



LES TYPES DE VARIABLES



01

DÉFINITION

Qu'est ce qu'une variable?

02

ASSIGNATION

Comment attribuer une valeur à une variable?

03

AFFECTATION MULTIPLE

Comment attribuer une valeur à plusieurs variables?

04

LES TYPES DE DONNÉES

Quels sont les types de données standard?

05


NUMBERS

Comment stocker des valeurs numériques?

06

STRING

Comment stocker des chaînes de caractères?





07

LIST

Qu'est-ce qu'une liste
en Python?

09

DICTIONARY

Qu'est-ce qu'un
dictionnaire en Python?

08


TUPLE

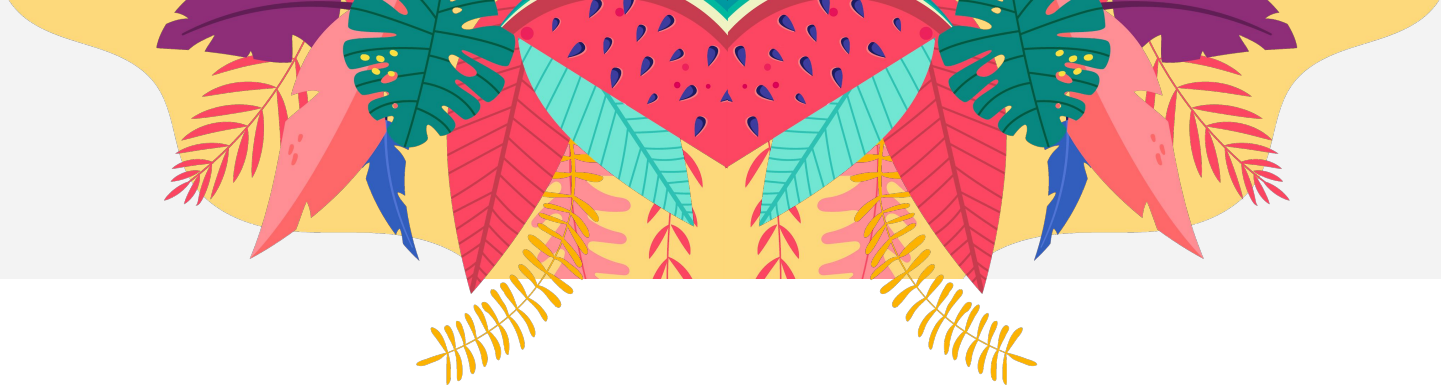
Qu'est-ce qu'un tuple
en Python?

10

CONVERSION

Comment convertir des
types de données?

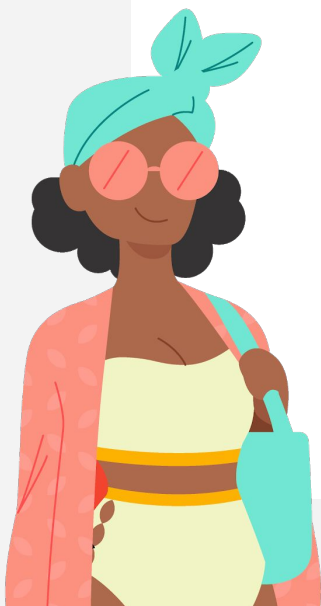




DÉFINITION

Les variables sont des emplacements de mémoire réservés pour stocker des valeurs. Cela signifie que lorsque vous créez une variable, vous réservez un certain espace en mémoire.

En fonction du type de données d'une variable, l'interpréteur alloue de la mémoire et décide de ce qui peut être stocké dans la mémoire réservée. Par conséquent, en attribuant différents types de données aux variables, vous pouvez stocker des nombres entiers, des décimales ou des caractères dans ces variables.





ASSIGNATION

Les variables Python n'ont pas besoin de **déclaration explicite** pour réserver de l'espace mémoire.

La déclaration se fait **automatiquement** lorsque vous **attribuez** une valeur à une variable. Le signe égal "=" est utilisé pour **attribuer des valeurs aux variables**.

L'opérande à **gauche de l'opérateur "="** est le **nom de la variable** et l'opérande à **droite de l'opérateur "="** est la **valeur stockée dans la variable**.

Assignment

```
number = 10  
weight = 2.5  
fruit = "apple"
```

```
print(number)  
print(weight)  
print(fruit)
```

10, 2.5 et "apple" sont les valeurs attribuées
respectivement aux variables number, weight
et fruit. Cela donne le résultat suivant :

```
10  
2.5  
apple
```



AFFECTATION MULTIPLE

Python vous permet d'attribuer **une seule valeur** à **plusieurs variables simultanément**.

Affectation multiple

```
a = b = c = 1
```

Ici, un objet **entier** est créé avec la valeur **1**, et les **trois variables** sont affectées au **même emplacement mémoire**.

Vous pouvez également affecter **plusieurs objets** à **plusieurs variables**. Par exemple :

Affectation multiple

```
a, b, c = "apple", "banana", "cherry"
```

```
print(a) # affiche : apple
```

```
print(b) # affiche : banana
```

```
print(c) # affiche : cherry
```

Ici, trois objets **chaînes de caractères** sont attribués aux variables **a, b et c** respectivement.

LES TYPES DE DONNÉES

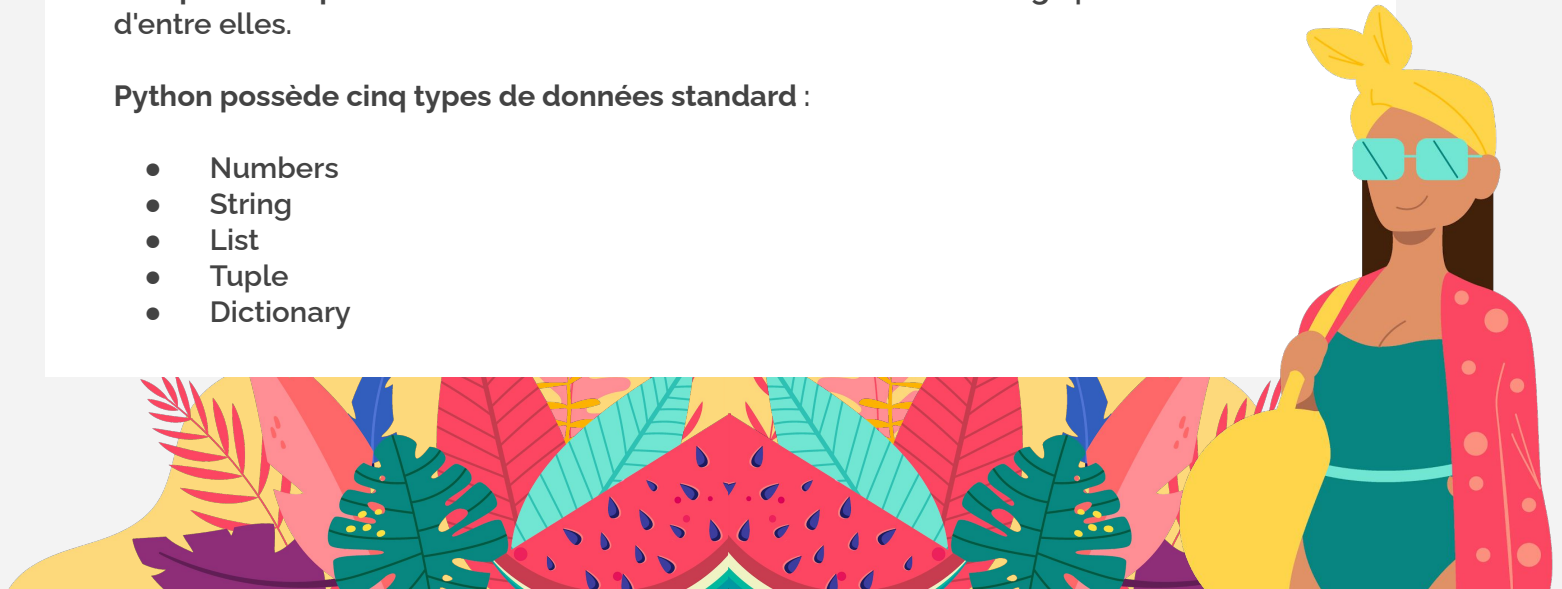
Les données stockées en mémoire peuvent être de **plusieurs types**.

Par exemple, l'âge d'une personne est stocké sous forme de valeur numérique et son adresse est stockée sous forme de caractères alphanumériques.

Python dispose de différents **types de données standard** qui sont utilisées pour définir les **opérations possibles** sur ces données et la **méthode de stockage** pour chacune d'entre elles.

Python possède cinq types de données standard :

- Numbers
- String
- List
- Tuple
- Dictionary





NUMBERS

Les types de données numériques stockent des valeurs numériques. Les objets numériques sont créés lorsque vous leur attribuez une valeur numérique.

Vous pouvez également supprimer la référence à un objet numérique en utilisant l'instruction `del`.

```
a = 1  
b = 10
```

```
del b  
print(a) # affiche : 1  
print(b) # erreur : "b" n'existe plus.
```

Vous pouvez supprimer un objet unique ou plusieurs objets en utilisant l'instruction `del`.

```
a, b, c = 1, 10, 100  
del a, b, c
```

Python supporte quatre types numériques différents :

- **int** (nombres entiers signés)
- **long** (entiers longs, ils peuvent aussi être représentés en octal et en hexadécimal)
- **float** (valeurs réelles en virgule flottante)
- **complex** (nombres complexes)



STRING

Les **strings** en Python sont identifiées comme un ensemble de caractères entre guillemets.

Des sous-ensembles de chaînes peuvent être pris en utilisant l'opérateur "[]" et "[:]" avec des indexes positifs ou négatifs.

Le signe plus "+" est l'opérateur de concaténation des chaînes de caractères et l'astérisque "*" est l'opérateur de répétition.

```
msg = 'Hello World!'
```

```
print(msg)           # affiche la string msg
print(msg[0])        # affiche le premier caractère de la string
print(msg[2:5])      # affiche la string du 3ème au 5ème caractère
print(msg[2:])       # affiche la string à partir du 3ème caractère
print(msg* 2)        # affiche deux fois la string
print(msg + " Test") # affiche la string concaténée
```

Résultats

```
Hello World!
H
llo
llo World!
Hello World!Hello World!
Hello World! Test
```



Une liste contient des éléments séparés par des virgules et placés entre crochets “[]”.

Tous les éléments appartenant à une liste peuvent être de types de données différents.

Les valeurs stockées dans une liste sont accessibles à l'aide de l'opérateur “[]” et “[:]” avec des index positifs ou négatifs.

Le signe plus “+” est l'opérateur de concaténation de la liste, et l'astérisque “*” est l'opérateur de répétition.

LIST

```
my_list= [ 'fruit', 10, 2.5, 'banana', 70.2 ]
short_list = [ 1000, 'apple' ]

print(my_list)                # affiche la liste
print(my_list[0])             # affiche le 1er élément de la liste
print(my_list[1:3])           # affiche le 2ème et 3ème élément de la liste
print(my_list[2:])            # affiche la liste à partir du 3ème élément
print(short_list * 2)         # affiche deux fois la liste
print(my_list + short_list )  # affiche la liste concaténée

# Résultats
[ 'fruit', 10, 2.5, 'banana', 70.2 ]
'fruit'
[ 10, 2.5 ]
[ 2.5, 'banana', 70.2 ]
[ 1000, 'apple', 1000, 'apple' ]
[ 'fruit', 10, 2.5, 'banana', 70.2, 1000, 'apple' ]
```



TUPLE

Un tuple est constitué d'un certain nombre de valeurs séparées par des virgules.

Les tuples sont entre parenthèses "()" et ne peuvent pas être mis à jour, ils sont immutables. Ils peuvent être considérés comme des listes en lecture seule.

```
tuple = ( 'fruit', 10, 2.5, 'banana', 70.2 )
my_tuple = ( 1000, 'apple' )

print(tuple)           # affiche le tuple
print(tuple[0])        # affiche le 1er élément du tuple
print(tuple[1:3])      # affiche le 2ème et 3ème élément du tuple
print(tuple[2:])       # affiche le tuple à partir du 3ème élément
print(my_tuple * 2)    # affiche deux fois le tuple
print(tuple + my_tuple) # affiche le tuple concaténé

# Résultats
( 'fruit', 10, 2.5, 'banana', 70.2 )
'fruit'
( 10, 2.5 )
( 2.5, 'banana', 70.2 )
( 1000, 'apple', 1000, 'apple' )
( 'fruit', 10, 2.5, 'banana', 70.2, 1000, 'apple' )
```

DICTIONARY

Les **dictionnaires** fonctionnent comme des **tableaux associatifs** et sont constitués de **paires clé-valeur**.

Une **clé** de dictionnaire peut être de n'importe quel type Python.

Les **valeurs** peuvent être n'importe quel objet Python.

Les **dictionnaires** sont entourés d'accolades "**{ }**" et les valeurs peuvent être attribuées et accessibles à l'aide de **crochets "[]"**.

Les **dictionnaires** n'ont aucune notion d'ordre entre les éléments.

```
dict = {}  
dict['one'] = "This is one"  
dict[2] = "This is two"  
my_dict = { 'name': 'apple', 'weight': 2.5, 'number': 7 }  
  
print(dict['one'])           # affiche la valeur pour la clés 'one'  
print(dict[2])              # affiche la valeur pour la clés 2  
print(my_dict)              # affiche le dictionnaire my_dict  
print(my_dict.keys())       # affiche toutes les clés  
print(my_dict.values())     # affiche toutes les valeurs
```

Résultats

```
This is one  
This is two  
{ 'name': 'apple', 'number': 7, 'weight': 2.5 }  
[ 'weight', 'number', 'name' ]  
[ 'apple', 7, 2.5 ]
```



CONVERSION DE TYPES

Parfois, vous devrez effectuer des **conversions entre les types**.

Pour cela, il suffit d'utiliser le **nom du type comme fonction**. Ces fonctions renvoient un nouvel objet représentant la valeur convertie.

En voici quelques-unes :

int(x [,base])	Convertit x en un entier. base spécifie la base si x est une chaîne.
float(x)	Convertit x en un nombre à virgule flottante.
str(x)	Convertit l'objet x en une chaîne de caractères.
tuple(s)	Convertit s en tuple.
list(s)	Convertit s en liste.
dict(d)	Crée un dictionnaire. d doit être une séquence de tuples (clé, valeur).

The background of the slide is a vibrant, colorful pattern of various tropical leaves and foliage. The leaves are in shades of green, yellow, orange, red, and purple, creating a dense and lively border around the central text area. Some leaves are large and detailed, while others are smaller and more stylized. The overall effect is a tropical and energetic aesthetic.

EXERCICES

EXERCICES

1. Quels sont les 5 types de données standard?
2. Assignez le nombre entier 777 à la variable *triple7* puis l'afficher.
3. Assignez la valeur “Cherry” aux 3 variables suivantes *slot_A*, *slot_B*, *slot_C* en utilisant l’assignation multiple.
4. Supprimez la variable *slot_B*.



EXERCICES

5. Assignez la valeur "Je profite de ma dernière journée à Las Vegas" à une variable *phrase*.

6. Afficher le sous-ensemble de caractères "Las Vegas".

7. Assigner à la variable *complément*, la valeur "car je repars demain."

8. Afficher la concaténation des variables *phrase* et *complément*.



EXERCICES

9. Les tuples sont-ils mutables ou immutables?

10. Créez un dictionnaire nommé *"vegas"* ayant pour clés *"casinos"* et pour valeur la liste suivante : Bellagio, The Venitian, Paris, Luxor, Excalibur.

La deuxième clé de ce dictionnaire est *"restaurants"* ayant pour valeur la liste suivante : STK, Edge Steakhouse, Kabuto, Le Cirque, Joe's Seafood.

11. Affichez la liste des clés du dictionnaire *"vegas"* et la liste des restaurants.

12. Convertissez *triple7* en nombre flottant et la variable *slot_A* en liste.



The background of the slide is a vibrant, colorful pattern of various tropical leaves and foliage. The leaves are in shades of green, yellow, orange, red, and purple, creating a dense and lively border around the central text. Some leaves are large and detailed, while others are smaller and more stylized. The overall effect is a tropical and energetic aesthetic.

SOLUTIONS

CORRECTION

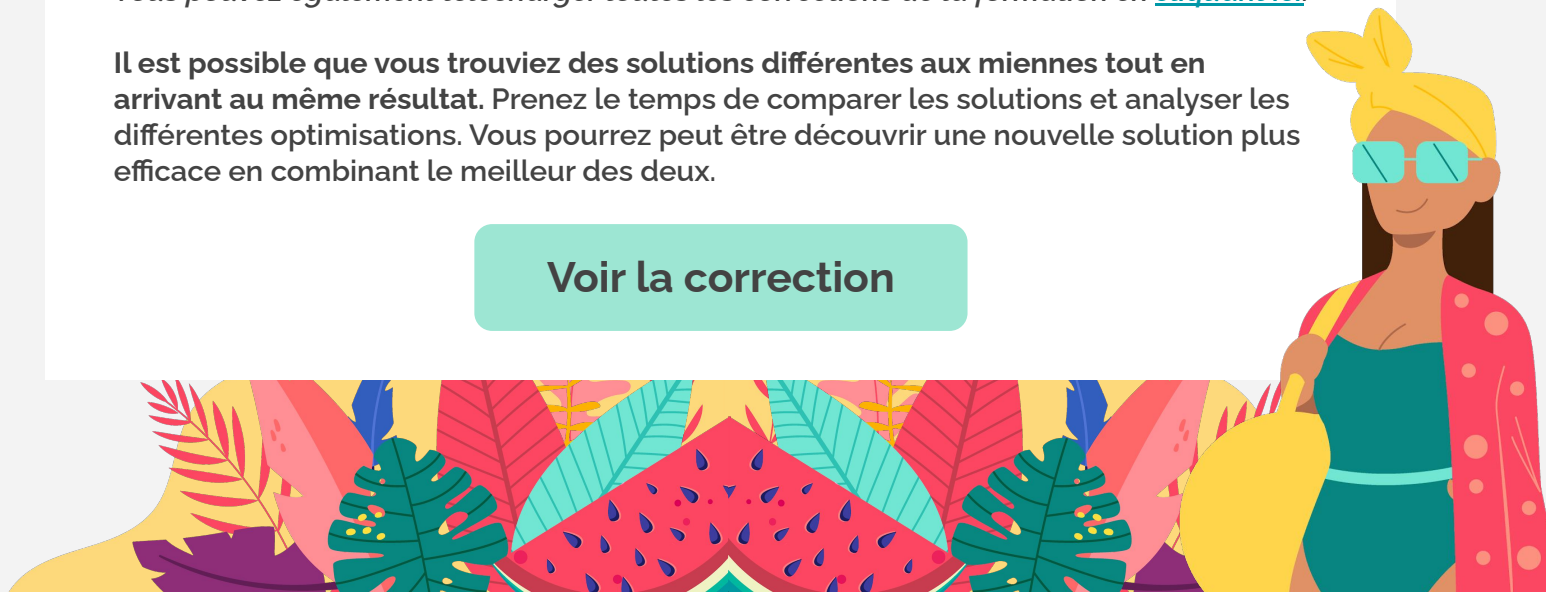
Pour visualiser la correction du chapitre cliquer sur le bouton ci-dessous.

*Le fichier de la correction s'ouvrira dans un nouvel onglet de votre navigateur préféré.
Pour cela vous devez avoir accès à une connexion Internet.*

Vous pouvez également télécharger toutes les corrections de la formation en [cliquant ici](#).

Il est possible que vous trouviez des solutions différentes aux miennes tout en arrivant au même résultat. Prenez le temps de comparer les solutions et analyser les différentes optimisations. Vous pourrez peut être découvrir une nouvelle solution plus efficace en combinant le meilleur des deux.

Voir la correction





**Félicitation vous avez terminé
le chapitre sur les types de
variables!**

A decorative border of various tropical leaves in vibrant colors like red, orange, yellow, green, and purple surrounds the central white area. The leaves include Monstera, palm, and other exotic foliage.

CRÉDITS

- Modèle de la présentation par [Slidesgo](#)
- Icônes par [Flaticon](#)
- Images et infographies par [Freepik](#)