OOP With Python

Unit 05 - Challenges

{<coding:lab}

Quick Review of Prior Sessions

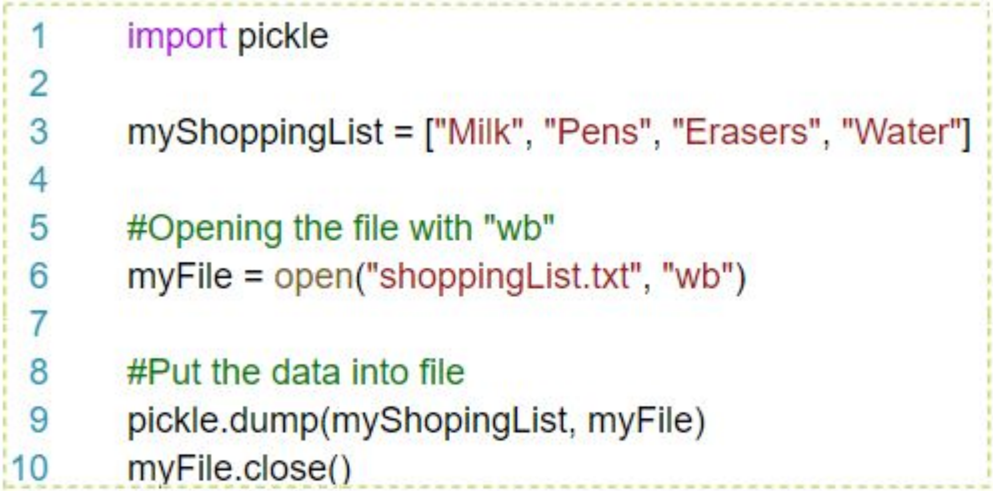
Get to know Python

Persistent Storage

- We can use files to achieve persistent storage of data
- Two key methods to write to file:
  - Writing as text
  - Using pickle module
- There are other methods to write to file
- There are also other methods to achieve persistent storage
  - Database

Using Pickle

- Import module => open file with “wb” => “Dump” data into file



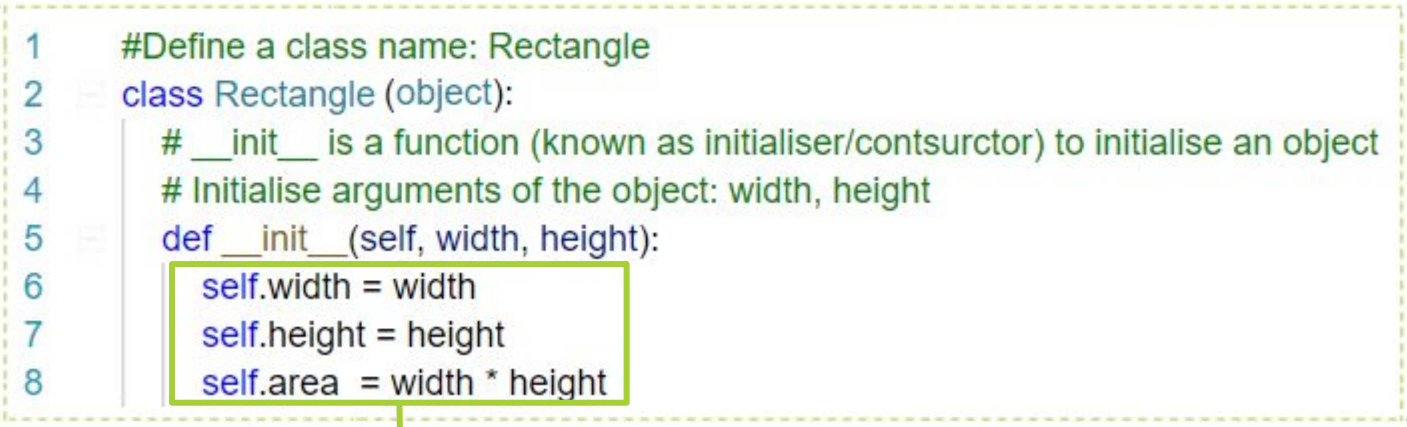
```python
import pickle

myShoppingList = ["Milk", "Pens", "Erasers", "Water"]

#Opening the file with "wb"
myFile = open("shoppingList.txt", "wb")

#Put the data into file
pickle.dump(myShopingList, myFile)
myFile.close()
```

What is Object Oriented Programming

- An approach to programming
- Groups functions and variables together to create classes.
- Each class can be used to create objects
- Objects will share the same variables and functions as the class
- A function in a class is called a method
- A variable that’s part of a class is called an attribute.

Defining a Class

- Create a class by doing the following



```
#Define a class name: Rectangle
class Rectangle (object):
    # __init__ is a function (known as initialiser/contsurctor) to initialise an object
    # Initialise arguments of the object: width, height
    def __init__(self, width, height):
        self.width = width
        self.height = height
        self.area  = width * height
```

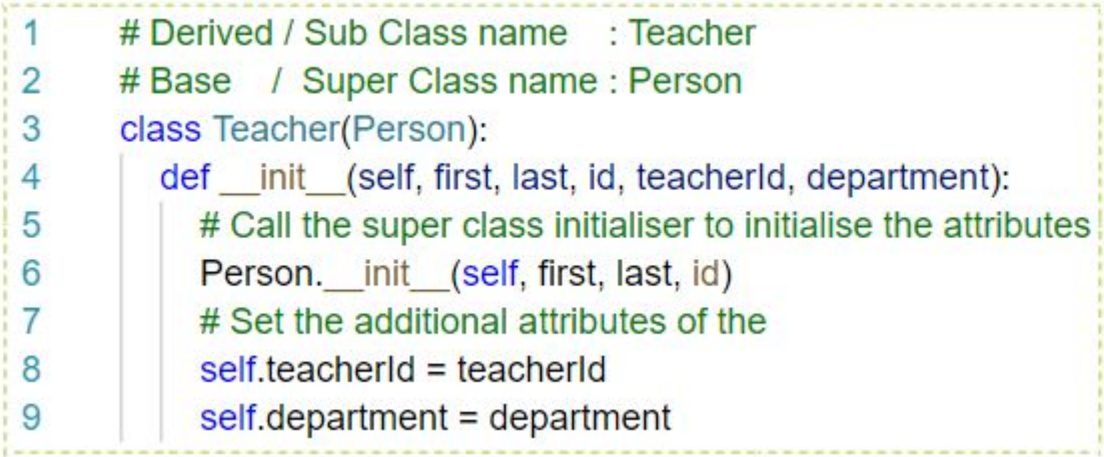
Setting the attributes of the object

What is Inheritance and Why use it?

- Enables a new object or class to take on (inherit) the properties of another existing class
- Enables more efficient coding

Define the Sub Class which Inherits from the Super Class

- Create a class by doing the following



```
# Derived / Sub Class name    : Teacher
# Base    /  Super Class name : Person
class Teacher(Person):
    def __init__(self, first, last, id, teacherId, department):
        # Call the super class initialiser to initialise the attributes
        Person.__init__(self, first, last, id)
        # Set the additional attributes of the
        self.teacherId = teacherId
        self.department = department
```

Challenges

The next few slides contain challenges that can be tackled with what you had learnt. Attempt as many as you can

Shapes

- Write a set of classes used to model various geometry shapes
  - The super class will be “Shape”
  - “Shape” has “Circle” and “Rectangle” as subclasses
  - “Square” in turn is a subclass of “Rectangle”
  - Implement the following methods
    - Calculate area
    - Calculate perimeter
    - Compare area
    - Compare perimeter
    - Plot the shape

Rational Numbers

- Write a class that supports rational numbers
  - The fields should be two variables, for numerator and denominator
  - Store the rational number in reduced form, with the denominator always non negative
  - Provide the following methods:
    - Add (to sum up two rational numbers)
    - Subtract (to subtract one rational number from another)
    - Multiply (to multiply one rational number to another)
    - Divide (to divide one rational number from another)
    - Equals (check if one rational number is equal to another)
    - Compares (to compare if one rational number is the same as another)
    - To String (to return a string representation of the rational number)

Complex Numbers

- Write a Complex number class
  - Complex number consists of a real part and an imaginary part
  - Support the following operations
    - Add (to sum up two rational numbers)
    - Subtract (to subtract one rational number from another)
    - Multiply (to multiply one rational number to another)
    - Divide (to divide one rational number from another)
    - Equals (check if one rational number is equal to another)
    - To String (to return a string representation of the rational number)

Congratulations

Congratulations

- Congratulations! You have completed OOP With Python!