Les variables

Principe

Une variable est une zone de la mémoire de l'ordinateur dans laquelle une valeur est stockée. Aux yeux du programmeur, cette variable est définie par un nom, alors que pour l'ordinateur, il s'agit en fait d'une adresse, c'est-à-dire d'une zone particulière de la mémoire.

En Python, la déclaration d'une variable et son initialisation se font *en même temps*

```
a = 5
```

Nous avons crée une variable nommée "a"
en lui attribuant une valeur "5"

Cours réalisé par le professeur Khalid +212 622176552

Nommer une variable

Les noms de variables peuvent avoir des lettres, en minuscule ou majuscule et des chiffres et ne doivent pas commencer par un chiffre ni porter un nom réservé par Python

Types de données

Les trois types principaux sont les entiers (integer ou int), les réels (float) et les chaînes de caractères (string ou str).

Python reconnaît certains types de variable automatiquement (entier, réel).
Par contre, pour une chaîne de caractères, il faut l'entourer de guillemets (*simples, doubles voire trois guillemets successifs simples ou doubles*) afin d'indiquer à Python le début et la fin de la chaîne.


```
>>> y = 3.14
>>> y
3.14
>>> a = "bonjour"
>>> a
'bonjour'
>>> b = 'salut'
>>> b
'salut'
>>> c = '''girafe'''
>>> c
'girafe'
```

Opérations sur les nombres

```
>>> x = 45
>>> x + 2
47
>>> y = 2.5
```

```
>>> x + y
47.5
>>> (x * 10) / y
180.0
```

```
>>> 2**3
8
>>> 5 % 4
1
>>> 8 % 4
0
```

important

⇒ l'utilisation de parenthèses permet de gérer les priorités
⇒ ** pour la puissance
⇒ % pour obtenir le reste de la division entière

Opérations sur les chaines de caractères

```
>>> chaine = "Salut"
>>> chaine
'Salut'
>>> chaine + " Python"
'Salut Python'
>>> chaine * 3
'SalutSalutSalut'
```

L'opérateur d'addition + permet de concaténer (assembler) deux chaînes de caractères et l'opérateur * permet de dupliquer plusieurs fois une chaîne.

Quelques méthodes sur les variables

La fonction type()

Si vous ne vous souvenez plus du type d'une variable, utilisez la fonction type() qui vous le rappellera

```
>>> x = 2
>>> type(x)
<class 'int'>
>>> y = 2.0
>>> type(y)
<class 'float'>
>>> z = '2'
>>> type(z)
<class 'str'>
```

Conversion de types

```
>>> i = 3
>>> str(i)
'3'
>>> i = '456'
>>> int(i)
456
>>> float(i)
456.0
>>> i = '3.1416'
>>> float(i)
3.1416
```

Les opérateurs mathématiques

```
x + y   # Addition
x - y   # Soustraction
x * y   # Multiplication
x / y   # Division
x // y  # Division entière
x % y   # Modulo (reste)
-x      # Opposé - opérateur unaire
abs(x)  # Valeur absolue - opérateur unaire
x ** y  # Puissance
```

Les opérateurs de comparaison

```
x < y        # Inférieur
x <= y       # Inférieur ou égal
x > y        # Supérieur
x >= y       # Supérieur ou égal
x == y       # Égal (attention !)
x != y       # Différent
x <> y       # Différent
x is y       # Identité
x is not y   # Non identité
```

Les opérateurs logique

```
x and y # Intersection
x or y  # Union
not y   # Négation
```

Autres méthodes pour chaines de caractères

Chaine.count(sub,*start,end*)

Retourne le nombre d'occurrences de la chaîne de caractères sub, dans la chaine "chaine" les paramètres par défaut start et end permettent de réduire la recherche entre les caractères d'indice start et end exclu.

Par défaut start est nul tandis que end correspond à la fin de la chaîne de caractères

```
>>> x="zzzaa"
>>> x.count('z')
3
```

```
x="azzzzz"
y= x.count("z",1,5)
print (y)
4
```

Chaine.find(sub,start,end)

Retourne la 1ere occurrence d'une chaîne de caractères sub recherchée dans "chaine", les paramètres par défaut start et end ont la même signification que ceux de la fonction count. Cette fonction retourne -1 si la recherche n'a pas abouti.

```
x="salut je suis un texte"
y= x.find("x")
print (y)
19
```

```
x="salut je suis un texte"
y= x.find("x",10,50)
print (y)
19
```

maj=chaine.upper()

Remplace les minuscules par les majuscules

Min=chaine.lower()

Remplace les majuscules par les miniscules

```
maj="chaine".upper()
print(maj)
CHAINE
```

```
min="cHaiNe".lower()
print(min)
chaine
```

Chaines de caractères et listes

Les chaînes de caractères peuvent être considérées comme des listes [avec des crochets] (un peu particulières).

```
liste =['chien', 'chat', 'souris']
print (liste)
['chien', 'chat', 'souris']
chaine =('chien chat souris')
print (chaine )
chien chat souris
print (liste[0])
chien
print(chaine[0])
c
print (len(chaine))
17
print (len(liste))
3
```

Cours réalisé par le professeur Khalid +212 622176552

les chaînes de caractères présentent toutefois une différence notable, ce sont des listes non modifiables. Une fois définie, vous ne pouvez plus modifier un de ses éléments. Le cas échéant, Python renvoie un message d'erreur

```
>>> animaux = "girafe tigre"
>>> animaux[4]
'f'
>>> animaux[4] = "F"
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
```

Les caractères spéciaux

Il existe certains caractères spéciaux comme **\n** que nous avons déjà vu (pour le retour à la ligne). Le caractère **\t** vous permet d'écrire une tabulation. Si vous voulez écrire un guillemet simple ou double (et que celui-ci ne soit pas confondus avec les guillemets de déclaration de la chaîne de caractères), vous pouvez utiliser **\' ou \"** ou utiliser respectivement des guillemets doubles ou simple pour déclarer votre chaîne de caractères.

```
>>> print("Un retour à la ligne\npuis une tabulation\t, puis un guillemet\"")

Un retour à la ligne
puis une tabulation	, puis un guillemet"
>>> print('J\'affiche un guillemet simple')

J'affiche un guillemet simple
>>> print("Un brin d'ADN")

Un brin d'ADN
>>> print('Python est un "super" langage de programmation')

Python est un "super" langage de programmation
```

La fonctions print

la fonction print() affiche l'argument qu'on lui passe entre parenthèses et un retour à ligne

```
>>> print("Hello world!")
Hello world!
```

La fonctions input

la fonction input() provoque une interruption dans le programme courant. L'utilisateur est invité à entrer des caractères au clavier et à terminer avec <Enter>. Lorsque cette touche est enfoncée, l'exécution du programme se poursuit, et la fonction fournit en retour une valeur correspondant à ce que l'utilisateur a entré.

```
print ('Veuillez entrer votre message : ')
nn = input()
print ("vous avez dit", nn)
```

Veuillez entrer votre message :
bonjour
vous avez dit bonjour

Application

Questionnaire

1- Définissez une variable? Une donnée?
2- Quels sont les 3 types principaux de données ?
3- Comment concaténer deux chaines de caractères?
4- Expliquez le rôle de chacune des méthodes:
   a. Str()
   b. Int()
   c. Float()

5- Donnez la signification des opérateurs et méthodes suivants:

/ // % ** abs() < >= <= == <> and or

\n \" \t print() input()

Chaine.count(sub,*start*,*end*) Chaine.find(sub,start,end)

maj=chaine.upper() Min=chaine.lower()

Application

Questionnaire

6- Comment définir une liste en python?

7- Donnez la signification des méthodes relatives aux listes suivantes:

   a. liste[0]
   b. len(liste)
   c. print(liste)

Application

Travaux pratiques

1. écrire un programme qui définit 3 variables de différents types et affiche leurs types avec "print"
2. Écrire un programme, qui donne le reste "***reste***" de la division entre deux variables "***dividende***" et "***diviseur***" que l'utilisateur rentre via la fonction "input", Afficher la solution sous forme...

   le résultat de la division entre ..... et ..... vaut .....

3. Écrire un programme, qui récupère le prénom de l'utilisateur et renvoie les sorties suivantes:

Votre prénom est ..... (retour à la ligne) Il contient .....caractères

(solution page suivante)

Application

```
x=12

y=1.3

st="bonjour"

print(type(x), type(y), type(st))
```

```
x=input("rentrez le diviseur")

y=input("rentrez le dividende")

z=int(y)%int(x)

print("le résultat de la division entre  " + y +" et " + x + " est " + str(z) )
```

Application

```
pr=input("votre prénom")

x=len(pr)

print("votre prénom est ",pr, '\n',"il contient ",str(x)," caractères")
```