Day 36

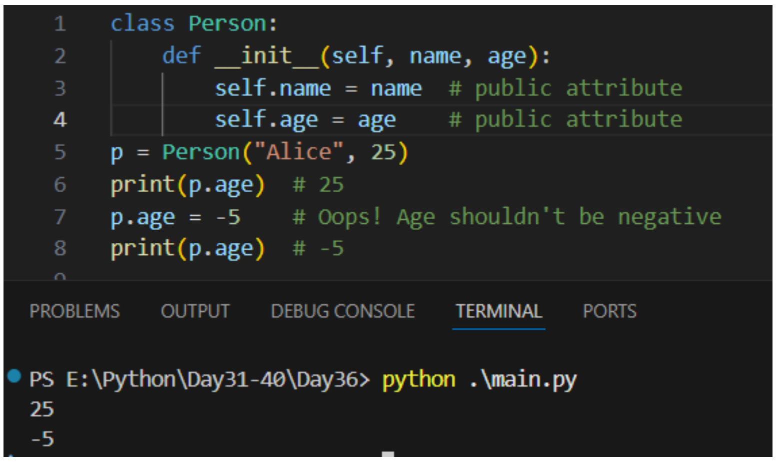
Getters and Setters in Python:

What are getters and setters?

- Getter: A method used to access the value of a private attribute.
- Setter: A method used to modify the value of a private attribute.

In Python, we usually make an attribute private by prefixing it with an underscore _ (convention) or double underscore __ (name mangling).

Example: without using getter and setter. Here, nothing prevents setting age to a negative number. That's why we use getters and setters.

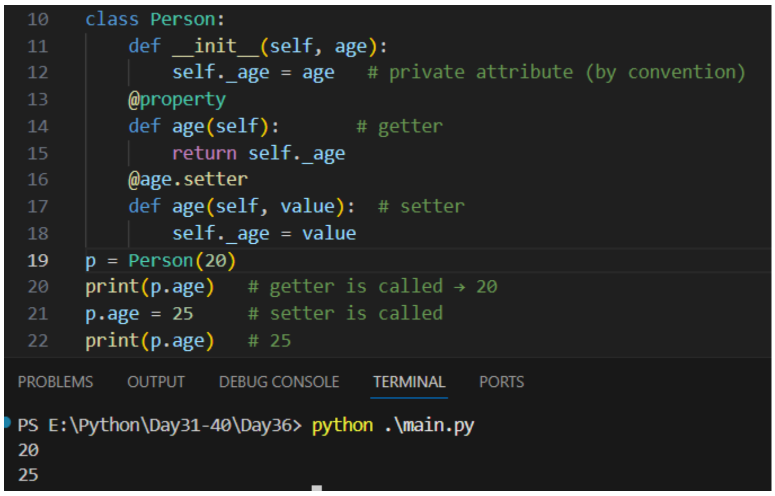


```python
class Person:
    def __init__(self, name, age):
        self.name = name  # public attribute
        self.age = age    # public attribute
p = Person("Alice", 25)
print(p.age)   # 25
p.age = -5     # Oops! Age shouldn't be negative
print(p.age)   # -5
```

```
PS E:\Python\Day31-40\Day36> python .\main.py
25
-5
```

This example shows a problem with public attributes. The Person class allows direct access to age, so after creating p = Person("Alice", 25), you can freely change p.age to -5. Python does not stop this, even though a negative age doesn't make sense. This is why properties (getters and setters) are useful—they let you add validation and protect data while still keeping the code easy to use.

Example:



```python
class Person:
    def __init__(self, age):
        self._age = age    # private attribute (by convention)
    @property
    def age(self):        # getter
        return self._age
    @age.setter
    def age(self, value):  # setter
        self._age = value
p = Person(20)
print(p.age)    # getter is called → 20
p.age = 25      # setter is called
print(p.age)    # 25
```

```
PS E:\Python\Day31-40\Day36> python .\main.py
20
25
```

This code shows how properties work in Python to control access to class attributes. The Person class stores the age in a “private” variable _age. The @property decorator makes the age() method act like a normal attribute, so p.age calls the getter and returns the value. The @age.setter decorator lets you update the value using p.age = 25, which calls the setter method. This way, you can safely get and set values while still using simple attribute-style access.

Summary:

A getter returns a value, a setter changes a value — and @property lets you use them like normal variables.

--The End--