

# [320] Inheritance

Tyler Caraza-Harter

Review

# Review Classes + Special Methods

```
class Dog:
    def __init__(self, name):
        self.name = name

    def bark(self, mult, ucase):
        msg = "bark " * mult
        if ucase:
            msg = msg.upper()
        print(self.name + ": " + msg)
```

```
sam = Dog("Fido")
fido = Dog("Sam")
```

```
fido.bark(5, False)           # 1
fido.bark(fido, 5, True)      # 2
fido.bark(fido, 5, True, None) # 3
```

which call produces the following error?

**TypeError:** bark() takes 3 positional arguments but 4 were given

# Review Classes + Special Methods

```
class Dog:
    def __init__(self, name):
        self.name = name

    def bark(self, mult, ucase):
        msg = "bark " * mult
        if ucase:
            msg = msg.upper()
        print(self.name + ": " + msg)
```

```
sam = Dog("Fido")
fido = Dog("Sam")
```

```
fido.bark(5, False)           # 1
fido.bark(fido, 5, True)      # 2
fido.bark(fido, 5, True, None) # 3
```

which call is correct?

# Review Classes + Special Methods

```
class Dog:
    def __init__(self, name):
        self.name = name

    def bark(self, mult, ucase):
        msg = "bark " * mult
        if ucase:
            msg = msg.upper()
        print(self.name + ": " + msg)
```

```
sam = Dog("Fido")
fido = Dog("Sam")
```

```
fido.bark(5, False) # 1
```

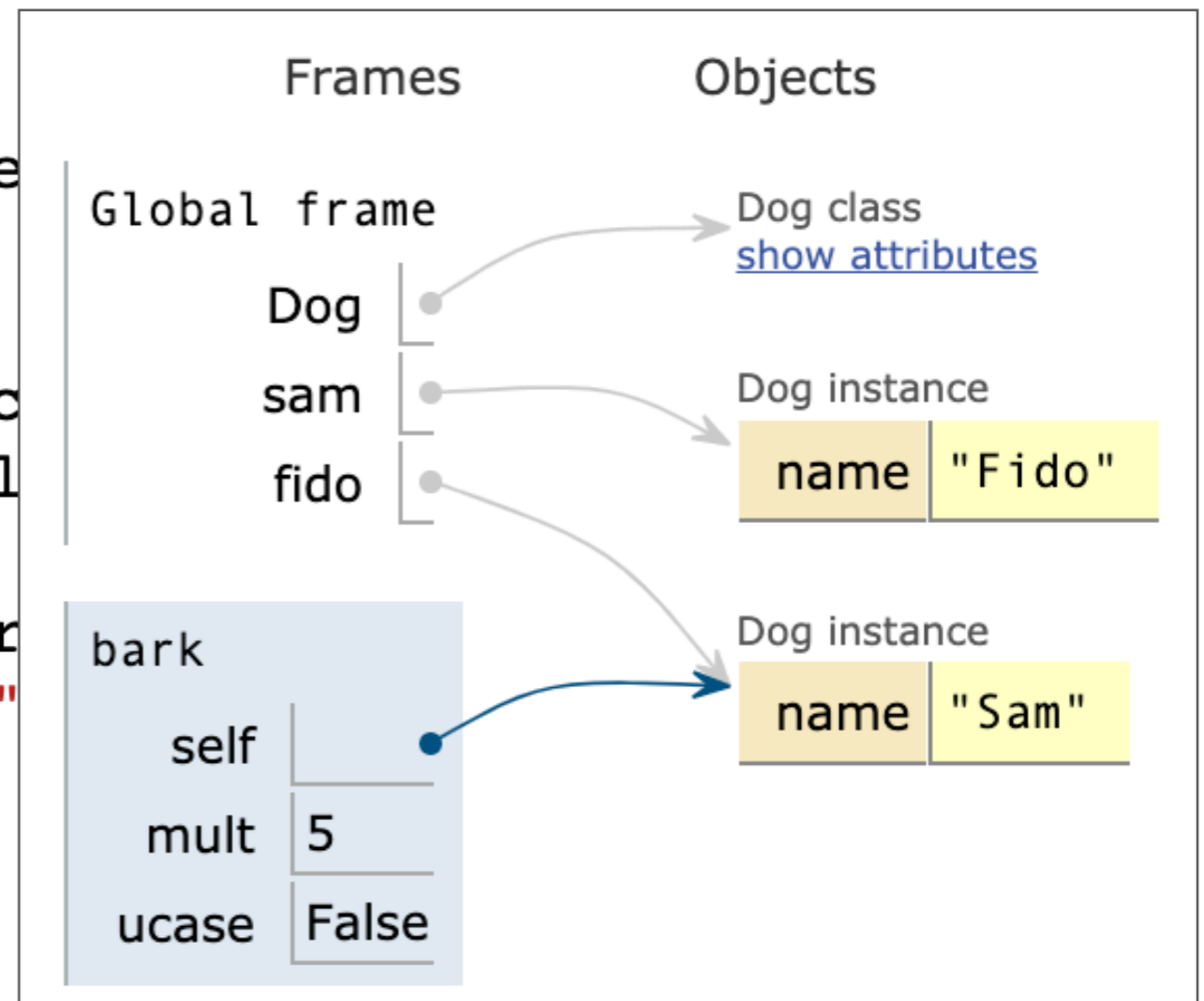
what is printed?

- (1) Fido: bark bark bark bark bark
- (2) Fido: BARK BARK BARK BARK BARK
- (3) Sam: bark bark bark bark bark

# Review Classes + Special Methods

```
class Dog:  
    def __init__(self, name):  
        self.name = name  
  
    def bark(self, mult, uc):  
        msg = "bark " * mult  
        if uc:  
            msg = msg.upper  
        print(self.name + " " + msg)
```

```
sam = Dog("Fido")  
fido = Dog("Sam")
```



```
fido.bark(5, False)
```

```
# 1
```

what is printed?

- (1) Fido: bark bark bark bark bark
- (2) Fido: BARK BARK BARK BARK BARK
- (3) Sam: bark bark bark bark bark

# Review Classes + Special Methods

Special methods usually get called

1. explicitly
2. implicitly

What does **print(...)** use to represent an object?

1. `__str__`
2. `__repr__`
3. `__repr_html__`

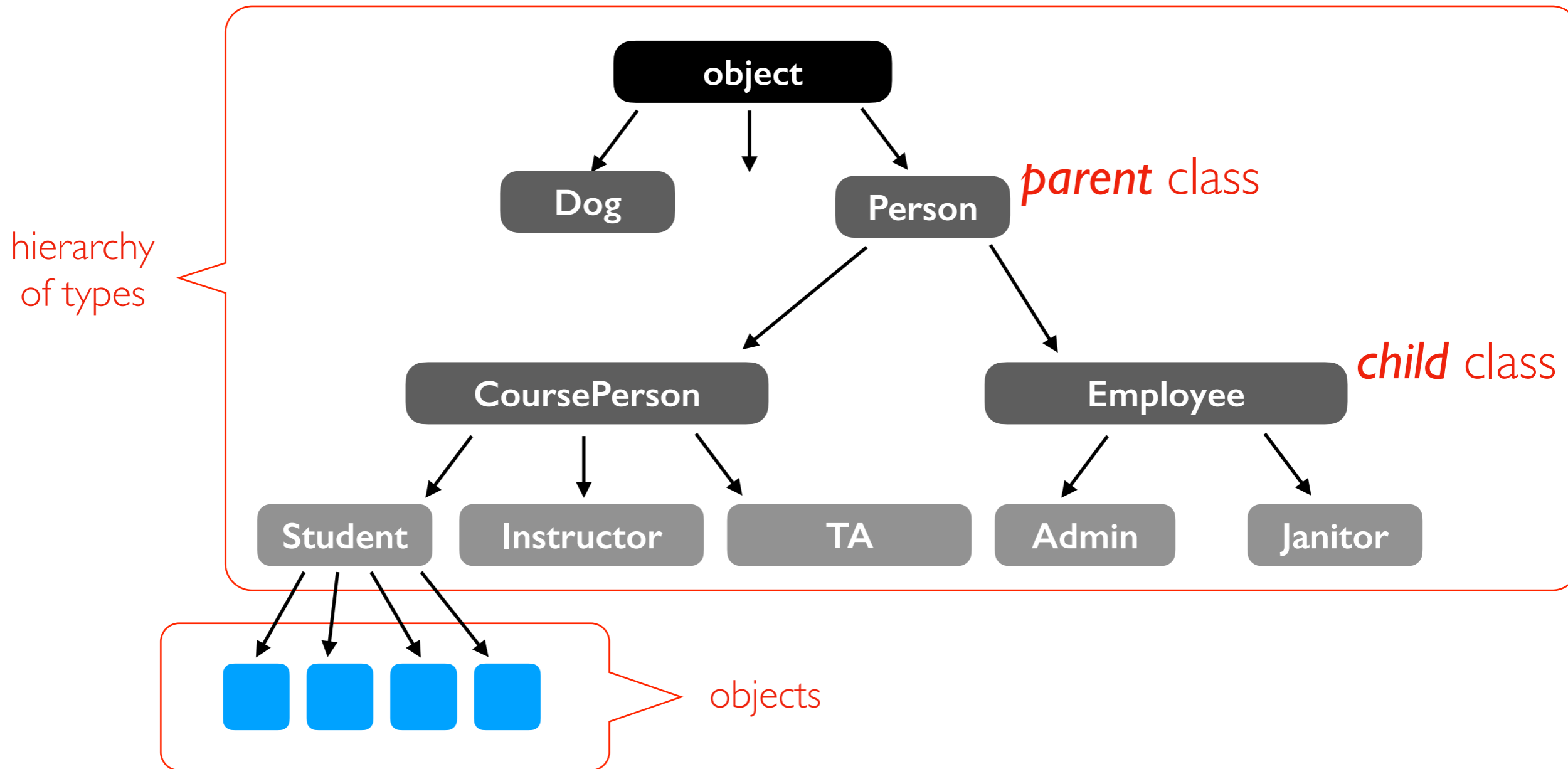
What special method must be implemented for **sorting** to work?

1. `__repr__`
2. `__order__`
3. `__lt__`
4. `__gt__`

# Inheritance

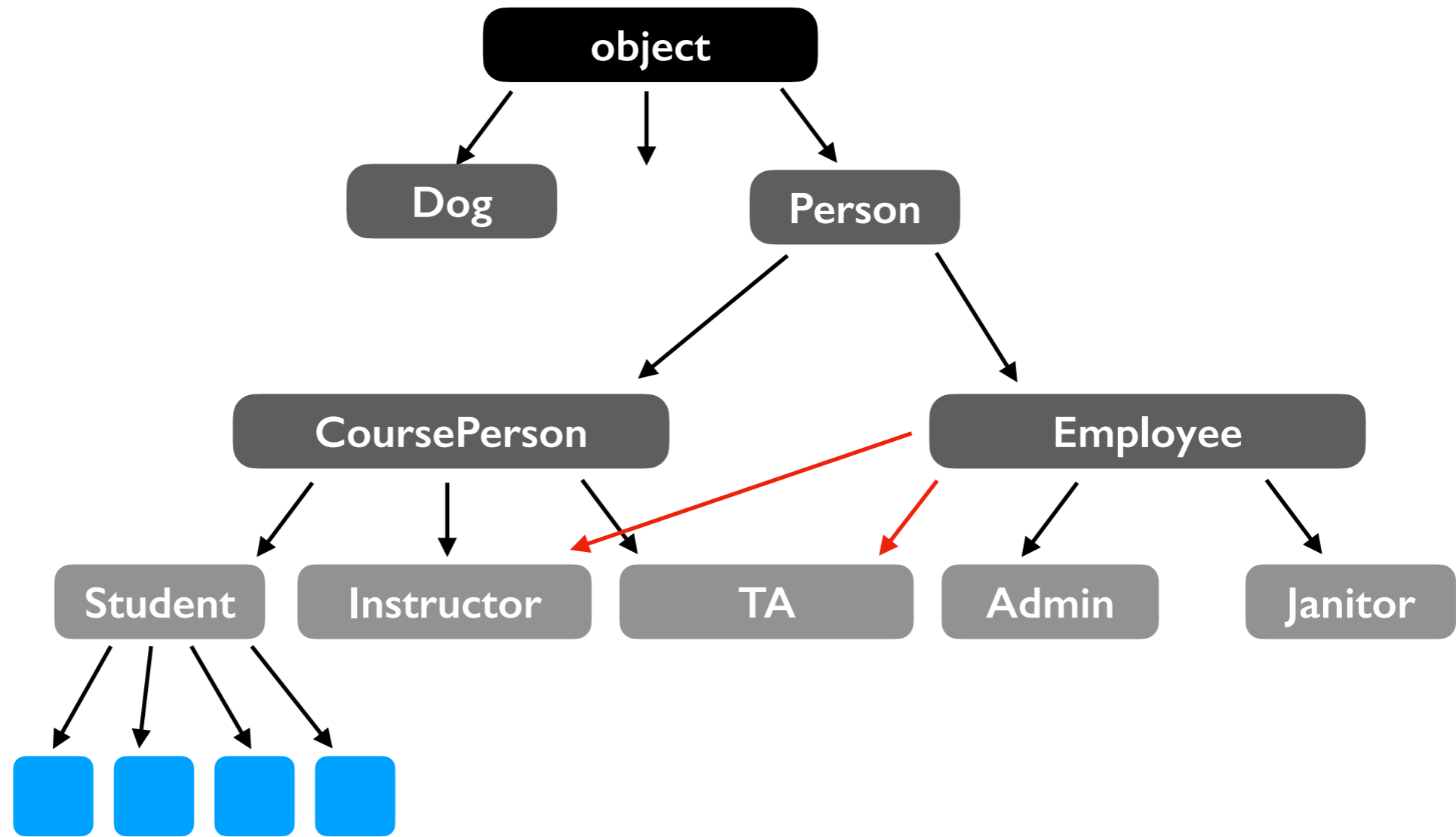


# Types, Sub Types, and Objects



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)