TP 17
Traitement de tables de données

C'est la Saint-Valentin ! On en vient à parler couple, amour, naissance, choix du prénom pour le bébé… quoi de plus romantique que… d'écrire une programme Python pour manipuler la table de données de tous les prénoms de nouveau-nés donnés en France et déclarés à l'état civil entre 2000 et 2009 ?

I Introduction

Le fichier `materiel.zip` contient un fichier `prenoms.csv` avec toute l'information dont nous avons besoin. Dans sa forme actuelle il contient 110 605 lignes, on n'y verra donc pas grand chose avec un éditeur de texte. Les premières lignes du fichier sont les suivantes :

```
"sexe","prenom","annee","nombre"
1,AARON,2000,118
1,ABBAS,2000,7
1,ABD,2000,6
1,ABD-ALLAH,2000,6
1,ABDALLAH,2000,68
1,ABDEL,2000,65
...
2,EMMA,2004,6634
2,EMMA-JANE,2004,3
2,EMMA-LISA,2004,4
2,EMMA-LOU,2004,10
2,EMMA-LOUISE,2004,4
2,EMMA-ROSE,2004,5
2,EMMANUELA,2004,5
2,EMMANUELLA,2004,21
2,EMMANUELLE,2004,219
```

Cela représente un tableau à 4 colonnes, comme leur nom l'indique. La colonne `"sexe"` vaut en fait 1 pour les garçons et 2 pour les filles (l'état-civil ne reconnaît pas d'autre genre et ce n'est pas la question), la colonne `"prenom"` est en majuscule, la colonne `"annee"` est l'année de naissance et la colonne `"nombre"` le nombre de naissances avec ce prénom cette année.
Le fichier est partiellement classé par ordre alphabétique : il l'est pour chaque sexe et année. Un même prénom va donc revenir plusieurs fois, chaque année. Nous n'aurons pas à nous préoccuper de l'organisation interne du fichier car nous le traitons à travers des boucles Python.
Il s'agit d'un **fichier CSV**, qui permet de représenter un tableau ou une base de données à la structure très simple, facile à traiter par l'ordinateur et dans une certaine mesure lisible par un humain : des colonnes décrites dans une en-tête, et dans chaque ligne les données correspondantes sont séparées par des virgules. Le mot CSV lui-même signifie « Comma-Separated Values » soit littéralement… « valeurs séparées par des virgules ». On ne s'est pas pris la tête pour trouver un nom ! On appellera chaque ligne une **entrée** de la **table**.
Le code Python de démarrage ne fait qu'ouvrir ce fichier en utilisant la bibliothèque `csv` et le charge dans une grosse liste nommée `liste_prenoms` :

```
>>> len(liste_prenoms)
110604
```

La longueur est exactement un de moins que le nombre de lignes du fichier… à cause de l'en-tête.
Affichons un extrait en vrac de cette liste : par exemple, tout entre les indices entre 30 000 et 40 000 mais en sautant avec des pas de 1 000 ce qui devrait afficher 10 prénoms :

```
>>> liste_prenoms[30000:40000:1000]
[{'sexe': '1', 'prenom': 'DAVIDSON', 'annee': '2003', 'nombre': '4'},
 {'sexe': '1', 'prenom': 'IRWIN', 'annee': '2003', 'nombre': '8'},
 {'sexe': '1', 'prenom': 'MARVIN', 'annee': '2003', 'nombre': '334'},
```

```
{'sexe': '1', 'prenom': 'SAMET', 'annee': '2003', 'nombre': '37'},
{'sexe': '2', 'prenom': 'AIMY', 'annee': '2003', 'nombre': '33'},
{'sexe': '2', 'prenom': 'CYNTHIA', 'annee': '2003', 'nombre': '245'},
{'sexe': '2', 'prenom': 'IRENA', 'annee': '2003', 'nombre': '13'},
{'sexe': '2', 'prenom': 'LYANE', 'annee': '2003', 'nombre': '9'},
{'sexe': '2', 'prenom': 'NOALIE', 'annee': '2003', 'nombre': '3'},
{'sexe': '2', 'prenom': 'SWANA', 'annee': '2003', 'nombre': '3'}]
```

Comme chaque prénom a droit à sa propre entrée, une grande partie de cette liste est composée de prénoms rares, et toutes les variantes orthographiques ont aussi leur propre entrée ; les prénoms courant apparaissent une seule fois, mais avec un grand `nombre`.
Chaque entrée de cette liste est nommée en Python un **dictionnaire** — nous y reviendrons. Écrivons par exemple

```
>>> liste_prenoms[46090]
{'sexe': '2', 'prenom': 'EMMA', 'annee': '2004', 'nombre': '6634'}
```

alors on accède au prénom correspondant par `liste_prenoms[46090]["prenom"]`, à l'année par `liste_prenoms[46090]["annee"]` et de même pour les deux autres colonnes. On peut aussi faire rentrer cette donnée toute entière dans une variable, et alors on y accédera simplement :

```
>>> x = liste_prenoms[46090]
>>> x["sexe"]
'2'
>>> x["prenom"]
'EMMA'
>>> x["annee"]
'2004'
>>> x["nombre"]
'6634'
```

Cela se passe comme si chaque entrée était une petite liste, dont les indices ne sont pas des nombres mais sont les noms des colonnes avec lesquelles nous travaillons.
On n'hésitera donc pas à écrire dans une boucle `x = liste_prenoms[i]` pour y voir plus clair, voir à itérer directement sur la liste `for x in liste_prenoms` quand il n'est pas nécessaire de connaître l'indice.
Quelques dernières remarques :

1. Respectez les majuscules et les accents, dans les prénoms comme dans les intitulés de nos colonnes, cela a son importance. De plus, chaque orthographe correspond à une entrée différente. Les noms des colonnes sont en minuscule et sans accents. Les prénoms sont en majuscule et avec certains accents.
2. Toutes les données présentes sont des chaînes de caractères, de type `str`, y compris quand cela représente des nombres entiers. Pour accéder au nombre correspondant au prénom `x` il faut donc écrire `int(x["nombre"])`. Quant aux années, tant qu'on ne fait pas de calculs dessus, on peut les garder de type `str`, mais alors quand on donne une année il faut bien la mettre entre guillemets. Ainsi il faudra écrire des choses telles que `if x["annee"] == "2004"`. Le sexe lui est soit `"1"` soit `"2"`.
3. Attention car le mot *prénom* peut désigner à la fois la chaîne de caractère, mais aussi l'entrée entière de liste (la combinaison sexe, prénom, année, âge). Cela crée des confusions. On insiste donc sur l'utilisation du mot *entrée*. Chaque prénom apparaît dans plusieurs entrées (plusieurs année, parfois aussi plusieurs sexes) et il faut en tenir compte.
4. Ne demandez pas à afficher toute la liste d'un seul coup ! Ne tapez pas `print(liste_prenoms)` !!! (Si vous l'avez fait quand même : et bien vous étiez prévenus.)

II Chercher et compter

Exercice 1. Écrire une fonction `cherche(prenom)` qui prend en argument un prénom, parcourt toute la liste, et si le prénom est trouvé dans la liste, affiche toute l'entrée correspondante. Cette fonction ne renvoie rien, mais elle affiche avec `print`.

Testez-la sur votre prénom, bien entendu, et observez un peu la liste.

Exercice 2. Écrire une fonction `cherche_indice(prenom, annee, sexe)` qui prend en argument un prénom, une année et un sexe, qui parcourt les indices de la liste et renvoie l'indice de l'entrée à laquelle se trouve ce prénom, avec cette année et ce sexe. Une telle entrée est soit unique soit n'existe pas, auquel cas la fonction renvoie `-1`.

Testez cette fonction en vérifiant la cohérence avec la fonction précédente : si elle renvoie un indice `i` alors `liste_prenoms[i]` doit bien être une entrée qui apparaît dans la fonction précédente.
On rappelle que pour compter quelque chose il faut initialiser une variable qui compte, à 0, qui augmente de 1 sous la condition qu'on veut compter.

Exercice 3. Écrire une fonction `nombre_prenoms(annee)` qui compte le nombre total de prénoms donnés pour l'année.

Maintenant il faut compter en utilisant la colonne `nombre`.

Exercice 4. Écrire une fonction `compte(prenom)` qui prend en argument un prénom et compte combien de fois il a été donné, dans toute la liste (sur toutes les années).

Exercice 5. Écrire une fonction `total_naissances(annee)` qui prend en argument une année, et qui compte le nombre total d'enfants nés cette année.

III Filtrer

Exercice 6. Écrire une fonction `maximum(annee, sexe)` qui renvoie le prénom le plus donné, sur l'année et le sexe passés en argument.

On rappelle que pour renvoyer une liste de prénoms satisfaisant une certaine condition, il faut initialiser une liste vide puis utiliser des instructions `append`.

Exercice 7. Écrire une fonction `liste_compte_superieur(n, annee, sexe)` qui prend en argument un entier n et qui renvoie la liste de toutes les entrées dont le nombre est au moins égal à n, dans les années et sexes donnés. Testez-la avec une valeur de n proche de 10 000 (jusqu'à 10 000 pour l'année 2000, mais vers 7 000 si vous regardez 2009).

IV Choix du prénom

On s'intéresse maintenant au choix du prénom au hasard. Pour cela il ne s'agit pas simplement de choisir une entrée de la liste au hasard : on voudrait choisir un prénom de façon proportionnelle à sa fréquence d'apparition. Dans un premier temps on cherche parmi toutes les années et sexes confondus. Cela nécessite donc d'abord de compter le nombre de naissances totales, appelons le N. Ensuite on tire au hasard un nombre entre 1 et N. L'idée à traduire dans un algorithme, qui parcourt toute la liste est la suivante…
Imaginons qu'un premier prénom soit donné 3 fois, un deuxième est donné 8 fois, et le dernier est donné 2 fois. Cela fait 15 naissances, on prend un nombre au hasard entre 1 et 15. Alors on veut choisir le premier prénom si le nombre tiré est $1, 2, 3$, le deuxième si le nombre tiré est entre 4 et 12, et le troisième si le nombre tiré est $13, 14, 15$. Autrement dit dans une boucle on a besoin de compter les cumuls de naissances, et comparer le cumul avec le nombre tiré au hasard : dès que le cumul dépasse notre nombre choisi, on s'arrête et on considère qu'on choisit ce prénom !

Exercice 8. Écrire une fonction `choix_prenom()` qui choisit un prénom au hasard par cette méthode, et renvoie le prénom.

Testez cette fonction dans une boucle, et observez si cela pourrait ressembler aux prénoms de votre entourage !
Il est facile de modifier la fonction pour qu'elle donne un prénom uniquement selon un sexe donné, ainsi que proportionnellement à une année donnée.

V Représentation graphique

La fonction `plt.bar(X, Y)` de la bibliothèque `matplotlib.pyplot` permet de tracer un diagramme en barres (histogramme) qui sera adapté à afficher le nombre de fois qu'un prénom a été donné chaque année. La liste `X` sera la liste des années, et la liste `Y` sera celle des nombres. La documentation ou les aides-mémoires de `matplotlib` permettent d'améliorer un peu l'aspect du graphique : titre, noms des axes, couleurs des barres etc.

Exercice 9. Écrire une fonction `histogramme(prenom, sexe)` qui affiche un diagramme en barres du nombre de fois qu'un prénom a été donné en fonction de l'année.