# Lists and tuples and sets

Lists are defined with square brackets[]. They are ordered datasets. Lists are mutable.

For example:

d= [2,2,3,5,4,7]

They also take methods with a (.)

For example, d.append(5) results in:

d= [2,2,3,5,4,7,5]

Sets are unordered data. They are immutable. They take methods.

For example:

d= (2, 2,3,5,4,7)

Dictionaries

Dictionaries take keywords and values. They are defined by {} curly brackets.

For example:

d= { keyword1= value1,

keyword2= value2,

keyword3= value3}

Tuples are defined with parentheses and are immutable. They do not take methods.

# 

# Classes

These are personally fashioned functions. Like a function, they are set and completed with a colon(:). Next line is indented and a def initiating line.

The def line looks like this: def \_\_init\_\_ (self). Which takes (self) as the first argument.

class Parent:

def \_\_init\_\_(self):

A function inside a class is called a method. A property is anything that comes after the dot(.) also called an attribute.

Like this:

class Parent:

def \_\_init\_\_(self, name):

self.name= name

‘name’ here is the property/attribute. Clue; it follows the dot(.)

Classes take in subclasses also called child. They are defined below the parent class and take parent class in round brackets.

For example:

class Child(Parent):

def \_\_\_init\_\_(self, name):

Some methods take in data and some take in functionality. This helps to subdivide a class method from becoming too long and cumbersome.

Getters access the value of an attribute

Setters set the value of an attribute

Deleters delete the value of an attribute