DS n° 03 – DS03

- Faire tous les exercices dans un même fichier NomPrenom.py à sauvegarder,
- mettre en commentaire l'exercice et la question traités (ex : # Exercice 1),
- ne pas oublier pas de commenter ce qui est fait dans votre code (ex : # Je crée une fonction pour calculer la racine d'un nombre),
- il est possible de demander un déblocage pour une question, mais celle-ci sera notée 0 (sauf pour les 10 et 11),
- il faut vérifier avant de partir que le code peut s'exécuter et qu'il affiche les résultats que vous attendez.

Classement du championnat de France de Football Féminin

L'objectif de ce travail est de générer le classement de la division 1 du Championnat de France de Football Féminin.
Pour cela, les résultats des 11 premières journées sont fournis dans le fichier `resultats.csv` présent dans le dossier `Ressources`.
En sachant que le calcul des points s'effectue de la manière suivante : 3pts pour un match gagné, 1pt pour un match nul et 0pt pour un match perdu. On donne la première ligne du fichier pour décrire sa structure.
`Bordeaux;1:1:1;3:6:0;0:1:2;3:1:0;0:1:4;0:2:5;3:3:0;3:3:1;0:0:1;1:1:1;0:0:1;`
Cette ligne s'interprète de la manière suivante, les éléments sont séparés par des ';', le premier étant le nom du club :

- `1:1:1`, pour la première journée, Bordeaux a fait un match nul (1pt), a marqué 1 but et encaissé 1 but,
- `3:6:0`, pour la deuxième journée, Bordeaux a gagné (3pts), a marqué 6 buts et encaissé 0 but,
- `0:1:2`, pour la troisième journée, Bordeaux a perdu (0pts), a marqué 1 buts et encaissé 2 buts,
- ...

1 Importer dans un script python le fichier `resultats.csv` (sans copier le fichier) et afficher la première ligne.

Solution 1.

```
file=open('resultats.csv','r')
contenu=file.read()
equipes=contenu.split('\n')

print(equipes[0])
```

2 Créer, à l'aide d'un script, une liste `resultats` qui contient les données du fichier mises sous la forme suivante (seule la première ligne apparaît ici). Afficher la première ligne.

```
['Bordeaux', '1:1:1', '3:6:0', '0:1:2', '3:1:0', '0:1:4', '0:2:5', '3:3:0',
 '3:3:1', '0:0:1', '1:1:1', '0:0:1']
```

Solution 2.

```
resultats=[]
for equipe in equipes[:-1]:
    resultats.append(equipe.split(';')[:-1])

print(resultats[0])
```

3 A partir de la liste précédente, créer, à l'aide d'un script, une liste `tableau_points` qui contient pour chaque journée, le cumul des points de l'équipe et la différence des buts cumulés. La première ligne correspond à celle-ci :

```
['Bordeaux', [1, 0], [4, 6], [4, 5], [7, 6], [7, 3], [7, 0], [10, 3], [13, 5],
 [13, 4], [14, 4], [14, 3]]
```

Elle a été obtenue en faisant le calcul suivant :

- Journée 1 : **1**pt, 1 but pour - 1 but contre = **0**,
- Journée 2 : 1pt (journée 1) + **3** pts (journée 2)=**4**pts, 0 buts (journée 1)+ 6 buts pour - 0 but contre =**6**buts,
- Journée 3 : 4pts (journées 1 et 2) + 0pt=**4**pts, 6 buts (journées 1 et 2) + 1 but pour - 2 buts contre = **5**buts,
- ...

Solution 3.

```
tableau_points=[]

for id,equipe in enumerate(resultats):
    tableau_points.append([equipe[0]])
    points_total=0
    buts_total=0
    for match in equipe[1:]:
        points_match,buts_pour,buts_contre=match.split(':')
        points_total+=int(points_match)
        buts_total+=int(buts_pour)-int(buts_contre)
        tableau_points[id].append([points_total,buts_total])

print(tableau_points[0])
```

4 Tracer pour l'équipe de Bordeaux, l'évolution du nombre de points en fonction des journées de championnat.

Solution 4.

```
import matplotlib.pyplot as plt

plt.plot([point for point,buts in tableau_points[0][1:]])
plt.show()
```

5 Tracer pour toutes les équipes, mais sur le même graphe, l'évolution du nombre de points en fonction des journées de championnat. On indiquera à quelle équipe correspond quelle courbe grâce au paramètre label, comme dans l'exemple suivant :

```
import matplotlib.pyplot as plt

plt.plot([1,2,3,4], label='Toto')
plt.legend()
plt.show()
```

Solution 5.

```
import matplotlib.pyplot as plt

for equipe in tableau_points:
    plt.plot([point for point,buts in equipe[1:]], label=equipe[0])
plt.legend()
plt.show()
```

Classement des équipes

6 Coder une fonction de tri (au choix mais pas de fonction python de type `sorted()`) et la tester sur la liste `[4,5,2,3,6,1]`.

Solution 6.

```
def tri_selection(liste):
    # Parcours de 1 à la taille de la liste
    for i in range(len(liste)-1):
        # Initialiser le min
        min=liste[i]
        jmin=i
        for j in range(i, len(liste)):
            # Chercher le min
            if liste[j]<min:
                jmin=j
                min=liste[j]
        # Permuter le min et l'élément i
        liste[i],liste[jmin]=liste[jmin],liste[i]

liste=[4,5,2,3,6,1]
tri_selection(liste)
print(liste)
```

7 Créer et afficher, à l'aide d'un script, une liste `points_derniere_journee` contenant la liste des points de chaque équipe à la dernière journée de championnat, dans l'ordre de la liste `tableau_points`. Le résultat doit être : `[14, 12, 21, 8, 4, 33, 19, 30, 22, 14, 5, 7]`.

Solution 7.

```
points_derniere_journee=[]

for equipe in tableau_points:
        points_derniere_journee.append(equipe[-1][0])
```

8 Créer et afficher, à l'aide d'un script, une liste `points_derniere_journee_tri`, copie de la liste précédente mais classée dans l'ordre croissant à l'aide de la fonction de tri créée précédemment.

Solution 8.

```
points_derniere_journee_tri=points_derniere_journee.copy()
tri_selection(points_derniere_journee_tri)
print(points_derniere_journee_tri)
```

9 Créer, à l'aide d'un script, la liste des équipes du championnat dans le même ordre que celui de la liste `tableau_points`.

Solution 9.

```
liste_equipes=[]

for equipe in tableau_points:
        liste_equipes.append(equipe[0])
print(liste_equipes)
```

10 Créer une fonction de tri, en se basant sur la précédente, permettant de classer la liste des villes en même temps que la liste `points_derniere_journee` pour la dernière journée du championnat (on ne se préoccupera pas du classement des villes ayant le même nombre de points, mais toutes doivent apparaître). Vous pourrez vérifier le résultat avec celui de la question 5.

Solution 10.

```
def tri_selection(liste,liste2):
    # Parcours de 1 à la taille de la liste
    for i in range(len(liste)-1):
        # Initialiser le min
        min=liste[i]
        jmin=i
        for j in range(i, len(liste)):
            # Chercher le min
            if liste[j]<min:
                jmin=j
                min=liste[j]
        # Permuter le min et l'élément i
        liste[i],liste[jmin]=liste[jmin],liste[i]
        liste2[i],liste2[jmin]=liste2[jmin],liste2[i]

points_derniere_journee_tri=points_derniere_journee.copy()
liste_equipes_tri=liste_equipes.copy()
tri_selection(points_derniere_journee_tri,liste_equipes_tri)
print(liste_equipes_tri)
```

11 Créer, à l'aide d'un script, la liste `classement_journees` qui contient la liste, journée par journée, des villes classées par ordre de points, pour toutes les journées du championnat (on ne se préoccupera pas du classement des villes ayant le même nombre de points, mais toutes doivent apparaître). L'affichage du premier élément de cette liste, correspondant à la première journée est :

```
['Dijon', 'Fleury', 'Guingamp', 'Issy', 'Reims', 'Bordeaux', 'Saint_Étienne',
 'PSG', 'Paris_FC', 'Lyon', 'Montpellier', 'Soyaux']
```

Solution 11.

```
classement_journees=[]

for journee in range(len(tableau_points[0])-1):
    liste_journee=[]
    for equipe in tableau_points:
        liste_journee.append(equipe[journee+1][0])
    liste_equipes_journee=liste_equipes.copy()
    tri_selection(liste_journee,liste_equipes_journee)
    classement_journees.append(liste_equipes_journee)

print(classement_journees[0])
```