Méthodes et effets de bord

La programmation orientée objet est une approche de programmation différente de la programmation fonctionnelle. Elle est hors programme. Mais l'usage des méthodes qu'elle propose est d'un intérêt indéniable.

1 Méthodes sur les listes et les strings

On rappelle qu'une liste est modifiable, et qu'une chaîne ne l'est pas.

1.1 Sur les listes

Les méthodes modifient la liste à laquelle elles sont appliquées. Soit `L` une liste quelconque.

- L.`append(x)` : ajoute l'élément `x` à la fin de la liste L
- L.`extend(T)` : concatène la liste `T` à la fin de la liste L
- L.`remove(x)` : retire l'élément `x` de la liste L, si `x` est présent
- L.`pop(i)` : retire le $i^{\text{ème}}$ élément et renvoie sa valeur
- L.`index(x)` : renvoie l'indice de l'élément `x` si `x` est présent
- L.`insert(i,x)` : insère l'élément `x` à l'indice `i` dans L.
- L.`count(x)` : renvoie le nombre de fois où l'élément `x` est présent dans la liste L
- L.`reverse()` : renverse l'ordre des éléments de la liste L
- L.`sort()` : trie les éléments de la liste L par ordre croissant
- L.`clear()` : vide la liste L de tous ses éléments
- L.`copy()` : renvoie une copie de la liste L (voir ci-dessous)

1.2 Méthodes sur les chaînes de caractères

Une chaîne de caractère est non modifiable. Les méthodes renvoient donc une nouvelle chaine, qu'il faudra affecter à une variable. On ne présente ici que celles utiles dans le cadre de ce cours. Soit `S` une chaîne de caractère quelconque.

- S.`isalpha()` : renvoie `True` si tous les caractères de S sont des lettres, `False` sinon.
- S.`isdigit()` : renvoie `True` si tous les caractères de S sont des chiffres, `False` sinon
- S.`upper()` : renvoie une chaîne en remplaçant les minuscules de S par des majuscules
- S.`lower()` : renvoie une chaîne en remplaçant les majuscules de S par des minuscules
- S.`find(sub)` : renvoie le plus petit indice de la chaîne de caractères `sub` recherchée dans la chaîne principale S. Cette fonction renvoie -1 si la recherche n'a pas abouti
- S.`count(sub)` : renvoie le nombre d'occurences de la chaîne de caractères `sub` dans la chaîne principale S
- S.`replace(old,new)` : renvoie une copie de la chaîne de caractères en remplaçant toutes les occurrences de la chaîne `old` par `new`.
- S.`split(sep)` : renvoie une liste de mots correspondante à S découpée selon le séparateur `sep`
- S.`lstrip(car)` : renvoie une copie de la chaîne S où les caractères `car` situé en début de liste ont été retirés

2 Effet de bord

2.0.1 Les listes en tant qu'objet

L'affectation d'une liste vers une liste n'est pas une copie du contenu, mais du référencement.

```
In [1]: L1 = [1, 2, 3]
[1, 2, 3]
In [2]: L2 = L1
[1, 2, 3]
In [3]: id(L1),id(L2)
(200563912, 200563912)
In [4]: L1.append(4)
In [5]: print(L2)
[1, 2, 3, 4]
```

Pour créer une nouvelle liste identique à l'originale mais indépendante, il faut utiliser les instructions nécessaires à la copie stricte, et/ou profonde :

- grâce à un transtypage : `copie1 = list(maListe)`
- grâce à une extraction de tous les éléments de maListe : `copie2 = maListe[:]`
- grâce à la méthode copye : `copie3 = maListe.copy()`
- avec le module copy :
  ```
  import copy
  copie4 = copy.deepcopy(maListe)
  ```

2.1 Mise en évidence de l'effet de bord

Cet effet secondaire (indésirable ou non) est à maitriser avec subtilité. Il apparait lors de l'usage combiné de variables évoluées (collections, images, etc.) et de fonctions.

```python
def ordre1(M):
    N=sorted(M)
    return(N)

def ordre2(M):
    M.sort()
    N=M
    return(N)

L=[2,1,4,3]
print('liste originelle : ',L)
print(ordre1(L))
print('liste originelle : ',L)
print(ordre2(L))
print('liste originelle : ',L)
```

Pour se protéger d'un effet de bord lors de l'usage d'un objet dans une fonction, il est nécessaire de travailler sur une copie de l'objet.