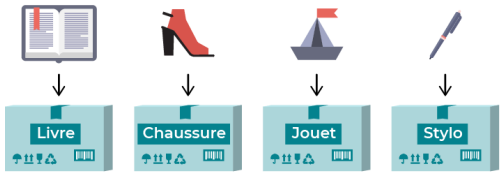
Les variables Python

Les variables

Une variable est comme une boîte : c'est l'élément qui permet de stocker des données.



Source : openclassrooms.com

Une variable

- Un nom
- Un type
- Une valeur

Types des variables Python

Types primitifs

Les types de variables les plus simples sont :

- les entiers (*integer*)
- les nombres à virgule (*float*)
- les booléens : 'vrai' ou 'faux' (*True* ou *False*)
- les chaînes de caractères (*string*)

Types des variables en Python : exemples

Entiers

- 5
- 2021
- 0

Types des variables en Python : exemples

Entiers

- 5
- 2021
- 0

Nombres à virgule

- `0.1`
- `7.0`
- `365.4`

Types des variables en Python : exemples

Entiers

- 5
- 2021
- 0

Nombres à virgule

- 0.1
- 7.0
- 365.4

Chaînes de caractères

- "Hello !"
- 'a'

En pratique

- En Python, il n'est pas nécessaire de déclarer explicitement le type des variables.
- Un nom de variable doit être sans espace, sans caractères spéciaux, et de préférence en minuscule. Pour lui attribuer une valeur, on écrit :

```
annee = 2021
```

- La valeur des chaînes de caractères s'écrit toujours entre guillemets :

```
salutation = "Bonjour !"
```

Opérations sur les variables

Avec les éléments et variables de type numérique (entiers ou nombres à virgule), il est possible d'effectuer les opérations mathématiques classiques : addition (+), soustraction (-), multiplication (*), et division (/).

Exemple

```
a = 1
produit = 5 * 2
operation = produit + a
```

Commandes utiles

- Pour afficher le contenu d'une variable, on utilise la commande *print()*
- Pour extraire le type d'une variable, on utilise la commande *type()*
- Pour attendre une valeur en entrée et la stocker dans une variable, on utilise la commande *input()*

Exercices

Expérimentations

`date = "2022"`

- Quel est le type de la variable date ?
- Essayer d'afficher l'expression `date + "1"`. Que se passe-t-il ?

Les variables Python : types complexes

Grouper les données : les listes

Il est possible de stocker des collections de données dans des listes.
Dans Python, on utilise des crochets [ ] pour indiquer une liste.

Liste

```
soupe = ['potimarron', 'pommes de terre', 'oignon']
cartes_a_jouer = ['as', 'roi', 'dame', 'valet', 10, 9,
8, 7]
```

Les listes

Pour accéder aux éléments d'une liste, il faut connaître leur position : le premier élément est à la position 0, le suivant à la position 1, etc...

Exemple

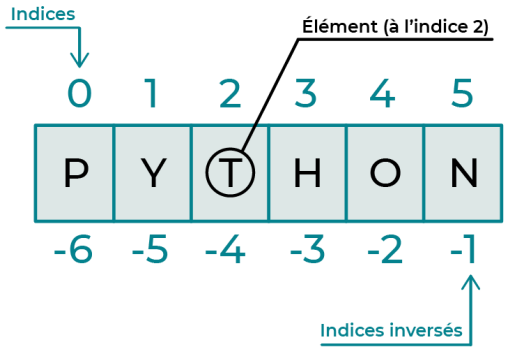
```
cartes_a_jouer = ['as', 'roi', 'dame', 'valet', 10, 9, 8, 7]

print(cartes_a_jouer[0])
» 'as'

print(cartes_a_jouer[4])
» 10
```

Listes et chaînes de caractères

Les chaînes de caractères sont en fait des listes ! On peut ainsi accéder à chaque caractère d'une chaîne de la même façon qu'à un élément de liste.



Source : openclassroom.com

Opérations sur les listes

On peut effectuer de nombreuses opérations sur les listes :

- Ajouter un élément à une liste existante : *append()*
- Supprimer un élément : *remove()*
- Calculer sa longueur (son nombre d'éléments) : *len()*
- Trier la liste : *sort()*

Les tuples

Les tuples sont assez semblables aux listes, mais sont définis par les parenthèses `()` au lieu des crochets `[]`. A la différence des listes, les tuples ne sont pas modifiables un fois définis : on peut seulement accéder à leur valeur.

Exemple

```
fruit_1 = ("orange", 3)
fruit_2 = ("pomme", 5)
```

Les dictionnaires

Les dictionnaires permettent de stocker des paires clé-valeur. La clé sert à référencer la valeur et à y accéder. On utilise des accolades `{}` pour définir un dictionnaire.

Dictionnaire

```
metadonnees = {
titre : "Le Roman inachevé",
auteur : "Louis Aragon",
annee : 1956
}
```

Pour accéder à une valeur contenue dans un dictionnaire, on doit préciser la clé associée à la valeur qu'on cherche.

```
metadonnees['titre']
```