

Affectation et type des variables



Les variables sont indispensables en programmation. Elles correspondent à des cases mémoires auxquelles on peut affecter des valeurs, chiffres, lettres, mots, ..., des données de toute sorte.

1. Coder les instructions suivantes qui définissent la variable **n** et lui affectent la valeur **7**.

```
n = 7  
print(n)
```

2. Essayer :

```
msg = "Quoi de neuf ?"  
pi = 3.14159  
print(msg)  
print(pi)
```

3. Taper et visualiser les lignes précédentes avec Thonny. Que peut-on observer ?

.....
.....

Le signe = en programmation signifie affectation et non égalité, le nom de variable doit être à gauche.

4. Essayer **7 = n**.

On peut changer la donnée d'une variable.

5. Essayer :

```
msg = "Quoi de neuf ?"  
print(msg)  
  
msg = "Ca roule"  
print(msg)
```

Dans la plupart des langages de programmation il est nécessaire de déclarer dans le programme le type de la variable et sa taille, par exemple un nombre entier, un nombre réel, un caractère, ... Sous Python le typage est automatique. On dit qu'il est dynamique.

6. L'instruction ci-dessous définit **a** comme un nombre entier. Qu'est-il affiché ?

```
a = 5  
print(type(a))
```

.....
.....

7. L'instruction suivante définit **b** comme un nombre réel, dénommé à virgule flottante.

```
b = 5.0
```

puis essayer : **c = '5'**

```
print(type(b))  
  
print(type(c))
```

.....
.....

8. Voici comment incrémenter une variable. Noter que la syntaxe juste en programmation serait complètement fautive en mathématiques.

```
a = a + 1 # illegal en Mathématiques car on a écrit 0 = 1  
print(a)  
a += 1 # astuce pratique  
print(a)  
a += 8  
print(a)
```

Quelles sont les valeurs prises par **a** ?

.....
.....

Le **#** en Python permet de mettre un commentaire qui ne sera pas pris en compte par le compilateur. Il est important de bien commenter un programme, pour s'y retrouver, ou que les autres développeurs comprennent rapidement comment fonctionne le programme.

9. Les affectations précédentes sont des affectations simples qui affectent une seule variable à la fois. On peut affecter plusieurs variables avec la même donnée dans des affectations multiples.

```
x = y = 7  
print(x)  
print(y)
```

10. On peut affecter plusieurs variables avec des données différentes dans des affectations parallèles. On utilise la virgule comme séparateur.

```
a, b = 4, 8.33  
print(a)  
print(b)
```

11. Taper la ligne d'instructions ci-dessous et ajouter une ligne qui permet d'échanger les contenus des variables a et c (sans écrire a = 3 et c = 1).

```
a, b, c, d = 1, 2, 3, 4
```

.....

12. Les noms des variables suivent des règles précises. Essayer les instructions suivantes, chacune leur tour.

```
longueur3 = 5
```

.....

```
3longueur = 5
```

.....

```
devise€ = 120
```

.....

```
devise en euros = 120
```

.....

```
devise_en_euros = 120
```

.....

```
hauteur = 5  
Hauteur = 10  
print(hauteur)
```

.....

```
print(Hauteur)
```

.....

Au fur et à mesure du développement des langages de programmation, les programmeurs se sont rendu compte qu'il fallait se conformer le plus possible à un style de codage. Pour python, vous le trouverez sur le lien suivant.

<http://docs.python.org/py3k/tutorial/controlflow.html#intermezzo-coding-style>

13. On peut utiliser n'importe quel mot à part ceux déjà utilisés par le langage. Essayer dans l'interpréteur :

```
print('coucou')  
print = 'toto'
```

```
print  
print('KC')
```

Que s'est-il passé et pourquoi ?

.....
.....
Les mots réservés peuvent être trouvés sur :

http://docs.python.org/py3k/reference/lexical_analysis.html#keywords

<http://docs.python.org/py3k/library/functions.html>

<http://docs.python.org/py3k/library/constants.html>

14. Enfin voici les types de variables les plus classiques à essayer dans l'interpréteur.

```
type(True)  
type(False)  
type(false)  
type(32)  
type(32.0)  
type(32.)  
type('32')  
type('coucou') # Le type string se déclare avec ' ou " ou ''' ou ''''  
type("coucou") # selon que l'on veuille utiliser ' ou " dans la chaîne de caractères  
type("coucou") # de caractères elle-même.  
type("""coucou""")  
type(''coucou'')  
type(''''coucou''')
```

15. On peut convertir les types entre eux.

```
float('32')  
bool(1)  
int('32')  
bool(0.5)  
str(32)  
bool(0)  
str(32.)  
bool('')  
float(32)  
bool('toto')  
int(32.45)  
float(True)  
float(False)
```