TP 8 – UTILISATION DE DEICTIONNAIRES

Préambule :

1. Ouvrir une session sur l'ordinateur en entrant Identifiant / Mot de passe ;
2. Lancer Spyder en cherchant Spyder dans la barre de recherche ;
3. Effacer le texte dans la fenêtre de gauche et écrire :

```
#NOM
#PRENOM
#TP8
```

Objectifs de ce TP :

- *Créer des dictionnaires ;*
- *Accéder aux valeurs d'un dictionnaire ;*
- *Ajouter / modifier les données d'un dictionnaire ;*
- *Utilisation des dictionnaires pour résoudre des problèmes.*

Toutes les réponses aux questions doivent écrites dans le programme qui sera à rendre dans le dossier Restitution de devoirs de votre classe avant 18h.

Avant de commencer les exercices à proprement parler utilisant des dictionnaires, vous allez utiliser la console Python et tester plusieurs commandes. Vous pourrez alors découvrir l'utilisation des dictionnaires.

Définition :
Un dictionnaire, objet de type dict, est une association entre des clés et des valeurs. Les clés sont des objets non mutables, les valeurs des objets quelconques. Ces clés ne sont pas ordonnées. On accède à une clé en temps constant selon un processus qu'il n'est pas nécessaire de décrire ici. On accède à une valeur par sa clé.

Pour créer un dictionnaire, on écrit entre des accolades les couples clé: valeur, une clé et la valeur sont séparées par le signe deux-points, les couples sont séparés par des virgules.

```
>>> jours={"dimanche":1, "lundi":2, "mardi":3, "mercredi":4, "jeudi":5,"vendredi":6, "samedi":7}
```

La méthode `keys` permet d'obtenir les différentes clés.

```
>>> jours.keys()
```

La méthode `items` permet d'obtenir l'ensemble des couples (clé, valeur).

```
>>> jours.items()
```

Le mot `in` permet de tester l'appartenance d'une clé à un dictionnaire, pas d'une valeur.

```
>>> 4 in jours

>>> "dimanche" in jours
```

Donc, avec `for elt in dic`, où dic est un dictionnaire, la variable d'itération `elt` est une **clé**.

```
>>> cle = []
>>> for elt in jours:
      cle.append(elt)
```

La variable `cle` a pour valeur :
`['dimanche', 'lundi', 'mardi', 'mercredi', 'jeudi', 'vendredi', 'samedi'].`

Mais il est aussi possible d'itérer sur les couples clés-valeurs en utilisant la méthode `items`.

```
>>> for cle, val in jours.items():
            print(cle, val)
```

L'accès à une valeur s'obtient comme avec les listes, les tuples ou les chaînes. Mais il faut préciser la clé à la place de l'indice.

```
>>>  jours['dimanche']
```

Comme pour les listes, il est possible de modifier une valeur par affectation :

```
>>> jours['dimanche'] = 0.
```

Pour insérer un élément, on écrit une instruction comme :

```
>>> jours['dimanche2'] = 8
>>> print(jours)
```

Si la clé c existe, l'instruction `d[c] = val` modifie la valeur associée à la clé c. Si la clé c n'existe pas, elle est créée et la valeur val lui est associée.

On peut avoir le nombre de clés dans un dictionnaire à l'aide de la fonction len.

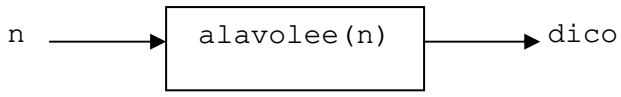
```
>>> len(jours)
```

<u>Exercice 1 : à la volée</u>

1. Ecrire à la suite du code le commentaire `#Exercice 1`.

L'objectif est d'écrire une fonction qui permet de créer à la volée un dictionnaire contenant n clés et n valeurs associées. Le nom des clés et les valeurs associées seront demandées à l'utilisateur à l'aide de la fonction `input`.

2. Ecrire une fonction `alavolee(n)` qui permet de créer le dictionnaire :



3. Utiliser votre fonction pour créer le dictionnaire suivant qui contient vos notes au concours :

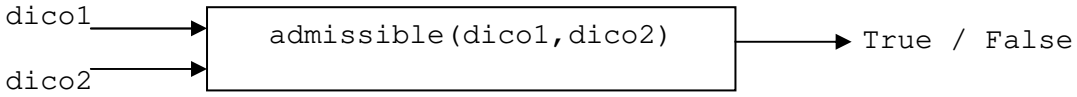
```
dico_notes={'maths': 10.0, 'phys': 12.0, 'si': 9.0, 'francais': 19.0, 'anglais': 5.0}
```

4. Utiliser votre fonction pour créer le dictionnaire suivant qui contient les coefficients des matières au concours :

```
dico_coeffs= {'maths': 4.0, 'phys': 3.0, 'si': 8.0, 'francais': 2.0, 'anglais': 2.0}
```

A l'aide de ces deux dictionnaires, nous allons savoir si vous êtes admissible en vérifiant que le total de vos points est supérieur à la moyenne (10/20).

5. Créer la fonction `admissible(dico1,dico2)` qui renvoie `True` si vous êtes admissible et `False` sinon :



6. Tester votre fonction sur les dictionnaires `dico_notes` et `dico_coeffs`.

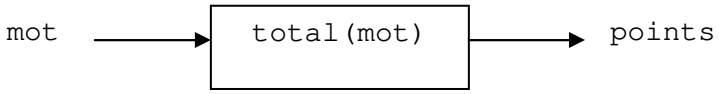
Exercice 2 : combien au scrabble ?

Le dictionnaire suivant (récupérer le en copier/coller), répertorie les points que rapportent les lettres d'un mot au scrabble :

```
scrabble={"A":1,"B":3,"C":3,"D":2,"E":1,"F":4,"G":2,"H":4,"I":1,"J":8,"K":10,"L":1,"M":2,"N":1,"O":
1,"P":3,"Q":8,"R":1,"S":1,"T":1,"U":1,"V":4,"W":10,"X":10,"Y":10,"Z":10}
```

L'objectif de cet exercice est de calculer le nombre de points que rapporte un mot au scrabble en utilisant le dictionnaire `scrabble`.

1. Ecrire à la suite du code le commentaire `#Exercice 2.`
2. Ecrire une fonction `total(mot)` qui détermine le nombre de `points` que rapporte le mot.



Exercice 3 : nombre d'occurences

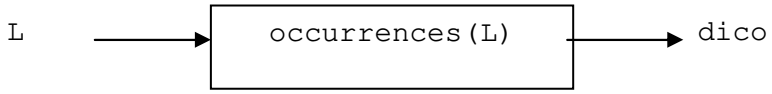
L'objectif de cet exercice est de calculer le nombre de fois qu'apparait une lettre dans une liste, appelée occurrence. Par exemple, combien de fois apparait 1 dans cette liste :

```
liste=[1,3,2,1,4,1,2,1]
```

Le résultat devra être présenté sous la forme d'un dictionnaire, comme celui-ci :

```
dico={1: 4, 3: 1, 2: 2, 4: 1}
```

1. Ecrire à la suite du code le commentaire `#Exercice 3.`
2. Ecrire la fonction `occurrences(L)` qui renvoie les occurrences des éléments de la liste `L` :



3. Tester votre fonction avec `L=liste`.

Exercice 4 : correction automatique de QCM

L'objectif de cet exercice est de créer un correcteur automatique de QCM. Les réponses justes au QCM sont contenues dans le dictionnaire reponse_justes ci-dessous :

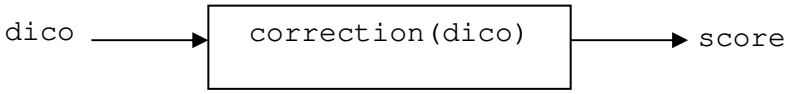
```
reponses_justes = {"Q1":"c","Q2":"a","Q3":"d","Q4":"c","Q5":"b"}
```

Les réponse de l'élève Alice sont contenues dans le dictionnaire ci-dessous ; remarquez qu'Alice n'a pas répondu à toutes les questions :

```
reponses_Alice = {"Q1":"b","Q2":"a","Q3":"d","Q5":"a"}
```

La notation du QCM est la suivante : 3 points par réponse correcte, -1 point par réponse incorrecte et 0 si l'on n'a pas répondu.

1. Ecrire à la suite du code le commentaire `#Exercice 4.`
2. Ecrire la fonction `correction(dico)` qui renvoie le score atteint par les réponses contenues dans `dico` :



3. Tester votre fonction sur le dictionnaire d'Alice.

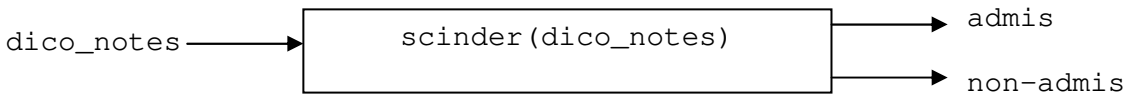
Exercice 5 : admis ou non-admis ?

Le dictionnaire suivant regroupe les notes d'étudiants à un examen final.

```
etudiants = {"etudiant_1" : 13 , "etudiant_2" : 17 , "etudiant_3" : 9 , "etudiant_4" : 15 ,
"etudiant_5" : 8 , "etudiant_6" : 14 , "etudiant_7" : 16 , "etudiant_8" : 12 , "etudiant_9" : 13 ,
"etudiant_10" : 15 , "etudiant_11" : 14 , "etudiant_112" : 9 , "etudiant_13" : 10 , "etudiant_14" :
12 , "etudiant_15" : 13 , "etudiant_16" : 7 , "etudiant_17" : 12 , "etudiant_18" : 15 ,
"etudiant_19" : 9 , "etudiant_20" : 17 ,}
```

L'objectif de cet exercice est de scinder ce dictionnaire en deux : un dictionnaire contenant les étudiants non-admis et leurs notes, un dictionnaire contenant les étudiants admis et leurs notes.

1. Ecrire à la suite du code le commentaire `#Exercice 5.`
2. Créer la fonction `scinder(dico_notes)` qui renvoie les deux dictionnaires souhaités :



3. Tester votre fonction sur le dictionnaire `etudiants`.