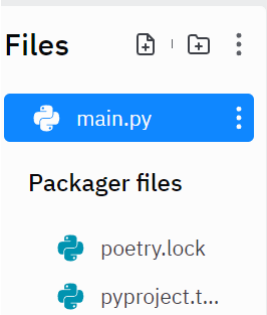
Activité traitement de données en tables

Afin d'explorer les possibilités offertes par le traitement de données en tables, nous allons utiliser le fichier countries.csv disponible dans le cahier de texte sur Pronote.

Afin d'avoir une base commune et de pouvoir utiliser certaines bibliothèques absentes des ordinateurs du lycée, nous allons utiliser le site replit.com pour écrire notre code.

Une fois sur replit, il sera nécessaire d'importer sur le site le fichier countries.csv



Pour importer un nouveau fichier, cliquez sur les trois petits points tout en haut à droite du menu de gauche, et cliquez sur "Upload file".

1 - Importer un fichier csv en Python

Pour charger le fichier au sein d'une structure de données lisible par Python, il va falloir ouvrir le fichier et le lire. On peut faire cela de différentes manières :

- En ouvrant le fichier nous-mêmes et en lisant les lignes une par une.

1 - Ouvrir le fichier countries.csv avec bloc-notes.
a) Ecrire la liste des descripteurs du fichier countries.csv. A quoi correspondent-ils comme données selon vous ?
b) Quel est le délimiteur utilisé pour séparer les données les unes des autres ?

2 - Ecrire le code suivant sur replit en remplacant le "X" par le bon délimiteur:

```
pays = []
with open('countries.csv', newline='') as csvfile:   # Ouverture d'un fichier
   reader = csv.DictReader(csvfile, delimiter='X')   # Création d'un lecteur de
fichier
   for row in reader:            # Pour chaque ligne du fichier
      pays.append(dict(row))     # Ajout de la ligne à notre liste sous forme de
dict
```

3 - Dans la console de replit, afficher le contenu de la variable "table" (`print(table)`)

a) Ecrire `print(table[0])` dans la console. Qu'est-ce qui s'affiche ? A quoi correspond donc une ligne de "table" ?

b) Ecrire print(`table[0][0]`) dans la console. Qu'est-ce qui s'affiche ? Quel est le problème ?

c) A quel type de variable correspond donc "table" ? Une liste ? Un dictionnaire ?...

d) Comment faire pour afficher le nom du pays contenu dans table[0] ? tester la solution sur la console.

- En utilisant une bibliothèque qui permet de faire cela automatiquement pour nous. Dans ce TD, nous utiliserons la bibliothèque csv.

```
import csv
f = open("countries.csv")   # ouverture du fichier
                            # "countries.csv" dans une var "f"
pays = list(csv.DictReader(f, delimiter = "X")) # création d'une var "table"
                                                # récupérant une liste de dicts
f.close()   # on referme le fichier après utilisation
```

Ces deux méthodes sont équivalentes. La deuxième méthode utilise juste une bibliothèque chargée de faire le travail à notre place.

Lire des données en table en Python

Requêtes de base

Maintenant que l'on possède les données dans notre programme, il est possible de "poser des questions" à notre jeu de données. Par exemple :

- Quelle est la liste des pays où l'on paye en euros ? on peut récupérer celle-ci en écrivant dans la console (ou dans notre programme) :

```
[p['Name'] for p in pays if p['Currency_Code'] == 'EUR']
```

Traduit en français, cela donne : "créer une liste en ajoutant les noms des pays existants dans la table `pays` si le descripteur `Currency_Code` correspondant contient la valeur "EUR".

La machine parcourt notre table et ne récupère que les lignes pour lesquelles le `Currency_Code` est "EUR". Pour ces lignes-ci, le programme récupère le champ `Name` correspondant et l'ajoute à une liste.

4 - Changer le code ci-dessus pour récupérer la liste des pays qui utilisent la monnaie 'Dollar'.

5 - Créer le code permettant de récupérer la liste des pays dont la population est de moins de 50000 habitants (attention, toutes les données chargées sont sous la forme de chaines de caractères. Pour pouvoir faire une comparaison entre la population et un chiffre, il faut transformer la population dans votre requêtes avec `int()`).

On peut récupérer plusieurs informations à la fois si on le souhaite :

```
[[p['Name'], p['Population']] for p in pays if p['Currency_Code'] == 'EUR']
```

Ce bout de code permet de récupérer la liste des pays payant en euros, ainsi que leur population.

Trier une table selon un critère

Pour pouvoir trier une table selon un critère, il est nécessaire d'utiliser une fonction renvoyant le critère de tri souhaité.

Par exemple, pour trier un pays selon sa population, il faut créer une fonction :

```
def get_population(p):
    return int(p['Population'])   # On doit transformer la population en entier
```

Il existe deux méthodes (fonctions) Python permettant de trier une table :

- sort() permet de trier notre liste (cela change l'ordre des éléments dans notre variable)
- sorted() crée une nouvelle liste et laisse notre liste initiale intacte.

6 - Tester le code suivant (n'oubliez pas d'écrire la fonction de l'exemple du dessus)

```
pays.sort(key=get_population)
```

a) Afficher la liste des pays. Dans quel ordre s'affichent les pays ?

Il est possible de trier une liste selon un unique critère dans l'ordre inverse :

b) Tester le code suivant et afficher la liste des pays. dans quel ordre s'affichent les pays ?

```
pays.sort(key=get_population, reverse=True)
```

7 - Trier la table pays en fonction de la superficie de chaque pays. N'oubliez pas de créer la fonction correspondante et de renvoyer le bon type de données (float).

Récupérer les premiers résultats d'une recherche

Parfois, on ne cherche qu'à récupérer que les premiers résultats d'une recherche (les 3 premiers, par exemple). Pour récupérer les 3 pays les plus grands (sans modifier notre liste d'origine), on procède ainsi :

```
sorted(pays, key=get_population)[:3]  # [:3] permet de récupérer les 3 premiers résultats
```

8 - Récupérer le nom et le continent des cinq derniers pays dans l'ordre alphabétique.

Manipuler des données en table en Python

L'utilisation des tables, en plus de permettre de récupérer facilement des informations, peut aussi permettre d'en générer de nouvelles.
Par exemple, dans notre fichier countries.csv, nous avons la population totale d'un pays ainsi que sa superficie. On pourrait donc déduire la densité de population de chaque pays.

La densité de population correspond au nombre d'habitants par kilomètre carré. La formule correspondante est donc, dans le cadre de notre fichier :
`population / superficie`

Pour ajouter une information à notre structure de données, il suffit de la parcourir et de rajouter l'information à chaque ligne. Si l'on souhaitait rajouter un descripteur "toto" contenant la valeur "coucou" à chacun de nos pays, il suffirait d'écrire :

```
for p in pays:
    p['toto'] = "coucou"
```

9 - Rajouter une clé "densite" à la table ayant pour valeur pour chaque pays sa population divisée par sa superficie.

Exporter un fichier csv en Python

Maintenant que nous avons modifié notre table, nous pouvons vouloir garder cette information en l'enregistrant dans un fichier. L'exemple ci-dessous écrit dans le fichier "monfichier.csv" le résultat d'une table appelée "table" selon la liste de ses descripteurs "liste_descripteurs".

```
fichier = open("monfichier.csv", "w")  # w : write, on ouvre le fichier en mode
écriture
liste_descripteurs = table[0].keys()   # astuce permettant de récupérer la liste
des
                                       # descripteurs
w = csv.DictWriter(fichier, liste_descripteurs)
w.writeheader()
w.writerows(table)
```

10 - Ecrire dans un fichier "countries2.csv" le contenu de la table "pays".

11 - Ouvrir le fichier et vérifier que celui-ci est correct.