TP 10 - Dictionnaires et ensembles

Exercice 1. Analyse d'un fichier texte

Récupérez les fichiers `miserables.txt` et `dico.txt` que vous allez utiliser pour tester vos fonctions.

1. Écrire une fonction `occurrencesLettres` qui prend en paramètre un nom de fichier et renvoie un dictionnaire qui associe à chaque lettre le nombre de fois où elle apparaît dans le fichier. Pour tester qu'un caractère `c` est une lettre, vous pouvez utiliser `c.isalpha()` qui renvoie `True` si `c` est une lettre et `False` sinon.

   Vous ne devez lire le fichier qu'une seule fois !

   Si vous testez votre fonction sur le fichier `miserables.txt`, vous devez obtenir :

   ```
   print(occurencesLettres('miserables.txt'))

   {'d': 17744, 'e': 88828, 'f': 5865, 'g': 4509, 'a': 44998, 'b': 4978,
    'c': 15904, 'l': 30212, 'm': 15440, 'n': 34706, 'o': 26010,
   'h': 5119, 'i': 39376, 'j': 2888, 'k': 20, 't': 38794, 'u': 32391,
    'v': 10174, 'w': 21, 'p': 13345, 'q': 6626, 'r': 31846, 's': 37652,
    'x': 2074, 'y': 1735, 'z': 962}
   ```

2. Écrire une fonction `pourcentages` qui prend en argument un dictionnaire associant à chaque clé un nombre d'occurrences et renvoie un nouveau dictionnaire associant à chaque clé le pourcentage que représente les occurrences de cette clé sur le total des occurrences du dictionnaire.

   Par exemple, si `d = { 'a' : 2 , 'b' : 1 , 'c' : 1}`, `pourcentages(d)` renverra { `'a' : 50.0 , 'b' : 25.0 , 'c' : 25.0`}, puisque les deux 'a' représentent 50 % des 4 occurrences du dictionnaire.

   Si vous testez votre fonction sur le fichier `miserables.txt`, vous devez obtenir :

   ```
   occL=occurencesLettres('miserables.txt')
   print(pourcentages(occL))

   {'y': 0.33872362690031765, 'x': 0.40490651423127305, 'z': 0.18781102540524816,
    'u': 6.32368703108253, 't': 7.573743159637419, 'w': 0.004099824878908744,
    'v': 1.986267538953217, 'q': 1.2935923641737779, 'p': 2.6053410956684373,
    's': 7.350790778127239, 'r': 6.217286813987041, 'm': 3.014347434778619,
   'l': 5.898281392456712, 'o': 5.077925957162687, 'n': 6.775643916543183,
    'i': 7.6873668777100335, 'h': 0.9993811216730409, 'k': 0.003904595122770232,
    'j': 0.5638235357280215, 'e': 17.34186877827171, 'd': 3.46415679292175,
    'g': 0.8802909704285489, 'f': 1.1450225197523705, 'a': 8.784948566720745,
    'c': 3.1049340416268887, 'b': 0.9718537260575107}
   ```

3. Écrire une fonction `triParOccurrences` qui prend un dictionnaire associant à chaque clé un nombre d'occurrences et renvoie la liste des clés triées par ordre décroissant d'occurrences.

   Si vous testez votre fonction sur le fichier `miserables.txt`, vous devez obtenir :

```
occL=occurencesLettres('miserables.txt'))
print(triParOccurrences(occL))

['e', 'a', 'i', 't', 's', 'n', 'u', 'r', 'l', 'o', 'd', 'c',
  'm', 'p', 'v', 'q', 'f', 'h', 'b', 'g', 'j', 'x', 'y', 'z', 'w', 'k']
```

4. Écrire une fonction `occurrencesMots` qui prend en paramètre un nom de fichier et renvoie un dictionnaire qui associe à chaque mot le nombre de fois où elle apparait dans le fichier. Pour tester qu'une chaîne `m` est bien un mot vous pouvez utiliser `m.isalpha()`.

   Vous ne devez lire le fichier qu'une seule fois !

   Si vous testez votre fonction sur le fichier `miserables.txt`, vous devez obtenir :

```
occMots=occurencesMots('miserables.txt'))
print(occMots['le'])
print(occMots['agreable'])

2563
1
```

5. Écrire une fonction `top25` qui prend un dictionnaire d'occurrences et renvoie la liste des 25 premières clés qui ont le plus grand nombre d'occurrences par ordre décroissant d'occurrences. Vous pouvez utiliser `triParOccurrences`.

   Si vous testez votre fonction sur le fichier `miserables.txt`, vous devez obtenir :

```
occMots=occurencesMots('miserables.txt'))
print(top25(occMots))

['de', 'la', 'il', 'et', 'a', 'le', 'l', 'un', 'les', 'que', 'd', 'une',
 'qui', 'qu', 'en', 'etait', 'dans', 'est', 'ce', 'des', 'avait', 'ne',
'pas', 'se', 'n']
```

6. Le fichier `dico.txt` contient une liste des mots de la langue française. Écrire une fonction `correcteur` qui prend en paramètre un nom de fichier et qui renvoie la liste des mots de ce fichier qui ne sont pas des mots apparaissant dans `dico.txt`.

```
print(correcteur('miserables.txt'))

['blondeau', 'greif', 'josefa', 'com', 'anywhere', 'ecria', 'xiv', 'xii',
 'ailly', 'nisi', 'dignifier', 'needham', 'barcelonnette', 'please',
 'cythere', 'caron', 'delaverderie', 'archidiaconats', 'arabie', 'toulon',
 'lebrun', 'beure', 'coural', 'parvulos', 'boujean', 'aurele', 'henri' ...
```

7. ★ Écrire une fonction `maxLongueur` prenant un nom de fichier en paramètre et affichant pour chaque taille la liste des mots de cette taille qui apparaissent le plus dans le texte.

```
maxLongueur('miserables.txt')

0 []
1 ['a']
2 ['de']
3 ['les']
4 ['dans']
5 ['etait']
6 ['eveque']
```

```
7 ['valjean']
8 ['monsieur']
9 ['madeleine']
10 ['thenardier']
11 ['monseigneur']
12 ['champmathieu']
13 ['conventionnel']
14 ['habituellement']
15 ['continuellement', 'malheureusement']
16 ['particulierement']
17 []
18 ['consciencieusement']
```

Exercice 2. Cherchez l'assassin !
Un meurtre horrible vient d'être commis et vous êtes chargé de l'enquête.

- **1ère partie de l'enquête :**
  Après plusieurs jours d'investigation, la liste de coupables potentiels est encore longue (le fichier `noms.txt`), mais vous avez tout de même réussi à déterminer que l'assassin se trouvait forcément au bar ou en train de faire une partie de poker à 15h et qu'il avait rendez-vous avec Erin à 20h. Ceci va vous permettre d'établir la liste des suspects.
- **2ème partie de l'enquête :**
  Chaque suspect a été interrogé sur son agenda le jour du meurtre. Une seule personne à intérêt à mentir : le coupable ! Recoupez les réponses des suspects et les informations données pour chaque activité et démasquez l'assassin...

Chaque fichier fourni ne peut être lu qu'UNE SEULE FOIS.

Le dossier de police se trouve dans l'archive `enquete.tgz`. Dans la ville où a été commis le meurtre, les différentes activités possibles sont listées ci-dessous :

```
activites = ["Bar", "Poker", "Golf", "Patinoire", "Cinema"]
```

Grâce au système de surveillance des établissements où elles ont lieu, vous avez recueilli les noms de toutes les personnes qui ont participé aux activités ainsi que les heures où elles étaient présentes (ces informations sont fiables). Ces données sont conservées dans les fichiers du répertoire `activites`, chaque ligne de fichier étant de la forme : `nom:heure`.

Vous avez également obtenu les agendas de toutes les personnes visées par l'enquête (ces informations proviennent des interrogatoires, elles ne sont pas fiables). Ils se trouvent dans le répertoire `activites`, chaque ligne de fichier étant de la forme : `activité:heure`.

Écrivez le programme permettant d'établir la liste de suspects de la première partie de l'enquête, puis de déterminer qui est le coupable.