

# Dictionnaire (type dict)

Partie 2 : Données structurées

## Quiz



Code : <https://wooclap.com>

XEIOYT

## Objectifs :

- Connaître l'interface et l'implémentation des dictionnaires,
- Créer un dictionnaire vide ou non,
- Accéder à une valeur à partir de sa clé,
- Ajouter un couple clé / valeur,
- Parcourir les clés, les valeurs d'un dictionnaire.

## Définition

### Définition

dictionnaire

En informatique, un **dictionnaire** est une structure de données représentant une séquence finie qui associe une clé et une valeur.

## Implémentation

### • Initialisation d'un dictionnaire

#### •• Représentation

En python, les dictionnaires sont contenus dans des accolades { ... }

Les clés sont séparées des valeurs par deux points : et les couples clés / valeurs sont séparés par des virgules ,

```
{ clé1 : valeur1 , clé2 : valeur2 }
```

#### •• Dictionnaire vide

- Il est possible de créer un dictionnaire vide :

```
1 # Créer un dictionnaire vide
2 dico = {}
```

### •• Dictionnaire non vide

– Il est aussi possible de l'initialiser avec des couples clés / valeurs.

```
1 # Créer un dictionnaire à partir de valeurs
2 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
```

On peut représenter les clés et valeurs associées dans ce dictionnaire par ce tableau :

Clé		Valeur
Alan Turing	→	1912
Ada Lovelace	→	1815
Grace Hopper	→	1906

### • Accéder, ajouter et modifier des éléments

#### •• Accéder à un élément

⚠ Contrairement au tableau, pour accéder à une valeur, on n'utilisera pas l'index de la valeur mais la **clé** qui lui est associée.

L'instruction pour accéder à une valeur est construite à partir du nom du dictionnaire et de la clé de la valeur souhaitée entre crochets.

nom\_dictionnaire[clé]

#### Exemple

Le dictionnaire s'appelle `informaticiens` et la valeur `1815` est associée à la clé "Ada Lovelace".

On accède donc à cet élément avec l'instruction :

**informaticiens["Ada Lovelace"]**

```
1 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2 print(informaticiens["Ada Lovelace"])
3 1815
```

- Ajouter / modifier un couple clé / valeur \_\_\_\_\_

Pour ajouter un élément, l'instruction est composée du nom du dictionnaire, de la clé entre crochets et de la valeur. L'affectation se fait par le signe =.

On a donc :

```
nom_dictionnaire[ clé ] = valeur
```

### Exemple

Pour ajouter le couple **Ian** 1973, au dictionnaire **informaticiens**, on utilisera l'instruction :

```
1  informaticiens["Ian Murdock"] = 1973
2  print(informaticiens)
3  {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906, "Ian Murdock":1973}
```

*i*

Il y a 2 cas possibles :

- Si la **clé** est déjà présente dans le dictionnaire, la **valeur** associée est **remplacée** par la nouvelle,
- Si la **clé** n'est pas déjà présente dans le dictionnaire, le couple clé / valeur est ajouté au dictionnaire.

- **Parcourir les dictionnaires** \_\_\_\_\_

- Parcourir les clés \_\_\_\_\_

*i*

Il existe deux méthodes pour parcourir les clés d'un dictionnaire :

- En itérant sur le dictionnaire,
- En itérant sur l'objet **dict\_keys** obtenu avec la méthode **keys()**

### Exemple

Itération sur le dictionnaire :

```
1  informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2  for cle in informaticiens:
3      print(cle)
```

## Exemple

Itération sur l'objet `dict_keys` obtenu par l'utilisation de la méthode `keys()` :

```
1 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2 for cle in informaticiens.keys():
3     print(cle)
```

Dans les 2 cas l'affichage sera :

```
Alan Turing
Ada Lovelace
Grace Hopper
```

---

### •• Parcourir les valeurs \_\_\_\_\_

*i*

Dans ce cas aussi, il existe deux méthodes pour parcourir les valeurs d'un dictionnaire :

- En utilisant les clés du dictionnaire,
- En itérant sur l'objet `dict_values` obtenu avec la méthode `values()`

## Exemple

Itération sur le dictionnaire :

```
1 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2 for cle in informaticiens:
3     print(informaticiens[cle])
```

Itération sur l'objet `dict_values` :

```
1 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2 for valeur in informaticiens.values():
3     print(valeur)
```

Dans les 2 cas l'affichage sera :

```
1912
1815
1906
```

---

## •• Parcourir les couples clé / valeurs \_\_\_\_\_

**i** Encore une fois, il existe deux méthodes pour parcourir les couples clé / valeur d'un dictionnaire :

- En utilisant les clés du dictionnaire,
- En itérant sur l'objet `dict_items` obtenu avec la méthode `items()`

### Exemple

Itération sur le dictionnaire :

```
1 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2 for cle in informaticiens:
3     print(f"{cle} est né.e en {informaticiens[cle]}")
```

Itération sur l'objet `dict_items` :

```
1 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2 for cle, valeur in informaticiens.items():
3     print(f"{cle} est né.e en {valeur}")
```

Dans les 2 cas l'affichage sera :

```
Alan Turing est né.e en 1912
Ada Lovelace est né.e en 1815
Grace Hopper est né.e en 1906
```

## •Supprimer un couple clé / valeur \_\_\_\_\_

Pour supprimer un couple clé / valeur d'un dictionnaire, on utilise l'instruction `del` suivi du nom du dictionnaire et de la clé du couple à supprimer entre crochets.

```
del nom_dictionnaire[ clé]
```

### Exemple

Dans cet exemple, nous allons supprimer le couple clé / valeur : `"Alan Turing":1912` du dictionnaire `informaticiens`.

```
1 informaticiens = {"Alan Turing":1912, "Ada Lovelace":1815, "Grace Hopper":1906}
2 del informaticiens["Alan Turing"]
3 print(informaticiens)
```

L'affichage obtenu sera :

```
{"Ada Lovelace":1815, "Grace Hopper":1906}
```