# 1) présentation des dictionnaires

Comme les listes, les dictionnaires permettent de “stocker” des données. Chaque élément d’un dictionnaire est composé de 2 parties, on parle de paires “clé/valeur”.

Voici un exemple de dictionnaire :

```
mon_dico = {"nom": "Durand", "prenom": "Christophe", "date de naissance": "29/02/
```

Comme vous pouvez le constater, nous utilisons des accolades {} pour définir le début et la fin du dictionnaire (alors que nous utilisons des crochets [] pour les listes et les parenthèses pour les tuples). Dans le dictionnaire ci-dessus, “nom”, “prenom” et “date de naissance” sont des clés et “Durand”, “Christophe” et “29/02/1981” sont des valeurs. La clé “nom” est associée à la valeur “Durand”, la clé “prenom” est associée à la valeur “Christophe” et la clé “date de naissance” est associée à la valeur “29/02/1981”. Les clés sont des chaînes de caractères ou des nombres. Les valeurs peuvent être des chaînes de caractères, des nombres, des booléens…

Pour créer un dictionnaire, il est aussi possible de procéder comme suit :

```
mon_dico = {}
mon_dico["nom"] = "Durand"
mon_dico["prenom"] = "Christophe"
mon_dico["date de naissance"] = "29/02/1981"
```

À noter qu’il est aussi possible d’écrire :

```
mon_dico = dict()
mon_dico["nom"] = "Durand"
mon_dico["prenom"] = "Christophe"
mon_dico["date de naissance"] = "29/02/1981"
```

Il est possible d’obtenir la valeur associée à une clé :

```
mon_dico = {"nom": "Durand", "prenom": "Christophe", "date de naissance": "29/02/

a = mon_dico['nom']
```

Dans le programme ci-dessus, la variable *a* aura pour valeur *Durand*.

Il est facile d'ajouter un élément à un dictionnaire (les dictionnaires sont mutables)

```
mon_dico = {"nom": "Durand", "prenom": "Christophe", "date de naissance": "29/02/

mon_dico['lieu de naissance'] = 'Contamine'
```

La deuxième ligne du programme ci-dessus a permis d'ajouter la clé *lieu de naissance* au dictionnaire *mon_dico*. Cette clé a pour valeur *Contamine*

L'instruction "del" permet du supprimer une paire "clé/valeur"

Soit le dictionnaire suivant :

```
mes_fruits = {"poire": 3, "pomme": 4, "orange": 2}
```

si on exécute la ligne :

```
del mes_fruits["pomme"]
```

le dictionnaire *mes_fruits* n'aura plus que 2 clés : *poire* et *orange*

Il est possible de modifier la valeur d'une clé :

```
mes_fruits = {"poire": 3, "pomme": 4, "orange": 2}
mes_fruits["pomme"] = mes_fruits["pomme"] - 1
```

Après l'exécution de ce programme, la clé *pomme* aura pour valeur 3

# 2) parcourir un dictionnaire avec la boucle for

## a) parcourir les clés

Il est possible de parcourir un dictionnaire à l'aide d'une boucle for. Ce parcours peut se faire selon les clés ou les valeurs. Commençons par parcourir les clés à l'aide de la méthode *keys*

Le programme suivant :

```
mes_fruits = {"poire": 3, "pomme": 4, "orange": 2}

for fruit in mes_fruits.keys():
    print(fruit)
```

permet d'afficher :

```
poire
pomme
orange
```

Attention : vous n'obtiendrez par forcement le même ordre que ci-dessus (surtout si vous utilisez une version un peu ancienne de Python). En effet, les paires clé/valeur ne sont pas ordonnées dans un dictionnaire.

À noter que le *.keys()* n'est pas obligatoire pour parcourir les clés, on obtient le même résultat avec simplement :

```
mes_fruits = {"poire": 3, "pomme": 4, "orange": 2}

for fruit in mes_fruits:
    print(fruit)
```

## b) parcourir les valeurs

La méthode *values* permet de parcourir le dictionnaire selon les valeurs :

```
mes_fruits = {"poire": 3, "pomme": 4, "orange": 2}
for qte in mes_fruits.values():
  print(qte)
```

Le programme ci-dessus permet d'obtenir :

```
3
4
2
```

## c) parcourir les clés et les valeurs en même temps

Il est possible de parcourir un dictionnaire à la fois sur les clés et les valeurs en utilisant la méthode *items* :

```
mes_fruits = {"poire": 3, "pomme": 4, "orange": 2}

for fruit, qte in mes_fruits.items():
    print (f"{fruit} : {qte}")
```

l'exécution du programme ci-dessus nous permet d'avoir :

```
poire : 3
pomme : 4
orange : 2
```

Vous avez sans doute remarqué l'utilisation de deux variables (*fruit* et *qte*) au niveau du *for*... *in*

## activité 3.1

Il est possible d'afficher le contenu d'un dictionnaire dans la console.

Tapez la ligne suivante dans la partie éditeur de Spyder ou de basthon :

```
mes_fruits = {"poire": 3, "pomme": 4, "orange": 2}
```

après avoir exécuté le "programme" ci-dessus, tapez *mes_fruits* dans la partie console. Vous devriez alors voir s'afficher le contenu (clés et valeurs) du dictionnaire.

## activité 3.2

```
d = {"voiture": 25, "vélo": 55, "train": 20}
tr = d['vélo']
```

Quelle est la valeur de la variable *tr* après l'exécution du programme ci-dessus. Vérifiez votre réponse à l'aide de la console.

## activité 3.3

```
tab = []
d = {"voiture": 25, "vélo": 55, "train": 20}
for t in d.values():
    if t < 40 :
        tab.append(t)
```

Quelle est la valeur de la variable *tab* après l'exécution de ce programme. Vérifiez votre réponse à l'aide de la console.

## activité 3.4

```
tab = []
d = {"voiture": 25, "vélo": 55, "train": 20}
for v,t in d.items():
    if t < 40 :
        tab.append(v)
```

Quelle est la valeur de la variable *tab* après l'exécution de ce programme. Vérifiez votre réponse à l'aide de la console.

## activité 3.5

```
tab = [{'nom': 'toto', 'num': 2}, {'nom': 'titi', 'num': 5},  {'nom': 'tata', 'nu
tab_nom =  []

for t in tab :
    if t['num'] > 3:
        tab_nom.append(t['nom'])
```

Quelle est la valeur de la variable *tab_nom* après l'exécution de ce programme. Vérifiez votre réponse à l'aide de la console.

## activité 3.6

cliquez sur ce lien pour faire l'activité

## activité 3.7

cliquez sur ce lien pour faire l'activité

## activité 3.8

cliquez sur ce lien pour faire l'activité

## Ce qu’il faut savoir

- un dictionnaire permet de “stocker” des données
- chaque élément d’un dictionnaire est composé de 2 parties, on parle de paires “clé/valeur” (exemple : *mon_dico = {“nom”: “Durand”, “prenom”: “Christophe”, “date de naissance”: “29/02/1981”})*
- pour afficher une “valeur” particulière, on utilise la notation *mon_dico[nom_de_la_clé ]*
- il est possible de parcourir l’ensemble des clés d’un dictionnaire à l’aide d’une boucle for en utilisant *keys*
- il est possible de parcourir l’ensemble des valeurs d’un dictionnaire à l’aide d’une boucle for en utilisant *values*
- il est possible de parcourir l’ensemble des clés et des valeurs (en même temps) d’un dictionnaire à l’aide d’une boucle for en utilisant *items*

## Ce qu’il faut savoir faire

- construire un dictionnaire
- utiliser la notation *mon_dico[nom_de_la_clé ]* afin d’utiliser une valeur particulière
- parcourir l’ensemble des valeurs
- parcourir l’ensemble des clés
- parcourir l’ensemble des clés et des valeurs (en même temps)