Portée des variables avec Python

TS spé ISN

lycée Les Eaux Claires

2015 - 2016

Rappel

Qu'est ce qu'une variable ?

Rappel

Qu'est ce qu'une variable ?

- Un nom
- Une adresse mémoire
- Une valeur (une suite d'octets...)

Espace de noms

- Un espace de noms fonctionne comme un index à la fin d'un livre
- Cet espace contient la liste de toutes les variables utilisées au cours du programme (cette liste est enrichie au cours de l'exécution du programme). A chaque variable référencée est associée une adresse mémoire (comme un numéro de page pour un livre) : on dit qu'une variable pointe vers une zone de la mémoire.
- La valeur de la variable est enregistrée en mémoire à l'adresse pointée.

Illustration

Espace de noms

Nom	Adresse
a	145785
toto	78541
maliste	817332
image_17	37154

Mémoire

Adresse	
37154	011100011 ...
78541	'Salut'
145785	19
817332	[4,11,25,3]

Remarques

- L'adresse mémoire d'une variable est donnée par la fonction id()
- Lorsqu'une affectation a lieu pendant l'exécution d'un programme :
  - l'adresse mémoire pointée par une variable est modifiée dans l'espace de noms
  - ou le contenu de l'espace mémoire pointé par la variable est modifié

Remarques

- Toute information en mémoire est codée sous forme binaire que l'on représente avec une suite de 0 et 1
- Le type d'une variable indique de quelle manière doit être interprétée la suite d'octets enregistrée en mémoire.

Programme 1

Script

```
1  def action():
2      a = 5
3      print('a = ', a)
4
5  a = 2
6  action()
7  print('a = ', a)
```

Quel est l'affichage obtenu ?

Programme 1

- Dans l'exemple ci-dessus, une première variable a est créée de valeur 2
- Lorsqu'une fonction est exécutée, un espace de noms (temporaire) propre à la fonction est construit.
- Lorsque la fonction action() est appelée, une seconde variable a est créée (interne à la fonction), avec la valeur 5
  Ces deux variables coexistent dans deux espaces de noms différents.
- Lorsque l'exécution de la fonction est terminée, l'espace de noms associé à cette fonction est détruit et les variables inscrites dans cet espace disparaissent.

Programme 2

Script

```
1  def action(a):
2      print('a = ', a)
3
4  z = 2
5  action(z)
```

Quel est l'affichage obtenu ?

Programme 2

- La fonction action() attend un paramètre.
- Lorsque la fonction est exécutée, la variable a est créée (espace de noms de la fonction). Sa valeur est égale au paramètre transmis lors de l'appel de la fonction, ici la valeur pointée par la variable z

Programme 3

Script

```
1  def action():
2      print('a = ', a)
3
4  a = 2
5  action( )
```

Quel est l'affichage obtenu ?

Programme 3

- La fonction action() n'attend pas de paramètre.
- Pendant l'exécution de la fonction, la variable a n'existe pas dans l'espace de noms de la fonction.
- Pour afficher sa valeur, on va chercher dans l'espace de noms principal qui est accessible en lecture.

Programme 4

Script

```
1  def action(a):
2      print('a = ', a)
3      a = a + 1
4      return a
5
6  z = 2
7  z = action(z)
8  print('z = ',z)
```

Quel est l'affichage obtenu ?

Programme 4

- Ce programme complète le programme 2
- Lorsque la fonction est exécutée, la variable a est créée (espace de noms de la fonction). Sa valeur est ensuite modifiée.
- L'instruction return permet de ne pas perdre la valeur calculée à l'intérieur de la fonction lorsque l'espace de noms de la fonction est détruit
- La ligne 7 permet d'affecter la variable z avec la valeur renvoyée par la fonction

Programme 5

Script

```
1  def action( ):
2      print('a = ', a)
3      a = a + 1
4
5  a = 2
6  action()
7  print('a = ', a)
```

Quel est l'affichage obtenu ?

Programme 5

- Ce programme complète le programme 3
- Pendant l'exécution de la fonction, la variable a n'existe pas dans l'espace de noms de la fonction.
- On peut aller chercher dans l'espace de noms principal la valeur de la variable a qui est accessible en lecture.
- Par contre, cette variable a n'est pas accessible en écriture : il n'est pas possible de modifier sa valeur 'depuis' la fonction
- Un message d'erreur apparait lors de l'exécution du programme.

Programme 6

Script

```
1  def action():
2      print(maliste)
3      maliste[0] = 7
4
5  maliste = [3,12]
6  action()
7  print(maliste)
```

Quel est l'affichage obtenu ?

Programme 6

- Pendant l'exécution de la fonction, la variable maliste n'existe pas dans l'espace de noms de la fonction.
- Contriramment à la variable a dans le programme précédent, la variable maliste est ici accessible en lecture. et en écriture. Il est possible de modifier sa valeur 'depuis' la fonction.
- On dire que cette variable a une portée globale

Une définition

On appelle portée d'une variable la partie du programme dans laquelle cette variable est accessible et modifiable.

A retenir

- Lors de l'exécution d'une fonction, les variables de l'espace de noms principal ne sont pas accessibles (uniquement en lecture)
- Cependant, les variables de type list ont une portée globale : elles sont utilisables et modifiables depuis une fonction.
- De même, les objets du module tkinter ont une portée globale.
- On peut étendre la portée d'une variable avec l'instruction global. Mais il est recommandé de ne pas utiliser cette instruction !