R1.01 Initiation au développement
Feuille de TP n°7
Les entrées/sorties

Objectifs de la feuille

Dans cette feuille vous allez travailler sur

- Les interactions avec l'utilisateur
- La sauvegarde et le chargement de données

Exercice 1 *Création d'un menu*

Le but de cet exercice est de permettre de créer un menu comme ci-dessous à partir de la liste des options du menu. Comme il n'est pas évident de faire des tests sur des fonctions d'affichage ou d'interaction avec l'utilisateur, les fonctions de test ne sont pas demandées dans cet exercice.

```
+-------------------------+
| MENU DE MON APPLICATION |
+-------------------------+
 1 -> Charger un fichier
 2 -> Rechercher la population d une commune
 3 -> Afficher la population d un département
 4 -> Quitter
Entrez votre choix [1-4]:
```

Pour cela on vous demande d'écrire les fonctions suivantes.

1.1 Écrire une fonction qui à partir d'un titre et d'une liste d'options données sous la forme d'une liste de `str` affiche le menu comme ci-dessus.

Attention la numérotation doit se faire automatiquement et la fonction n'affiche pas le message d'invite (Entrez ...).

1.2 Écrire une fonction qui prend en paramètre un message d'invite et un entier positif `borne_max` et affiche à l'utilisateur le message et attend sa réponse. La fonction doit retourner `None` si l'entrée de l'utilisateur est incorrecte. Sinon elle retourne le nombre entier compris entre 1 et `borne_max`

Pour cette fonction vous pouvez utiliser la méthode `isdecimal()` qui permet de tester si une `str` contient un nombre décimal ou utiliser le mécanisme d'exceptions de Python.

1.3 Écrire une fonction qui à partir d'un titre et d'une liste d'options données sous la forme d'une liste de `str` affiche le menu et demande à l'utilisateur sa réponse. La fonction retourne le numéro de l'option du menu choisie par l'utilisateur ou `None` si celui-ci a entré une valeur incorrecte.

1.4 Vous pouvez tester votre menu avec la fonction `programme_principal()` donnée dans le fichier `tp7_source.py`

Cette fonction est une boucle infinie qui après avoir défini les options du menu va

1. afficher le menu et attendre la réponse de l'utilisateur
2. traiter dans la conditionnelle la réponse de l'utilisateur
3. si le choix de l'utilisateur est l'option `Quitter`, l'instruction `break` permet de sortir de la boucle.

Exercice 2 *Population Française*

L'objectif de cette exercice est de créer une petite application permettant de consulter des informations sur la population française par commune. Cette application va utiliser des fichiers CSV issus des recensements effectués par l'INSEE. Ces fichiers contiennent les informations suivantes séparées par des points-virgules :

- `DEPCOM` qui est une chaîne de caractères dont les deux premiers caractères représentent le numéro de département de la commune et les suivants un numéro d'ordre de la commune.
  - ⚠ Notez que pour les DOM, le département est codé sur 3 caractères mais pour simplifier les traitements on ne tiendra pas compte de cette particularité.
  - ⚠ Notez aussi que le numéro de département est décimal sauf pour les deux départements corses (2A et 2B).
  - ⚠ Dans les fichiers les arrondissements des villes comme Paris, Lyon ou Marseilles seront considérées comme des communes à part entière. On ne tiendra pas compte de cette particularité.
- `COM` qui est le nom de la commune
- `PNUM` qui est la population municipale
- `PCAP` qui est le population comptée à part
- `PTOT` qui est la population totale de la commune

Pour notre programme seules les colonnes `DEPCOM`, `COM` et `PTOT` nous intéressent.

2.1 Écrire une fonction qui prend en paramètres un nom de fichier et qui produit la liste de tuples `(DEPCOM, COM, PTOT)` contenue dans le fichier.

Associés au TP vous avez des exemples de fichiers de l'INSEE (deux extraits et deux fichiers complets).

2.2 Modifier la fonction `menu_principal()` pour que lorsque l'utilisateur choisit la 1ere option on lui demande le nom du fichier qu'il souhaite charger, charge le fichier en stockant la liste de tuples dans une variable locale et affiche le nombre de communes trouvées.

📎 **ATTENTION ! ! !**

1. Dans la suite vous devez écrire les fonctions de tests associées à vos fonctions dans le fichier `test_tp7.py`
2. Toutes les fonctions sur les communes auront en paramètre une liste de tuples `(DEPCOM,COM,PTOT)`

2.3 Écrire une fonction qui permet de retrouver la population d'une commune en fonction du nom de la commune. Modifier la fonction `menu_principal()` pour permettre à l'utilisateur d'utiliser cette fonctionnalité.

2.4 Écrire une fonction qui permet de retrouver la liste des noms de commune qui commencent par une certaine chaîne de caractères dans une liste.

Par exemple si on veut la liste des noms de communes commençant par "Orl" on appellera

```
1    liste_noms = liste_des_communes_commencant_par(population, "Orl")
```

N'oubliez pas d'ajouter ce qu'il faut dans le programme principal pour permettre l'utilisation de cette fonctionnalité.

2.5 Écrire une fonction qui permet de connaître la commune la plus peuplée d'un département donné par son numéro.

Par exemple si on veut la commune la plus peuplée du Loiret on appellera

```
1    commune_top_= commune_plus_peuplee_departement(population, "45")
```

⚠ On rappelle que les numéros de département sont des chaînes de caractères

💡 La méthode `chaine.startswith("45")` indique si `chaine` débute par `"45"`

Intégrez cette fonctionnalité dans votre menu.

2.6 Écrire une fonction qui permet de retourner le nombre de communes qui appartiennent à une certaine tranche de population.

Par exemple si on veut le nombre de communes ayant entre 1000 et 2000 habitants on appellera

```
1    nb_communes = nombre_de_communes_tranche_pop(population, 1000, 2000)
```

Intégrez cette fonctionnalité dans votre menu.

2.7 Écrire une fonction qui à partir d'une liste de communes triées en ordre décroissant des populations et d'un tuple `(DEPCOM, COM, PTOT)` va ajouter cette commune au bon endroit dans la liste. Vous écrirez deux fonctions, celle déterminant le placement et celle insérant le tuple.

⚠ Le paramètre `taille_max` de la fonction `ajouter_trier` indique que la liste de communes ne doit pas avoir une longueur supérieure cette valeur.

2.8 Écrire une fonction qui permet de créer la liste des n communes les plus peuplées de France en utilisant la fonction précédente.

Intégrez cette fonctionnalité dans votre application.

2.9 Écrire une fonction qui permet de calculer le nombre total d'habitants dans chaque département en supposant que la liste de tuples est triée par département (c-à-d toutes les communes d'un même département sont toutes à la suite dans la liste).

Cette fonction doit retourner une liste de couples `(NUMDPT, POPDPT)`.

2.10 Écrire une fonction qui à partir d'un nom de fichier et d'une liste de couples `(NUMDPT, POPDPT)` va sauvegarder la liste de couples dans une fichier CSV.

Intégrez les deux fonctions précédentes dans une même option de votre application.