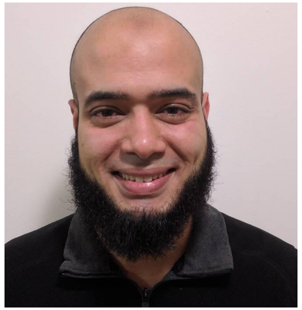
Python Programming

Static Variables

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

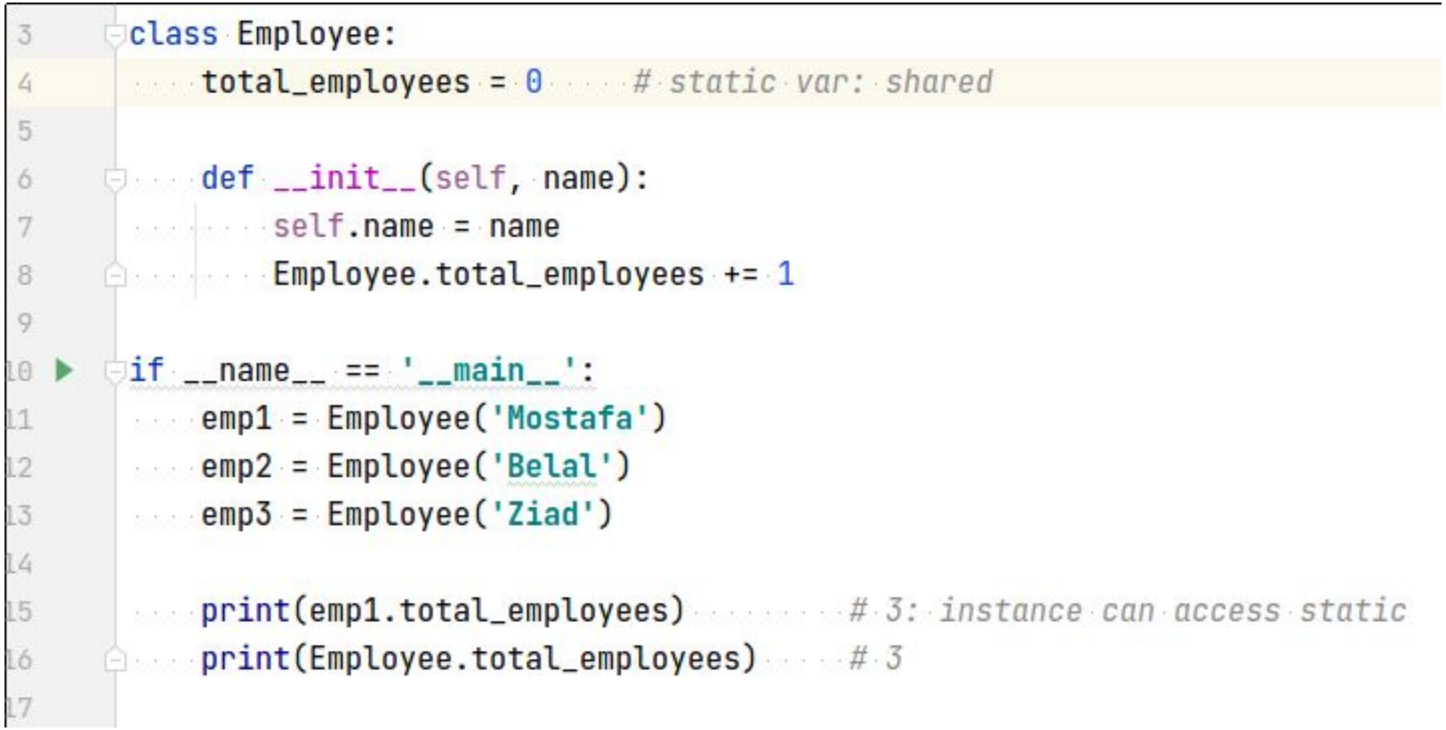
Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Static Variables

- What if we need a **shared** variable among all objects?
- So defined once and used by all?
- This is called **static** attribute
  - Created on class level and aren't instantiated.
  - With any change ⇒ all objects see the effect

Creating static variables

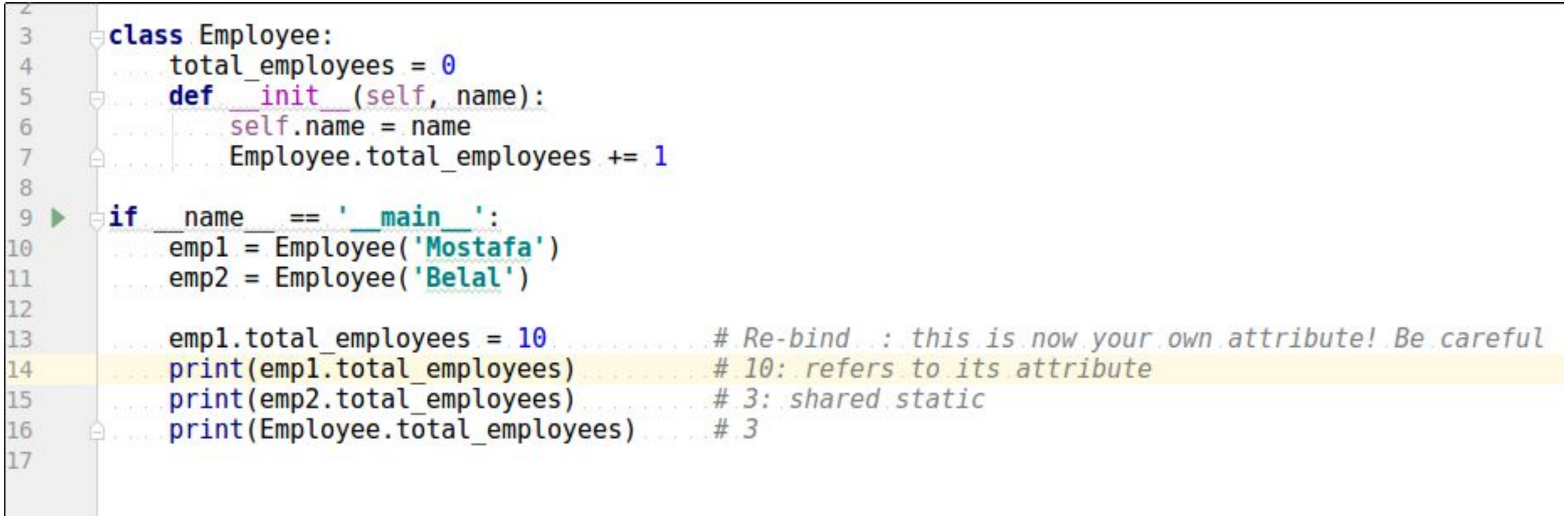


```python
class Employee:
    total_employees = 0    # static var: shared

    def __init__(self, name):
        self.name = name
        Employee.total_employees += 1

if __name__ == '__main__':
    emp1 = Employee('Mostafa')
    emp2 = Employee('Belal')
    emp3 = Employee('Ziad')

    print(emp1.total_employees)        # 3: instance can access static
    print(Employee.total_employees)    # 3
```

Confusion is coming!

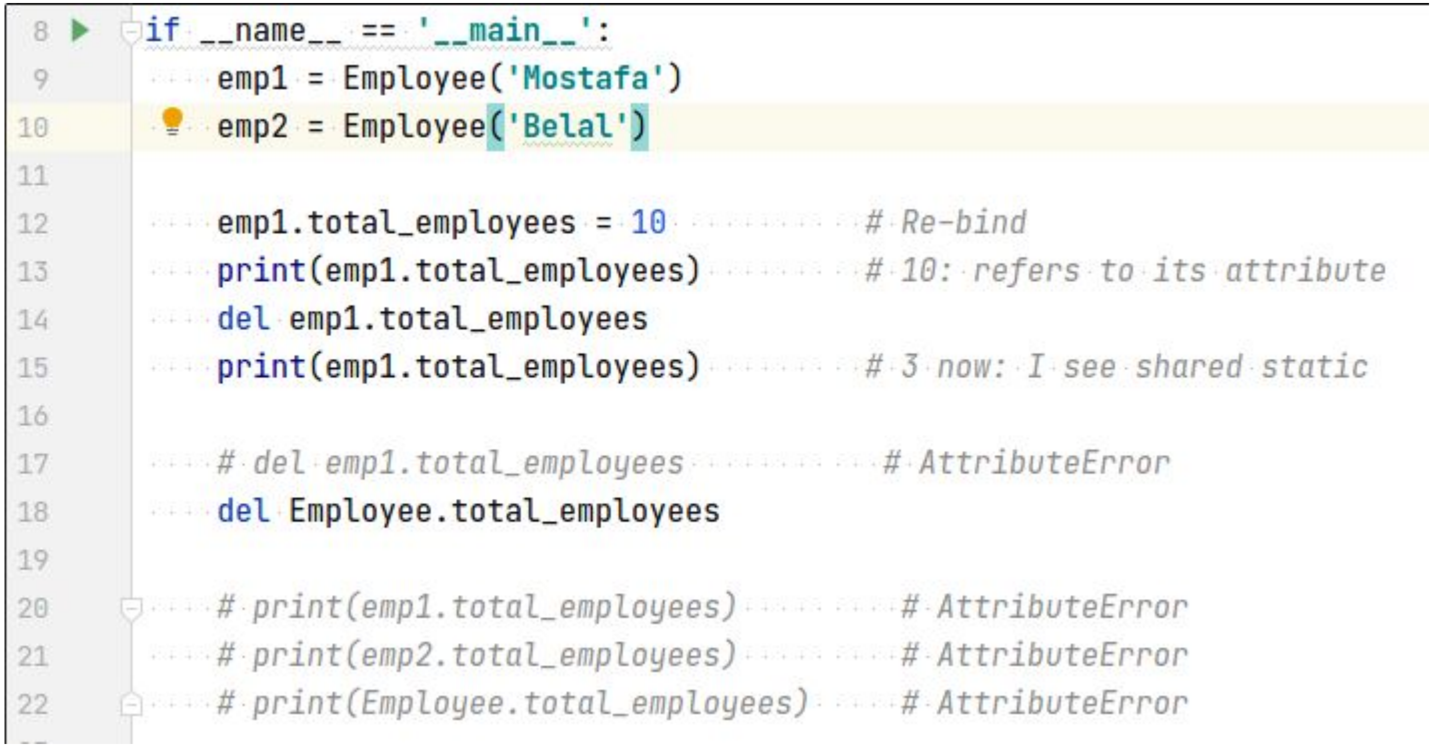
- Static variables are nice as long as you used them carefully
- As long as you use the Class to access/modify the static var ⇒ Perfect
- Once you use the object to modify the static var issues may occur
  - We need to understand instance namespace vs class namespace
  - We need to take into consideration: mutable vs immutable objects
- Similar issue if you have an attribute with same name as static var!
- Before next session
  - Practice what we learned
  - Take a few minutes min to guess the behaviour of the next 2 slides
    - No need to play with code or Google

Mixing the usage



```
class Employee:
    total_employees = 0
    def __init__(self, name):
        self.name = name
        Employee.total_employees += 1

if __name__ == '__main__':
    emp1 = Employee('Mostafa')
    emp2 = Employee('Belal')

    emp1.total_employees = 10              # Re-bind  : this is now your own attribute! Be careful
    print(emp1.total_employees)            # 10: refers to its attribute
    print(emp2.total_employees)            # 3: shared static
    print(Employee.total_employees)        # 3
```

Deleting attributes and vars



```
if __name__ == '__main__':
    emp1 = Employee('Mostafa')
    emp2 = Employee('Belal')

    emp1.total_employees = 10           # Re-bind
    print(emp1.total_employees)         # 10: refers to its attribute
    del emp1.total_employees
    print(emp1.total_employees)         # 3 now: I see shared static

    # del emp1.total_employees            # AttributeError
    del Employee.total_employees

    # print(emp1.total_employees)         # AttributeError
    # print(emp2.total_employees)         # AttributeError
    # print(Employee.total_employees)     # AttributeError
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."