Les listes

En Python, la liste est un type d'objet pouvant contenir plusieurs objets de tout autre type. Voici comment se définit une liste en Python:

In [ ]:
```
jours = ['Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi', 'Samedi', 'Dimanche']

print(f'Le 4e jour de la semaine est {jours[3]}')
```

```
Le 4e jour de la semaine est Jeudi
```

Intervalle

La fonction `range()` permet de créer une liste d'élément partant d'un élément à un autre suivant l'intervale précis.

In [ ]:
```
intervale = range(0, 20, 2) #définit un intervalle d'élément allant de 0 inclus à 20 exclus, avec une raison
 de 2 pas
print(f'Les éléments de mon intervale sont {list(intervale)}')
```

```
Les éléments de mon intervale sont [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

Vous pouvez accéder au N-ième élément d'une liste en précisant l'indice numérique N-1. Comme dans l'exemple ci-dessus, nous avons pu accéder le 4e jour de la semaine grâce à l'indice 3. Rappel: `jours[3]` affiche `Jeudi`

Notez que l'indice peut être un entier inférieur à zéro; dans ce cas, le comptable commence par le dernier élément. Voici quelques exemples:

- `jours[-1]` affiche `Dimanche` car il est le dernier élément, soit premier élément en partant de la droite à la gauche.
- `jours[-2]` affiche `Samedi` , soit deuxième élément en comptant de droite à gauche

Illustration

In [ ]:
```
jours[-1]
jours[-2]
jours[0:-1]
```

Out[ ]:
```
['Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi', 'Samedi']
```

N.B: Vous pouvez également utiliser les indices numériques sous forme d'intervale, séparés par deux points. Dans ce cas les éléments pris en compte seront ceux compris entre les indices. Voici un exemple:

In [ ]:
```
jours[0:2] #Lundi, Mardi
jours[2:4] #Mercredi, Jeudi
jours[0:-2] #Lundi, Mardi, Mercredi, Jeudi, Vendredi
```

Out[ ]:
```
['Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi']
```

Vous pouvez vérifier l'existence d'un élément dans une liste grâce à l'opérateur `in` . Celui-ci retournera un booléen `True` si l'élément est dans la liste et `False` au cas contraire.

In [ ]:
```
'Lundi' in jours #True
'Mon jour préféré' in jours #False
```

Out[ ]:
```
False
```

In [ ]:
```
joursOuvrables = ['Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi']
fruits = ['Orange', 'Mangue', 'Avocat', 'Pomme', 'Banane']
apprenant = ['Hervé', 'Mvwezolo', 14, 4.85, (1,3,5,7,9)]
```

Dans le code ci-dessus, vous pouvez accéder au dernier jour ouvrable en utilisant l'indice -1.

Vous pouvez vérifier si le fruit 'Fraise' est contenu dans la liste des fruits en utilisant l'instruction `'Fraise' in fruits` Ceci retournera `False` car la chaine `'Fraise'` ne se retrouve pas dans la liste des fruits définis dans la liste.

Les tuples et les ensembles

Les tuples sont des listes immutables, c'est à dire que leur contenu ne peut pas être changés par une fonction. Voici omment définir un tuple.

In [ ]:
```
 #Définition d'un tuple
chiffres = (1,3,5,7,9,9)

#Définition d'un ensemble. Ne contient que des éléments uniques
chiffresImpairs = set(chiffres)

print(f'Les chiffres dans le tuple sont: {chiffres}')
print(f'Les chiffres impairs sont: {chiffresImpairs}')
```

```
Les chiffres dans le tuple sont: (1, 3, 5, 7, 9, 9)
Les chiffres impairs sont: {1, 3, 5, 7, 9}
```

Toutes les notions apprises sur les listes, à l'exception de l'immutabilité, s'appliquent aussi aux tuples.

Déterminer l'indice d'un élément dans une liste, ou un tuple

In [ ]:
```
chiffresMagics = [1, 48, 6, 2, 10]
chiffresMagics.index(48) #retourne 1

maTuple = (5, 10, 8, 45)
maTuple.index(8) #retourner 2
```

Out[ ]:
```
2
```