Définition

Les variables permettent de **stocker des données** pour les réutiliser plus tard.

```
ma_variable = 4
print(ma_variable + 3) # 7
```

> 📌 Référence : https://www.learnpython.org/en/Variables_and_Types

Cycle de vie d'une variable

Les grandes étapes du cycle de vie d'une variable en python sont les suivantes :

1. **Déclaration** se son nom. L'interpréteur python se charge alors de lui donner un type *(typage dynamique)*
2. **Affection** ou **Assignation** à une valeur. Son nom va pointer vers une zone mémoire qui stocke sa valeur. Celà permet aussi de modifier son contenu si elle était déjà affecté
3. **Lecture** de la variable
4. **Suppression** automatique gérer par python dès qu'elle n'est plus utlisé.

Il va supprimer toutes les variables que nous n'utilisons plus grâce au "garbage collector" ou "ramasse-miettes".

Cette facilité d'utilisation et ce travail réalisé dans l'ombre donne à python un confort de développement en contrepartie d'une perte en performance.

Les affectations de variable

Python proposent **plusieurs manières** d'affecter des valeurs à des variables. Ces méthodes sont pertinentes lorsqu'elle sont utilisées avec bon sens, en privilégiant toujours la lisibilité du code.

- Les affectations simples

  ```
  a = 2
  b = 3
  ```

Règles de nommages

Python n'est **pas entièrement permissif** sur le nommage des variables. Certains noms de variables vont soulever une erreur.

- Ne peut pas commencer par un chiffre
- Ne peut pas contenir d'espaces
- Ne peut contenir que des caractères alphanumériques (A-z, 0-9) ainsi que le `_`
- Certains mot sont réservés comme `print`, `bool`, `break`...

Le module `keywords`

Pour obtenir la liste complète des mots réservés par python vous pouvez utiliser le module `keywords`.

```
import keyword
print (keyword.kwlist)
```

Par ailleurs, il existe une **convention** parmi les développeurs python s'agissant des bonnes pratiques de nommage des variables. Ainsi le format Snake case est à privilégier. Alors que dans d'autre langages on préfèrera utiliser le Camel case. ou d'autre type de syntaxe.

Toute variable en Python est un `objet`.

C'est pour cela que le code suivant s'exécute sans erreur :

```
hello = "Hello World"
print(hello.__len__())
```

Exercice: "1. Les variables"