MOI0201 – Projet Python 2023-2024

Les principaux objectifs de ce projet sont la lecture de fichiers au format `CSV`, la manipulation des dictionnaires et des listes ainsi que l'écriture de fichiers au format `CSV`.

Consignes générales

- Le but étant de programmer soi-même les fonctions qui vont vous être demandées, il est interdit d'utiliser d'autres bibliothèques que celles utilisées en TD/TP
- Les fonctions ne doivent pas nécessairement être le plus optimisées possible, simplement claires et fonctionnelles
- Les chemins d'accès aux fichiers doivent fonctionner sur n'importe quelle machine
- Vous êtes libre de créer des fonctions/procédures qui ne sont pas demandées si vous jugez cela pertinent
- Bien que cela ne soit pas obligatoire, il peut être judicieux de séparer vos codes dans différents fichiers (par exemple, un fichier qui contient les fonctions relatives aux `CSV`)
- **Il est impératif de commenter vos codes**

1 Lecture et affichage

Le but de cette première partie est la lecture (et l'affichage) des fichiers `Inventaire.csv` et `Recettes.csv`. Les questions qui suivent sont là pour vous guider mais vous pouvez tout à fait utiliser d'autre types de variables que ceux présentés dans les exemples. **Attention** : comme nous l'avons vu dans la partie tableur, les nombres sont en français. Il est donc indispensable de transformer le séparateur des décimales. Deux options s'offrent à vous, modifier directement dans un tableur ou bien écrire une petite fonction Python qui corrige cela et convertit une chaîne de caractère en réel (indication : utiliser la fonction `replace`).

1. Écrire une fonction `lireInventaire` qui prend en paramètre le chemin du fichier `Inventaire.csv` (donc une chaîne de caractères) et qui retourne le contenu du fichier dans un format adapté. Par exemple, un dictionnaire `inventaire` dont les clés sont les ingrédients et les valeurs sont des listes :

   `inventaire = {'Patates':[30,2,'23-févr',15],'Gruyère': ... }.`

   Dans cet exemple, l'entrée `Patates` correspond à la deuxième ligne du fichier, `Gruyère` à la troisième, etc

	A	B	C	D	E
1	Ingrédients	Quantité	Prix	Date de péremption	Minimale
2	Patates	27,75	2	23-févr	15
3	Gruyère	1,625	17	31-janv	1

2. Écrire une fonction `lireRecette` qui prend en paramètre le chemin du fichier `Recettes.csv` et qui retourne le contenu de celui-ci. Comme précédemment, libre à vous d'utiliser un format que vous trouvez adapté au contenu du fichier. Nous pouvons par exemple utiliser un dictionnaire `menu` qui contient lui-même des dictionnaires :

```
menu = {'Hachis parmentier': {'Ingrédients':['Gruyère',...],'Quantité':[0.0125,...], ...},
  'Chili con carne': {'Ingrédients':['Haricots rouges',...],'Quantité':[0.1,...], ...} ,
  'Quiche lorraine': {'Ingrédients':['Crème fraîche',...],'Quantité':[0.05,...], ...} }.
```

3. Écrire une procédure `afficheInventaire` (resp. `afficheMenu`) qui prend en paramètre la variable `inventaire` (resp. `menu`) et qui permet d'afficher proprement son contenu (voir l'exemple ci-dessous).

Ingrédients	Quantité	Minimale	Péremption
Patates	30.00	15.00	23-févr
Gruyère	2.00	1.00	31-janv
Beurre	2.00	2.00	31-janv
Farine	10.00	7.00	27-févr
Tomates	15.00	10.00	31-janv
Oignons	15.00	12.00	31-janv
Bœuf haché	25.00	21.00	31-janv
Ail	25.00	15.00	31-janv
Œufs	25.00	25.00	31-janv
Haricots rouges	25.00	18.00	07-juin
Crème fraîche	2.00	1.50	31-janv
Lardons	3.00	2.00	31-janv
Lait	10.00	8.00	13-févr

4. Écrire un script `loadCSV.py` qui permet de charger les fichiers `Inventaire.csv` et `Recettes.csv` et affiche l'inventaire et le menu en utilisant les fonctions précédentes. Pour rappel, la partie de votre script qui appelle les différentes fonctions et procédures commence par

```
if __name__ == '__main__':
```

2 Gestion du restaurant

1. Écrire une fonction `prendreCommande` qui prend en arguments le nom d'un plat présent dans le menu, l'inventaire et éventuellement un entier pour le nombre de commande. Cette fonction permet de décompter les ingrédients utilisés lorsqu'un plat est commandé et retourne l'inventaire mis à jour.
2. Écrire une fonction `remplissageInventaire` qui remplie l'inventaire en fin de journée en utilisant la valeur `minimale` qui provient du `CSV`. Si la quantité d'un ingrédient est inférieure à la quantité minimale, alors il faut en racheter et donc ajouter les dépenses dans la caisse. Cette fonction prend en arguments la variable `inventaire` et une variable `caisse`. et les retourne une fois actualisées. La variable `caisse` contient les gains bruts, le coût de fabrication des plats, les bénéfices et les dépenses.
3. Écrire une fonction `finDeJournee` qui permet de décompter tous les ingrédients utilisés pendant la journée et remplir l'inventaire si la quantité minimale est atteinte pour un ingrédient. Cette fonction prend en arguments une variable `commandes` (un dictionnaire ?), `menu`, `inventaire` ainsi qu'une variable `caisse` et retourne l'inventaire et la caisse en fin de journée.
4. Écrire un script `gestionRestaurant.py` qui simule la gestion de l'inventaire et de la caisse du restaurant pendant une journée. Votre script commence donc par importer l'inventaire et les différentes recettes, puis décompte/réapprovisionne l'inventaire et ajuste

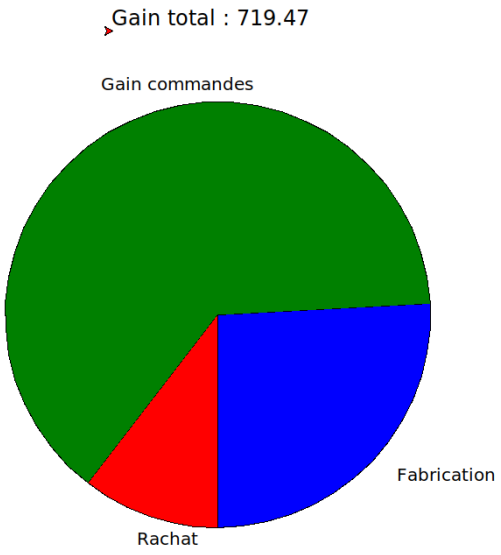
la caisse en conséquence. On pourra par exemple considérer une journée où les commandes sont :
— 30 hachis parmentier,
— 45 chili con carne,
— 16 quiches lorraine.

3 Écriture de fichier

1. Écrire une procédure `exportInventaire` qui prend en paramètre `inventaire` (éventuellement le chemin du nouveau fichier) et qui écrit un fichier au format `CSV` (et qui n'écrase pas `Inventaire.csv`). Vérifiez que votre fichier s'ouvre correctement avec un tableur.
2. Écrire un script `exportCSV.py` qui permet d'exporter l'inventaire au format `CSV` après la fin de journée. Votre script doit donc importer les différents `CSV`, simuler une journée (par exemple les commandes de l'étape précédente) et exporte l'inventaire en fin de journée.

4 Graphique avec Turtle

1. Écrire une procédure `graphCaisse` qui prend en argument la variable `caisse` et dessine un graphique (voir l'exemple ci-dessous) représentant les dépenses, les bénéfices et les gains totaux d'une journée. Libre à vous de dessiner un diagramme en bâtons ou un diagramme circulaire. Vous pouvez légender vos dessins en utilisant la fonction `write`.



2. Écrire un script `grapheRestaurant.py` qui permet de tracer les dépenses, les bénéfices et les gains sur une journée. Votre script doit donc importer les différents `CSV`, simuler une journée et tracer graphiquement le contenu de la caisse à la fin de la journée.