# TD 0 Introduction à Python

**Lycée Louis Thuillier - Informatique - PCSIB -**

## 1 Opérations élémentaires sur les nombres

Avec $a$ et $b$ deux nombres :

♡ **Commande ♡. Opérations élémentaires**

- multiplication : $a * b$
- division : $a/b$
- puissance : $a**b$
- division euclidienne : quotient $a//b$ et reste $a\%b$

*Application 1 : Toutes les applications sont à réaliser avec l'ordinateur.*

- *Calculer :* $2 \times 3$, $\sqrt{2}$, $5^{1/3}$, $\left(\dfrac{2+\sqrt{6*4}}{1,4}\right)^{1.4}$.
- *Calculer le quotient et le reste de la division euclidienne de* 87 *par* 34
- *Convertir :* 369 *minutes en heures et minutes*

On fera **Attention !** aux parenthèses **parenthèses** Parenthèses PARENTHÈSES **PARENTHÈSES**!!!

## 2 Affectation

### 2.1 Affectation simple

**Définition. Affectation**

Lorsqu'on affecte une variable on relie deux grandeurs :

- un nom (il peut comporter des minuscules, des majuscules, des chiffres (jamais au début ! ), ainsi que _)
- un objet informatique (un nombre, une fonction, une chaîne, une liste, ...)

♡ **Commande ♡. Affectation**

L'affectation se réalise à partir de la commande " = ".

nom_variable = valeur à affecter

**Attention ! ce n'est pas la même chose que l'égalité en mathématique ! !**

Il faut plutôt voir cela comme "→"

*Exemple 1 :*

```
>>> heure=369//60
```

*j'ai créé une variable "heures" qui désigne le nombre d'heures dans 369 minutes*

*Exemple 2 :*

```
>>> a=2

>>> print(a)
2

>>> 2*a
4
```

▷ *avec la première ligne je décide que la variable a "pointe" désormais vers le nombre entier 2*

▷ *je peux demander d'afficher sa valeur, avec la commande print()*

▷ *je calcule $2 \times a$, Python va chercher dans sa mémoire vers quelle valeur a pointe. Ici $a \to 2$ donc*

$$2 \times a \to 2 \times 2 \text{ : il affiche } 4$$

**Attention ! L'ordre compte ! !**
C'est pour cela qu'il est utile d'avoir en tête que = en informatique c'est comme → : il y a un sens !

```
>>> 3=a
  File "<console>", line 1
SyntaxError: cannot assign to literal
```

Ici on cherche à affecter à la variable appelée 3 la valeur de $a$. Comme 3 n'est pas un nom valide, et que la variable $a$ n'existe pas c'est impossible.

*Application 2 : Que va afficher Python lors de l'exécution du programme suivant (évidemment, on n'utilisera pas l'ordinateur : on le fait à la main !)*
*On prendra le temps de préciser à chaque ligne ce que fait l'ordinateur.*

```
1 a=1
2 b=2
3 b=a
4 a=b
5 print(a,b)
```

**Attention !** A partir de maintenant on arrête d'écrire directement dans le terminal : on va écrire dans le fichier nos lignes de codes, les unes à la suite des autres. Python les exécutera toutes de haut en bas.

Deux intérêts : on peut sauvegarder notre programme (au format *.py*) et, si on fait une erreur, on peut la corriger sans avoir à tout retaper.

### ▶ Notion d'incrémentation

L'incrémentation consiste à réassigner une nouvelle valeur à une variable déjà existante en utilisant son ancienne valeur.

```
1 n=1
2 n=n+2
3 n=2*n
```

▷ *ligne 2* : dans la mémoire, $n \to 1$ donc $n+1$ vaut 2. On affecte à $n$ une nouvelle valeur 3

▷ *ligne 3 :* de la même façon, on réassigne à $n$ la valeur $2 \times 3$ soit 6.

*Application 3 : La variable $x$ pointe vers le nombre 2. Qu'obtiens-je si je tape 10 fois la commande*

1. $x = x + 1$
2. $x = 2 * x$
3. *(plus dur)* $x = x + x$
4. *(plus dur)* $x = x * x$

## 2.2 Nature d'une variable

Une variable (ou tout autre objet informatique) est sauvegardé dans le disque dur suivant un procédé **propre à la nature de cette variable**. On ne sauvegarde pas de la même façon un nombre réel, un entier naturel, un mot, ...
Il est utile d'avoir en tête la nature des variables qu'on manipule. On en distinguera, pour commencer, 3 types :

1. les nombre entier, appelé *int*
2. les nombres réels, appelé *float*
3. les chaines de caractère (*i.e.* un mot/phrase), appelé *str*

On force la nature d'une variable en utilisant le type associé :

```
1 x=2
2 y=int(2)
3 z=float(2)
```

- ▷ sans précision, Python considère 2 comme un nombre entier
- ▷ je force $y$ à pointer vers l'entier 2
- ▷ je force $z$ à pointer vers le réel 2

*Application 4 : Créer trois variables appelées nombre_entier, nombre_réel et mot qui pointent vers le chiffre 7 sous la forme d'un entier, d'un nombre réel et d'une chaîne de caractères.*

*Application 5 : Comment reconnaît-on visuellement le nombre entier 2 du réel 2 ?*

## 2.3 Affectations multiples

*Partie plus difficile, à travailler peut-être en deuxième approche.*

Il est possible en informatique de réaliser deux choses en même temps, notamment affecter des variables.

**♡ Commande ♡. Affectation multiple**
On peut affecter deux variables en même temps via la commande :

nom_variable_1 , nom_variable_2 = valeur_1 , valeur_2

*Application 6 : Que va afficher Python lors de l'exécution des deux programme suivants :*

```
1 a=1
2 b=2
3 a=b
4 b=a
5 print(a,b)
```

```
1 a=1
2 b=2
3 a,b=b,a
4 print(a,b)
```

*On prendra le temps d'expliquer la différences entre les deux.*

*Astuce* : très utile pour
- ▷ les suites comme $u_{n+1} = u_n + v_n$ et $v_{n+1} = u_n - v_n$ ou bien $u_{n+2} = u_{n+1} - u_n$
- ▷ les méthodes d'Euler d'ordre supérieur à 1

# 3 Dialoguer avec l'utilisateur

## 3.1 Poser une question : commande *input*

La commande *input*(...) permet de dialoguer avec l'utilisateur.

♡ **Commande ♡. input(...)**
La commande *input*(".....") affiche le message écrit entre les guillemets "....". L'utilisateur doit alors taper une réponse. La réponse peut être alors affecter à une variable.

nom_variable = input ( " Question " )

**Attention !** Suivant la nature de la réponse attendu, trois cas possibles :
1. la réponse est un mot :

nom_variable = input ( " Question " )

2. la réponse est un nombre entier :

nom_variable = int( input ( " Question " ) )

3. la réponse est un nombre réel :

nom_variable = float( input ( " Question " ) )

**Attention !** On fera **Attention !** aux parenthèses **parenthèses** Parenthèses PARENTHÈSES **PARENTHÈSES** !!!

**Attention !** La commande *input*(".....") toute seule ne sert à rien, il faut toujours l'affecter !

▷ input ( " Question " ) : poser une question sans écouter la réponse
▷ nom_variable = input ( " Question " ) : poser une question et enregistrer la réponse

> ***Application 7 :*** *Écrire un programme qui demande à l'utilisateur son année de naissance et affiche son âge.*

## 3.2 Afficher une réponse : commande *print*

Si vous taper les commandes :

```
1 x=2
2 z=4
3 x+z
```

Python n'affichera rien ... Pourquoi ? Car vous lui avez demander d'affecter les deux variables et de calculer leur somme. Vous ne lui avez pas demander d'**afficher** la réponse ...

♡ **Commande ♡. print(...)**
La commande *print*(...) permet d'afficher la valeur de la variable préciser entre parenthèse. On peut demander d'afficher plusieurs valeurs à la suite en les séparant par des virgules.
Il est possible également d'ajouter des bouts de phrases, encadrées par des guillemets "...."

```
1 prenom=input("Quel est votre prénom ? ")
2 print("Bonjour ",prenom,", j'espère que vous allez bien.'")
```