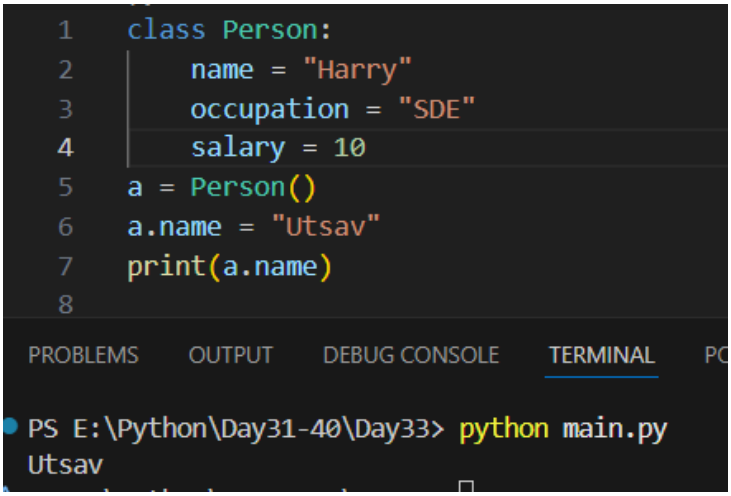
```
car1 = Car()
car2 = Car()
```

car1 and car2 are objects of class Car

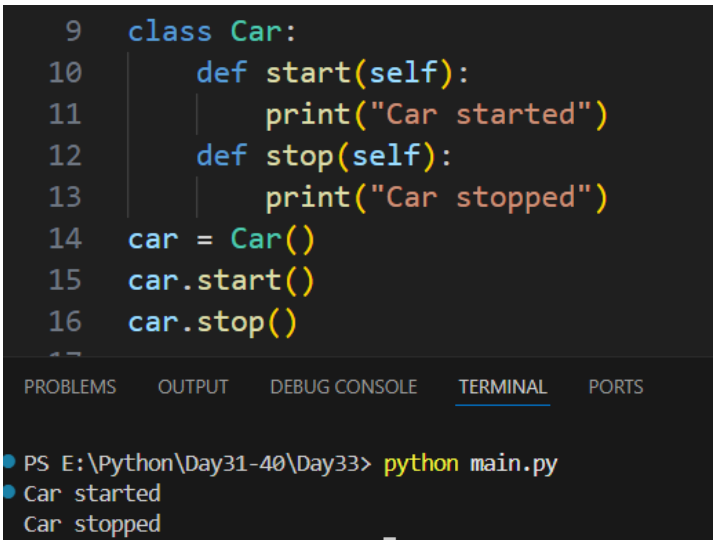
Example: creating a class, an object, and printing one of its attributes.



Example: updating the attributes.



Example: Methods (Functions inside a Class)



self Keyword (VERY IMPORTANT)

- Represents the current object
- Allows access to object variables and methods

Example: self is missing.

```
 9  class Car:
10      def start():
11          print("Car started")
12      def stop():
13          print("Car stopped")
14  car = Car()
15  car.start()
16  car.stop()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\Python\Day31-40\Day33> python main.py
Traceback (most recent call last):
  File "E:\Python\Day31-40\Day33\main.py", line 15, in <module>
    car.start()
    ~~~~~~~~~~^^
TypeError: Car.start() takes 0 positional arguments but 1 was given
```

--The End--