Tuples

They are similar to lists, but where lists are defined between square brackets:

mylist = [12, 7.2, 'porcupine ']

tuples are defined between parentheses or nothing at all:

mytuple = (12, 7.2, 'porcupine')

or

mytuple = 12, 7.2, 'porcupine'

The crucial difference is that a tuple is immutable, meaning you cannot change a tuple once it has been defined, so the following does not work:

mytuple[2] = 'hedgehog' # bad!

but you can read individual elements of a tuple:

print(mytuple[2])

outputs

'porcupine'

Note that if you include a list as one of the tuple elements, you can change the contents of the list:

anothertuple = (3, 4, ['horse', 'meringue'])

anothertuple[2][1] = 'cream cake'

Dictionaries

A dictionary consists of key-value pairs. The key consists of any immutable type, and it is separated from the value by a colon. Pairs are comma separated, and appear between braces (curly brackets):

examscore = { "John":74, "Susan":83, "Sam":37 }

Elements are accessed using the dictionary name and the key in square brackets:

print(examscore["Susan"])

Dictionaries are not immutable; you can change values:

examscore["Susan"] = 79

You can also add values:

examscore["Chris"] = 23

and delete values:

del examscore["John"]

Note that keys should be unique, but if they are not, the rightmost one overwrites the previous one.

Sets

A set consists of unique values of any type, comma separated, between braces.

myset = { 12, "sandcastle", 9.2 }

Duplicate values are ignored. Python does not store the values in any particular order. You can add values:

myset.add(1000)

or remove them:

myset.remove("sandcastle")