

Test 4

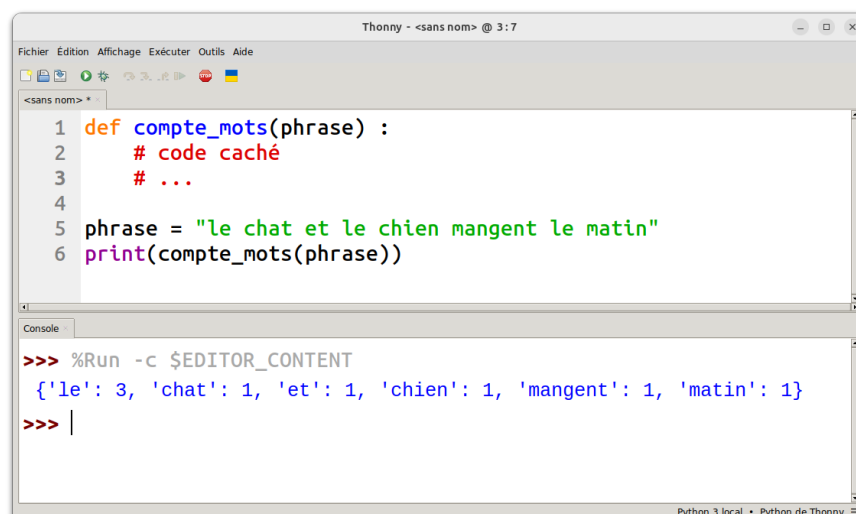
1. Écrivez un programme qui lit un mot donné par l'utilisateur et construit un dictionnaire associant à chaque lettre de ce mot son nombre d'occurrences.

```
1 mot = input("votre mot ? ")
2 occ = {}
3 for lettre in mot :
4     if lettre in occ :
5         occ[lettre] += 1
6     else :
7         occ[lettre] = 1
8
9 print(occ)
```

2. Écrivez une fonction **compte_mots(phrase)** qui prend en paramètre une chaîne de caractères **phrase** et qui retourne un dictionnaire dont les clés sont les mots de la phrase et les valeurs le nombre d'occurrences de chaque mot.

Les mots sont séparés par des espaces. On suppose qu'il n'y a aucune apostrophe ni aucune ponctuation.

Par exemple :



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python function definition for `compte_mots`. The function takes a `phrase` as an argument and contains a placeholder for the implementation code. Below the editor, the console window shows the execution of the code, displaying the resulting dictionary for the input phrase "le chat et le chien mangent le matin".

```
1 def compte_mots(phrase) :
2     # code caché
3     # ...
4
5 phrase = "le chat et le chien mangent le matin"
6 print(compte_mots(phrase))
```

```
>>> %Run -c $EDITOR_CONTENT
{'le': 3, 'chat': 1, 'et': 1, 'chien': 1, 'mangent': 1, 'matin': 1}
>>> |
```

```
1 def compte_mots(phrase) :
2     occ = {}
3     for mot in phrase.split(" ") :
4         if mot in occ :
5             occ[mot] += 1
6         else :
7             occ[mot] = 1
8     return occ
```

3. Voici un stock de fruits dans un tableau associatif. Écrivez des instructions qui suppriment de ce dictionnaire tous les fruits dont le stock est inférieur à 5, et qui en même temps crée une liste contenant tous les noms des fruits supprimés.

```
1 stock = {  
2     "pommes" : 17,  
3     "poires" : 4,  
4     "prunes" : 53,  
5     "ananas" : 3,  
6     "pêches" : 7  
7 }
```

```
supprimes = []  
  
for fruit in stock :  
    if stock[fruit] < 5 :  
        supprimes.append(fruit)  
  
for fruit in supprimes :  
    del stock[fruit]
```

4. Voici un dictionnaire de notes d'étudiants. Écrivez des instructions qui permettent de construire un nouveau dictionnaire dont les clés sont les prénoms et les valeurs sont les moyennes des étudiants (on suppose que les coefficients des notes sont tous 1).

```
1 notes = {  
2     "Alice": [12, 15, 14],  
3     "Bob": [10, 9],  
4     "Claire": [18, 17, 19, 16]  
5 }
```

```
moyennes = {}  
for prenom, liste_notes in notes.items() :  
    moyenne = sum(liste_notes) / len(liste_notes)  
    moyennes[prenom] = moyenne
```

5. Écrivez une fonction `anagramme(mot1, mot2)` qui dit si oui ou non les mots passés en paramètres sont anagrammes l'un de l'autre.

```
def anagrammes(mot1, mot2) :  
    if len(mot1) != len(mot2) :  
        return False  
    for lettre in mot1 :  
        if mot1.count(lettre) != mot2.count(lettre) :  
            return False  
    return True
```

6. Voici 3 groupes de personnes.

```
1 groupe1 = {"Alice", "Bob", "Claire"}  
2 groupe2 = {"Bob", "David", "Emma"}  
3 groupe3 = {"Alice", "Emma", "Fred"}
```

a) Écrivez des instructions qui construisent l'ensemble des personnes apparaissant au moins une fois dans un groupe.

```
u = groupe1.union(groupe2).union(groupe3)  
print(u)
```

b) Écrivez des instructions qui permettent de dire quelles personnes sont dans au moins 2 groupes.

```
v12 = groupe1.intersection(groupe2)  
v13 = groupe1.intersection(groupe3)  
v23 = groupe2.intersection(groupe3)  
  
v = v12.union(v23).union(v13)  
print(v)
```