Dictionnaires du jour 8

Dictionnaires

Un dictionnaire est une collection de types de données non ordonnés et modifiables (mutables) (clé: valeur).

Créer un dictionnaire

Pour créer un dictionnaire, nous utilisons des supports bouclés, {} ou la fonction intégrée *dict()*.

```
# syntax vide_dict = {} # dictionnaire avec valeurs de données dct = {'key1': 'value1', 'k
ey2': 'value2', 'key3': 'value3', 'key4': 'value4'}
```

Exemple:

```
Personne = {'premier_name': 'asabeneh', 'last_name': 'yetayeh', 'age': 250, 'country': '
Finland', 'is_marred': true, 'compétences': 'javascript' 'Zipcode': '02210'}}
```

Le dictionnaire ci-dessus montre qu'une valeur pourrait être n'importe quel type de données: ch aîne, booléen, liste, tuple, ensemble ou dictionnaire.

Longueur de dictionnaire

Il vérifie le nombre de paires «clés: valeur» dans le dictionnaire.

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} print (
len (dct)) # 4
```

Exemple:

```
personne = {'premier_name': 'asabeneh', 'las
t_name': 'encoreayeh', 'age': 250, 'country': '
Finlande',
```

```
'is_married': true, 'compétences': ['javascript', 'react', 'node', 'mongodb', 'python'], 'adr
esse': {'street': 'space street', 'zipcode': '02210'}} print (Len (personne)) # 7
```

Accéder aux articles du dictionnaire

Nous pouvons accéder aux éléments du dictionnaire en faisant référence à son nom de clé.

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} print (
dct ['key1']) # value1 print (dct ['key4']) # value4
```

Exemple:

```
Personne = {'First_name': 'Asabeneh', 'last_name': 'yetayeh', 'age': 250, 'country': 'Fi
nland', 'is_marred': true, 'compétences': 'javascript' 'Zipcode': '02210'}}
```

```
print (personne ['premier_name']) # asabeneh print (personne ['country']) # Finland Print (pers
onne ['compétences']) # ['javascript', 'react', 'node', 'mongodb', 'python'] print (personne [" "] [0
]) # javascript print (Person'] [«STREE
```

L'accès à un élément par nom de clé soulève une erreur si la clé n'existe pas. Pour éviter d'abo rd cette erreur, nous devons vérifier s'il existe une clé ou si nous pouvons utiliser la méthode *get*. La méthode GET en renvoie aucune, qui est un type de données d'objet non-type, si la clé n'existe pas.

```
personne = {'first_name': 'asabeneh', 'last_n
ame': 'encoreayeh',
```

```
'Age': 250, 'Country': 'Finland', 'is_marred': true, 'compétences': ['javascrip print (Pers
on.get ('country')) # Finland Print (Person.get ('Skills')) # ['html', 'css', 'javascrip
```

Ajout d'articles à un dictionnaire

Nous pouvons ajouter de nouvelles paires de clés et de valeur à un dictionnaire

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} dct ['k
ey5'] = 'value5'
```

Exemple:

```
Personne = {'premier_name': 'asabeneh', 'last_name': 'yetayeh', 'age': 250, 'country': '
Finland', 'is_marred': true, 'compétences': 'javascript' 'Zipcode': '02210'}}
```

```
personne ['job_title'] = 'instructeur'
personne [«compétences»]. Ajouter («html») im
primer (personne)
```

Modification des articles dans un dictionnaire

Nous pouvons modifier des articles dans un dictionnaire

```
# syntaxe
```

```
dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} dct ['key1'] = 'va
lue-one'
```

Exemple:

```
Personne = {'premier_name': 'asabeneh', 'last_name': 'yetayeh', 'age': 250, 'country': '
Finland', 'is_marred': true, 'compétences': 'javascript' 'Zipcode': '02210'}}
```

```
Person ['First_name'] = 'eyoB' personne ['age'
] = 252
```

Vérification des clés dans un dictionnaire

Nous utilisons l'opérateur *in* pour vérifier si une clé existe dans un dictionnaire

```
# syntaxe
dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3',
'key4': 'value4'}
print ('key2' dans dct) # true print ('key5' da
ns dct) # false
```

Supprimer les paires de clés et de valeur d'un dictionnaire

- *pop(key)*: supprime l'élément avec le nom de clé spécifié:
- *popitem()*: supprime le dernier élément
- *del*: supprime un élément avec un nom de clé spécifié

```
# syntaxe
dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3',
'key4': 'value4'}
dct.pop ('key1') # supprime l'élément KEY1
dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3',
'key4': 'value4'}
dct.popitem () # supprime le dernier élément del dct ['key
2'] # supprime l'élément de clé2
```

Exemple:

```
personne = {'first_name': 'asabeneh', 'last_n
ame': 'encoreayeh',
```

```
'Age': 250, 'Country': 'Finland', 'is_marred': true, 'compétences': ['javascript', 'react', '
node', 'MongoDB', 'Python'], 'Address': {'Street': 'Space Street', 'Zipcode': '02210'}}}
```

```
Person.pop ('First_name') # Supprime l'élément FirstName Person.popitem () # supprime l'élé
ment d'adresse del Person ['is_married'] # supprime l'élément is_married
```

Changer le dictionnaire en une liste d'éléments

La méthode *items()* change le dictionnaire en une liste de tuples.

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'}


print (dct.items ()) # dict_items ([('key1', 'value1'), ('key2',
'value2'), ('key3', 'value3'), ('key4', 'value4')]))
```

Nettoyer un dictionnaire

Si nous ne voulons pas que les éléments d'un dictionnaire, nous pouvons les effacer en utilisant la méthode *cle*

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} print (
dct.clear ()) # aucun
```

Suppression d'un dictionnaire

Si nous n'utilisons pas le dictionnaire, nous pouvons le supprimer complètement

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} del dct
```

Copier un dictionnaire

Nous pouvons copier un dictionnaire en utilisant une méthode *copy()*. En utilisant la copie, n ous pouvons éviter la mutation du dictionnaire d'origine.

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'}


dct_copy = dct.copy () # {'key1': 'value1', 'key2': 'value2',
'key3': 'value3', 'key4': 'value4'}
```

Obtenir des clés de dictionnaire comme liste

La méthode *keys()* nous donne toutes les clés d'un dictionnaire A comme liste.

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} touches = dc
t.keys () print (clés) # dict_keys (['key1', 'key2', 'key3', 'key4'])
```

Obtenir des valeurs de dictionnaire comme liste

La méthode *values* nous donne toutes les valeurs d'un dictionnaire comme une liste.

```
# syntaxe dct = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3', 'key4': 'value4'} valeurs = dct.
values () print (valeurs) # dict_values (['value1', 'value2', 'value3', 'value4'])
```

Vous êtes étonnant. Maintenant, vous êtes super chargé du pouvoir des dictionnaires. Vous venez de terminer les défis du jour 8 et vous êtes 8 étapes de la grandeur à la grandeur. Faites maintenant quelques exercices pour votre cerveau et vos muscles.

Exercices: Jour 8

1. Créez un dictionnaire vide appelé chien

2. Ajouter le nom, la couleur, la race, les jambes, l'âge au dictionnaire pour chiens 3. Créez un dictionnaire étudiant et ajoutez d'abord_name, last_name, sexe, âge, état ma trimonial, compétences, pays, ville et adresse comme clés pour le dictionnaire 4. Obt enez la longueur du dictionnaire étudiant 5. 8. Obtenez les valeurs du dictionnaire co mme une liste 9. Changez le dictionnaire en une liste de tuples à l'aide de la méthode *items()* 10. Supprimer l'un des éléments du dictionnaire 11. Supprimer l'un des dictio nnaires

Félicitations!