Object Oriented Python

Unit 2 Lab 2

What is OOP?

Object-oriented programming (**OOP**) is a programming paradigm based on concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or *properties*), and code, in the form of procedures (often known as *methods*).

https://en.wikipedia.org/wiki/Object-oriented_programming

Objects are instances of a “class”

Classes are the definitions for an object. An example:

Dog Class
name - string age - int weight - int breed - string

bark()
eat()
sleep()
run()

The Dog class has attributes and methods.

We could create and instance of the Dog class called “rover”.

At creation “rover” would immediately get some defining attributes:

rover.name = “rover”
rover.breed = “dalmation”
rover.weight = 45
rover.age = 1

And, “rover” could do some things:

rover.sleep()
rover.bark()
rover.eat()

What are these attributes?

Attributes are nothing more than variables that are associated with the instance of the class – the object.

Each instance of the Dog class needs to have a name.

What are these methods?

Methods are the functions/procedures/abilities that are associated with the instance of the class.

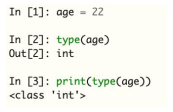
Each instance of the Dog class can do certain things when created. It can sleep, run, eat or bark.

Python is Object Oriented

Python does not force you to use Object Oriented Programming but it does fully support OOP.

If you have ever used a Python variable, you have used an "object."

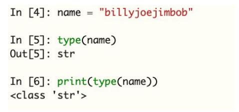
Use the built-in function type()



```
In [1]: age = 22

In [2]: type(age)
Out[2]: int

In [3]: print(type(age))
<class 'int'>
```

Printing the output of type(age) above shows us that "age" is not just an int, but it is of type <class 'int'>. i.e. age is an instance of the 'int' class → an object.

Use the built-in function type()



```
In [4]: name = "billyjoejimbob"

In [5]: type(name)
Out[5]: str

In [6]: print(type(name))
<class 'str'>
```

Printing the output of type(name) above shows us that “name” is not just a string, but it is of type <class ‘str’>. i.e. name is an instance of the ‘str’ class → an object.

You have likely used string “methods” in the past like .upper(), .lower() or .find()

Python is Object Oriented

Everytime you used a string “method” or a list “method” you were using the power of OOP in your programs. Python does not force it upon you but you were using it nonetheless.

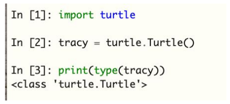
This class will not dive into the creation of new Python classes but you will use Python objects of many different class types.

Tracy the Turtle

Turtle Graphics is a Python module that can be imported and used in your programs. Commonly it is called Tracy the Turtle.

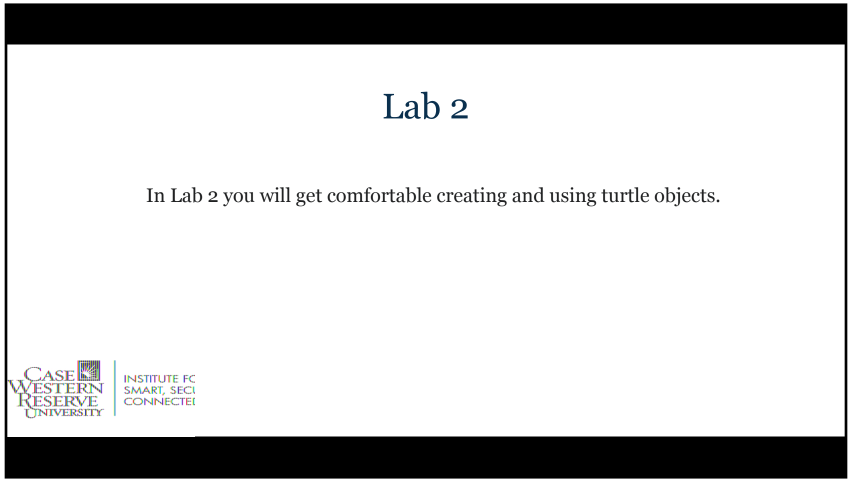
When you import the module you are importing the “class” definitions which you can then use to create objects.

Tracy the Turtle



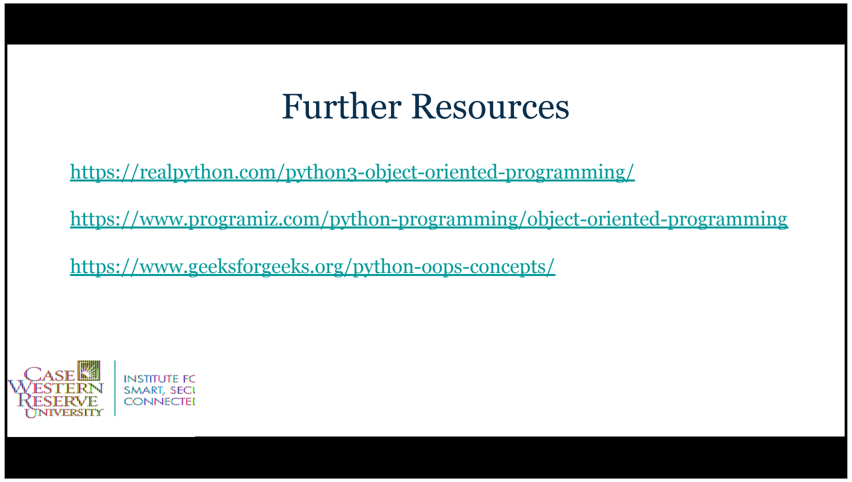
```
In [1]: import turtle

In [2]: tracy = turtle.Turtle()

In [3]: print(type(tracy))
<class 'turtle.Turtle'>
```

After importing the turtle module, we can create an instance of the turtle.Turtle() class called “tracy”. “tracy” has all the turtle attributes and methods from the imported class definition.



Lab 2

In Lab 2 you will get comfortable creating and using turtle objects.



Further Resources

https://realpython.com/python3-object-oriented-programming/

https://www.programiz.com/python-programming/object-oriented-programming

https://www.geeksforgeeks.org/python-oops-concepts/