T2.2 Tuples

Comme les listes, en Python les **tuples** sont des séquences ordonnées d'éléments (de n'importe quel type).

▸ On les déclare avec des parenthèses:

Script Python

```
>>> t1 = (1, 2, 4, 8, 16, 32, 64, 128)
>>> t2 = ("Gon", "Kirua", "Leolio")
>>> type(t1)
<class 'tuple'>
```

▸ Comme ils sont ordonnés, on peut désigner les éléments du tuple par leur indice, comme une liste. La fonction `len` est également valable.

Script Python

```
>>> t1[2]
4
>>> t2[0]
'Gon'
>>> t1[-1]
128
>>> len(t2)
3
```

▸ Un tuple est un objet *iterable*, on peut donc le parcourir:

Script Python

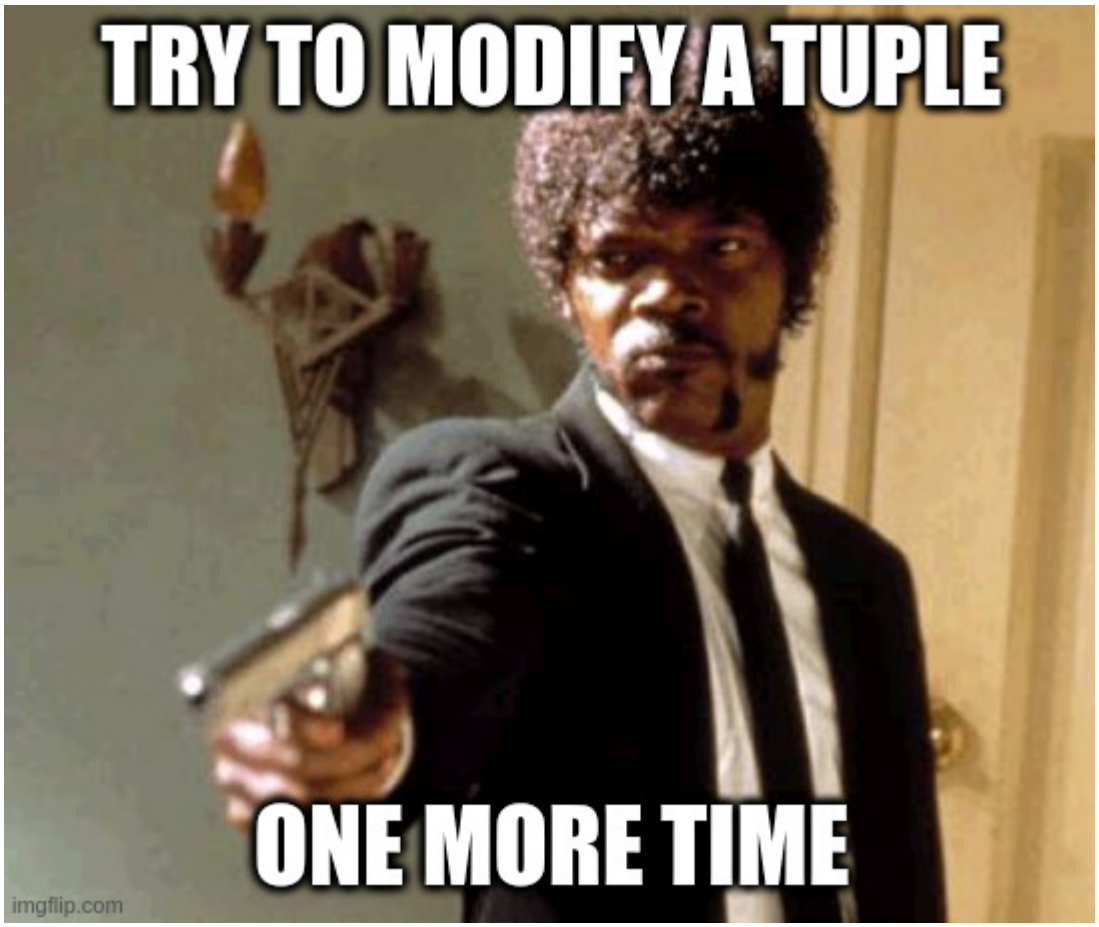
```
>>> for nom in t2:
...     print(nom, "est un hunter")
...
...
Gon est un hunter
Kirua est un hunter
Leolio est un hunter
```

Différence entre tuple et liste

Un tuple est un objet **non mutable**, c'est-à-dire qu'on ne peut pas le modifier : ni réaffecter ses éléments, ni ajouter ou supprimer d'élément.

Script Python

```
>>> t = (1, 2, 3)
>>> t[0] = 0
Traceback (most recent call last):
File "<console>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
>>> t.append(4)
Traceback (most recent call last):
File "<console>", line 1, in <module>
AttributeError: 'tuple' object has no attribute 'append'
>>> t.remove(1)
Traceback (most recent call last):
File "<console>", line 1, in <module>
AttributeError: 'tuple' object has no attribute 'remove'
```



Utilisation d'un tuple

On se sert d'un tuple pour représenter une séquence de données **qui n'a pas vocation à être modifiée au cours du programme**, comme par exemple des constantes.

On verra plus tard dans l'année une vertu du caractère non mutable de ce type d'objet.