Thème : Traitement des données

19 Manipulation de fichiers CSV

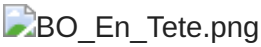
> Les fichiers **CSV (pour Comma Separated Values)** sont des fichiers-texte (ils ne contiennent aucune mise en forme) utilisés pour stocker des données, séparées par des virgules (ou des points-virgules, ou des espaces...). Il n'y a pas de norme officielle du CSV.

0.0.1 Table des matières

1. Ouverture d'un fichier CSV par des logiciels classiques

- Ouvrir avec le Bloc-Notes le fichier `Les colleges_Puy_De_dome_2022.csv`.
- Ouvrir le fichier avec LibreOffice.

2. Exploitation avec Python



2.1 Exercice n°1 : Exploitation d'un fichier CSV en Python avec le module CSV

L'utilisation d'un tableur peut être délicate lorsque le fichier CSV comporte un très grand nombre de lignes. Python permet de lire et d'extraire des informations d'un fichier CSV même très volumineux, grâce à des modules dédiés, comme le bien-nommé `csv` (utilisé ici) ou bien `pandas` (qui sera vu plus tard).

```
import csv
f = open('Les_colleges_Puy_De_Dome_2022.csv', "r", encoding = 'utf-8') #
le "r" signifie "read", le fichier est ouvert en lecture seule
donnees = csv.reader(f)  # donnees est un objet (spécifique au module
csv) qui contient des lignes

for ligne in donnees:
    print(ligne)

f.close()    # toujours fermer le fichier !
```

2.1.1 Problèmes :

1. Les données ne sont pas structurées : la première ligne est la ligne des «descripteurs» (ou des «champs»), alors que les lignes suivantes sont les valeurs de ces descripteurs.
2. La variable `donnees` n'est pas exploitable en l'état.

2.1.2 Améliorations

Au lieu d'utiliser la fonction `csv.reader()`, utilisons `csv.DictReader()`. Comme son nom l'indique, elle renverra une variable contenant des dictionnaires.

```
import csv
f = open('Les_colleges_Puy_De_Dome_2022.csv', "r", encoding = 'utf-8')
donnees = csv.DictReader(f)

for ligne in donnees:
    print(ligne)

f.close()
```

C'est mieux ! Les données sont maintenant des dictionnaires. Comment ré-accéder au premier d'entre eux ?

```
donnees[0]
```

Erreur normal, nous allons donc créer une liste de dictionnaires.

```
import csv
global colleges
f = open('Les_colleges_Puy_De_Dome_2022.csv', "r", encoding = 'utf-8')
donnees = csv.DictReader(f)
colleges = []
for ligne in donnees:
    colleges.append(dict(ligne))


descripteurs=colleges[0]  # on stocke les descripteurs dans une variable

colleges.pop(0)   # on enlève les descripteurs

f.close()    # toujours fermer le fichier
```

```
descripteurs
```

```
print(colleges[1]['Nom du Canton'])
```

```
print(colleges[2]['Nom'])
```

```
print(colleges[2]['Nom du (de la) Principal(e)'])
```

Question 1 : Compléter la fonction `plus_grand_college(colleges)` qui renvoie la commune, le nom du collège et le nom du (de la) principal(e)

```
def plus_grand_college():
    maxi=0
    nom=''
    commune=''
    principal=''
    for col in ...:
        if int(...)>maxi:
            maxi=...
            nom=...
            commune=...
            principal=...
    return nom,commune,principal,maxi

plus_grand_college()
```

Question 2 : Ecrire une fonction `plus_petit_college(colleges)` qui renvoie la commune, le nom du collège et le nom du (de la) principal(e)

```
def plus_petit_college():
    mini=1E99
    nom=''
    commune=''
    principal=''
    pass
    return nom,commune,principal,mini

plus_petit_college()
```

Question 3 : Ecrire une fonction `moyenne()` qui renvoie le nombre d'èléves moyen dans les collèges.

```
def moyenne():
    somme=0


round(moyenne(),1)
```

Question 4 : Ecrire une fonction `renseignement(college)` qui renvoie les renseignements sur le collége choisi.

```
def renseignement(college):
    for ... in ...:
        if ...==...:
            return col
    return ...
```

```
renseignement('Collège I. et F. Joliot Curie')
```

Question 5 : Ecrire une fonction `renseignement(commune)` qui renvoie les renseignements sur le(s) colléges de la commune ainsi que leur nombre.

```
def renseignement_commune(commune):
    rep=[]
    pass
    return nb,rep

renseignement_commune('ISSOIRE')
```

2.2 Exercice n°2 : les joueurs de rugby du TOP14

- Ouvrir avec le Bloc-Notes le fichier `Les colleges_Puy_De_dome_2022.csv`.
- Ouvrir le fichier avec LibreOffice.

Le fichier `Top14.csv` contient tous les joueurs du Top14 de rugby, saison 2019-2020, avec leur date de naissance, leur poste, et leurs mensurations.

Ce fichier a été généré par Rémi Deniaud, de l'académie de Bordeaux.

Question 1. Stocker dans une variable `joueurs` les renseignements de tous les joueurs présents dans ce fichier csv.

```
import csv
f = open('data/Top14.csv', "r", encoding = 'utf-8')
donnees = csv.DictReader(f)
joueurs = []
for ligne in donnees:
    joueurs.append(dict(ligne))

f.close()
```

Question 2. Combien de joueurs sont présents dans ce fichier ?

```
len(joueurs)
```

Question 3. Quel est le prénom du joueur n°486 ?

```
joueurs[486]['Nom']
```

Question 4. Ecrire une fonction permettant d'obtenir le numero d'un joueur connaissant son nom.

```
def numero(nom):
    pos=0
    for n in joueurs:
        if n['Nom']==nom:
            return pos
        pos+=1

numero('SERIN Baptiste')
```

```
89
```

2.2.1 Extraction de données particulières

Question 5. Ecrire une fonction `clubJoueur(joueur)` permettant de savoir où joue un joueur ?
A appliquer avec Baptiste SERIN ?
La méthode la plus naturelle est de parcourir toute la liste jusqu'à trouver le bon joueur, puis d'afficher son équipe :

```

```

Une méthode plus efficace est d'utiliser une liste par compréhension incluant un test.
Rappel :

```
exemple = [k for k in range(1,50) if k % 3 == 0]
print(exemple)
```

Procéder de même pour retrouver le club de Baptiste SERIN.

```
clubSerin = [k['Club'] for k in joueurs if k['Nom'] == 'SERIN Baptiste']
print(clubSerin)
```

Question 6. Ecrire une fonction `joueurPoid(poid)` qui renvoie la liste des joueurs de plus de poid kg ?
A tester avec les joueurs de plus de 140 kg

```

```

Il est impératif de convertir la chaine de caractère renvoyée par `k['Poids']` en entier, par la fonction `int()`.

Question 7. Ecrire une fonction `taille_moyenne(poste)` renvoyant la taille moyenne des joueurs en fonction du poste.

Question 8. Ecrire une fonction `poid_moyenne(poste)` renvoyant le poid moyen des joueurs en fonction du poste.

Question 9. Ecrire une fonction renvoyant la taille et le poid moyen des joueurs par équipe.

2.2.2 Exploitation graphique

Nous allons utiliser le module Matplotlib pour illustrer les données de notre fichier csv.

2.2.3 Exemple

```
import matplotlib.pyplot as plt
plt.close()
X = [0,1,3,6]
Y = [12,10,7,15]
plt.plot(X,Y,'ro') # r pour red, o pour un cercle. voir
https://matplotlib.org/api/markers_api.html
plt.show()
```

2.2.4 Application

Question 10. Afficher sur un graphique tous les joueurs de rugby du top14, en mettant le poids en abscisse et la taille en ordonnée.

```
plt.close()
X=[]
Y=[]
for k in joueurs:
    p=int(k['Poids'])
    X.append(p)
    t=int(k['Taille'])
    Y.append(t)
```

```
plt.plot(X,Y,'ro') # r pour red, o pour un cercle. voir
https://matplotlib.org/api/markers_api.html
plt.show()
```

Question 11. Faire apparaître ensuite les joueurs évoluant au poste de Centre en bleu (b), les 2ème lignes en vert (g), les demis de Mêlée en magenta (m), les Piliers en jaune et les Arrières en cyan (c).

```
plt.close()
#tous les joueurs
X = [int(k['Poids']) for k in joueurs]
Y = [int(k['Taille']) for k in joueurs]
plt.plot(X,Y,'ro')

#on recolorie les Centres en bleu
X = [int(k['Poids']) for k in joueurs if k['Poste'] == 'Centre']
Y = [int(k['Taille']) for k in joueurs if k['Poste'] == 'Centre']
plt.plot(X,Y,'bo')

#on recolorie les 2ème ligne en vert
X = [int(k['Poids']) for k in joueurs if k['Poste'] == '2ème ligne']
Y = [int(k['Taille']) for k in joueurs if k['Poste'] == '2ème ligne']
plt.plot(X,Y,'go')

#on recolorie les Pilier en jaune
X = [int(k['Poids']) for k in joueurs if k['Poste'] == 'Pilier']
Y = [int(k['Taille']) for k in joueurs if k['Poste'] == 'Pilier']
plt.plot(X,Y,'yo')

#on recolorie les demi de Mêlée en magenta
X = [int(k['Poids']) for k in joueurs if k['Poste'] == 'Mêlée']
Y = [int(k['Taille']) for k in joueurs if k['Poste'] == 'Mêlée']
plt.plot(X,Y,'mo')

#on recolorie les Arrières en cyan
X = [int(k['Poids']) for k in joueurs if k['Poste'] == 'Arrière']
Y = [int(k['Taille']) for k in joueurs if k['Poste'] == 'Arrière']
plt.plot(X,Y,'co')
plt.show()
```

3. Trier des données

3.0.5 Créer une fonction filtre

Question 12.
L'objectif est de créer une fonction `joueursClub(equipe)` qui renvoie une liste contentant tous les joueurs de l'équipe `equipe`.

Le paramètre `equipe` sera donnée sous forme de chaîne de caractères.
La valeur renvoyée sera de type liste.

```
def joueursClub(equipe):
    ret = []

    for k in joueurs :
        if k['Club'] == equipe :
            ret.append(k)

    return ret

joueursClub("Clermont")
```

Question 13.
Ecrire une fonction `nomJoueursClub(equipe)` qui cette fois ne renvoie que les noms des joueurs pour le club.

```
def nomJoueursClub(equipe):
    rep=joueursClub(equipe)
    jrs=[]
    for k in rep:
        jrs.append(k['Nom'])


    return jrs

print(len(nomJoueursClub("Clermont")))
nomJoueursClub("Clermont")
```

Question 14.
Définir de la même manière une fonction `joueursPoste(poste)`. pour l'ensemble des joueurs puis écrire une fonction pour le nom des joueurs club par poste.

```

```

3.1 Utilisation d'une fonction de tri

Comment classer les joueurs suivant leur taille ?
La fonction `sorted(liste)` est efficace sur les listes : elle renvoie une nouvelle liste triée dans l'ordre croissant.

```
mylist = [4,2,8,6]
mynewlist = sorted(mylist)
print(mynewlist)
```

```
[2, 4, 6, 8]
```

Mais comment trier un dictionnaire ?

```
test = sorted(joueurs)
```

```
Traceback (most recent call last):
  File "<input>", line 1, in <module>
TypeError: '<' not supported between instances of 'dict' and 'dict'
```

Il est normal que cette tentative échoue : un dictionnaire possède plusieurs clés différentes. Ici, plusieurs clés peuvent être des critères de tri : la taille, le poids.

3.1.1 un exemple de tri de dictionnaire

```
Simpsons = [{"Prenom" : "Bart", "age estimé": "10"},
            {"Prenom" : "Lisa", "age estimé": "8"},
            {"Prenom" : "Maggie", "age estimé": "1"},
            {"Prenom" : "Homer", "age estimé": "38"},
            {"Prenom" : "Marge", "age estimé": "37"}]
```

```
def age(personnage):
    return int(personnage["age estimé"])
```

```
age(Simpsons[0])
```

```
10
```

La création de cette fonction `age()` va nous permettre de spécifier une clé de tri, par le paramètre `key` :

```
triSimpsons = sorted(Simpsons, key = age)
```

```
triSimpsons
```

```
triSimpsons = sorted(Simpsons, key = age, reverse = True)
```

```
triSimpsons
```

Question 15. Trier les joueurs du top14 par taille.
Adapter ensuite pour afficher que certains renseignements.

```
def taillePlayer(player) :
    return int(player['Taille'])
```

```
joueurs_taille_croissant = sorted(joueurs, key = taillePlayer)
print(joueurs_taille_croissant)
```

```
for nom in  joueurs_taille_croissant:
    print(nom['Nom'],nom['Taille'])
```

```

```

Question 16. Trier les joueurs de Clermont par taille.

```

```

Question 17.
Trier les joueurs de Clermont suivant leur Indice de Masse Corporelle (IMC)

```

```

Recherche des joueurs de profil physique similaire

3.1.2 Distance entre deux joueurs

Construire une fonction `distance(joueur1,joueur2)` qui renvoie la somme des carrés des différences de tailles et de poids entre les joueurs `joueur1` et `joueur2` : $$ d = (p_1-p_2)^2 + (t_1-t_2)^2$$

```
def distance(joueur1,joueur2):
    p1 = int(joueur1['Poids'])
    p2 = int(joueur2['Poids'])
    t1 = int(joueur1['Taille'])
    t2 = int(joueur2['Taille'])
    return (p1-p2)**2+(t1-t2)**2

def distanceJ(joueur1,joueur2):
    n1=numero(joueur1)
    n2=numero(joueur2)
    d=distance(joueurs[n1],joueurs[n2])
    return d
```

3.1.3 Distance des joueurs avec Baptiste Serin

Retrouvons d'abord le numéro de Baptiste Serin et de joueur2 dans notre classement de joueurs. (Fonction déja faite)

```
distanceJ('PARRA Morgan','SERIN Baptiste')
```

```
16
```

Pour pouvoir utiliser le tri dans un dictionnaire, il faut intégrer la recherche du numero dans la fonction

```
def IMC(player):
    masse = int(player['Poids'])
    taille_m = int(player['Taille']) / 100
    return masse / taille_m**2

joueursASM = [k for k in joueurs if k['Club'] == 'Clermont']
joueursASM

def IMCJoueur(player):
    n1=numero(player)
    j=joueurs[n1]
    imc=IMC(j)
    return imc


joueursASM_tri = sorted(joueurs, key = IMC)


def distance(joueur1,joueur2):
    p1 = int(joueur1['Poids'])
    p2 = int(joueur2['Poids'])
    t1 = int(joueur1['Taille'])
    t2 = int(joueur2['Taille'])
    return (p1-p2)**2+(t1-t2)**2



def distanceSerin(joueur2):
    p1 = int(joueurs[89]['Poids'])
    p2 = int(joueur2['Poids'])
    t1=int(joueurs[89]['Taille'])
    t2 = int(joueur2['Taille'])
    return (p1-p2)**2+(t1-t2)**2
```

```
joueurs_VS_Serin = sorted(joueurs, key=distanceSerin)
```

```
joueurs_VS_Serin
```

```
for j in joueurs_VS_Serin:
    print(j['Nom'],distanceSerin(j))
```

```

```