TP n° 04 – Lecture et écriture d'un fichier CSV et traitement statistique des données

I Import de données

Jusqu'à présent, lorsque vous avez traité des données avec un programme écrit en python, elles ont disparu à la fermeture du programme car elles étaient stockées dans la mémoire vive de l'ordinateur (Random Access Memory). Pour conserver des données pour une utilisation ultérieure, il faut les stocker sur des mémoires non volatiles (disque dur, carte mémoire, CD Rom,...). Elles sont stockées sur ces supports dans une zone physique identifiée par le système d'exploitation [1]. La suite de donnée structurée est appelée fichier (ou *file* en anglais).
Il existe deux types de fichier de données : les fichiers de type texte (encore appelé ASCII) et de type binaire.
Les fichiers de type texte sont constitués de caractères codés par un nombre entier, écrit en code binaire sur 8 bits (1 octet), compris entre 0 et 255.
Les 128 premiers caractères sont communs aux différents codes ASCII.
Les 128 caractères suivant permettre de décrire les différents caractères nationaux (exemple : é, è, à,... pour le français). Chaque code ASCII fait l'objet d'une norme ISO. Pour les langues d'Europe de l'ouest, la norme est ISO 8859-1 appelé Latin-1. Le codage est donné dans le tableau I.

ISO/CEI 8859-1

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	*positions inutilisées*															
1x																
2x	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8x	*positions inutilisées*															
9x																
Ax	NBSP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
Bx	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
Fx	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

L'encodage universel des caractères est réalisé soit par l'Unicode ou l'UTF-8 (Universal Transformation Format).
Les fichiers de type binaire ne contiennent pas (exclusivement) du texte. Et ils ne peuvent être convenablement traités que par des logiciels spécialisés. Un fichier PDF, une image JPEG ou un mp3 sont quelques exemples de fichiers binaires.
Dans la suite, on n'utilisera que des fichiers de type texte.

I.1 Les fichiers CSV

Le sigle CSV signifie Comma-Separated Values et désigne un fichier informatique de type tableur, dont les valeurs sont séparées par des virgules.
Le format CSV est un format de texte simple qui est utilisé dans de nombreux contextes lorsque de grandes quantités de données doivent être fusionnées sans être directement connectées les unes aux autres.
L'extension de ce type de fichiers est .csv, et ils peuvent être utilisés entre différents outils informatiques et bases de données, lorsqu'on souhaite déployer le contenu d'une base de données sur une feuille de calcul.
Des tableurs tels qu'Excel (Microsoft) ou Calc (LibreOffice) et des bases de données telles que MySQL et Oracle sont capables d'importer et exporter des fichiers CSV. Toutefois, en raison de sa structure basique, le format de fichier CSV ne convient que pour des données structurées simples.

I.2 Import des données du TP

Les données qui seront utilisées dans ce TP sont issues du site `https://www.data.gouv.fr` qui partage des données gouvernementales en accès libre.

1. En anglais « *OS* » (pour *Operating system.*)

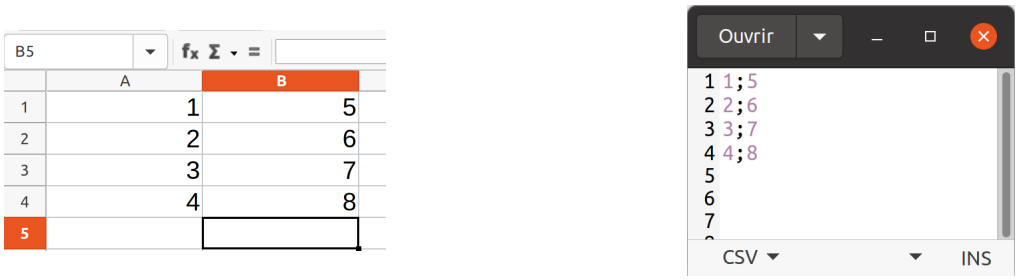

FIGURE 1 – Même fichier CSV vu dans un tableur et dans un éditeur de texte

Nous allons plus particulièrement utiliser celles de la page suivante, qui concerne la température quotidienne par département en France.

`https://www.data.gouv.fr/fr/datasets/temperature-quotidienne-departementale-depuis-janvier-2018/`

Le fichier */home/eleve/Ressources/PTSI/TP/TP04/temperature-quotidienne-departementale.csv* contient les données téléchargées sous la forme d'un fichier CSV. Copiez-le dans votre répertoire personnel.

Exercice 1.
Taper les lignes suivantes dans un fichier script et vérifier le contenu de `contenu` en ajoutant `print(contenu)` à la fin du script (cette commande pourra être retirée par la suite).

```
file=open('temperature-quotidienne-departementale.csv','r')
contenu=file.read()
file.close()
```

Solution 1.

```
file=open('temperature-quotidienne-departementale.csv','r')
contenu=file.read()
file.close()
print(contenu)
```

La fonction `open` ouvre le fichier avec le paramètre :
- `r`, pour *reading*, en lecture seule,
- `w`, pour *writting*, en écriture (le fichier est alors totalement écrasé à l'ouverture du fichier),
- `a`, pour *appending*, en modification, tout sera alors écrit à la suite du contenu déjà présent dans le fichier.

La fonction `read()` lit le contenu du fichier et la fonction `close()` ferme le fichier pour qu'il puisse être utilisé par un autre logiciel.
`contenu` est alors une chaîne de caractères que nous allons transformer en liste.
La commande `split` découpe les chaînes de caractères selon un séparateur. La commande suivante permet de générer la liste `lignes` qui contient toutes les lignes du fichier CSV.

```
lignes=contenu.split('\n')
```

Exercice 2.
Taper la ligne précédent dans le script et vérifier le contenu de `lignes` en ajoutant `print(lignes[0:2])` à la fin du script (cette commande pourra être retirée par la suite).

Solution 2.

```
file=open('temperature-quotidienne-departementale.csv','r')
contenu=file.read()
file.close()

lignes=contenu.split('\n')
print(lignes)
```

Nous allons maintenant pouvoir découper chacune des ces lignes en colonnes à l'aide de la même fonction.

Exercice 3.
Utiliser une boule `for` pour parcourir toutes les lignes. Découper ensuite chaque ligne à l'aide du séparateur ';' et stocker le résultat dans une liste `data` qui sera écrasée à chaque itération mais nous réglerons ce problème plus tard.

Solution 3.

```
for ligne in lignes[1:]:
    data=ligne.split(';')
```

Exercice 4.
Au début du script créer deux variables `dep='75'` et `year='2018'` qui permettront de stocker le numéro du département et l'année pour lesquels nous allons effectuer l'étude.

Solution 4.

```
dep='75'
year='2018'
```

Exercice 5.
Créer une liste appelée `temperatures` dans laquelle vous allez stocker les dates et les valeur des températures **moyennes** (converties en float) pour ce département et cette année. Faire attention au format des données, il faudra par exemple retirer les "". `print(temperatures[0])` devra retourner `['2018-01-01', 8.0]`.

Solution 5.

```
lignes=contenu.split('\n')
temperatures=[]
for ligne in lignes[1:]:
    data=ligne.split(';')
    if data[0][1:5]==year and data[1]==dep:
        temperatures.append([data[0][1:-1],float(data[5][1:-1])])
print(temperatures[0])
```

I.3 Export de données

Nous allons maintenant exporter le contenu de cette liste dans un fichier CSV. En utilisant le script suivant :

```
file2=open('fichier_export.csv','w')
for date,temp in temperatures:
    file2.write(date+';'+str(temp)+'\n')
file2.close()
```

Solution 6.

```
file2=open('fichier_export.csv','w')
for date,temp in temperatures:
    file2.write(date+';'+str(temp)+'\n')
file2.close()
```

Exercice 6.
Insérer le code suivant dans le script et double-cliquer sur le fichier `fichier_export.csv` ainsi créé afin de vérifier son contenu.

II Traitement des données

La suite du code peut s'écrire après le script précédent. Elle utilise la liste `temperatures` précédemment créée.

Exercice 7.
Proposer une solution pour déterminer la moyenne arithmétique des températures de Paris en 2018. Rappel : $\mu = \frac{\sum_{i=1}^{n} x_i}{n}$ pour la moyenne d'une série x_i de n éléments. Vous devez trouver `13.91`.

Solution 7.

```
t_total=0
for date,temp in temperatures:
    t_total+=temp
t_moy=t_total/len(temperatures)
print(t_moy)
```

Exercice 8.
Proposer une solution pour déterminer la médiane des températures de Paris en 2018. Rappel : En théorie des probabilités et en statistiques, la médiane est la valeur qui sépare la moitié inférieure de la moitié supérieure d'un ensemble (échantillon, population, distribution de probabilités). Intuitivement, la médiane est ainsi le point milieu de l'ensemble. Vous devez trouver `13.75`.

Solution 8.

```
temperatures_classe=sorted(temperatures)
print(temperatures_classe[len(temperatures)//2])
```

Exercice 9.
Proposer une solution pour déterminer l'écart type des températures de Paris en 2018. On rappelle que la formule de l'écart type est $\sigma = \sqrt{\frac{\sum_{i=1}^{n} |x_i - \mu|^2}{n}}$, avec μ la moyenne arithmétique de la série de données. Vous devez trouver `7.3`.

Solution 9.

```
sigma=0
for date,temp in temperatures:
    sigma+=(temp-t_moy)**2
sigma=m.sqrt(sigma/len(temperatures))
print(sigma)
```

On souhaite tracer l'histogramme de ces valeurs.

Exercice 10.
Importer la bibliothèque `matplotlib.pyplot` sous l'alias `plt` puis exécuter le code suivant :

```
plt.hist([temp for date,temp in temperatures],range=(-4,30),bins=34)
plt.show()
```

Solution 10.

```
import matplotlib.pyplot as plt
plt.hist([temp for date,temp in temperatures],range=(-4,30),bins=34)
plt.show()
```

Exercice 11.
Proposer une solution pour effectuer le calcul de ces trois opération en boucle pour 2018, 2019 et 2020.

Pour compléter, vous pourrez placer un `fig = plt.figure()` avant la boucle, puis à l'endroit où vous souhaiter tracer les histogrammes le code suivant :

```
ax = fig.add_subplot(4,1,num+1)
ax.hist([temp for date,temp in temperatures],range=(-4,30),bins=34)
```

Solution 11.

```
fig = plt.figure()
for num,year in enumerate(['2018','2019','2020']):
    temperatures=[]
    for ligne in lignes[1:]:
        data=ligne.split(';')
        if data[0][1:5]==year and data[1]==dep:
            temperatures.append([data[0][1:-1],float(data[5][1:-1])])

    # Calcul de la moyenne
    t_total=0
    for date,temp in temperatures:
        t_total+=temp
    t_moy=t_total/len(temperatures)


    # Calculer la médiane
    temperatures_classe=sorted([temp for date,temp in temperatures])
    t_med=temperatures_classe[len(temperatures)//2]

    # Calcul de l'écart type
    sigma=0
    for date,temp in temperatures:
        sigma+=(temp-t_moy)**2
    sigma=m.sqrt(sigma/len(temperatures))

    print("Pour l'année {} dans le {} :\n - La moyenne est {:.2f}\n - La médiane est {:.2f}\n - L'é

    # Tracer un histogramme
    ax = fig.add_subplot(4,1,num+1)
    ax.hist([temp for date,temp in temperatures],range=(-4,30),bins=34)
```

Exercice 12.
Effectuer le même travail en prenant dep='6', conclure quant à la douceur du climat de Nice.

Solution 12.

```
dep='6'
```

Les moyennes/médianes sont plus hautes et l'écart type est moins important.