Object Oriented Programming

Every value in Python is an object. Objects are a way to combine data and the functions that handle that data. This combination is called *encapsulation*. The data items and functions of objects are called *attributes*, and in particular the function attributes are called *methods*. For example, the operator + on integers calls a method of integers, and the operator + on strings calls a method of strings.

Functions, modules, methods, classes, etc are all first class objects. This means that these objects can be

* stored in a container
* passed to a function as a parameter
* returned by a function
* bound to a variable

Attribute of an object can be accessed using the *dot operator*: object.attribute. For example: if L is a list, we can refer to the method append with L.append. Modules are objects in Python, math.pi as accessing the data attribute pi of module object math.

An example of class definition:

**class** **MyClass**(object):

*"""Documentation string of the class"""*

**def** \_\_init\_\_(self, param1, param2):

"This initialises an instance of type ClassName"

self.b = param1 *# creates an instance attribute*

c = param2 *# creates a local variable of the function*

*# statements ...*

**def** f(self, param1):

*"""This is a method of the class"""*

*# some statements*

a=1 *# This creates a class attribute*

Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.