Python Programming

Inheritance 1: Motivation

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)

Common Vs Unique

- Assume we are modeling a system for a Teacher
  - There are many classes such as Student and Teacher
- Think in classes for both of them
  - What might be **common** attributes and methods?
  - What might be **unique** attributes and methods?
  - Any critical observation?

Common Vs Unique

```python
class Student:
    def __init__(self):
        self.name = None
        self.email = None
        self.address = None
        self.national_id = None
        self.starting_study_year = None
        self.GPA = None
        self.studied_courses = []

    def is_valid_email(self, email)
        pass
    def add_course_frade(self, course_id, grade):
        pass
    def print_grades(self):
        pass
```

```python
class Teacher:
    def __init__(self):
        self.name = None
        self.email = None
        self.address = None
        self.national_id = None
        self.starting_employement_year = None
        self.current_salary = None
        self.teaching_courses = []

    def is_valid_email(self, email):
        pass

    def add_course(self, course_id):
        pass
```

Common Vs Unique

```python
class Student:
    def __init__(self):
        self.name = None
        self.email = None
        self.address = None
        self.national_id = None
        self.starting_study_year = None
        self.GPA = None
        self.studied_courses = []

    def is_valid_email(self, email)
        pass
    def add_course_frade(self, course_id, grade):
        pass
    def print_grades(self):
        pass
```

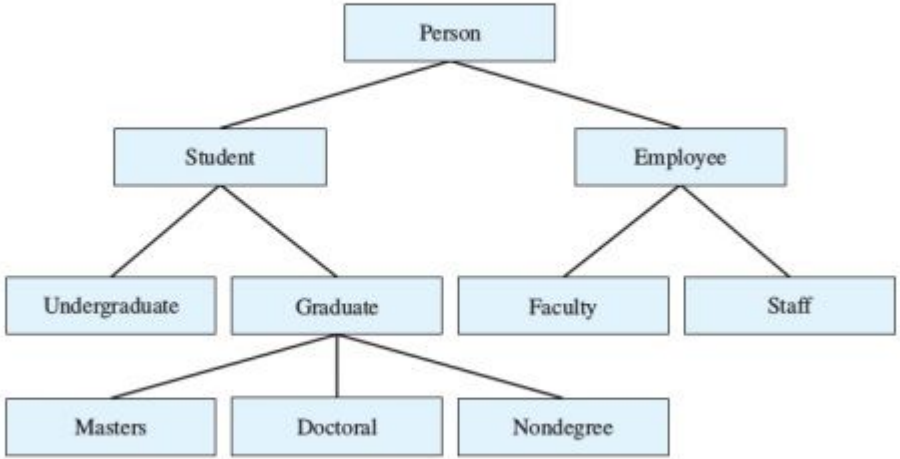
```python
class Teacher:
    def __init__(self):
        self.name = None
        self.email = None
        self.address = None
        self.national_id = None
        self.starting_employement_year = None
        self.current_salary = None
        self.teaching_courses = []

    def is_valid_email(self, email):
        pass

    def add_course(self, course_id):
        pass
```

- What is the clear problem in this code?
  - Some attributes and methods are **duplicated**!

Is-a relationship

- Student is-a person. Teacher is-a person. Dean is-a person
  - So some **common** attributes/methods + some **unique** attributes/methods
- Circle **is-a** shape.
  - Rectangle is-a shape. Triangle is-a shape.
- Software Engineer **is-an** employee.
  - Manager is-an employee.
  - Office Boy is-an employee
- Apple **is-a** fruit. Orange is-a fruit. Watermelon is-a fruit
- (Wagon / Bicycle / Motor vehicle / Railed vehicle) **is-a** vehicle

Is-a relationship: Hierarchy!



Img Src

Back to the Student vs Teacher

```python
class Student:
    def __init__(self):
        self.name = None
        self.email = None
        self.address = None
        self.national_id = None
        self.starting_study_year = None
        self.GPA = None
        self.studied_courses = []

    def is_valid_email(self, email)
        pass
    def add_course_frade(self, course_id, grade):
        pass
    def print_grades(self):
        pass
```

```python
class Teacher:
    def __init__(self):
        self.name = None
        self.email = None
        self.address = None
        self.national_id = None
        self.starting_employement_year = None
        self.current_salary = None
        self.teaching_courses = []

    def is_valid_email(self, email):
        pass

    def add_course(self, course_id):
        pass
```

- How can we avoid **duplicating** code in this problem?
  - Inheritance allow us to reuse code!

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."