FEUILLE n°8 *Révisions*

Exercice 1. Min, max et sorted

1.1. Quelle est la valeur de `mystere1` et `mystere2` après l'exécution du code suivant ?

```
exemple = [(6, 6), (12, 3), (9, 1), (2, 3) (5, 7), (5, 6)]

def critere1(couple):
    return couple[1]

def critere2(couple):
    return couple[0]+couple[1]

mystere1 = min(exemple, key = critere1)
mystere2 = sorted(exemple, key = critere2)
```

1.2. Proposer un code de la fonction `tri_selon_nombre_de_villes` :

```
centre = {'Loiret' : {'Orleans', 'Trainou', 'Chécy', 'Ardon', 'Amilly'},
    'Cher' : {'Vierzon', 'Quincy', 'Bourges'},
    'Indre' : {'Chateauroux', 'Issoudin', 'Déols', 'Le Blanc'}}

bretagne = {'Morbihan': {'Lorient', 'Vannes', 'Ploemeur'},
    'Finistère' : {'Cast', 'Brest', 'Quimper', 'Concarneau', 'Landerneau'},
    'Cotes Armor': {'Lannion'}}

def tri_selon_nombre_de_villes(region):
    """
    region est un dictionnaire dont les clés sont des régions/départements et
    la valeur associée un ensemble de villes
    renvoie la liste des clés rangées dans l'ordre croissant de leur
    nombre de villes
    """
    pass

assert tri_selon_nombre_de_villes(centre) == ['Cher', 'Indre', 'Loiret']
assert tri_selon_nombre_de_villes(bretagne) == ['Cotes Armor', 'Morbihan',
    'Finistère']
```

Exercice 2. Représentation de la mémoire

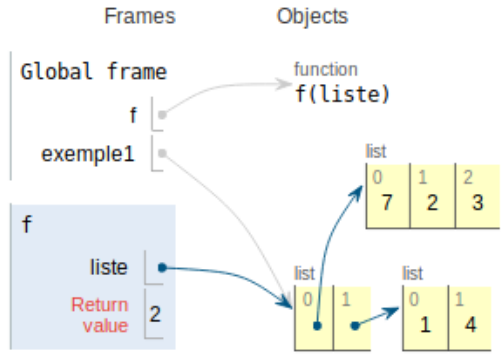
2.1. On exécute le script suivant. Représenter l'état de la mémoire au premier passage par l'endroit indiqué. Préciser la valeur de `dernier` à l'issue du script.

```
def reste(nombre):
  exemple = nombre % 3
  # ICI
  return exemple

def truc(liste):
   somme = 0
   for elem in liste:
       somme = somme + reste(elem)
   return somme

def ajoute(liste):
    copie = liste.copy()
    copie.append(7)
    return copie

exemple = [10, 5, 9]
suivant = ajoute(exemple)
dernier = truc(suivant)
```

Exercice 3. Représentation de la mémoire

```
# Script à compléter
exemple1 = [[7, 2, 3], [1, 4]]
```

3.1. Compléter le script donné sachant qu'il provoque les représentations de la mémoire ci-dessous. Préciser à quel endroit dans le script on se trouve.



Exercice 4. Bibliothèque On représente les livres d'une bibliothèque par un dictionnaire dont les clés sont les titres des livres, et les valeurs sont des couples contenant le nom de l'auteur et le genre du livre. Par exemple :

```
exemple = {
    'Aurélien' : ('Aragon', 'Roman'),
    'Brocéliande' : ('Aragon', 'Poésie'),
    "L'Avare" : ('Molière', 'Théâtre'),
    '1984' : ('Orwell', 'Roman'),
    'La Ferme des animaux' : ('Orwell', 'Roman'),
    'Elsa' : ('Aragon', 'Poésie')}
```

4.1. Compléter le code de la fonction `cherche_auteurs` qui, à partir d'une bibliothèque et d'un genre renvoie l'ensemble des auteurs qui ont écrit un livre de ce genre dans la bibliothèque. Inutile d'écrire la documentation, mais préciser la complexité de la fonction.

```
assert cherche_auteurs(exemple, 'Encyclopédie') == set()
assert cherche_auteurs(exemple, 'Roman') == {'Aragon', 'Orwell'}
```

4.2. Compléter le code de la fonction `auteurs` qui, à partir d'une bibliothèque construit un dictionnaire dont les clés sont le nom des auteurs et la valeur associée le nombre d'ouvrages qu'ils ont écrit présents dans la bibliothèque. Inutile d'écrire la documentation, mais préciser la complexité de la fonction.

```
assert auteurs(exemple) == {'Aragon': 3, 'Molière': 1, 'Orwell': 2}
```

4.3. Compléter le code de la fonction `ouvrages_par_genre` qui transforme une bibliothèque en un dictionnaire dont les clés sont les genres, et les valeurs sont l'ensemble des ouvrages de ce genre. Inutile d'écrire la documentation, mais préciser la complexité de la fonction.

```
assert ouvrages_par_genre(exemple) == {
    'Roman': {'1984', 'Aurélien', 'La Ferme des animaux'},
    'Poésie': {'Brocéliande', 'Elsa'},
    'Théâtre': {"L'Avare"}}
```

4.4. Compléter le code de la fonction `le_moins_représenté` qui, à partir d'une bibliothèque renvoie le nom de l'auteur le moins représenté dans la bibliothèque.

```
assert le_moins_représenté(exemple) == 'Molière'
```