TD — EXTRACTION DE DONNÉES D'UN FICHIER CSV ET TRI VIA PYTHON

Lien : https://icn-isn-boissy.yj.fr/wp/2019/12/12/extraction-de-donnees-dun-fichier-csv-et-tri-simple-sur-une-liste/

Voici mon code python complet qui traite toutes les activités de la première jusqu'au Bonus avec la prise en charge de l'écriture dans un nouveau fichier :

```
# Auteur         : Romain MELLAZA
# Date           : 19/03/2022
# Description    : Programme qui réalise plusieurs fonctions sur un
fichier csv donné : importation, conversion, manipulation de
données, ré-emploi de données pour un classement plus spécifique

import csv

csvfile="csv\liste_origine.csv"    # Définition de l'emplacement
du fichier .csv étudié

def import_lignes(file,count):
    '''
     Fonction qui prend en argument un fichier et qui renvoie la
     liste des lignes du fichier, c'est-à-dire une liste de
     chaînes de caractères, où chaque élément de la liste
     correspond à une ligne du fichier.
    '''
    lines1=[]
    lines2=[]
    for_count=count
    for_count_2=0
    f=open(file)
    lines1=f.readlines
    for lines1 in f:
        lines2=lines1
        for_count_2+=1
        if for_count_2 == for_count:
            break
    f.close()
    count+=1
```

```
    ###
    # Le fonctionnement de cette fonction est assez complexe mais
    # pour faire simple je lis ligne par ligne le fichier csv
    # importé en sachant que j'ai mis plusieurs compteurs pour
    # être sûr de ne pas dépasser le nombre de colonnes et de
    # lignes.
    ###
    return lines2

def separe_lignes(listecsv):
    '''
    Fonction qui prend en argument la chaîne de caractère créée
    par la fonction "import_lignes" et qui renvoie une liste
    contenant les chaînes de caractères correspondants aux champs
    de notre fichier.
    '''
    lines_aft=[]
# Je définis une liste qui va me permettre de recueillir les
données après modifications...
    listecsv= listecsv.replace(u'\xa0', u' ')
# Je supprime des balises éventuelles dû à l'encodage
    worked=str(listecsv)
# Je convertis la liste en chaîne de caractères, car certaines
opérations ne sont possibles que pour ce type de valeurs
    worked=worked.strip()
# Je supprime des espaces inutiles à la fin de la chaîne de
caractère.
    worked=worked.split(";")
# Je sépare la ligne à chaque fois qu'un point-virgule est
rencontré
    lines_aft=worked
# Je remets la chaîne de caratère maintenant exploitable dans une
liste.
    return lines_aft

def creation_dico(lines_for_dico):
    '''
    Fonction qui crée un dictionnaire qui contient toutes les
    données que j'ai définies (Classement; Nom; Date; Temps)
    et cela pour chaque cycliste !
    '''
    dico1={'Classement':lines_for_dico[0], 'Nom du
cycliste':lines_for_dico[1], 'Date':lines_for_dico[2], 'Temps en
secondes':lines_for_dico[8]}
    return dico1
```

```
def classement_annee(cycliste_etudie,annee_select):
    '''
     Fonction qui prend en argument la liste de dictionnaire
     précédemment créée et qui renvoie une nouvelle liste
     contenant uniquement les performances de l'année sélectionnée
     par l'utilisateur via le paramètre "annee_select".
    '''
    cycliste_list_for_slice=[]
# Je définis une liste qui va me permettre de recueillir les datez
de performance sans le "/"

    for i in cycliste_etudie:
        cycliste_list_for_slice+=i.split("/")
# Je sépare les années du reste de la date
    annee_search=str(annee_select)
# Je convertis l'année saisie par l'utilisateur en chaîne de
caractères afin de la rechercher par la suite dans les dicos.

    if annee_search in cycliste_list_for_slice:
# Je regarde si le dictionnaire contient l'année recherchée par
l'utilisateur

        return cycliste_list_for_slice
# Si c'est le cas, il renvoie le dictionnaire du cycliste car cela
signifie que la performance date bien de l'année recherchée.

    else:

        pass
# Si ce n'est pas le cas, il ne renvoie rien.

def nouveau_fichier(leaderboard,keys_for_csv):
    '''
     Fonction qui permet a l'utiliateur de créer un nouveau
     fichier csv avec seulement les performances de l'année qu'il
     a sélectionné.
     Les performances sont reclassées en fonction de cette année
     en particulier.
    '''
    columns = ['Classement', 'Nom du cycliste', 'Date', 'Temps en
secondes']
# Définition des colonnes que le fichier csv doit garder
    keys = keys_for_csv
# Importation des clées qui identifient l'ordre dans lequel les
valeurs du dictionnaire sont écrites
```

```
    with open('csv\classement_user.csv', 'w', newline='',
encoding='UTF8') as output_file:
# Je crée le nouveau fichier classement_user.csv et je le définis
comme fichier d'écriture en sortie
        writer = csv.DictWriter(output_file, fieldnames=columns)
# J'utilise la fonction DictWriter pour écrire le dictionnaire qui
contient les données voulues

# j'importe aussi l'entête des colonnes définies précédemment.
        writer.writeheader()
# J'écris l'entête des colonnes
        for keys in leaderboard:
            writer.writerow(keys)
# J'écris les différents dictionnaires ligne par ligne (clé par
clé)

    output_file.close()
# Je ferme le fichier csv après les différentes modifications


annee_user=int(input("Saisissez l'année du classement voulue :\
n"))
# Je demande à l'utilisateur quelle année il veut voir en
particulier

liste_perf = []
# Je définis deux listes vides qui contiendront des donnée par la
suite
liste_for_user=[]

classement_user=1
# Je définis que le classement annuel commencera à la première
place ("1") et pas ("0") comme il le ferait par défaut
compteur = 2
# Je définis le compteur de lignes/colonnes sur 2 pour que le
programme ne prenne pas l'entête pour un cycliste !

while compteur < 30 :
# Je définis une boucle jusqu'à la dernière ligne du fichier
    lines_bef=import_lignes(csvfile,compteur)
    print("\n")
    lines_comp=separe_lignes(lines_bef)
    dico_coureur=creation_dico(lines_comp)
```

```
    liste_perf.append(dico_coureur)
# Après avoir exécuté les différentes fonctions, j'ajoute les
dictionnaires de perfs. de chaque cycliste dans une liste
    classement_annuel=classement_annee(lines_comp,annee_user)
    if classement_annuel != None :
# Comme on l'a vu précédemment ma fonction renvoie une donnée
seulement quand la performance a lieu dans l'année voulue, donc
ici je n'exécuterai le code suivant seulement s'il y a bien un
résultat obtenu pour être sûr et certain.
        print("---------------------------------- Ce cycliste a
réalisé une perf. en",annee_user,":
----------------------------------")
        print(lines_comp)
        print(dico_coureur)
        temp_var=dico_coureur
# Je mets le dico. du coureur dans une variable temporaire.
        temp_var["Classement"] = classement_user
# Je change la valeur de la clé classement pour que le classement
soit seulement sur une année et pas globale comme avant
        liste_for_user.append(temp_var)
# Je remplie la liste qui contient tous les dictionnaires des
performances de l'année recherchées par l'utilisateur
        classement_user+=1
# J'incrémente le classement des performances annuelles
    else:
        print(lines_comp)
# Sinon je n'affiche que les performances d'autres années pour les
cyclistes qui n'ont pas réalisé de performances l'année voulue.
        print(dico_coureur)
    compteur+=1
    print("\n"*4)


print("Liste contenant les dictionnaires des résultas des
cyclistes :")
# Ici, j'affiche une liste des dictionnaires de TOUS les coureurs
du fichier csv
print('\n')
print(liste_perf)

keys = liste_for_user[0].keys()
nouveau_fichier(liste_for_user, keys)
# J'appelle ma fonction pour créer un nouveau fichier avec
seulement les résultats de l'année voulue et classé par rapport
aux perfs. de cette année en particulier.
```

Pour rendre l'utilisation de mon programme plus simple et plus détaillée que sur ce compte-rendu, je fournis ci-joint dans le fichier zip le programme en .py ainsi qu'un dossier contenant tous les fichiers csv étudiées et utilisées pour l'extraction, le tri et enfin l'écriture !

Voici quelques exemples de résultats que l'on peut obtenir en sortie avec mon programme :

```
Saisissez l'année du classement voulue :
2019


['1', 'Laurens ten Dam', '14/07/2013', '20,0km/h', '166bpm', '388W', '1 547,6', '00:58:17', '3497']
{'Classement': '1', 'Nom du cycliste': 'Laurens ten Dam', 'Date': '14/07/2013', 'Temps en secondes': '3497'}




---------------------------------- Ce cycliste a réalisé une perf. en 2019 : ----------------------------------
['2', 'Romain Bardet', '17/06/2019', '19,9km/h', '-', '-', '1 539,7', '00:58:35', '3515']
{'Classement': '2', 'Nom du cycliste': 'Romain Bardet', 'Date': '17/06/2019', 'Temps en secondes': '3515'}




---------------------------------- Ce cycliste a réalisé une perf. en 2019 : ----------------------------------
['3', 'Rein Taaramae', '17/06/2019', '19,5km/h', '164bpm', '353W ', '1 510,4', '00:59:43', '3583']
{'Classement': '3', 'Nom du cycliste': 'Rein Taaramae', 'Date': '17/06/2019', 'Temps en secondes': '3583'}




---------------------------------- Ce cycliste a réalisé une perf. en 2019 : ----------------------------------
['4', 'El Fares Julien', '17/06/2019', '19,4km/h', '169bpm', '397W ', '1 500,4', '01:00:07', '3607']
{'Classement': '4', 'Nom du cycliste': 'El Fares Julien', 'Date': '17/06/2019', 'Temps en secondes': '3607'}




---------------------------------- Ce cycliste a réalisé une perf. en 2019 : ----------------------------------
['5', 'Elie Gesbert', '17/06/2019', '19,2km/h', '-', '331W ', '1 482,3', '01:00:51', '3651']
{'Classement': '5', 'Nom du cycliste': 'Elie Gesbert', 'Date': '17/06/2019', 'Temps en secondes': '3651'}




---------------------------------- Ce cycliste a réalisé une perf. en 2019 : ----------------------------------
['6', 'Javi Moreno', '17/06/2019', '19,1km/h', '-', '-', '1 479,5', '01:00:58', '3658']
{'Classement': '6', 'Nom du cycliste': 'Javi Moreno', 'Date': '17/06/2019', 'Temps en secondes': '3658'}




---------------------------------- Ce cycliste a réalisé une perf. en 2019 : ----------------------------------
['7', 'Pierpaolo Ficara', '17/06/2019', '19,1km/h', '-', '350W ', '1 475,0', '01:01:09', '3669']
{'Classement': '7', 'Nom du cycliste': 'Pierpaolo Ficara', 'Date': '17/06/2019', 'Temps en secondes': '3669'}




---------------------------------- Ce cycliste a réalisé une perf. en 2019 : ----------------------------------
['8', 'Oscar Rodriguez', '17/06/2019', '19,1km/h', '-', '315W ', '1 474,2', '01:01:11', '3671']
{'Classement': '8', 'Nom du cycliste': 'Oscar Rodriguez', 'Date': '17/06/2019', 'Temps en secondes': '3671'}
```

```
Liste contenant les dictionnaires des résultas des cyclistes :


[{'Classement': '1', 'Nom du cycliste': 'Laurens ten Dam', 'Date': '14/07/2013', 'Temps en secondes': '3497'}, {'Classement': 1, 'Nom du cycliste': 'Romain Bardet', 'Date': '17/06/2019'
, 'Temps en secondes': '3515'}, {'Classement': 2, 'Nom du cycliste': 'Rein Taaramae', 'Date': '17/06/2019', 'Temps en secondes': '3583'}, {'Classement': 3, 'Nom du cycliste': 'El Fares
Julien', 'Date': '17/06/2019', 'Temps en secondes': '3607'}, {'Classement': 4, 'Nom du cycliste': 'Elie Gesbert', 'Date': '17/06/2019', 'Temps en secondes': '3651'}, {'Classement': 5, '
Nom du cycliste': 'Javi Moreno', 'Date': '17/06/2019', 'Temps en secondes': '3658'}, {'Classement': 6, 'Nom du cycliste': 'Pierpaolo Ficara', 'Date': '17/06/2019', 'Temps en secondes':
'3669'}, {'Classement': 7, 'Nom du cycliste': 'Oscar Rodriguez', 'Date': '17/06/2019', 'Temps en secondes': '3671'}, {'Classement': '9', 'Nom du cycliste': 'Julien Antomarchi', 'Date':
'08/06/2017', 'Temps en secondes': '3685'}, {'Classement': '10', 'Nom du cycliste': 'thomas rostollan', 'Date': '18/06/2015', 'Temps en secondes': '3726'}, {'Classement': 8, 'Nom du cyc
liste': 'Adrien Guillonnet', 'Date': '17/06/2019', 'Temps en secondes': '3757'}, {'Classement': 9, 'Nom du cycliste': 'tobias ludvigsson', 'Date': '17/06/2019', 'Temps en secondes': '37
98'}, {'Classement': 10, 'Nom du cycliste': 'Romain Hardy', 'Date': '17/06/2019', 'Temps en secondes': '3817'}, {'Classement': 11, 'Nom du cycliste': 'Joe Dombrowski', 'Date': '17/06/20
19', 'Temps en secondes': '3829'}, {'Classement': '15', 'Nom du cycliste': 'Yoann Bagot', 'Date': '04/05/2018', 'Temps en secondes': '3837'}, {'Classement': '16', 'Nom du cycliste': 'Th
omas LEMAITRE', 'Date': '01/11/2017', 'Temps en secondes': '3892'}, {'Classement': '17', 'Nom du cycliste': 'Freddy Ovett', 'Date': '07/08/2016', 'Temps en secondes': '3906'}, {'Classem
ent': '18', 'Nom du cycliste': 'Jason Bakke', 'Date': '31/07/2013', 'Temps en secondes': '3907'}, {'Classement': '19', 'Nom du cycliste': 'Jason Osborne', 'Date': '11/01/2015', 'Temps e
n secondes': '3911'}, {'Classement': '20', 'Nom du cycliste': 'Frederic Glorieux', 'Date': '08/08/2014', 'Temps en secondes': '3914'}, {'Classement': '21', 'Nom du cycliste': 'Loïc Ruff
aut', 'Date': '13/04/2015', 'Temps en secondes': '3923'}, {'Classement': '22', 'Nom du cycliste': 'Philibert Nicolas', 'Date': '11/08/2013', 'Temps en secondes': '3932'}, {'Classement':
 12, 'Nom du cycliste': 'Danilo Celano', 'Date': '17/06/2019', 'Temps en secondes': '3932'}, {'Classement': '24', 'Nom du cycliste': 'Joris Ronflet', 'Date': '21/09/2012', 'Temps en sec
ondes': '3935'}, {'Classement': '25', 'Nom du cycliste': 'Kenny Nijssen', 'Date': '05/06/2016', 'Temps en secondes': '3944'}, {'Classement': '26', 'Nom du cycliste': 'Kasper Svendsen',
'Date': '04/07/2017', 'Temps en secondes': '3945'}, {'Classement': '27', 'Nom du cycliste': 'David Swan', 'Date': '10/09/2016', 'Temps en secondes': '3954'}, {'Classement': '28', 'Nom d
u cycliste': 'David POLVERONI', 'Date': '21/09/2012', 'Temps en secondes': '3956'}]
```

Ici par exemple j'ai demandé à mon programme de me réaliser un classement seulement avec les performances de 2019 :

	A	B	C	D
1	Classement	Nom du cycliste	Date	Temps en secondes
2	1	Romain Bardet	17/06/2019	3515
3	2	Rein Taaramae	17/06/2019	3583
4	3	El Fares Julien	17/06/2019	3607
5	4	Elie Gesbert	17/06/2019	3651
6	5	Javi Moreno	17/06/2019	3658
7	6	Pierpaolo Ficara	17/06/2019	3669
8	7	Oscar Rodriguez	17/06/2019	3671
9	8	Adrien Guillonnet	17/06/2019	3757
10	9	tobias ludvigsson	17/06/2019	3798
11	10	Romain Hardy	17/06/2019	3817
12	11	Joe Dombrowski	17/06/2019	3829
13	12	Danilo Celano	17/06/2019	3932

Mais cela marche tout aussi bien avec n'importe quelle année :

2013 :

	A	B	C	D
1	Classement	Nom du cycliste	Date	Temps en secondes
2	1	Laurens ten Dam	14/07/2013	3497
3	2	Jason Bakke	31/07/2013	3907
4	3	Philibert Nicolas	11/08/2013	3932

2015 :

	A	B	C	D
1	Classement	Nom du cycliste	Date	Temps en secondes
2	1	thomas rostollan	18/06/2015	3726
3	2	Jason Osborne	11/01/2015	3911
4	3	Loïc Ruffaut	13/04/2015	3923

2017 :

	A	B	C	D
1	Classement	Nom du cycliste	Date	Temps en secondes
2	1	Julien Antomarchi	08/06/2017	3685
3	2	Thomas LEMAITRE	01/11/2017	3892
4	3	Kasper Svendsen	04/07/2017	3945