LESSON 10

main.py

```
@dataclass
class Student:
  name: str
  score: float
```

Py

Polymorphism & ABC

Learn it all @MyWebUniversity



A FREE MYWEBUNIVERSITY COURSE

PYTHON SCRIPTING & OOP

LESSON 10

OOP: Polymorphism, Abstraction and Dataclasses

Duck typing, abstract base classes, dunder methods and @dataclass

Polymorphism

shapes.py

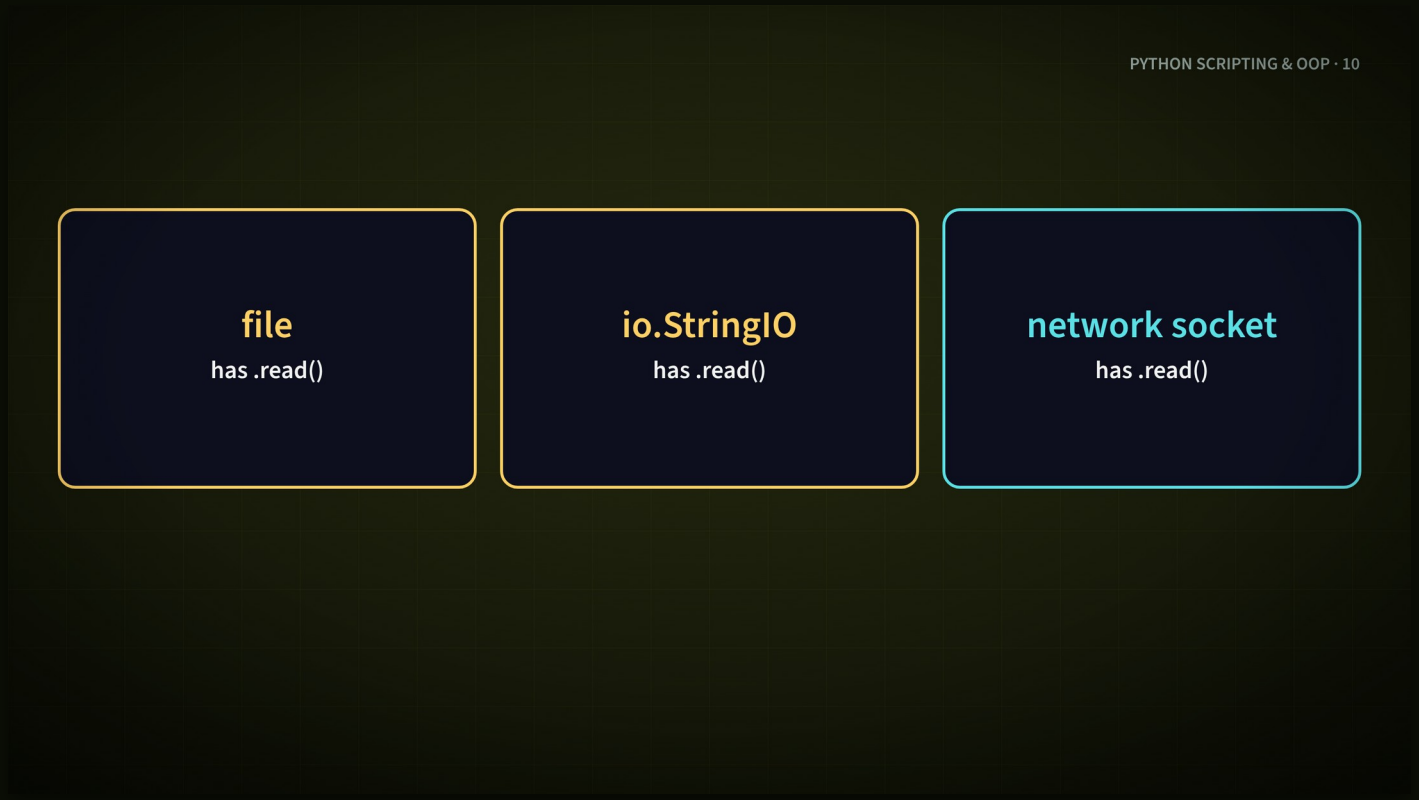
```python
import math
class Circle:
    def __init__(self, r): self.r = r
    def area(self): return math.pi * self.r ** 2
class Rectangle:
    def __init__(self, w, h): self.w, self.h = w, h
    def area(self): return self.w * self.h

for s in [Circle(1), Rectangle(2, 3)]:
    print(type(s).__name__, round(s.area(), 2))
```

- Polymorphism means one interface, many forms. Different classes can have a method with the …
- A loop over shapes calls area on each one: circles, rectangles and triangles each compute …

01

Duck Typing



- file — has .read()
- io.StringIO — has .read()
- network socket — has .read()

02

Abstraction with ABC

abc_shapes.py

```python
from abc import ABC, abstractmethod

class Shape(ABC):
    @abstractmethod
    def area(self): ...

class Square(Shape):
    def __init__(self, s): self.s = s
    def area(self): return self.s ** 2

Shape()          # TypeError: can't instantiate abstract class
```

- An abstract base class defines what methods every child must have, without implementing …
- You cannot create an object from the abstract class itself, and a child that forgets a …

03

Dunder Methods: Making Objects Feel Built-in

money.py

```python
class Money:
    def __init__(self, amount): self.amount = amount
    def __add__(self, other): return Money(self.amount + other.amount)
    def __eq__(self, other): return self.amount == other.amount
    def __lt__(self, other): return self.amount < other.amount
    def __repr__(self): return f"Money({self.amount})"

Money(5) + Money(7)              # Money(12)
sorted([Money(9), Money(2)])     # [Money(2), Money(9)]
```

- Special methods let your objects work with Python's operators and functions. double ...
- This is operator overloading, Python style, and it is what makes libraries like NumPy feel ...

04

@dataclass

student.py

```python
from dataclasses import dataclass, field

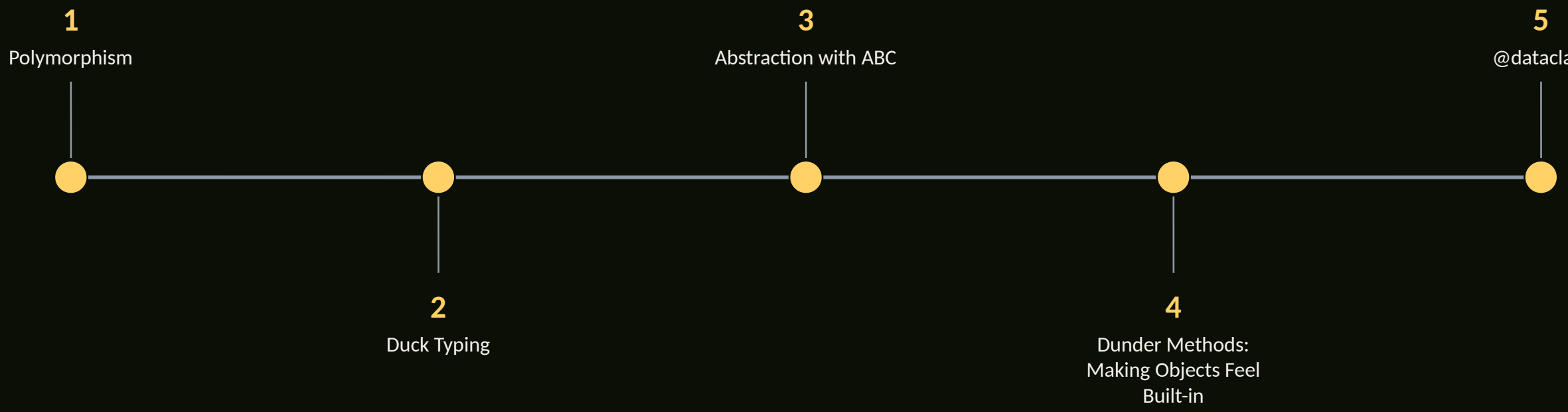
@dataclass(order=True)
class Student:
    score: float
    name: str
    courses: list[str] = field(default_factory=list)

s = Student(95, "Mariam")
print(s)        # Student(score=95, name='Mariam', courses=[])
```

- Many classes just hold data. The dataclass decorator writes double underscore init, repr …
- Add frozen equals True to make objects immutable, and order equals True to make them …

05

Lesson Map


1
Polymorphism
2
Duck Typing
3
Abstraction with ABC
4
Dunder Methods: Making Objects Feel Built-in
5
@dataclass

Sources and Further Reading

shapes.py

```python
import math
class Circle:
    def __init__(self, r): self.r = r
    def area(self): return math.pi * self.r ** 2
class Rectangle:
    def __init__(self, w, h): self.w, self.h = w, h
    def area(self): return self.w * self.h

for s in [Circle(1), Rectangle(2, 3)]:
    print(type(s).__name__, round(s.area(), 2))
```


abc_shapes.py

```python
from abc import ABC, abstractmethod

class Shape(ABC):
    @abstractmethod
    def area(self): ...

class Square(Shape):
    def __init__(self, s): self.s = s
    def area(self): return self.s ** 2

Shape()          # TypeError: can't instantiate abstract class
```

Thank you for watching

MyWebUniversity.com

Free technical education since 2000 — English and Dari

@MyWebUniversity · @MyWebUniversity-Dari