Examen Programmation avancée 2022_2023 SRI2A

Durée 1h30, Documents et machines autorisées, barême indicatif

1 Exercice 1

On a pour objectif de faire un dictionnaire de scrabble, à partir d'une liste de mots. Un dictionnaire de scrabble permet de trouver les mots faisables à partir d'un ensemble de lettres donné.

On va donc essayer de rentrer l'information sous forme d'un dictionnaire python, où les clefs seront des chaines avec les lettres, classées alphabétiquement. par exemple "aacdef" est une clef, à laquelle est associée le seul mot "facade". A la clef "cehin" sont associés les mots "chien", "chine" et "niche".

Vous pourrez utiliser les fonctions suivantes sur les chaines:

- len(s) donne la longueur d'une chaine. ex len("chien") -> 5
- sorted(s) renvoie une liste triée avec les lettres de la chaine s. ex sorted("chien") -> ["c","e","h","i","n"]
- c.join(liste) renvoie une chaine constitué des éléments de liste séparés par le caractère c. ex "-".join(["a","b","c"]) -> "a-b-c"

Questions (il n'est pas forcément besoin d'avoir écrit une fonction pour s'en servir ensuite):

1. Ecrivez une fonction **mot2clef** qui associe une clef triée à une chaine, par exemple mot2clef("chien") renvoie "cehin"

2. Ecrivez une fonction **faire_dico** qui prend une liste de mots en entrée et renvoie un dictionnaire de clef: pour chaque clef le résultat stocke la liste des mots qui correspondent à la clef.

```
Exemple si input = ["chien","chine","banane","niche","tri","rit"]

le résultat de faire_dict(input) pourrait être:

{"cehin":["chien","niche","chine"], "aabenn":["banane"], "irt":["tri","rit"]}
```

3. Faire une fonction **par_taille** qui prend en entrée un dictionnaire similaire à ceux renvoyés par faire_dico, et renvoie un dictionnaire de dictionnaires indexés par la longueur de leurs clefs.

Sur la sortie de l'exemple précédent on aurait donc comme résultat (attention à faire quelque chose de bien général si on avait par exemple tous les mots du dictionnaire en entrée) :

{5: {'cehin': ['chien', 'chine', 'niche']}, 6: {'aabenn': ['banane']}, 3: {'irt': ['tri', 'rit']}}

2 Exercice 2 : objets

On va définir une classe MultiSet qui représente un type d'ensemble où un élément peut être présent plusieurs fois. Par exemple:

$$\{(A,2),(B,4),(C,5)\}$$

(un multiset qui contient A 2 fois, B 4 fois, C 5 fois).

Vous pouvez choisir comment vous voulez stocker les informations dans la classe (attributs).

Ecrivez les méthodes suivantes:

- __init__ pour créer un multiset vide
- addElem pour ajouter un élément, avec le nombre de fois où il est présent
- add, qui fait l'ajout de 2 multisets: chaque élément est présent la somme de fois qu'il est présent dans les 2
- intersection: qui prend les éléments de 2 multisets, comptés avec le minimum des comptes dans les 2 multisets
- union: similaire à intersection, avec le max comme comptage.

3 Exercice 3: calcul vectoriel

Indice: en plus des opérations de base numpy, il sera utile de faire appel aux méthodes suivantes

- .reshape: méthode sur un vecteur/matrice/... numpy qui change les dimensions si le nombre d'éléments est compatible
- .means(axis=...) qui fait la moyenne des éléments d'une matrice selon la dimension donnée par axis. Par exemple 0 pour les "lignes", "1" pour les colonnes

Pour ces exercices, **vous ne devez faire aucun boucle**.

L'usage d'une boucle entraine 0 à la question

1. Définir une fonction qui crée une matrice numpy carrée NxN à partir d'un N donnée, avec les entiers de 1 à NxN

Par exemple carre(5) renverrait

```
```array([[ 1,  2,  3,  4,  5],
   [ 6,  7,  8,  9, 10],
   [11, 12, 13, 14, 15],
   [16, 17, 18, 19, 20],
   [21, 22, 23, 24, 25]])```
```

2. Ecrire une fonction qui prend une matrice et renvoie le vecteur des moyennes des 2 dernières colonnes.

Par exemple pour la matrice exemple de la question 1 le résultat serait :

```
```array([14., 15.])```
```

3. Ecrire une fonction qui calcule tous les termes de la suite U_n ci-dessous depuis n=1 à n=K et les renvoie dans un vecteur numpy:

$$u_n = (1 + \frac{1}{n})^n$$