[320] Special Methods and Inheritance

Department of Computer Sciences
University of Wisconsin-Madison

Special Methods

Special Methods

__init__ is a special method, with non-standard behavior

```
class Dog:
    def __init__(dog, name, age):
        print("created a dog")
        dog.name = name
        dog.age = age

    def speak(dog, mult):
        print(dog.name + ": " + "bark!"*mult)

fido = Dog("Fido", 9)


fido.speak(5)
```

Special Methods

There are MANY special method names:
https://docs.python.org/3/reference/datamodel.html#special-method-names

We'll learn a few:

`__str__, __repr__, _repr_html_`

`__eq__, __lt__`

`__len__, __getitem__`

`__enter__, __exit__`

control how an object looks when we print it or see it in Out[N]

generate HTML to create more visual representations of objects in Jupyter. Like tables for DataFrames

Special Methods

There are MANY special method names:
https://docs.python.org/3/reference/datamodel.html#special-method-names

We'll learn a few:

`__str__, __repr__, _repr_html_`

`__eq__, __lt__`

define how **==** behaves for two different objects

`__len__, __getitem__`

define how a list of objects should be sorted

```
c = (a==b) # type of c?
```

`__enter__, __exit__`

Special Methods

There are MANY special method names:
https://docs.python.org/3/reference/datamodel.html#special-method-names

We'll learn a few:

`__str__, __repr__, _repr_html_`

`__eq__, __lt__`

`__len__, __getitem__`

build our own sequences that we index, slice, and loop over:

```
val = obj[idx]
vals = obj[3:7]
for x in obj:
    print(x)
```

what goes in brackets?

`__enter__, __exit__`

Special Methods

There are MANY special method names:
https://docs.python.org/3/reference/datamodel.html#special-method-names

We'll learn a few:

`__str__, __repr__, _repr_html_`

`__eq__, __lt__`

`__len__, __getitem__`

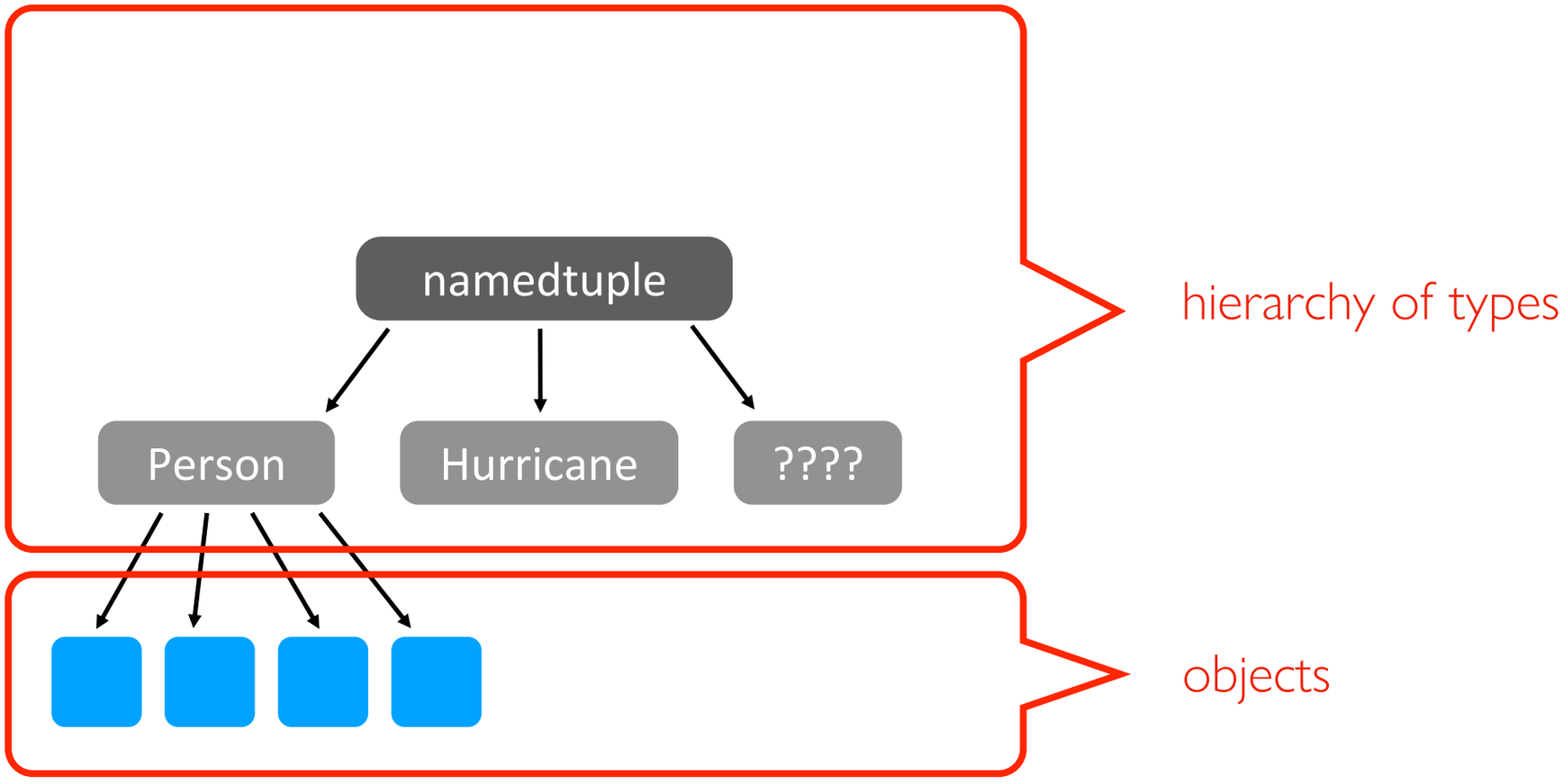
`__enter__, __exit__`

context managers

```
with open("file.txt") as f:
    data = f.read()
# automatically close
```

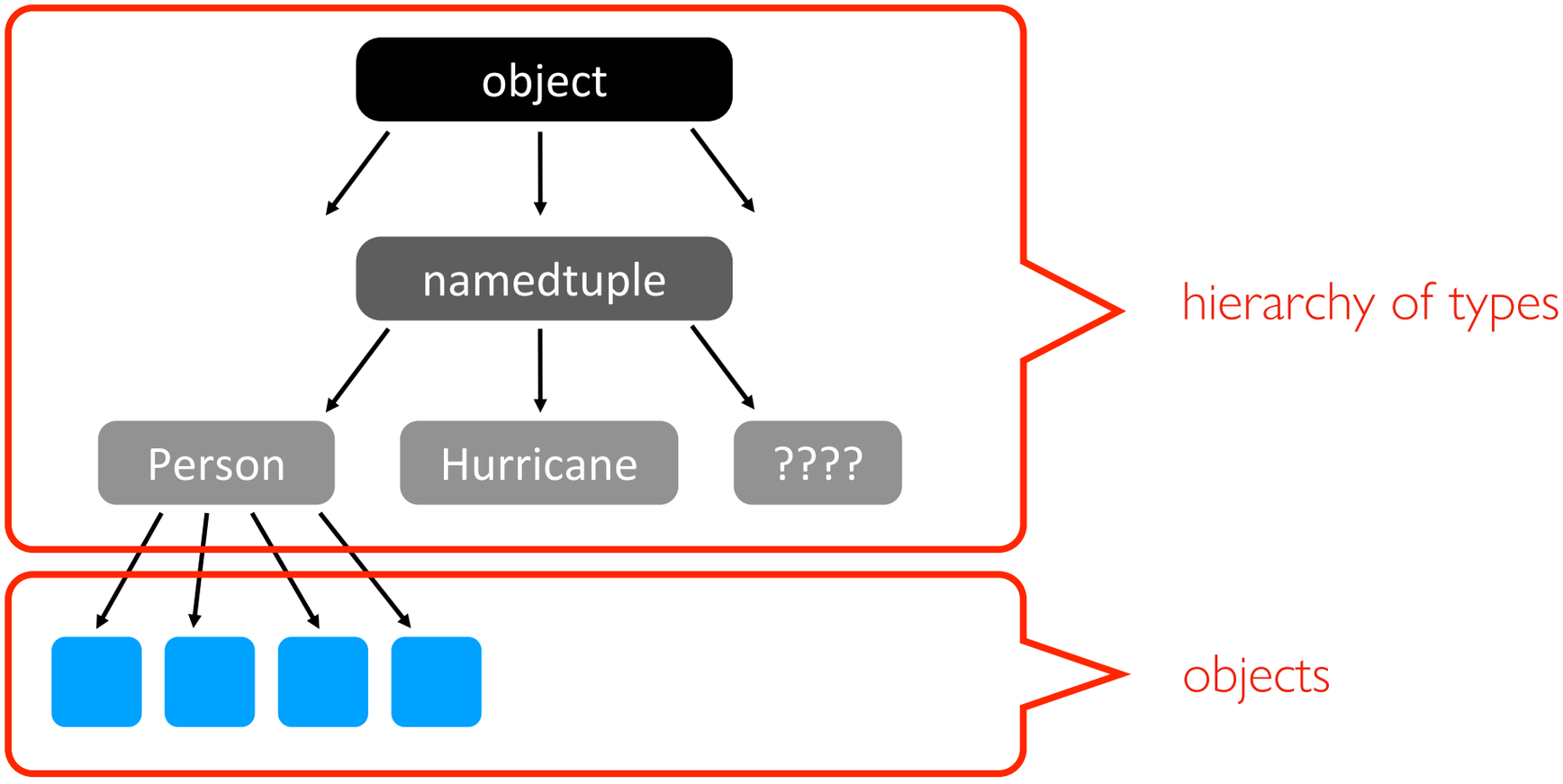
Inheritance

Types, Sub Types, and Objects



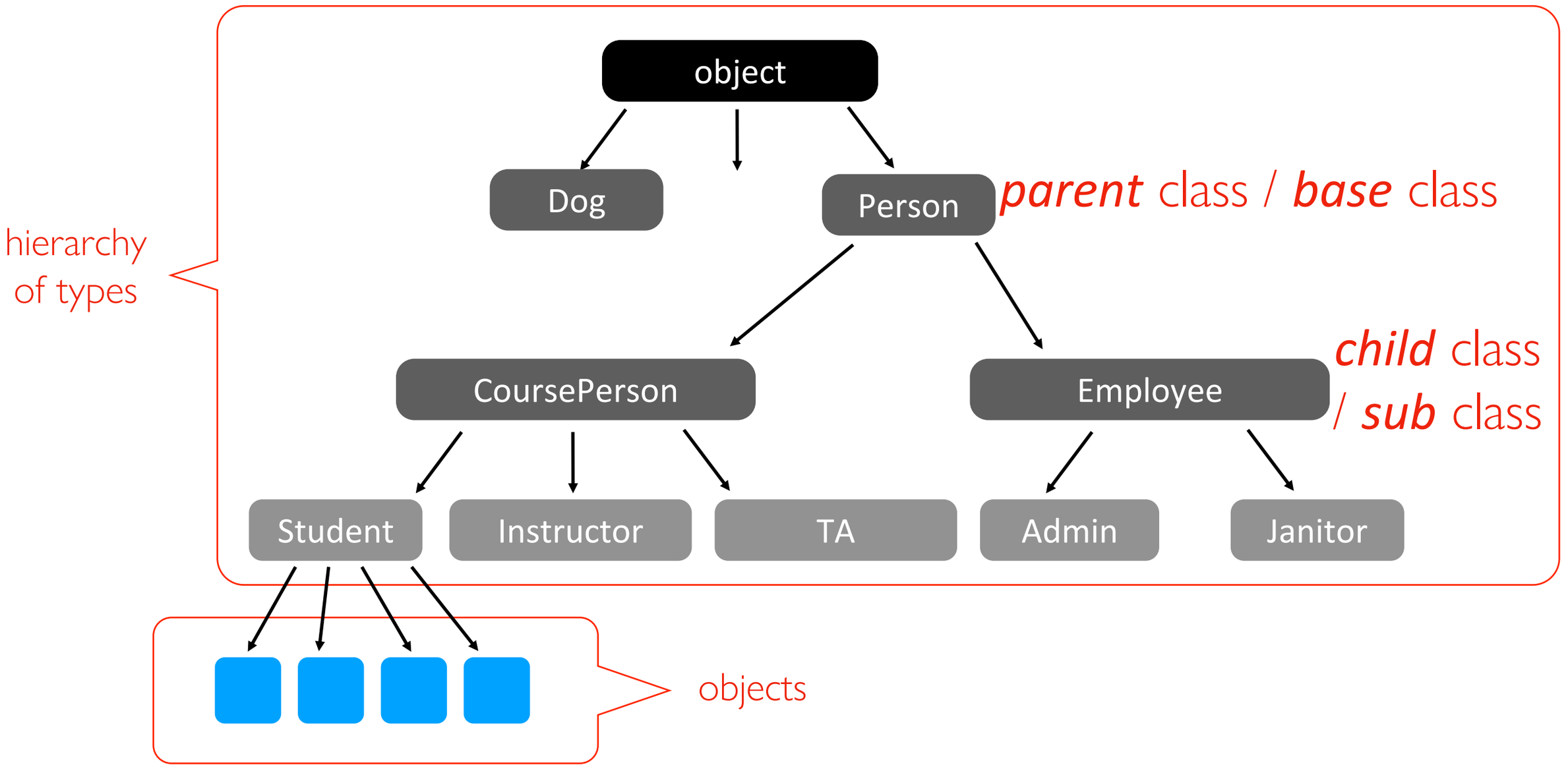
classes (and types in general) form a hierarchy

Types, Sub Types, and Objects



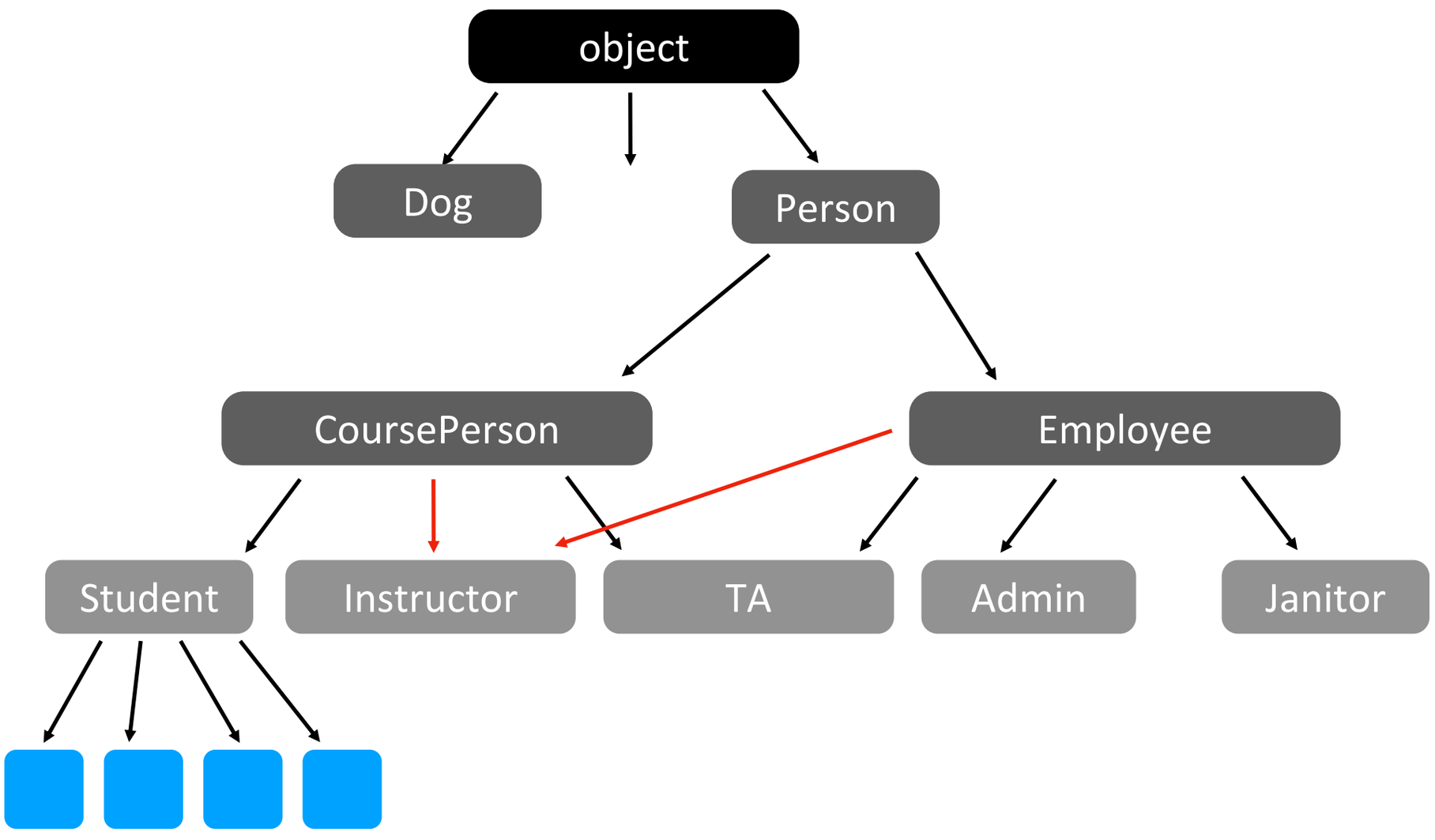
weird naming: the top type is called ***"object"***

Types, Sub Types, and Objects



we can design the hierarchy with ***inheritance***

Types, Sub Types, and Objects



multiple inheritance

Coding Examples

Principals

- method inheritance
- method resolution order
- overriding methods, constructor
- calling overridden methods
- abc's (abstract base classes)