dictionnairesisn-nsi

September 8, 2019

#
Les dictionnaires.

Le dictionnaire est un type de données qui se rapproche des listes, mais, contrairement aux listes, qui n'ont pas de structure ordonnée.

Un dictionnaire est composé de clés auxquelles sont associées une valeur. Aussi bien les clés que les valeurs peuvent être de tout type !

0.1 I Création d'un dictionnaire

Le dictionnaire se crée à l'aide d'une accolade (quand les listes sont créées avec des crochets et les tuples avecs des parenthèses).

[ ]:
```
dico={}
type(dico)
```

Pour créer une clé, on la nomme entre guillements, le tout entre crochet. On lui affecte sa valeur à l'aide du signe =.

[ ]:
```
dico = {}
dico ["Nom "]= "Chan"
dico ["Prénom "]= "Jacky"
dico
```

On peut aussi le créer directement :

[ ]:
```
dico1={ "Nom": "Chan", "Prénom":"Jacky"}
dico1
```

La syntaxe n'est pas tout à fait la même, la clé n'étant plus entre crochet et l'affectation se faisant avec :.

0.2 II Opérations sur les dictionnaires

0.2.1 1. Parcourir un dictionnaire

[ ]:
```
for cle in dico1.keys():
    print(cle)
```

Ou bien :

[ ]:
```
for cle in dico1: # même résultat
    print(cle)
```

Pour parcourir la liste des valeurs, on utilise la méthode `values` :

[ ]:
```
for valeurs in dico1.values():
    print(valeurs)
```

Pour avoir les clés et les valeurs on utilise la méthode `items` :

[ ]:
```
for cle, valeur in dico1.items():
    print(cle, valeur)
```

Remarque, on peut avoir une présentation plus soignée :

[ ]:
```
for cle, valeur in dico1.items():# A l'aide d'un fprint ou de la méthode format
    print(f'le {cle} est {valeur}')
for cle, valeur in dico1.items():
    print('le {} est {}'.format(cle, valeur))
```

Pour avoir la valeur d'une clé, on utilise la méthode `get` :

[ ]:
```
print(dico1.get("Nom"))
```

Attention ! On peut utiliser la méthode `get` avec deux paramètres : le nom de la clé et 0 au cas où la clé n'aurait pas de valeurs.

0.2.2 2. Supprimer, ajouter un élément.

Pour ajouter un élément, il suffit de créer une nouvelle clé.

[ ]:
```
dico1["Age"] = 24
dico1
```

Pour modifier la valeur d'une clé, on procède comme pour une variable :

[ ]:
```
dico1["Nom "]="Kennedy"
dico1
```

Pour supprimer une clé, on utilise la méthode `del` (ou la méthode `pop` si on veut conserver la valeur associée à la clé supprimée en mémoire) :

[ ]:
```
del(dico1["Age"])
dico1.pop("Prénom")
dico1
```

Enfin, pour copier un dictionnaire de façon indépendante, on utilise la méthode `copy`, sinon la modification d'un dictionnaire modifiera l'autre. Cette remarque fonction aussi pour les listes.

[ ]:
```
dico1={ "Nom": "Chan", "Prénom":"Jacky"}
dico2=dico1
dico3=dico1.copy()
dico1["Nom"]="Kennedy"
print(dico1,dico2,dico3)
```

Exercice

A partir d'une phrase entrée par l'utilisateur, créer un dictionnaire qui ait les lettres en clés et leur occurence en valeurs. Transformer ce dictionnaire en liste (ou tuple) et afficher le résultat par ordre alphabétique.

Prolongation : afficher ce résultat sous forme d'histogramme avec le module matplotlib

[ ]:
```
"""
Created on Sun Apr 14 16:46:40 2019

@author: jeanericrichard
"""



import matplotlib.pyplot as plt
import numpy as np

phrase = str(input('Phrase svp :'))
dico={}
for lettre in phrase:
    dico[str(lettre)]=phrase.count(lettre)
print(dico)

L=[]
for lettre in dico.keys():
    L.append(lettre)
    L.sort()
print(L)
nbrL=[]
for lettre in L:
    nbrL.append(dico[lettre])
print(nbrL)

x = np.arange(len(L))
plt.bar(x, height= nbrL)
plt.xticks(x, L)
```

[ ]: