PROGRAMMATION PYTHON

Chapitre 2 : Les Variables

Sommaire

Définition

Parfois dans un programme on a besoin d'utiliser à plusieurs reprises une donnée pour cela on stockera la donnée dans une sorte de boite à laquelle on associe un nom. Cette case est appelée variable:

- Qui est une donnée de programme stockée dans l'ordinateur.
- Qui pourra être utilisée à plusieurs reprises.
- Qui est identifiée par une étiquette qui est son nom (choisir un nom compréhensible).
- Qui peut contenir une valeur.

Certains langages comme le langage C impose qu'une variable soit créée avec un type précis qui spécifie le format de valeur que peut contenir la variable. Python de base n'impose pas l'association de type à une variable il fait partie des **langages non-typés**.
Avec Python une variable de nom **age** peut contenir une valeur comme 12 et après contenir une valeur comme « Christophe » ce qui causera une erreur dans les **langages typés**.

Création de variable

Pour créer une variable en Python, il suffit de respecter le format suivant:

nom_de_la_variable = valeur

```
Python 3.9.2 (v3.9.2:1a79785e3e, Feb 19 2021, 09:06:10)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> age = 12
>>> age
12
>>> age="Christophe"
>>> age
'Christophe'
>>>
```

Le nom d'une variable doit respecter les règles suivantes:

- Ne doit pas contenir d'espace doit être en un mot (uniquement de lettres majuscules, minuscules, chiffres et symbole underscore _).
- Ne doit pas commencer par un chiffre.

Le langage Python est sensible à la casse il différencie les majuscules des minuscules la variable **age** sera considérée différente de **AGE…**

Convention de notation

En programmation il existe plusieurs conventions de nommage des variables parmi lesquelles on a:

Le camel case: est une notation qui consiste à écrire un ensemble de mots en les liant sans espaces ni ponctuation mais en mettant en lettre majuscule la première lettre de chaque mot. Le nom d'une variable en camel case sera par exemple ***nomUtilisateur*** pour désigner une variable contenant le nom d'utilisateur.

Le snake case: est une notation qui consiste à écrire un ensemble de mots en les liant par un tiret bas « _ ». Le nom d'une variable en snake case sera par exemple ***nom_utilisateur***.

Le nom d'une variable doit être compréhensible éviter des noms comme « **a** » mais « **age** » pour désigner une variable qui va contenir un âge.

Dans la suite du cours nous utiliserons comme convention la notation Camel Case mais libre à chaque programmeur d'utiliser sa convention tout en restant cohérent.

Mots réservés

Certains mots ne peuvent être utilisés comme nom de variable car réservés par le langage python:

and	del	from	None	True
as	elif	global	nonlocal	try
assert	else	if	not	while
break	except	import	or	with
class	False	in	pass	yield
continue	finally	is	raise	
def	for	lambda	return	

Source: https://python.developpez.com/cours/apprendre-python-3/?page=la-calculatrice-python

L'utilisation de l'un de ces mots comme nom de variable entraine une erreur:

```
Python 3.9.2 (v3.9.2:1a79785e3e, Feb 19 2021, 09:06:10)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> and=2
SyntaxError: invalid syntax
>>> del=3
SyntaxError: invalid syntax
>>> from=3
SyntaxError: invalid syntax
>>>
```

Type de données

Python a besoin de savoir quel est le type de valeur de variable pour savoir quelle opération faire. La somme de deux variables contenant des nombres sera différente de la somme de deux variables contenant des lettres.

```
Python 3.9.2 (v3.9.2:1a79785e3e, Feb 19 2021, 09:06:10)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> nombre1 = 1
>>> nombre2 = 2
>>> resultat= nombre1 + nombre2
>>> resultat
3
>>> debutNom = "Chris"
>>> finNom = "tophe"
>>> resultat = debutNom + finNom
>>> resultat
'Christophe'
>>>
```

En Python, une variable peut contenir toute sorte de valeur qui est associée à un type bien précis.

Les types sont classés en deux catégories les types simples (primitifs) et les types complexes (que nous ne traiterons pas pour le moment).

Types usuels

En dehors des types complexes qui seront traités individuellement les principaux types usuels qu'on rencontrera sont:

- Les nombres entiers nommé ***int*** en Python pour ***integer*** exemple 12.
- Les nombres flottants (à virgule) nommé ***float*** en Python pour ***flottant*** exemple 12.5 (la virgule en Python est caractérisée par un « . » notation anglaise)
- Les types booléens (vrai/faux) nommé ***bool*** en Python pour ***boolean*** constitué de deux valeurs possible ***True/False.***
- Les chaines de caractères nommé ***str*** en Python pour ***string*** exemple « Toto » (en Python une chaine de caractère est délimitée par des guillemets ou apostrophes).

```
>>> nbre = 12
>>> nbre
12
>>> nbre2 = 12.5
>>> nbre2
12.5
>>> estSuperieur = 12 > 3
>>> estSuperieur
True
>>> estInferieur = 12 < 3
>>> estInferieur
False
>>> nom="Christophe"
>>> nom
'Christophe'
>>>
```

Fonctions *type* et *print*

Python dispose d'une fonction interne qui permet d'afficher le type d'une variable avec pour syntaxe:

type(nom_de_la_variable)

```
>>> variableMulti = 12
>>> type(variableMulti)
<class 'int'>
>>> variableMulti = 12.5
>>> type(variableMulti)
<class 'float'>
>>> variableMulti = True
>>> type(variableMulti)
<class 'bool'>
>>> variableMulti = "toto"
>>> type(variableMulti)
<class 'str'>
>>>
```

Pour afficher le contenu d'une variable on utilisera la fonction interne de Python:

print(nom_de_la_variable)

```
>>> nbre = 12
>>> print(nbre)
12
>>> chaine="toto"
>>> print(chaine)
toto
>>> print(chaine, nbre)
toto 12
>>>
```

Echappement

Les guillemets et l'apostrophe en Python sont considérés comme des caractères spéciaux pour encadrer une chaine de caractères. Pour afficher le caractère apostrophe ou les guillemets on utilisera l'échappement qui consiste à précéder le caractère spécial d'un anti-slash « \ ». Python traitera le guillemet ou l'apostrophe qui suit comme un caractère normal.

```
>>> action = 'J'allume'
SyntaxError: invalid syntax
>>> action = 'J\'allume'
>>> action
"J'allume"
>>> action = "J'allume"
>>> action
"J'allume"
```

```
>>> action = "J"allume"
SyntaxError: invalid syntax
>>> action = "J\"allume"
>>> action
'J"allume'
>>> action = 'J\"allume'
>>> action
'J"allume'
>>>
```

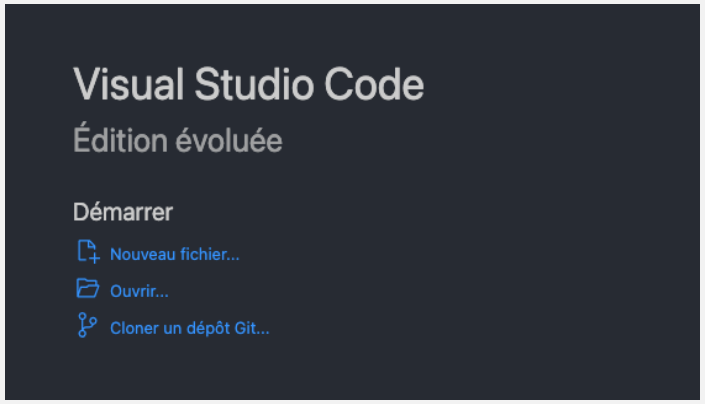
Certains caractères attachés à l'anti-slash ont une signification spécifique pour Python comme \n (pour passer à la ligne) et \t (pour mettre une tabulation).

```
>>> print("Ligne1\nLigne2")
Ligne1
Ligne2
>>> print("Ligne1\tLigne2")
Ligne1  Ligne2
>>> action = 'J\allume'
```

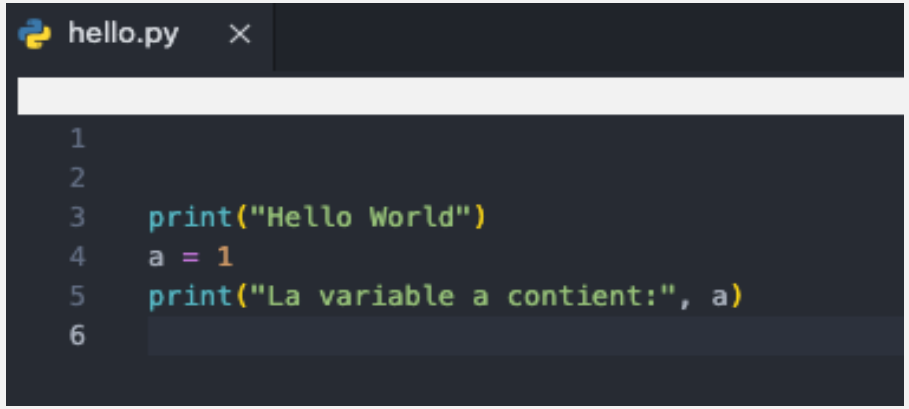
Utilisation d'un éditeur de code

Pour les projets nous utiliserons un éditeur de code au lieu de Idle. Pour écrire et exécuter un code python avec un éditeur il faut suivre les étapes suivantes:

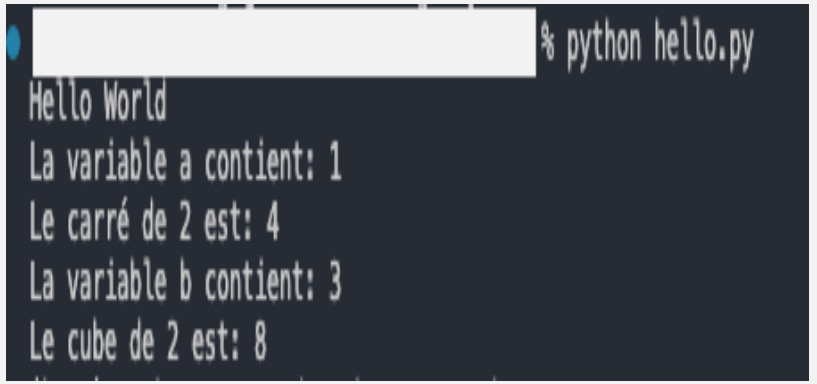
1. Ouvrir ou créer un nouveau fichier **.py** avec l'éditeur dans un dossier de notre choix
2. Ecrire le code
3. Exécuter le code avec un terminal avec la commande ***python nom_script.py***



Etape 1: Ouverture de VSCode



Etape 2: Création du fichier hello.py



Etape 3: Exécution du fichier hello.py

On peut aussi utiliser Idle pour créer des fichiers de script Python il suffit de cliquer sur File > New File pour créer un nouveau fichier de script.

Exercices

Ecrire les scripts Python des algorithmes suivants et donner la réponse:

Exercice 1

Quelles seront les valeurs des variables A et B après exécution des instructions suivantes ?

```
Début
    Variables A, B en Entier
    A <- 1
    B <- A + 3
    A <- 3
Fin
```

Exercice 2

Quelles seront les valeurs des variables A, B et C après exécution des instructions suivantes ?

```
Début
    Variables A, B, C en Entier
    A <- 5
    B <- 3
    C <- A + B
    A <- 2
    C <- B - A
Fin
```

Exercices

Exercice 3

Quelles seront les valeurs des variables A et B après exécution des instructions suivantes ?

```
Début
    Variables A, B en Entier
    A <- 5
    B <- A + 4
    A <- A + 1
    B <- A - 4
Fin
```

Exercice 4

Quelles seront les valeurs des variables A, B et C après exécution des instructions suivantes ?

```
Début
    Variables A, B, C en Entier
    A <- 3
    B <- 10
    C <- A + B
    B <- A + B
    A <- C
Fin
```

Exercices

Exercice 5

Quelles seront les valeurs des variables A et B après exécution des instructions suivantes ?

```
Début
    Variables A, B en Entier
    A <- 5
    B <- 2
    A <- B
    B <- A
Fin
```

Les deux dernières instructions permettent-elles d'échanger les deux valeurs de B et A ? Si l'on inverse les deux dernières instructions, cela change-t-il quelque chose ?

Exercice 6

Ecrire un algorithme permettant d'échanger les valeurs de deux variables A et B, et ce quelque soit leur contenu préalable.

Exercices

Exercice 7

Soit trois variables A, B et C de valeurs respectives 12, 13, 3.
Ecrire un algorithme qui transfère à B la valeur de A, à C la valeur de B et à A la valeur de C.
Tester l'algorithme avec différentes valeurs de A, B, et C.

FIN CHAPITRE 2