Chapitre 5

Variables globales/locales

5.1 Passage d'une valeur à une fonction

5.1.1 Exemple avec des scalaires

• Une variable utilisée dans la définition d'une fonction est créée à l'appel de la fonction, et détruite à la sortie.

• Lorsque l'on passe la valeur d'une variable en paramètre à une fonction, tout se passe comme si c'était une copie de cette valeur qui lui était passée :

```
def incremente(x):  
    x = x+1  
    return x
```

```
>>> x = 1  
>>> print(incremente(x))  
2  
>>> print(x)  
1  
>>> x = x+1  
>>> print(x)  
2
```

5.1.2 Exemples avec des listes

Autre exemple avec des listes :

```
def annule(L): # On passe à annule() copie de l'adresse de L  
    L = [0] * len(L) # Une nouvelle liste est créée !  
    return L
```

```
>>> L = [1, 2]  
>>> L1 = annule(L) ; print(L1)  
[0,0]  
>>> print(L)  
[1,2]
```

```
def Rallonge(L): # On passe à Rallonge() copie de l'adresse de L  
    L.append(0)  
    return L
```

```
>>> L = [1, 2]  
>>> L1 = Rallonge(L) ; print(L1)  
[1,2,0]  
>>> print(L)  
[1,2,0]
```

5.2 Variables locales, variables globales

5.2.1 Définition

- Une variable peut être globale ou locale selon qu'elle est définie :
 1. Dans l'environnement de l'appel d'une fonction : variable globale
 2. Dans une définition de fonction : variable locale

```
# Environnement du programme  
x = 10 # x variable globale  
  
# Définition d'une fonction fonct()  
def fonct():  
    y = 1 # y variable locale
```

• Une variable locale est créée à l'appel d'une fonction et détruite à la sortie de la fonction :

```
>>> fonct()  
>>> # Sortie de la fonction fonct()  
>>> print(y)  
NameError: name 'y' is not defined
```

L'appel de la variable à l'extérieur de la fonction la définissant produit une erreur.

5.2.2 Portée et modifiabilité d'une variable globale

- Une variable globale est accessible et modifiable dans l'environnement du programme où elle est définie :

```
# Environnement du programme
x = 10          # x variable globale
# Boucle while dans l'environnement
while (x > 2):
    x = x-1
# Définition d'une fonction fonct()
def fonct():
    print(x)    # on peut accéder à la valeur de x
    x = x + 1  # on ne peut pas modifier x => ERREUR
```

- Une variable globale est accessible par une fonction définie dans le même environnement, mais elle est non modifiable par cette fonction :

```
>>> fonct()    # Appel de fonct()
2
UnboundLocalError: local variable 'x' referenced before
assignment
```

- Si dans une fonction on crée une variable (locale) ayant même nom qu'une variable globale déjà définie, il s'agit d'une nouvelle variable qui est créée :

```
x = 10          # variable globale x
print(id(x))    # id() retourne l'identifiant de la variable x
def fonct():
    x = 1        # Variable locale créée
    print(id(x))
    x = x + 1    # Modifiable
    print(x)
```

Au lancement du programme :

```
4298180768 # Identifiant id(x) de x : variable globale
>>> fonct() # Appel de la fonction
4298180984 # identifiant id(x) de x : variable locale de même
nom
2        # La valeur de x retournée est celle de la variable locale
```

- Eviter de donner un même nom à une variable locale et une variable globale !

5.2.3 Création d'une variable globale dans une fonction

- Pour créer ou pour modifier une variable globale à l'intérieur d'une fonction, utiliser l'instruction `global` :

```
x = 10          # Variable globale : définie dans
l'environnement
def fonct():
    global x    # Déclaration de la variable globale x
    x = x + 1   # Elle devient modifiable
    global y    # Déclaration d'une nouvelle variable globale y
    y = 2
```

```
>>> print(x)
10
>>> fonct()
>>> print(x)
11
>>> print(y)
2
```

- On évitera tant que possible d'utiliser l'instruction `global`.