Représentation des données - Dictionnaires

Première Générale NSI - 2021/2022

1 Introduction des dictionnaires

1.1 Qu'est-ce que c'est ?

- Un dictionnaire est une structure de données qui associe des valeurs à des clés.
- Chaque élément d'un dictionnaire est donc constitué de deux parties, on parle de paire clé/valeur.

Exemple
Voici un exemples de dictionnaire: `{"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}`

Dans cet exemple le dictionnaire contient 4 clés : `"NSI"`, `"Français"`, `"Math"`, `"Anglais"` qui sont respectivement associées aux valeurs : `18`, `12`, `17`, `15`.

Remarques

- Plusieurs clés peuvent avoir la même valeur, mais chaque clé dans un dictionnaire est **unique**.
- De manière générale, les clés doivent être de type **non-mutable**. Dans ce cours, on demandera à ce que les clés soient uniquement du type **entier** ou **chaine de caractères**.
- En revanche, les valeurs peuvent être de n'importe quel type (`int`,`float`, `str`, `bool`, `list`, `dict` etc.).

1.2 Créer un dictionnaire en Python

Il existe plusieurs façons de créer un dictionnaire en Python.

1.2.1 En donnant explicitement les paires clé/valeur

```
d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
```

- On indique les entrées du dictionnaire avec la forme `clé:valeur` entre accolades `{}`.
- Les paires clé/valeur sont séparées par des virgules (`,`).

1.2.2 À partir d'un dictionnaire vide

On commence par créer un dictionnaire vide puis on le remplit petit à petit en utilisant une affectation de la forme `d[cle] = valeur`.

```
d = {} OU d = dict()
d["NSI"] = 18
d["Français"] = 12
```

Remarque

- L'ordre d'insertion n'a pas d'importance, les paires clé/valeur ne sont en principe pas ordonnées dans un dictionnaires (même si l'ordre d'insertion est en fait conservé depuis la version Python 3.7).

1.2.3 En compréhension

Tout comme les tableaux, il est possible de créer un dictionnaire en compréhension.

```
d = {i: i*i for i in range(5)} ## {0: 0, 1: 1, 2: 4, 3: 9, 4: 16}
```

1.3 Manipulation de dictionnaire

1.3.1 Accès

On accède à la valeur associée à une clé en utilisant la construction `d[cle]`.

```
d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
d["Math"] ## Renvoie la valeur associée à la clé "Math" (17)
```

Remarques

- Attention si la clé n'existe pas dans le dictionnaire, Python renverra une erreur :

```
>>>d = {"NSI" : 18, "Français": 12}
>>>d["Math"]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
KeyError: 'Math'
```

- On peut vérifier si une clé existe dans un dictionnaire avec la construction suivante : `clé in d`.

```
>>>d = {"NSI" : 18, "Français": 12}
>>>"NSI" in d
True
>>>"Math" in d
False
```

1.3.2 Ajout/Modification

Pour ajouter ou modifier une clé dans un dictionnaire on utilise la construction d'affectation suivante : `d[cle] = valeur`.

```
d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
d["Espagnol"] = 13 ## ajoute l'entrée "Espagnol:13"
d["Math"] = 11 ## modifie la valeur associée à la clé "Math"
```

1.3.3 Suppression

Pour supprimer une clé d'un dictionnaire on utilise le mot-clé `del` avec la construction suivante `del d[cle]`.

```
d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
del d["Français"] ## Supprime la clé "Français"
```

1.3.4 Longueur

On peut connaitre la longueur d'un dictionnaire, c'est à dire le nombre de paires clé/valeur qu'il contient avec la fonction `len()` :

```
>>>d = {"prénom": "Pierre", "âge": 17}
>>>len(d)
2
```

2 Parcourir un dictionnaire

Il existe plusieurs façons de parcourir un dictionnaire: par clés, par valeurs ou bien par clés et valeurs en même temps. On utilisera dans tous les cas une boucle `for`.

2.1 Parcours par clés

Pour parcourir les clés d'un dictionnaire on peut utiliser une boucle `for` et la méthode `.keys()`

```
d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
for matiere in d.keys():
 print(matiere) ## Affiche successivement "NSI" "Français" "Math" "Anglais"
```

Remarque

- Lorsqu'on veut itèrer uniquementsur les clés, on peut omettre la méthode `.keys()`.

```
>>>d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
>>>for matiere in d:
    print(matiere)
```

2.2 Parcours par valeurs

De même que pour les clés, on peut parcourir un dictionnaire par ses valeurs. Pour cela on utilise une boucle `for` et la méthode `.values()`.

```
d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
for note in d.values():
 print(note) ## Affiche successivement 18 12 17 15
```

Remarque

- Si une méme valeur est associée à plusieurs clés, celle-ci apparaitra autant de fois qu'elle est associée:

```
>>>d = {"NSI": 18, "Français": 12, "Math": 12}
>>>for note in d.values():
    print(note)
```

2.3 Parcours par clés et valeurs en même temps

Enfin, il est également possible de parcourir à la fois les clés et leurs valeurs associées en même temps. Pour cela on utilise une boucle `for` et la méthode `.items()`

```
d = {"NSI": 18, "Français": 12, "Math": 17, "Anglais": 15}
for matiere, note in d.items():
 print(matiere, note)
```