Chapitre 1 : Variables, types, opérateurs

1 Type int (integer) : nombres entiers

Pour **affecter** (on dit aussi assigner) la valeur 17 à la variable nommée Age :

```
>>> Age = 17
```

La fonction `print()` affiche la valeur de la variable :

```
>>> print(Age)
17
```

La fonction `type()` retourne le type de la variable, ici, c'est une entier :

```
>>> print(type(Age))
<class 'int'>
```

Le signe = n'a pas la même signification qu'en mathématique !

```
>>> Age = Age + 1      # en plus court : Age += 1
>>> print(Age)
18
```

L'opérateur `//` donne la **division entière :**

```
>>> d = 450//360
>>> print(d)
1
```

L'opérateur `%` donne le **reste de la division (opération modulo) :**

```
>>> reste = 450 % 360
>>> print(reste)
90
```

L'opérateur `**` donne la **puissance** :

```
>>> Mo = 2**20
>>> print(Mo)
1048576
```

2 Type float : nombres à virgule flottante

```
>>> b = 17.0  # le séparateur décimal est un point
>>> print(b)
17.0
>>> print(type(b))
<class 'float'>

>>> c = 14.0/3.0
>>> print(c)
4.66666666667
```

Les fonctions mathématiques

Pour utiliser les fonctions mathématiques, il faut commencer par importer le module `lycee`:

```
>>> from lycee import *
```

les fonctions que nous utiliserons le plus seront :
sqrt() pour racine carrée (square root), **cos**, **sin**, **tan**, **pi...**

```
>>> print(pi)
3.14159265359
>>> print(sin(pi/4.0))
0.707106781187
>>> print(sqrt(2.0))
1.41421356237
```

3- Le type `str` (string : chaîne de caractères)

```
>>> Nom = 'Dupont'          # entre apostrophes
>>> Prenom = "Pierre"       # ou guillemets
>>> print(Nom,Prenom)       # attention la virgule
Dupont Pierre
```

La **concaténation** désigne la mise bout à bout de plusieurs chaînes de caractères, on utilise l'opérateur **+**

```
>>> print(Nom + Prenom)
DupontPierre
>>> print(Prenom + ' ' + Nom+ ' 18 ans')
Pierre Dupont 18 ans
```

La fonction `len()` retourne la **longueur** (length) de la chaîne de caractères :

```
>>> print(len(Nom))
6

>>> print(chaine[0])        # premier caractère (indice 0)
P
>>> print(chaine[1]) # deuxième caractère (indice 1) i
```

4- Interaction avec l'utilisateur :

Demande d'un texte :

```
>>> nom=input('Ecrivez votre nom')
```

Demande d'un nombre entier:

```
>>> age=int(input('Donnez votre âge'))
```

Demande d'un nombre flottant (réel):

```
>>> taille=float(input('Donnez votre taille en mètre'))
```

5- Le type `list` (liste)

Une liste est une structure de données.
Le premier élément d'une liste possède l'indice (l'index) 0.
Dans une liste, on peut avoir des éléments de plusieurs types.

```
>>> infoperso = ['Pierre', 'Dupont', 17, 1.75, 72.5]
>>> # la liste infoperso contient 5 éléments de types str, str, int, float et float
>>> print type(infoperso)
<type 'list'>
>>> print infoperso
['Pierre', 'Dupont', 17, 1.75, 72.5]
>>> print 'Prénom : ', infoperso[0]         # premier élément (indice 0)
Prénom :  Pierre
>>> print 'Age : ', infoperso[2]      # le troisième élément a l'indice 2
Age :  17
>>> print 'Taille : ', infoperso[3] # le quatrième élément a l'indice 3
Taille :  1.75
```

La fonction `range()` crée une liste d'entiers régulièrement espacés :

```
>>> maliste = range(10)
>>> print maliste
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> print type(maliste)
<type 'list'>
>>> maliste = range(1,10,2)  # range(début,fin non comprise,intervalle)
>>> print maliste
[1, 3, 5, 7, 9]
>>> print maliste[2]         # le troisième élément a l'indice 2
5
```

On peut créer une liste de listes, qui s'apparente à un tableau à 2 dimensions (ligne, colonne) :

```
 0  1  2
10 11 12
20 21 22
```

```
>>> maliste = [[0, 1, 2], [10, 11, 12], [20, 21, 22]]
>>> print maliste[0]
[0, 1, 2]
>>> print maliste[0][0]
```

```
0
>>> print maliste[2][1]                # élément à la troisième
ligne et deuxième colonne
21
>>> maliste[2][1] = 69                 # nouvelle affectation
>>> print maliste
[[0, 1, 2], [10, 11, 12], [20, 69, 22]]
```

6- Le type bool (booléen)

Deux valeurs sont possibles : `True` et `False`

```
>>> a = True
```

Les opérateurs de comparaison :

Opérateur	Signification	Remarques
<	strictement inférieur	
<=	inférieur ou égal	
>	strictement supérieur	
>=	supérieur ou égal	
==	égal	Attention : deux signes ==
!=	différent	
is	identique	Deux conditions : égal et même type
is not	non identique	

```
>>> b = 10
>>> print(b>8) True
>>> print(b==5) False
>>> print(b!=10) False
>>> print(0<= b <=20) True
```

Les opérateurs logiques suivans peuvent être utilisés : `and, or, not`

```
>>> print(0<=b and  b<=20) True
```

Operateur in :

L'opérateur `in` s'utilise avec des chaînes (type `str`) ou des listes (type `list`) :

```
>>> chaine = 'Bonsoir'
>>> # la sous-chaîne 'soir' fait-elle partie de la chaîne
'Bonsoir' ?
>>> resultat = 'soir' in chaine
>>> print resultat
True
>>> print 'b' in chaine
False
>>> maliste = [4, 8, 15]
>>> # le nombre entier 9 est-il dans la liste ?
>>> print 9 in maliste
False
>>> print 8 in maliste
True
>>> print 14 not in maliste
True
```

8- Programmation Orientée Objet (POO)

Python est un langage de programmation **orienté objet** (comme les langages C++, Java, PHP, Ruby...).
Une variable est en fait un **objet** d'une certaine **classe**.

Par exemple, la variable `amis` est un objet de la classe `list`.
On dit aussi que la variable `amis` est une **instance** de la classe `list`.
L'**instanciation** (action d'instancier) est la création d'un objet à partir d'une classe (syntaxe : **NouvelObjet = NomdelaClasse(arguments)**) :

```
>>> # instanciation de l'objet amis de la classe list
>>> amis = ['Nicolas', 'Julie']
>>> print type(amis)
<type 'list'>
```

Une classe possède des fonctions que l'on appelle **méthodes** et des données que l'on appelle **attributs**.

La méthode **`append()`** de la classe `list` ajoute un nouvel élément en fin de liste :

```
>>> # instanciation d'une liste vide
>>> amis = []
>>> amis.append('Nicolas')#synthase générale : objet.méthode(arguments)
>>> print amis
['Nicolas']
```

```
>>> amis.append('Julie') # ou bien : amis = amis + ['Julie']
>>> print amis
['Nicolas', 'Julie']
>>> amis.append('Pauline')
>>> print amis
['Nicolas', 'Julie', 'Pauline']
>>> amis.sort()                 # la méthode sort() trie les éléments
>>> print amis
['Julie', 'Nicolas', 'Pauline']
>>> amis.reverse()       # la méthode reverse() inverse la liste des
    éléments
>>> print amis
['Pauline', 'Nicolas', 'Julie']
```

La méthode `lower()` de la classe `str` retourne la chaîne de caractères en casse minuscule :

```
>>> # la variable chaine est une instance de la classe str
>>> chaine = "BONJOUR"          # ou bien : chaine = str("BONJOUR")
>>> chaine2 = chaine.lower()    # on applique la méthode lower() à
    l'objet chaine
>>> print chaine2
bonjour
>>> print chaine
BONJOUR
```

Exercices du chapitre 1

Niveau 1

Exercice 1 ★

Ecrire un programme qui demande le nom, le prénom et l'âge de l'utilisateur, et affiche les initiales suivies de l'âge (par exemple LM18 pour Léa Martin qui a 18 ans).

Exercice 2 ★

Ecrire un programme qui demande le rayon et la hauteur d'un cylindre, et affiche son volume

Exercice 3 ★

Écrire un programme qui invite l'utilisateur à entrer un nombre entier, puis qui affiche le carré de ce nombre:

```
>>>
n? 7
Le carré de 7 est 49
```

Exercice 4 ★

Écrire un programme qui invite l'utilisateur à entrer deux notes l'une après l'autre, puis qui affiche la moyenne de ces deux notes:

```
>>>
Note 1? 11
Note 2? 12
La moyenne de ces deux notes est 11.5
```

Exercice 5 ★

Ecrire un programme qui demande l'heure (heure, minutes, secondes) et affiche les secondes qui se sont écoulées depuis minuit

Exercice 6 ★

Ecrire un programme qui lit le prix Hors Taxe d'un article, le nombre d'articles et le taux de TVA, et qui fournit le prix total TTC correspondant. Faire en sorte que des libellés apparaissent clairement.

Exercice 7 ★

1. Créer une variable liste égale à `[1, 2, 7, 3, 'a']`
2. Remplacer le troisième élément (7) par 8 et afficher la liste
3. Ajouter l'élément `'toto'` à la fin de la liste, et l'afficher

Niveau 2

Exercice 8 ★★

Ecrire un programme qui demande le nom, le prénom et l'âge de l'utilisateur, et affiche les premières et dernières lettres du nom et prénom, suivies du nombre total de lettres que contiennent le nom et prénom (par exemple ***LaMn9*** pour Léa Martin qui a 18 ans).

Exercice 9 ★★

L'identifiant d'accès au réseau à l'école est construit de la manière suivante : initiale du prénom puis les 8 premiers caractères du nom (le tout en minuscule).
Exemple : Alexandre Lecouturier → alecoutur
A partir des deux variables prenom et nom, construire l'identifiant.

A propos de l'écriture algorithmique, ou "pseudo-code"

L'affectation se fait à l'aide du symbole ← :

A ← 18 # on stocke la valeur 18 dans la variable A

Voici un exemple d'algorithme en pseudo-code :

```
Variable A en Numérique
Début
A ← 34
A ← 12
Fin
```

Vous remarquerez que contrairement à Python, **il faut déclarer les variables nécessaires au début du programme**. Cela permet doit vous permettre de bien penser, au tout début de votre réflexion, à quelle variables vous allez avoir besoin.

Exercice 10 ★

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

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

Exercice 11 ★

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

```
Variables A, B, C en Entier
Début
```

```
A ← 5
B ← 3
C ← A + B
A ← 2
C ← B − A
Fin
```

Exercice 12 ★

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

```
Variables A, B en Entier
Début
A ← 5
B ← A + 4
A ← A + 1
B ← A − 4
Fin
```

Exercice 13 ★

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

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

Exercice 14 ★

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

```
Variables A, B en Entier
Début
A ← 5
B ← 2
A ← B
B ← A
Fin
```

Moralité : 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 ?

Exercices supplémentaires

Exercice 15 ★

Un classique absolu, qu'il faut absolument maitriser : écrire un programme permettant d'échanger les valeurs de deux variables A et B.

Exercice 16 ★★

Une variante du précédent : on dispose de trois variables A, B et C. Ecrivez un programme transférant à B la valeur de A, à C la valeur de B et à A la valeur de C.