Python TP2 Les fichiers

Écrire dans un fichier

1/ Une fonction qui écrit une chaîne dans un fichier :

```
def writeToFile(s,filename):
                    pass
```

2/ Une fonction qui ajoute une chaîne à la fin d'un fichier :

```
def appendToFile(s,filename):
                    pass
```

3/ Une fonction qui écrit les valeurs de la fonction $f(x)=x^2+2x+4$ pour x compris entre -10 et 10 dans un fichier au format csv (chaque ligne de la forme x;f(x)) :

```
-10;84
-9;67
-8;52
….
8;84
9;103
10;124
```

```
def writeData(filename):
    pass
```

4/ La fonction précédente mais pour laquelle la fonction à calculer ainsi que les limites sont passées en paramètre :

```
def writeFunction(filename,function,mini,maxi):
                    pass
```

La fonction passée en paramètre doit retourner un réel et être définie avant (ou être importée).

Lire dans un fichier

5/ Une fonction qui compte les caractères d'un fichier :

```
def nbCarFile(filename):
    pass
```

6/ Une fonction qui compte les octets d'un fichier :

```
def nbBytesFile(filename):
    pass
```

7/ Une fonction qui compte les lignes d'un fichier :

```
def nbLinesFile(filename):
    pass
```

8/ Une fonction qui enlève dans une liste les éléments présent dans une autre :

```
def removeChar(firstList,toRemoveList):
        pass
```

Exemple :

```
firstList=['un','deux','cinq','trois','quatre','cinq'
,'six','','trois','quatre','']
toRemoveList=['trois','cinq','six']
```

Résultat : `['un','deux','quatre','six','quatre']`

Cette exercice ne concerne pas les fichiers mais sera utile pour la suite.
Pour la suite il faut aussi enlever la chaîne si elle est présente.

9/ Une fonction qui compte les mots d'un fichier texte.

```
def compteMots(filename):
    pass
```

On va travailler ligne après ligne. On considère que les mots sont séparés par des espaces. Un mot composé avec une apostrophe ou un tiret compte pour un seul mot. Un mot à cheval sur deux lignes sera compté pour deux mots. Il ne faut pas compter les éléments de ponctuations isolés (! ?:). S'ils restent accolés à un mot cela ne change pas le décompte des mots ('Monsieur,' 'fin.', 'OH!')

Algorithme :

Pour chaque ligne du fichier :
- enlever le '\n' à la fin de la ligne (voir la fonction **strip()**)
- créer une liste des éléments séparés par des espaces (utiliser **split()**)
- éliminer dans la liste tout ce qui n'est pas un mot (utiliser **removeChar**).
- accumuler la longueur de la liste → nombre de mots

10/ Une fonction qui enlève les ponctuations (!?, ;:.) éventuelles à la fin d'un mot :

```
def removePunct(mot):
    pass
```

(utile pour l'exercice suivant)

11/ Une fonction retourne un dictionnaire avec les mots du fichier comme clé et le nombre d'occurrence en valeur :

```
def dicoMots(filename) :
        pass
```

Exemple : `{ 'maître':8, 'corbeau':12, 'renard':11,....}`

La casse des mots ne doit pas intervenir : passer tous les mots en minuscule.
('Corbeau' = 'CORBEAU' = 'corbeau')
Il faut enlever la ponctuation des mots : pour que 'corbeau' et 'corbeau,' soit compté pour le même mots.

12/ Une fonction qui retourne la clé pour laquelle la valeur est maximale (dans un dictionnaire où les valeurs sont ordonables : entier, réel, chaîne de caractère,..).

```
def maxDico(d) :
    pass
```

Exemple : avec `{'maître':8, 'corbeau':12, 'renard':11}`
`la fonction retourne 'corbeau'`

Utiliser cette fonction et la précédente pour trouver le mot le plus présent dans un texte.

Le plus simple est de créer une liste avec les éléments (clé,valeur) du dictionnaire d : `items = list(d.items())`
Ensuite on travaille sur la liste de couples `items`