

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

### Exemples

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

- Un dictionnaire se note entre accolades : { et }

### Exemples

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

- Un dictionnaire se note entre accolades : { et }
- Les paires clés/valeurs sont séparés par des virgules.

### Exemples

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

- Un dictionnaire se note entre accolades : { et }
- Les paires clés/valeurs sont séparés par des virgules.
- Le caractère ":" sépare une clé de la valeur associée.

### Exemples

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

- Un dictionnaire se note entre accolades : { et }
- Les paires clés/valeurs sont séparés par des virgules.
- Le caractère ":" sépare une clé de la valeur associée.

### Exemples

- Un dictionnaire contenant des objets et leurs prix :

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

- Un dictionnaire se note entre accolades : { et }
- Les paires clés/valeurs sont séparés par des virgules.
- Le caractère ":" sépare une clé de la valeur associée.

### Exemples

- Un dictionnaire contenant des objets et leurs prix :  

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16}
```

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

- Un dictionnaire se note entre accolades : { et }
- Les paires clés/valeurs sont séparés par des virgules.
- Le caractère ":" sépare une clé de la valeur associée.

### Exemples

- Un dictionnaire contenant des objets et leurs prix :  
`prix = { "verre":12 , "tasse" : 8, "assiette" : 16}`
- Un dictionnaire traduisant des couleurs du français vers l'anglais

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Les dictionnaires de Python

- Les dictionnaires de Python permettent de stocker des données sous forme de tableau associant une clé à une valeur :

Valeurs	val1	val2	val3	val4	val5
↑	↑	↑	↑	↑	↑
Clés	'cle1'	'cle2'	'cle3'	'cle4'	'cle5'

- Un dictionnaire se note entre accolades : { et }
- Les paires clés/valeurs sont séparés par des virgules.
- Le caractère ":" sépare une clé de la valeur associée.

### Exemples

- Un dictionnaire contenant des objets et leurs prix :  
`prix = { "verre":12 , "tasse" : 8, "assiette" : 16}`
- Un dictionnaire traduisant des couleurs du français vers l'anglais  
`couleurs = { "vert":"green" , "bleu" : "blue", "rouge" :  
"red" }`



### Opérations sur un dictionnaire

- On accède aux éléments d'un dictionnaire avec la syntaxe  
`nom_dictionnaire[cle]`

### Opérations sur un dictionnaire

- On accède aux éléments d'un dictionnaire avec la syntaxe

`nom_dictionnaire[cle]`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" :  
30 }
```

Par exemple, `prix["verre"]` contient 12

### Opérations sur un dictionnaire

- On accède aux éléments d'un dictionnaire avec la syntaxe

`nom_dictionnaire[cle]`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" :  
30 }
```

Par exemple, `prix["verre"]` contient 12

- On peut ajouter une clé à un dictionnaire existant en effectuant une affectation `nom_dictionnaire[nouvelle_cle]=nouvelle_valeur`

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Opérations sur un dictionnaire

- On accède aux éléments d'un dictionnaire avec la syntaxe

`nom_dictionnaire[cle]`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" :  
30 }
```

Par exemple, `prix["verre"]` contient 12

- On peut ajouter une clé à un dictionnaire existant en effectuant une affectation `nom_dictionnaire[nouvelle_cle]=nouvelle_valeur`

On ajoute un nouvel objet avec son prix :

```
prix["couteau"]=20
```

### Opérations sur un dictionnaire

- On accède aux éléments d'un dictionnaire avec la syntaxe

`nom_dictionnaire[cle]`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" :  
30 }
```

Par exemple, `prix["verre"]` contient 12

- On peut ajouter une clé à un dictionnaire existant en effectuant une affectation `nom_dictionnaire[nouvelle_cle]=nouvelle_valeur`

On ajoute un nouvel objet avec son prix :

```
prix["couteau"]=20
```

- On peut modifier la valeur associée à une clé avec une affectation

`nom_dictionnaire[cle]=nouvelle_valeur`

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Opérations sur un dictionnaire

- On accède aux éléments d'un dictionnaire avec la syntaxe

`nom_dictionnaire[cle]`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" : 30 }
```

Par exemple, `prix["verre"]` contient 12

- On peut ajouter une clé à un dictionnaire existant en effectuant une affectation `nom_dictionnaire[nouvelle_cle]=nouvelle_valeur`

On ajoute un nouvel objet avec son prix :

```
prix["couteau"]=20
```

- On peut modifier la valeur associée à une clé avec une affectation

`nom_dictionnaire[cle]=nouvelle_valeur`

Le pris d'une tasse passe à 10 :

```
prix["tasse"]=10
```

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Présence dans un dictionnaire

- Attention, essayer d'accéder à une clé qui n'est pas dans un dictionnaire renvoie une erreur !

### Présence dans un dictionnaire

- Attention, essayer d'accéder à une clé qui n'est pas dans un dictionnaire renvoie une erreur !

Il n'y a pas de clé 'fourchette' dans le dictionnaire prix, donc `prix['fourchette']` renvoie une erreur (**KeyError**).



### Présence dans un dictionnaire

- Attention, essayer d'accéder à une clé qui n'est pas dans un dictionnaire renvoie une erreur !

Il n'y a pas de clé 'fourchette' dans le dictionnaire prix, donc `prix['fourchette']` renvoie une erreur (**KeyError**).

- On teste la présence d'une clé dans un dictionnaire avec `cle in nom_dictionnaire`

### Présence dans un dictionnaire

- Attention, essayer d'accéder à une clé qui n'est pas dans un dictionnaire renvoie une erreur !

Il n'y a pas de clé 'fourchette' dans le dictionnaire prix, donc `prix['fourchette']` renvoie une erreur (`KeyError`).

- On teste la présence d'une clé dans un dictionnaire avec `cle in nom_dictionnaire`

la fourchette n'est pas dans le dictionnaire prix

Le test `fourchette in prix` renvoie `False`

# C2 Programmation dynamique

## 1. Les dictionnaires de Python

### Présence dans un dictionnaire

- Attention, essayer d'accéder à une clé qui n'est pas dans un dictionnaire renvoie une erreur !

Il n'y a pas de clé 'fourchette' dans le dictionnaire prix, donc `prix['fourchette']` renvoie une erreur (`KeyError`).

- On teste la présence d'une clé dans un dictionnaire avec `cle in nom_dictionnaire`

la fourchette n'est pas dans le dictionnaire prix

Le test `fourchette in prix` renvoie `False`

- On peut supprimer une clé existante dans un dictionnaire avec `del nom_dictionnaire[cle]`

### Présence dans un dictionnaire

- Attention, essayer d'accéder à une clé qui n'est pas dans un dictionnaire renvoie une erreur !

Il n'y a pas de clé 'fourchette' dans le dictionnaire prix, donc `prix['fourchette']` renvoie une erreur (**KeyError**).

- On teste la présence d'une clé dans un dictionnaire avec `cle in nom_dictionnaire`

la fourchette n'est pas dans le dictionnaire prix

Le test `fourchette in prix` renvoie **False**

- On peut supprimer une clé existante dans un dictionnaire avec `del nom_dictionnaire[cle]`

On supprimer le couteau :

```
del prix["couteau"]
```

### Parcours d'un dictionnaire

- Le parcours par clé s'effectue directement avec `for cle in nom_dictionnaire`

### Parcours d'un dictionnaire

- Le parcours par clé s'effectue directement avec `for cle in nom_dictionnaire`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" : 30 }
```

Par exemple, `for objet in prix` permettra à la variable `objet` de prendre successivement les valeurs des clés : "verre", "tasse", "assiette" et "plat".

### Parcours d'un dictionnaire

- Le parcours par clé s'effectue directement avec `for cle in nom_dictionnaire`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" : 30 }
```

Par exemple, `for objet in prix` permettra à la variable `objet` de prendre successivement les valeurs des clés : "verre", "tasse", "assiette" et "plat".

- Le parcours par valeur s'effectue en ajoutant `.values()` au nom du dictionnaire : `for valeur in nom_dictionnaire.values()`

### Parcours d'un dictionnaire

- Le parcours par clé s'effectue directement avec `for cle in nom_dictionnaire`

```
prix = { "verre":12 , "tasse" : 8, "assiette" : 16, "plat" : 30 }
```

Par exemple, `for objet in prix` permettra à la variable `objet` de prendre successivement les valeurs des clés : "verre", "tasse", "assiette" et "plat".

- Le parcours par valeur s'effectue en ajoutant `.values()` au nom du dictionnaire : `for valeur in nom_dictionnaire.values()`

Par exemple, `for p in prix.values()` permettra à la variable `p` de prendre successivement les valeurs du dictionnaire : 12, 8 , 16 et 30.