OBJECTIFS : L'objectif de ce TP est d'appliquer ce qui a été vu en cours sur le chapitre des fichiers.

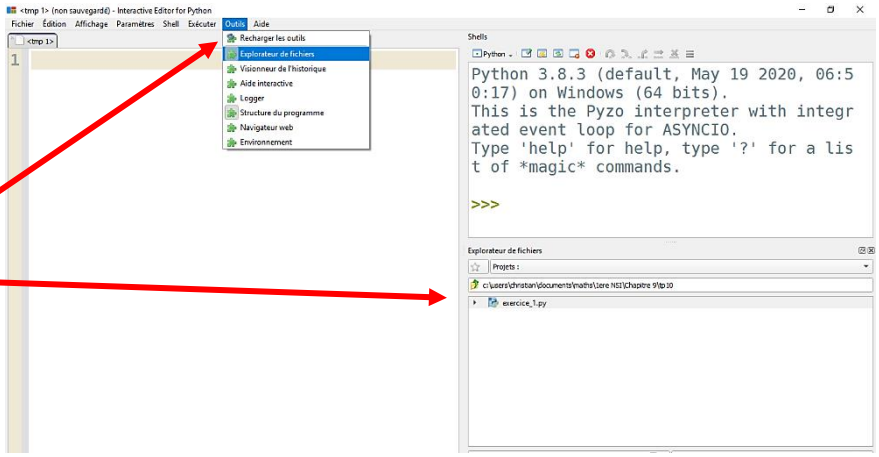
En fin de Tp, vous uploaderez **uniquement** les 4 fichiers constitués *exercice_1.py, exercice_2A.py, exercice_2B.py, exercice_3.py et exercice_4.py* avec le code **tp10** . Aucun autre compte-rendu n'est demandé.

1. Ecrire dans un fichier texte :

⇨ Ouvrir un nouveau fichier *.py* et le sauvegarder sous le nom *exercice_1.py* .

⇨ Dans Pyzo, dans l'onglet *Outils,* afficher la fenêtre *Explorateur de fichier*. Aller dans répertoire de travail dans lequel se trouve ce fichier *exercice_1.py.*

Ne pas oublier de cocher la case dans l'onglet *Exécuter.*



⇨ Utiliser ce qui a été vu en cours pour écrire dans ce fichier *exercice_1.py,* le code python qui permet de créer un fichier nommé *exercice_1.txt* et qui contiendra :

Vous pouvez bien sûr modifier les noms proposés.

exercice_1.py | exercice_1.txt

```
1 nom, prenom, note_en_nsi
2 Bonnepart,Hector,18
3 Bravard,Paul,14
4 Bretagne,Hans,16
```

Le code incomplet est :

exercice_1.py | exercice_1.txt

```
1 f = open("exercice_1.txt","w",encoding="utf-8")
2
3
4
5 f.write("Bretagne,Hans,16\n")
6 f.close()
```

Permet d'écrire dans le fichier les caractères é è ê ç ù …. (norme utf-8)

⇨ Vérifier que le fichier *exercice_1.txt* contient bien les 4 lignes souhaitées.

2. Lire dans un fichier texte :

⇨ Ouvrir un nouveau fichier *.py* et le sauvegarder sous le nom *exercice_2A.py* .

2.1. Utilisation de la fonction *read()* :

Rappel du cours : comme le montre l'exemple ci-dessous, si *fichier* est l'objet python de ' *Communes.csv* ', la commande *fichier.read()* retourne une chaine de caractère qui contient **tout le contenu** du fichier :

```
fichier = open("Communes.csv",'r')  # Ouvre le fichier
print(fichier.read())               # Le print écrit le contenu du fichier dans le terminal
fichier.close()                     # Ferme le fichier
```

Application :

Le code ci-dessous incomplet, permet d'afficher dans le shell le contenu du fichier *exercice_1.txt* crée précédemment :

```
>>> (executing file "exercice_2A.py")

 <class 'str'>

nom, prenom, note_en_nsi
Bonnepart,Hector,18
Bravard,Paul,14
Bretagne,Hans,16
```

⇨ Compléter le fichier *exercice_2A.py* ci-dessous :

exercice_1.txt | exercice_2A.py

```
1 f = open("exercice_1.txt","r",encoding='utf8')
2
3 f.close()
4
5 print("\n",type(contenu),"\n")
6 print(contenu)
```

2.2. Utilisation de la fonction *readline()* :

Rappel du cours : Si *fichier* est l'objet python de ' *Communes.csv* ', la commande *fichier.readline()* retourne une chaine de caractère qui contient **une seule ligne** du fichier. Cette ligne est celle sur laquelle se trouve le pointeur de lecture.

2.2.1. Lecture de la seconde ligne du fichier *exercice_1.txt* :

⇨ Ecrire le code ci-contre et le sauvegarder sous le nom *exercice_2B.py* :

```
1 f = open("exercice_1.txt","r",encoding='utf8')
2 c = f.readline()
3 print(c)
4 c = f.readline()
5 print(c)
6 f.close()
```

Le résultat de l'exécution est le suivant :

On constate que les chaines de caractères contenues dans la variable ***c*** comprennent un saut de ligne

```
>>> (executing file "exercice_2B.py")
nom, prenom, note_en_nsi

Bonnepart,Hector,18
```

⇨ Modifier le code précédent pour **ne pas** afficher le dernier caractère de ***c*** qui est égal au saut de ligne :

```
1 f = open("exercice_1.txt","r",encoding='utf8')
2 c = f.readline()
3 print(c[:-1])
4 c = f.readline()
5 print(c[:-1])
6 f.close()
```

⇨ Tester ce code modifié …

2.2.2. Lecture de la seconde ligne et séparation des éléments dans une liste avec *split()* :

⇨ Modifier le code précédent pour utiliser la méthode ***split()*** sur la chaine de caractère ***c*** :

```
1 f = open("exercice_1.txt","r",encoding='utf8')
2 c = f.readline()
3 c = f.readline()
4 l = c.split(",")
5 print(l)
6 f.close()
```

⇨ Tester ce code modifié …

2.2.3. Modification du dernier élément de la liste pour enlever le saut de ligne :

⇨ Modifier encore ce code pour enlever le saut de ligne sur la chaine de caractère ***c*** :

```
1 f = open("exercice_1.txt","r",encoding='utf8')
2 c = f.readline()
3 c = f.readline()
4 c = c[:-1]
5 l = c.split(",")
6 print(l)
7 f.close()
```

⇨ Tester ce code modifié.

2.2.4. Lecture des 2 premières lignes dans une liste double :

⇨ Modifier enfin ce code pour lire les 2 premières lignes dans une liste double :

```
1  f = open("exercice_1.txt","r",encoding='utf8')
2  lst=[]
3  for i in range(2) :
4      c = f.readline()
5      c = c[:-1]
6      l = c.split(",")
7      lst.append(l)
8  f.close()
9
10 print(lst)
```

⇨ Tester ce code modifié.

3. Lecture du fichier complet dans une liste double :

A l'ouverture du fichier, on ne connait pas le nombre de lignes qu'il contient. Il est ainsi nécessaire de lire chacune d'elles avec une boucle ***while*** jusqu'à ce que le pointeur de lecture se retrouve sur une ligne vide.

⇨ **Compléter** le code ci-contre qui permet de lire la totalité du fichier dans une liste double :

```
1  # Lecture du fichier
2  f = open("exercice_1.txt","r",encoding='utf8')
3  lst = []
4  c = f.readline()
5  while c != "" :
6      c = c[:
7      l =
8      lst.append(
9      c = f.readline()
10 f.close()
11
12 # Affichage des valeurs lues
13 for l in lst :
14     print(l)
```

L'exécution de ce code doit conduire à l'affichage de :

```
>>> (executing file "exercice_2B.py")
['nom', ' prenom', ' note_en_nsi']
['Bonnepart', 'Hector', '18']
['Bravard', 'Paul', '14']
['Bretagne', 'Hans', '16']
```

4. Utilisation de cette liste pour effectuer un calcul de moyenne :

⇨ Ouvrir un nouveau fichier *.py* et le sauvegarder sous le nom *exercice_3.py* .

⇨ Compléter le code qui suit et qui permet :

- à partir de la fonction *lecture()* de lire le fichier *exercice_1.txt*,
- à partir de la fonction *moyenne()* de calculer la moyenne des élèves figurant dans le fichier,
- à partir de la fonction *eleve_moyen()* de déterminer quel élève à une note qui est la plus proche de la moyenne calculée.

L'exécution donnera dans le shell :

```
>>> (executing file "exercice_3.py")
La moyenne est de :  16.0
Bretagne Hans a comme note :  16
```

Le programme principal sera le suivant :

```
# Main
eleves = lecture("exercice_1.txt")
moy = moyenne(eleves)
print(f"La moyenne est de : {round(moy,2)}")
i = eleve_moyen(eleves,moy)
print(f"{eleves[i][0]} {eleves[i][1]} a comme note :  {eleves[i][2]}")
```

A vous de définir les différentes fonctions. Ne pas oublier d'indiquer au début de chaque fonction, un commentaire indiquant les entrées / sorties de celle-ci :

```
def lecture(nom) :
    '''
        Lit le fichier dont le nom est en argument.
        Retourne une liste double
    '''
    f = open(nom,"r",encoding='utf8')
    lst = []
    [redacted]
    f.close()
    return lst
```

```
def moyenne(liste) :
    '''
        Renvoie la moyenne des notes des élèves
        enregistrés dans la liste qui est en
        argument.
    '''
    [redacted]
    return moy
```

```
def eleve_moyen(liste,moy) :
    '''
        Renvoie l'indice de l'élève de la liste
        mise en argument pour lequel la note est la
        plus proche de la valeur moyenne des notes
        mise en argument.
    '''
    min = float(liste[1][2]) - moy
    if min < 0 : min = -min
    indice = 1
    for i in range(1,len(liste)) :
        [redacted]
```

5. *Utilisation du code de exercice_3.py sur un fichier plus volumineux* :

⇨ Télécharger le fichier *tp10.zip* à partir de *nsibranly.fr* et décompresser le fichier *world_nsi.txt* **dans votre répertoire** de travail.

⇨ Ce fichier *world_nsi.txt* contient 150 000 lignes correspondants à tous les élèves de ***Nsi*** de notre planète. Jeter un coup d'œil dans ce fichier pour vérifier que les données y sont écrites dans le même format que dans le fichier *exercice_1.txt*. Utiliser votre code pour déterminer la moyenne de ces notes et les nom et prénom de l'élève dont la note est la plus proche de la moyenne.

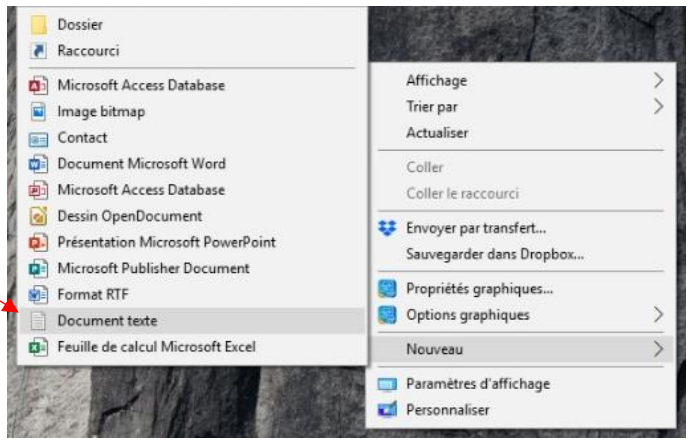
Le résultat de l'exécution sera normalement :

```
>>> (executing file "exercice_3.py")
La moyenne est de : 10.04
EZZAKI winnie a comme note :  10.04
```

6. Traitement d'un fichier .csv téléchargé sur un data public. :

⇨ Aller sur le site github : https://github.com/fivethirtyeight/data qui propose des datas U.S. originaux.

Aller dans la catégorie births et copier le fichier US_births_2000-2014_SSA.csv en cliquant sur Raw pour ensuite copier l'ensemble de la page (ctrl A , puis ctrl C) dans un fichier texte que vous nommerez ***naissance.txt***



Ce fichier contient pour chacune des années entre 2000 et 2014, pour chaque mois de l'année et jour du mois, le nombre de naissance aux U.S.A. On trouve aussi sur chacune des lignes, juste avant le nombre de naissance, le jour de la semaine : 1 pour lundi, 2 pour mardi, …. et 7 pour dimanche.

```
year,month,date_of_month,day_of_week,births
2000,1,1,6,9083
2000,1,2,7,8006
2000,1,3,1,11363
2000,1,4,2,13032
2000,1,5,3,12558
```

⇨ Créer un code composé du programme principal suivant :

```
# Main
naissance = lecture("naissances.txt")
n_total = total(naissance)
print(f"Il y a eu {n_total} naissances aux U.S. entre 2000 et 2014")
pourcentage = stat(naissance,"mardi",n_total)
print(round(pourcentage,1)," % des naissances, c'était le mardi")
```

A l'exécution il doit donner dans le shell :

```
>>> (executing file "exercice_4.py")
Il y a eu  62176233  naissances aux U.S. entre 2000 et 2014
16.5  % des naissances, c'était le mardi
```

Ce code sera enregistré sous le nom *exercice_4.py* .