09- Portée des variables locales (Programmation)

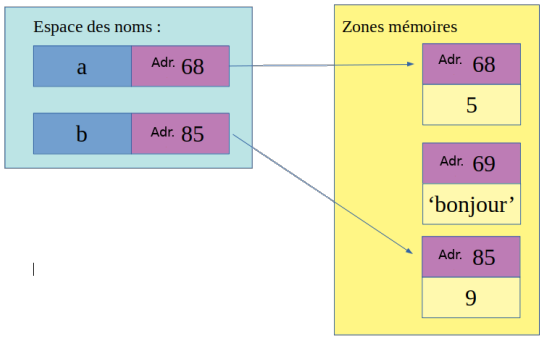
I – Identifiants des variables

1.1 Espace des noms

Les variables Python ne peuvent pas être vues comme de simples boîtes.

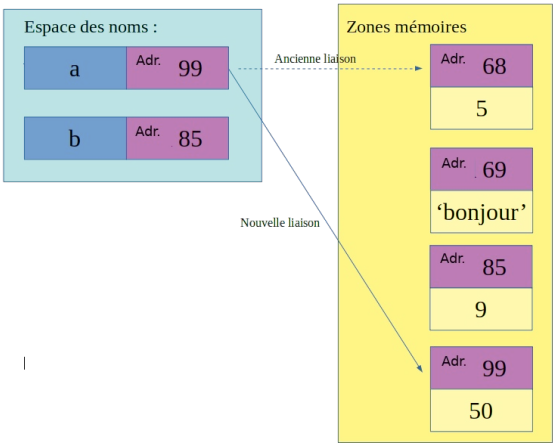
L'interpréteur Python dispose pour le programme principal et pour chaque lancement de fonction de tableaux qu'on nommera ESPACES DES NOMS.

Chacun de ces tableaux associe un nom de variable à un identifiant permettant de localiser une zone mémoire.



1.2 Affectation

Une affectation correspond donc en Python à la création ou la modifcation d'une des lignes dans l'un des tableaux.



1.3 Fonction native id()

```
>>> a = 50
>>> id(a) 99 # Valeur imaginaire ci-dessus
```

II – Espace des noms

```
1       def positif(x1):
2           if x1 < 0:
3              x1 = -x1
4           return x1
5
6       def pas_negatif(x2):
7           if x2 < 0:
8               x2 = 0
9           return x2
10
11      xpp = -5
12      y = positif(xpp)
13      z = pas_negatif(xpp)
```

Sur cet exemple, j'ai rajouté des indications derrière les variables notées x mais elles ne sont pas nécessaires. On pourrait garder les mêmes instructions en ne notant que x à chaque fois sans que l'interpréteur Python ne se trompe.

Il suffit de se souvenir que le programme principal ainsi que chaque lancement de fonction a son propre espace des noms.

Ainsi le x des lignes 1, 2, 3 et 4 est la variable x de la fonction positif. Et il n'a rien à voir avec le x des lignes 11, 12 et 13 qui correspond à la variable x du programme principal.

III – f-string

```
def exemple_f():
    a = 5
    return f"La variable a contient {a}"
```

Le résultat dans la console :

```
>>> exemple_f()
"La variable a contient 5"
```

IV – Portée de l'espace des noms

L'espace des noms d'une fonction est détruit dès que la fonction est terminée.