

## Comment fonctionnent les variables en python

Une variable sert à stocker une valeur à un endroit précis pour pouvoir la réutiliser ensuite. Pour ça on va lui donner un nom et la valeur que l'on veut lui donner.

Par exemple :

```
jour = 10
mois = "Décembre"
```

Ensuite dans le code on va pouvoir changer la valeur stocker dans ces variables de la même façon qu'on l'a fait plus haut.

Plusieurs opérations sont possible sur les valeurs de ces variables, voici une petite liste qui va vous être utile :

- '+' permet d'additionner deux valeurs
- '-' permet de soustraire deux valeurs
- '\*' permet de multiplier deux valeurs
- '/' permet de diviser deux valeurs

En jouant avec des parenthèses, on peut maîtriser la priorité des différentes opérations, comme lors du calcul normal...

## Comment fonctionnent les conditions en python

Pour que l'ordinateur puisse faire des choix, il existe plusieurs instructions qui vont permettre d'exécuter des morceaux de code en fonction d'une variable ou d'une expression.

Ici nous allons voir l'instruction la plus utilisée, if. Elle s'utilise avec une expression qui va décrire dans quel cas exécuter le code dans le corps ou quand passer à la suite. Voici un exemple avec une expression qui vérifie si la variable "mois" est égale à "Décembre" et si oui il affiche "Joyeux Noël !" dans la console :

```
if mois == "Décembre":
    print("Joyeux Noël !")
```

En plus de cette structure très simple on peut faire en sorte que du code soit exécuté si la condition donnée dans le if n'est pas vraie. Pour ça on utilise l'instruction else qui s'utilise un peu de la même façon mais sans condition.

```
if mois == "Décembre":
    print("Joyeux Noël !")
else:
    print("C'est pas Noël...")
```

Pour pouvoir faire des choix logiques, plusieurs opérateurs sont disponibles pour créer des expressions de condition, voici une petite liste qui va vous être utile :

- '==' (avec bien deux égal) permet de comparer deux valeurs sur leur égalité, si elles sont égal alors l'expression est vrai, sinon fausse

- ‘>’ permet de tester si la valeur de gauche est plus grande que la valeur de à droite
- ‘>=’ permet de tester si la valeur de gauche est plus grande ou égale à la valeur de droite
- ‘<’ permet de tester si la valeur de gauche est plus petite que la valeur de droite
- ‘<=’ permet de tester si la valeur de gauche est plus petite que la valeur de droite

On peut aussi ajouter avant la comparaison le mot clé “not” qui va inverser le résultat de celle-ci. Et finalement on peut combiner plusieurs comparaisons avec “and” qui est vrai seulement si les deux comparaisons le sont ou “or” qui est vrai seulement si les deux comparaisons le sont aussi.

## Comment fonctionnent les fonctions en python

Une fonction est un morceau de code réutilisable, cela permet d’effectuer plusieurs fois le même code depuis des endroits différents du code sans devoir répéter le même code. Pour ça on va lui donner un nom comme pour une variable et pour plus de flexibilité, on peut aussi passer des paramètres à une fonction pour qu’elle effectue des actions plus complexes.

Par exemple on va définir, avec le mot clé “def” la fonction bonjour qui va prendre en paramètre un nom et utiliser la fonction “print” pour l’afficher dans le terminal :

```
def bonjour(nom) :  
    print("Bonjour " + nom)
```

On peut voir ci-dessus que j’utilise la fonction print alors qu’elle n’est défini nul part et c’est parce qu’en fait elle fait partie du langage en lui-même. Une autre fonction qui vous sera utile comme ça c’est “input” qui permet de demander à l’utilisateur d’entrer du texte dans la console.