Labo 7

Tuples et dictionnaires

Objectifs de ce laboratoire

Exercices avec:

- Tuples, dictionnaires et autres structures de données.
- Algorithmes qui utilisent des structures de données.

Tuples

- Comme les liste, mais pas modifiables

```
>>> t1 = (1,2,3,4,5)
>>> len(t1)
5
>>> t1[0]
1
>>> t1[0] =7
Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    t1[0] =7
TypeError: 'tuple' object does not support
item assignment
```

Dictionnaires

- Les ***dictionnaires*** ne sont pas des séquences.
- On a des valeurs (de n'importe quel type) et pour accéder a ces valeurs on utilise un index spécifique que l'on appellera une ***clé*** (alphabétique, numérique, ou autre).

```
>>> d1 = {'computer': 'ordinateur',
        'keyboard': 'clavier', 'mouse': 'souris'}
>>> d1['printer'] = 'imprimante'
>>> d1
{'keyboard': 'clavier', 'printer': 'imprimante',
'computer': 'ordinateur', 'mouse': 'souris'}
>>> len(d1)
4
```

Dictionnaires (suite)

- Les dictionnaires ne sont modifiables. On peut ajouter ou enlever des éléments. On peut changer la valeur d'une clé.

```
>>> d1['computer'] = 'ordi'
>>> d1.keys()
dict_keys(['keyboard', 'printer', 'computer', 'mouse'])
>>> d1.values()
dict_values(['clavier', 'imprimante', 'ordi', 'souris'])
>>> 'printer' in d1
True
```

Construction d'un histogramme à l'aide d'un dictionnaire

```
>>> texte ="les saucisses et saucissons secs sont dans le saloir"
>>> lettres ={}
>>> for c in texte:
            lettres[c] =lettres.get(c, 0) + 1

>>> print(lettres)
{'t': 2, 'u': 2, 'r': 1, 's': 14, 'n': 3, 'o': 3, 'l': 3, 'i': 3, 'd': 1, 'e': 5, 'c': 3, ' ': 8, 'a': 4}
>>> lettres_triees = list(lettres.items())
>>> lettres_triees.sort()
>>> print(lettres_triees)
[(' ', 8), ('a', 4), ('c', 3), ('d', 1), ('e', 5), ('i', 3), ('l', 3), ('n', 3), ('o', 3), ('r', 1), ('s', 14), ('t', 2), ('u', 2)]
```

Exercice 1

- Créez une fonction qui prend une chaire de caractères et retourne un histogramme sous la forme d'un dictionnaire
- Le programme principal doit prendre le dictionnaire et l'afficher en ordre alphabétique.

Note d.get(val1, val2) va retourner la valeur de la clé val1 dans le dictionnaire d, ou val2 sinon. Si on fait accès direct d[val1], ça donne erreur si la clé val1 n'existe pas

Exercice 2

- Créez une fonction qui prends un tuple et retourne un dictionnaire qui est une histogramme pour compter combien de fois chaque nombre arrive dans le tuple:
- Le programme principal doit prendre le dictionnaire, transformer les items dans une liste, utiliser sort() pour ordonner la liste (en ordre ascendant) et après ça afficher la liste.

```
def histo_n(x):
    '''(tuple)->dict
    Retourne un dictionnaire
    Precondition: le tuple contient des entiers
    >>> t = (1,2,-3,3,4,-3,3,3)
    >>> histo_n(t)
    {1: 1, 2: 1, 3: 3, 4: 1, -3: 2}
    '''
```

Exercice 3

- Créez une fonction avec la description suivante:

```
def somme_de_trois(x):
    '''(tuple)->bool
    Retourne True si la somme de 3 elements
    consecutive est zero
    Precondition: le tuple a au moins 3
    elements
    >>> t = (1,2,-3,4,-1,3)
    >>> somme_de_trois(t)
    True
    '''
```

Exercice 4

Créez une fonction move_zeros qui prend comme paramètre une liste des entiers et déplace tous les zéros à la fin de la liste. Par exemple, si la liste est [1, 0, 3, 0, 0, 5, 7] le résultat devrait être [1, 3, 5, 7, 0, 0, 0]

Préparez TROIS solutions

- move_zeros_v1 utilise une autre liste tmp pour calculer la nouvelle liste et la retourne comme résultat (problème facile) . La liste initiale n'est pas changée.
- move_zeros_v2 modifies la liste initiale à l'intérieur de la fonction. La fonction retourne rien.
- move_zeros_v3 déplace les éléments dans la liste initiale sans utiliser des listes temporaires (probleme plus difficile). La fonction retourne rien. On peut utiliser une variable temporaire pour échanger deux éléments, mais on peut utiliser l'échange Python `a,b=b,a`

Exercice 4 (suite)

```
>>> x = [1, 0, 3, 0, 0, 5, 7]
>>> y=move_zeros_v1(x)
>>> print(x, y)
[1, 0, 3, 0, 0, 5, 7] [1, 3, 5, 7, 0, 0, 0]

>>> x = [1, 0, 3, 0, 0, 5, 7]
>>> z=move_zeros_v2(x)
>>> print(x, z)
[1, 3, 5, 7, 0, 0, 0] None

>>> x = [1, 0, 3, 0, 0, 5, 7]
>>> t=move_zeros_v2(x)
>>> print(x, t)
[1, 3, 5, 7, 0, 0, 0]  None
```