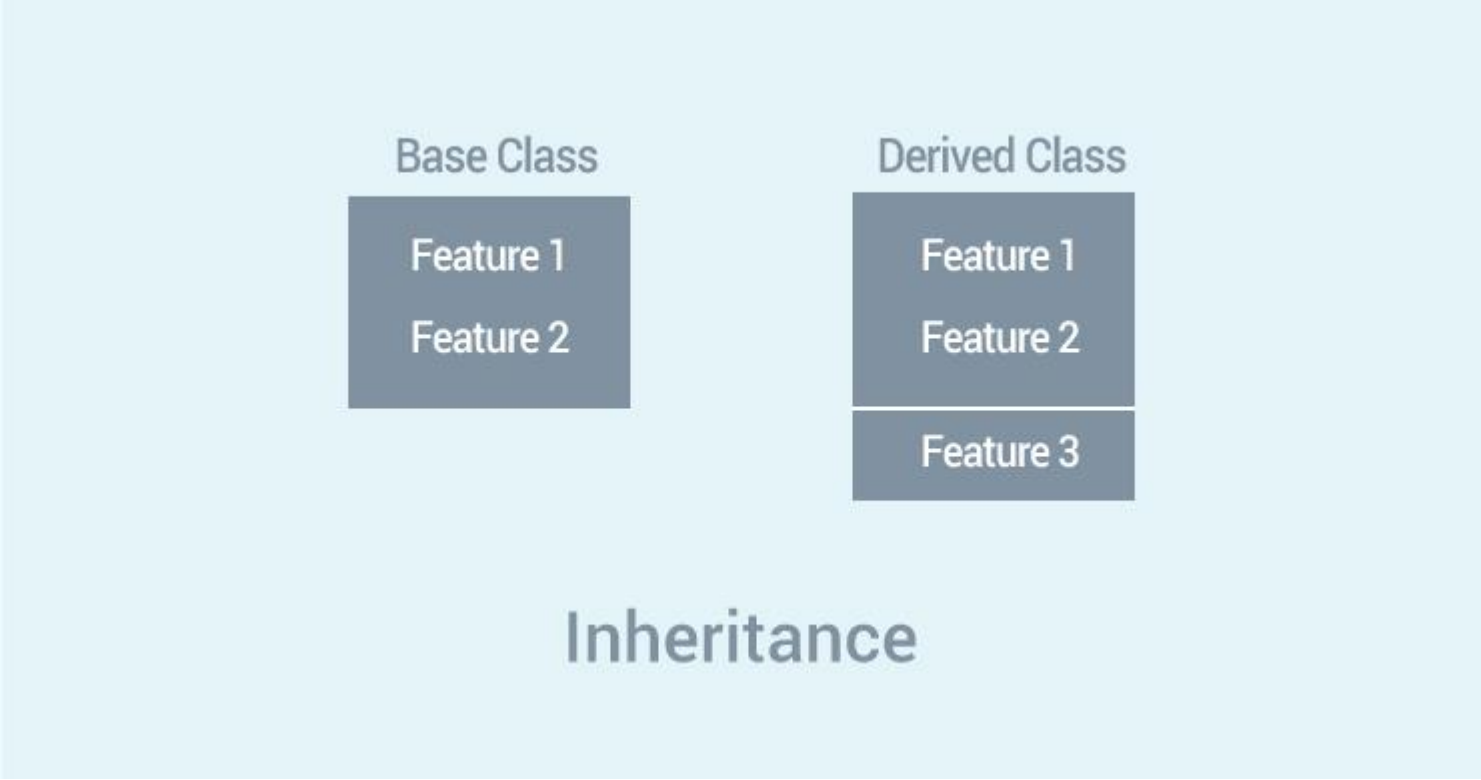
Python Inheritance

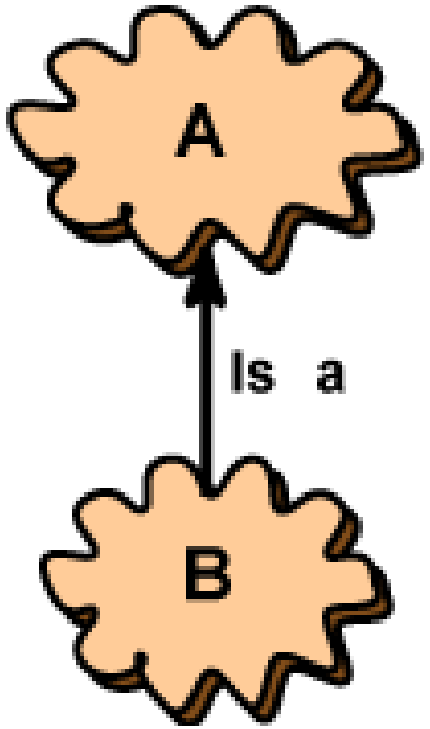
Inheritance

- **Inheritance enable us to define a class that takes all the functionality from parent class and allows us to add more.**

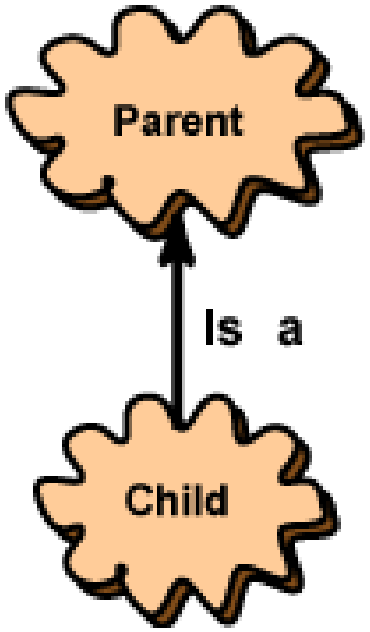


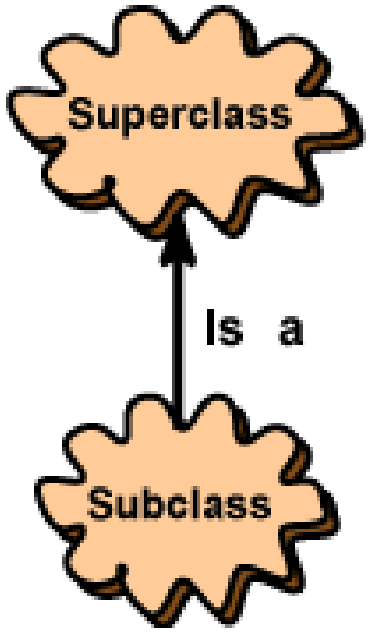
Is-a Relationship

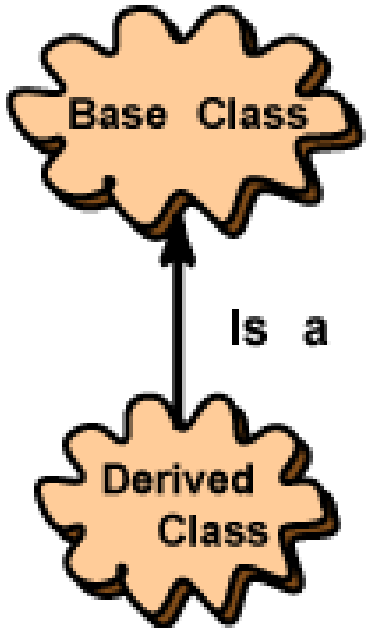
- The subclass and the super class form an “is-a” relationship
- Since the subclass is a specialized “version” of the super class, the subclass object can be viewed “as an” object of the same type as the super class.

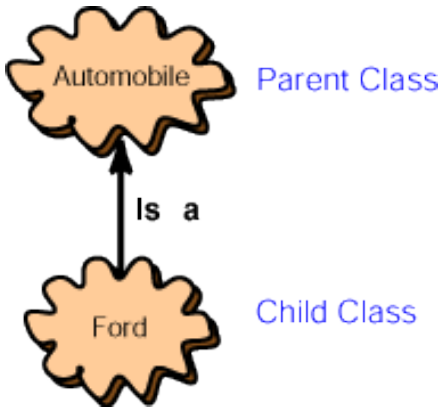


Interchangeable Phrases









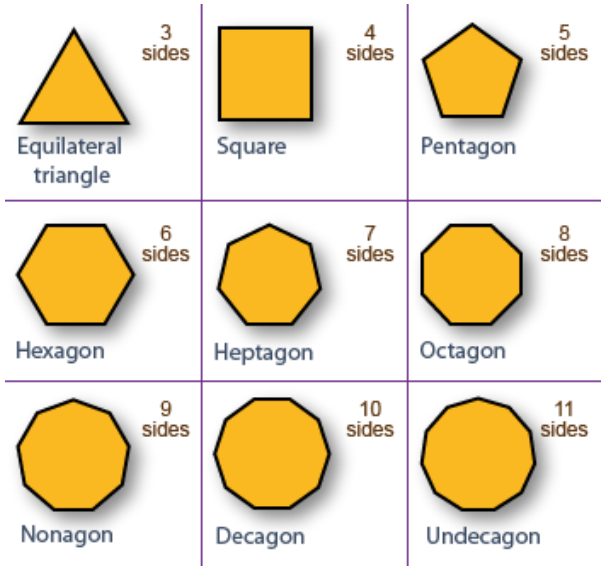
Syntax

```
class BaseClass:
        Body of base class
class DerivedClass(BaseClass):
        Body of derived class
```

- Derived class inherits features from the base class, adding new features to it. This results into re-usability of code.

Example : Polygon

- A polygon is a closed figure with 3 or more sides.




```
class Polygon:
        def __init__(self, no_of_sides):
                self.n = no_of_sides
                self.sides = [0 for i in range(no_of_sides)]

        def inputSides(self):
                self.sides = [float(input("Enter side "+str(i+1)+" : ")) for i in range(self.n)]

        def dispSides(self):
                for i in range(self.n):
                        print("Side",i+1,"is",self.sides[i])
```

Example: Triangle inherits from Polygon

- A triangle is a polygon with 3 sides.

```
class Triangle(Polygon):
        def __init__(self):
                Polygon.__init__(self,3)
        def findArea(self):
                a, b, c = self.sides
                # calculate the semi-perimeter
                s = (a + b + c) / 2 area = (s*(s-a)*(s-b)*(s-c)) ** 0.5
                print('The area of the triangle is %0.2f' %area)
```

Some useful functions

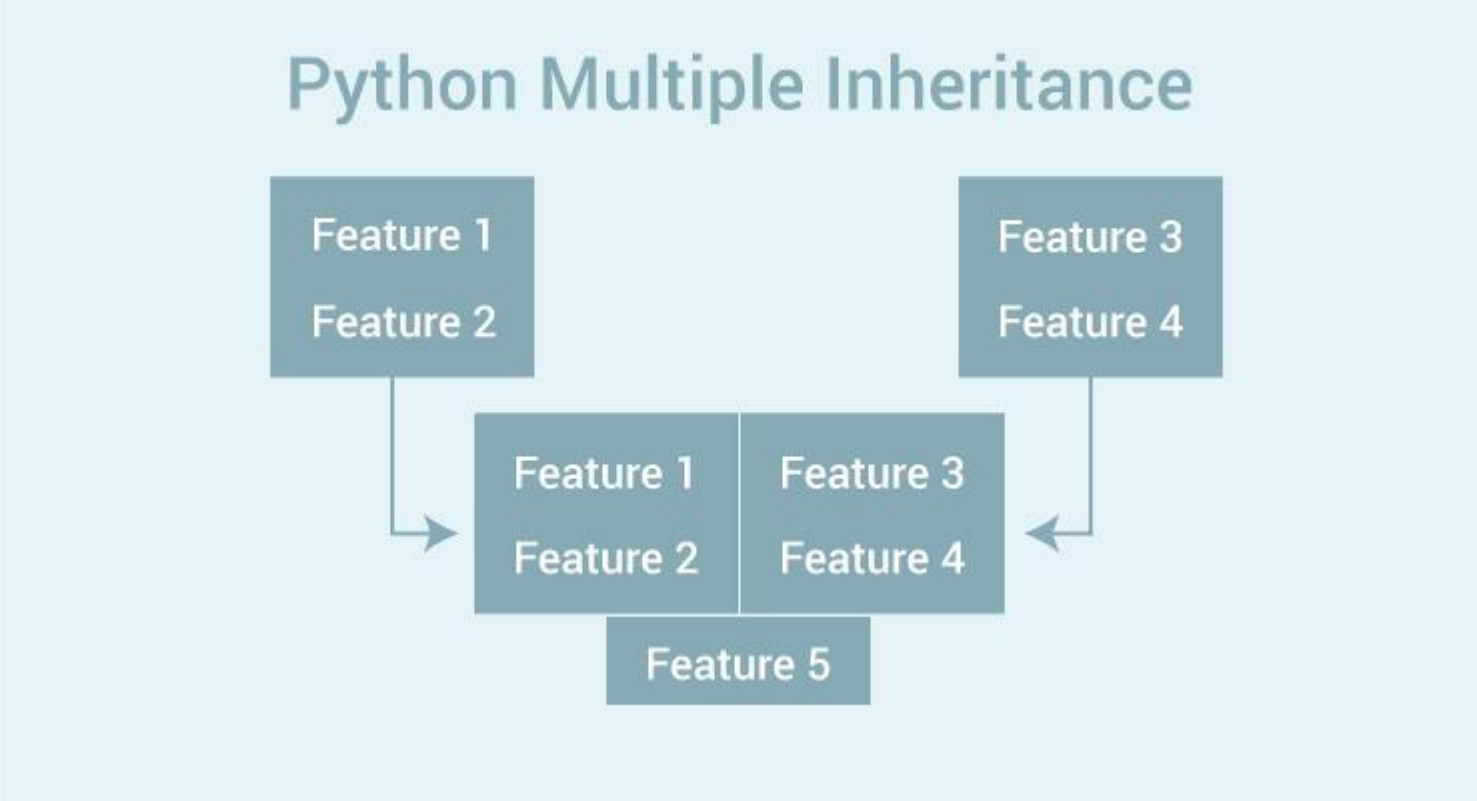
- super()
- isinstance()
  - returns True if the object is an instance of the class or other classes derived from it.
- issubclass()
  - is used to check for class inheritance.

Note: Each and every class in Python inherits from the base class object.

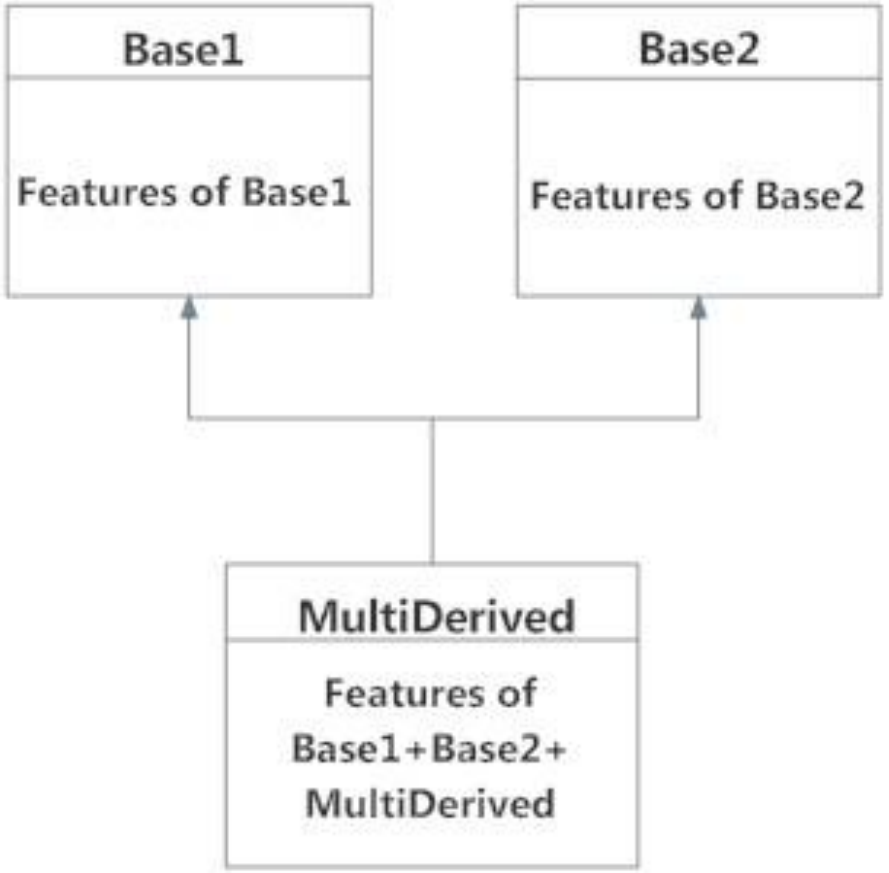
Note: **Inheritance is between <u>classes</u>, not between objects.**

Multiple Inheritance

- A class can be derived from more than one base classes in Python.



Example

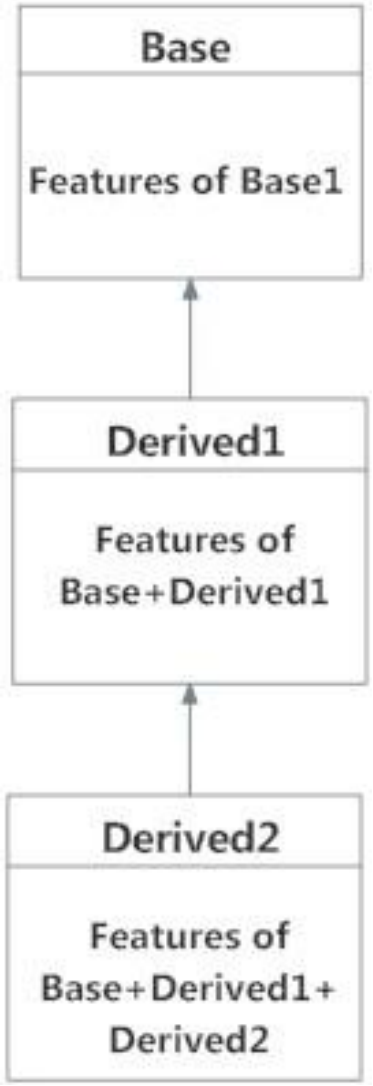



```
class Base1:
      pass
class Base2:
      pass
class MultiDerived(Base1, Base2):
      pass
```

Multilevel Inheritance

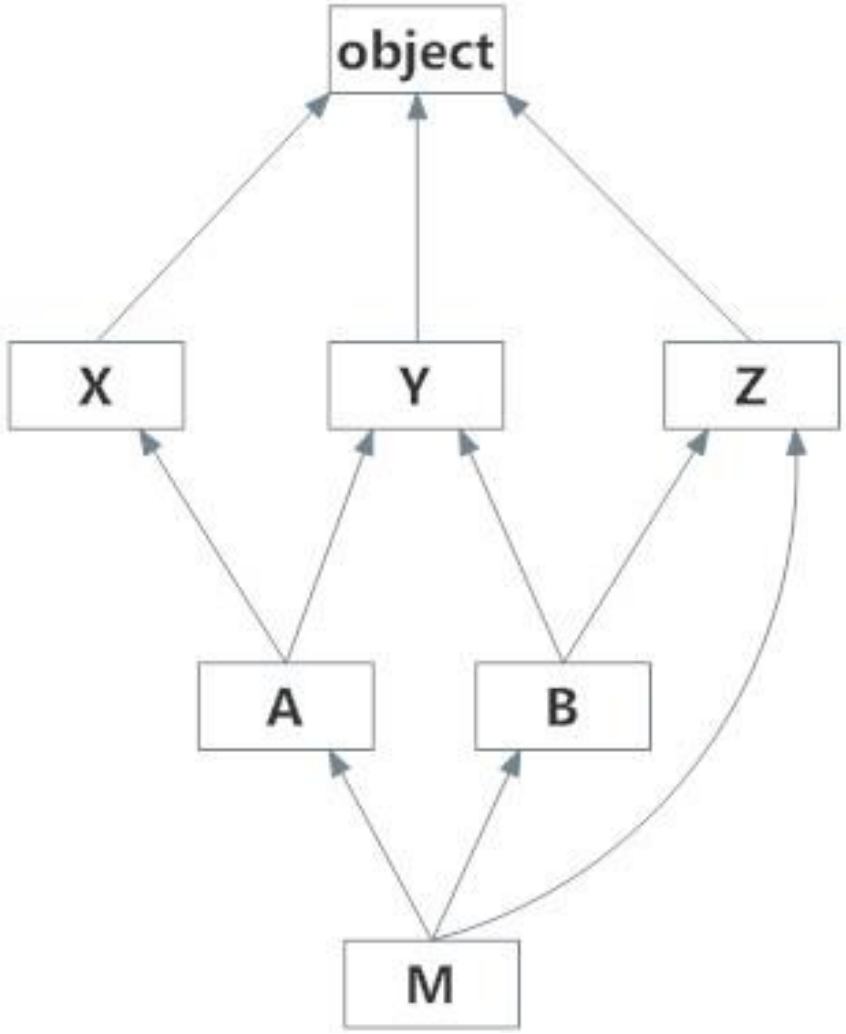
- Features of the base class and the derived class is inherited into the new derived class.

```
class Base1:
      pass
class Derived1(Base):
      pass
class Derived2(Derived1):
      pass
```



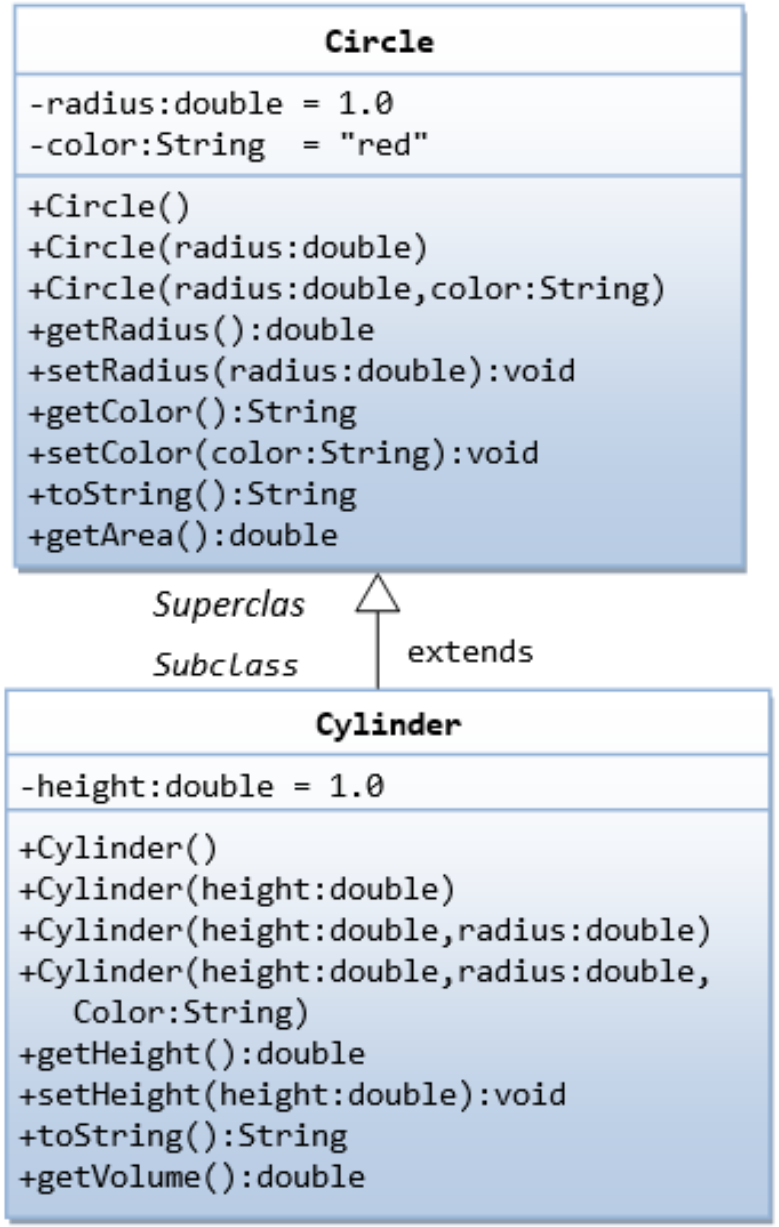
Easy Challenge

- Implement the Python code for the given scenario



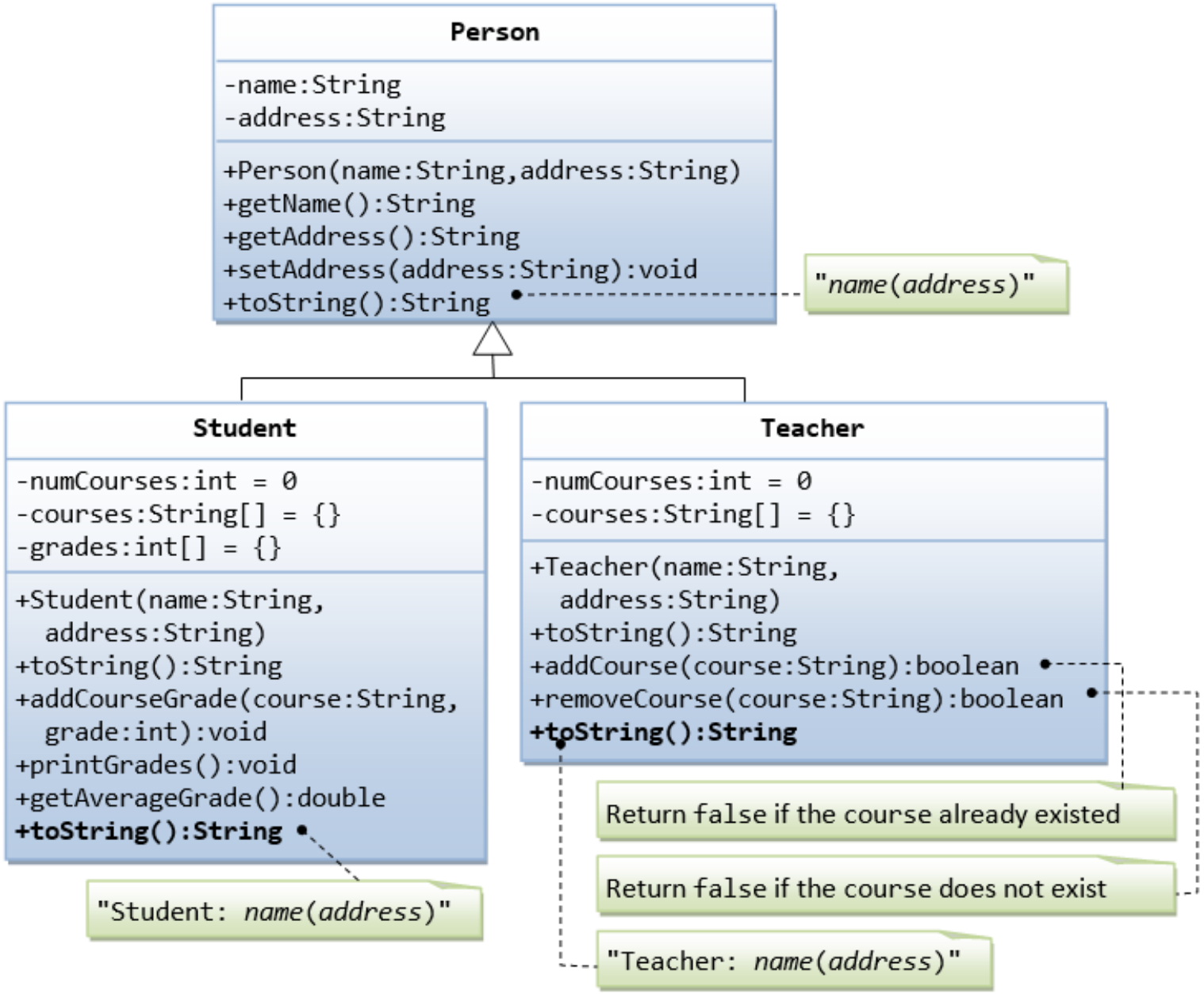
Another Easy Challenge

- Implement the Python code for the given scenario



Moderately Challenging

- Implement the Python code for the given scenario



End