I - Rappels : statistiques descriptives univariées

Exercice 1

1. **Écrire une fonction `moyemp(X)` qui prend en paramètre une série statistique (sous forme de liste) et qui renvoie sa moyenne empirique.**

```
def moyemp(X) :
    n = len(X)
    # calcule de la moyenne empirique de X
    S = 0
    for i in range(n) :
        S = S + X[i]
    MoyX = S/n
    return MoyX
```

2. **Écrire une fonction `varemp(X)` qui prend en paramètre une série statistique et qui calcule et renvoie sa variance empirique.**

```
def varemp(X) :
    n = len(X)
    # on appelle moyemp pour obtenir la moyenne empirique
    MoyX = moyemp(X)
    # calcule de la variance empirique de X
    S = 0
    for i in range(n) :
        S = S + (X[i] - MoyX)**2
    Var = S/n
    return Var
```

3. **Écrire une fonction `med(X)` qui prend en paramètre une série statistique rangée par ordre croissant et qui calcule et renvoie sa médiane.**

```
def med(X) : # avec X liste rangée dans l'ordre croissant
    n = len(X)
    if n%2 == 1 : # si n est impair, la série se découpe bien en deux
        milieu = int(n/2) # indice du milieu de la liste
        return X[milieu]
    else : # si n est pair, on renvoie la moyenne des termes centraux
        milieu = int(n/2)-1 # indice du "milieu" de la liste
        return (X[milieu]+X[milieu+1])/2
```


II - La bibliothèque pandas de Python

Exercice 2

Le fichier `"Notes.csv"` contient des données sur les notes aux concours d'une classe du sud ouest de la France.
On exécute les instructions suivantes.

```
>>> import pandas as pd #  On charge la bibliothèque pandas
>>> Notes = pd.read_csv ("Notes.csv",sep = ',')}
>>> Notes.columns
['Etudiant', 'DCG EDHEC/ESSEC', 'DCG EML/HEC', 'ESH ESCP', 'ESH ESSEC/HEC',
'LVA ELVi', 'LVB ELVi', 'Maths EML', 'Maths Edhec', 'Maths ESSEC II',
 'Maths HEC', 'Synthèse ESCP']
```

1. **Pour une épreuve de votre choix, écrire des instructions permettant d'afficher la meilleure note ainsi que la moyenne et l'écart-type de cette épreuve.**
Choisissons l'épreuve de LV1 ELVi.

```
>>> # infos sur la série de notes 'Maths EML'
>>> Matiere = Notes['Maths EML']
>>> print('Max : ',Matiere.max())
Max :  17.7
>>> print('Moyenne : ',Matiere.mean())
Moyenne :  12.6725
>>> print('Ecart type : ',Matiere.std())
Ecart type :  3.250797732470585
```

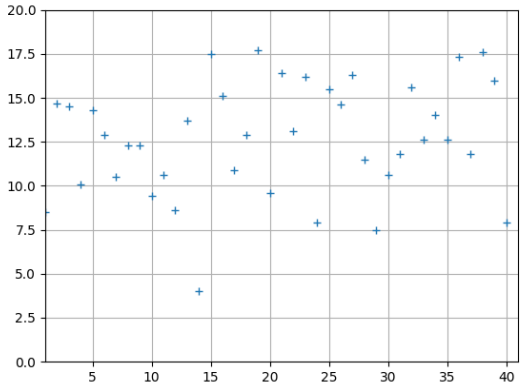
2. **Écrire une fonction Python d'entête `"def NbrNotes(M,eps,Matiere) :"` qui renvoie le nombre de notes de la matiere `"Matiere"` qui sont entre `"M-eps"` et `"M+eps"`.**

```
def NbrNotes(M, eps, matiere) :
    N = 0
    t = len(Matiere)
    for i in range(t) :
        if Matiere[i] <= M+eps and Matiere[i] >= M-eps :
            N += 1
    return N
```

3. **Compléter le script ci-dessous pour qu'il affiche un graphe représentant les notes obtenues à cette épreuve. *On utilisera en abscisses une liste d'entiers numérotant les élèves.***
On ajoute les instructions suivantes :

```
import matplotlib.pyplot as plt
Matiere = Notes['Maths EML']
N = len(Matiere) # la taille de la liste correspond aux nombre d'élèves
plt.plot (range(1,N+1), Matiere, '+') # '+' pour tracer des points
# sous forme de croix sans les relier
plt.show()
```

On obtient alors le graphe suivant (en abscisses, le numéro de l'élève, en ordonnée sa note).



III - Statistiques descriptives bivariées

Exercice 3

1. **Démontrer la formule suivante :** $\text{cov}(x,y) = \dfrac{1}{n}\sum_{i=1}^{n} x_i y_i - \overline{xy}$.

   On a :

$$\begin{aligned}
\text{cov}(x,y) &= \frac{1}{n}\sum_{i=1}^{n}\left(x_i-\overline{x}\right)\left(y_i-\overline{y}\right)\\
&= \frac{1}{n}\left[\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i\overline{y} - \sum_{i=1}^{n}\overline{x}y_i + \sum_{i=1}^{n}\overline{xy}\right]\\
&= \frac{1}{n}\sum_{i=1}^{n} x_i y_i - \overline{y}\underbrace{\frac{1}{n}\sum_{i=1}^{n} x_i}_{=\overline{x}} - \overline{x}\underbrace{\frac{1}{n}\sum_{i=1}^{n} y_i}_{=\overline{y}} + \underbrace{\frac{1}{n}\sum_{i=1}^{n}\overline{xy}}_{=\overline{xy}} \quad \text{car } \overline{y},\overline{x} \text{ constantes}\\
&= \frac{1}{n}\sum_{i=1}^{n} x_i y_i - \overline{yx} - \overline{xy} + \overline{xy}\\
&= \frac{1}{n}\sum_{i=1}^{n} x_i y_i + \overline{xy}
\end{aligned}$$

2. **En déduire une fonction Python d'entête** `"def Covariance(x,y) :"` **qui calcule la covariance empirique de la série double** `(x,y)` **stockée dans deux tableaux** `numpy` **donnés en paramètre.** ***On pourra utiliser la méthode*** *`mean`***.**

```
# version sans mean(x*y)
def Covariance(x,y) :
    n = len(x)
    # calcule de la moyenne empirique de x*y
    S = 0
    for i in range(n) :
        S = S + x[i]*y[i]
    Moy_xy = S/n
    return Moy_xy - np.mean(x)*np.mean(y)

# version avec mean(x*y)
def Covariance(x,y) :
    return np.mean(x*y) - np.mean(x)*np.mean(y)
```

3. **Utiliser la fonction précédente pour calculer la variance empirique de la série statistique** $(x_i)_{i\in[\![1;n]\!]}$**.**

```
def VarEmp(x) :
    return Covariance(x,x)
```

4. **Écrire une fonction Python d'entête** `"def CoeffCor(x,y) :"` **qui calcule le coefficient de corrélation linéaire empirique de la série double** `(x,y)` **stockée dans deux tableaux** `numpy` **donnés en paramètre.**

```
def CoeffCorr(x,y) :
    return Covariance(x,y)/(np.std(x)*np.std(y))
```

Exercice 4

On considère la série statistique double contenant les notes aux épreuves de Maths EDHEC et de Maths EML des élèves de notre classe du sud ouest de la France.
Compléter le programme suivant pour qu'il trace le nuage de points ainsi que le point moyen associés à cette série double.

```
Notes = pd.read_csv ("Notes.csv",sep = ',')}
X = Notes['Maths Edhec']
Y = Notes['Maths EML']
plt.plot (X, Y, '+')  # nuage de points : X en abscisses, Y en ordonnees
plt.plot (X.mean(), Y.mean(), '*')  # point moyen
plt.show()
```

IV - Droite de régression linéaire

Exercice 5

1. On reprend le programme de l'exercice 4 et on ajoute les lignes ci-dessous.
   Compléter ce programme pour qu'il détermine les coefficients a et b de la droite de régression linéaire de Y en X puis expliquer ce qu'il fait.

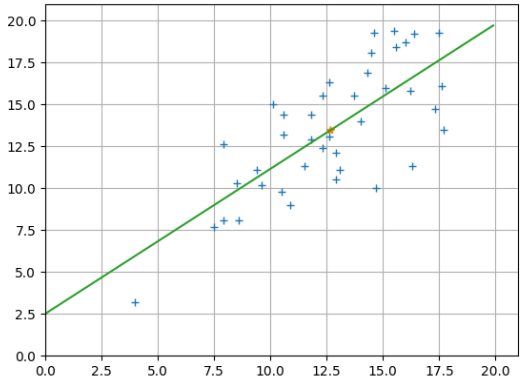
```
cov = Covariance(X,Y)  # Fonction écrite plus haut
a = cov / np.var(X)  # coeff de la droite de régression (cf theoreme)
b = np.mean(Y) - a*np.mean(X)   # coeff de la droite de régression
# on trace la droite de régression linéaire d'équation y =ax+b
abs = np.arange(0, 20 ,0.1)
plt.plot(abs, a*abs+b )
plt.show()
print(CoeffCorr(X,Y))  # Fonction écrite plus haut
```

   Ce programme calculer les coefficients de la droite de régression linbéaire et trace cette droite.

2. **On obtient le graphique suivant.**

Parmi les trois propositions, quel coefficient de corrélation linéaire correspond à cette série statistique?

$\rho_{x,y} \approx 0.74$; $\rho_{x,y} \approx -0.39$; $\rho_{x,y} \approx 0.54$



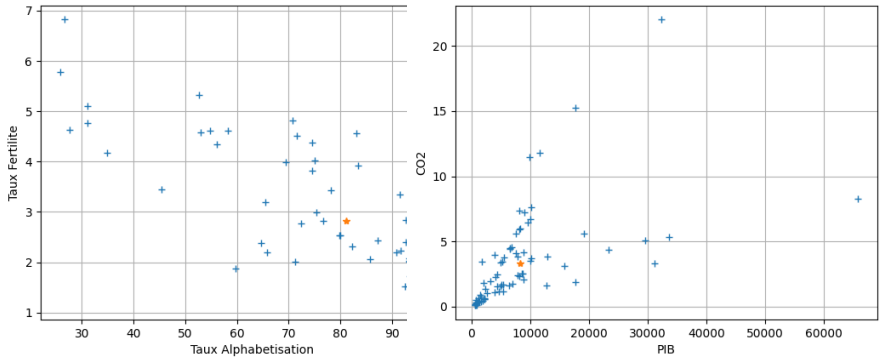
Le coefficient de corrélation linéaire sera positif car les deux variables varient dans le même sens.
Il semble y avoir une bonne corrélation donc $\rho_{x,y}$ sera plutôt proche de 1. Ainsi, $\rho_{x,y} \approx 0.78$.

Exercice 6

Le fichier "Pays.csv contient les données suivantes sur certains pays du monde : Émissions de CO2, taux de fertilité des femmes, PIB, taux d'alphabétisation des femmes.
On exécute les instructions suivantes.

```
>>> import pandas as pd #  On charge la bibliothèque pandas
>>> Pays = pd.read_csv ("Pays.csv",sep = ',')}
>>> Pays.columns
['Pays', 'CO2', 'Fertilite', 'PIB', 'Alphabetisation']
```

1. On trace les nuages de points suivants.



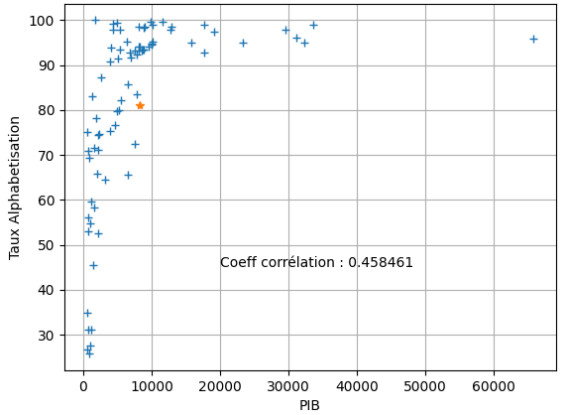
Associer à chaque graphique le coefficient de corrélation linéaire correspond et commenter.

$\rho_{x,y} \approx 0.56$; $\rho_{x,y} \approx -0.79$; $\rho_{x,y} \approx 0.85$ $\rho_{x,y} \approx -0.25$

- Graphique 1 :
  Le coefficient de corrélation linéaire sera négatif car les deux variables varient en sens opposé.
  Il semble y avoir une bonne corrélation donc $\rho_{x,y}$ sera plutôt proche de -1. Ainsi, $\rho_{x,y} \approx -0.79$.
- Graphique 2 :
  Le coefficient de corrélation linéaire sera positif car les deux variables varient dans le même sens.
  Il ne semble y avoir une forte corrélation donc $\rho_{x,y}$ sera plutôt éloigné de 1. Ainsi, $\rho_{x,y} \approx 0.56$.
  La valeur du coefficient de corrélation linéaire est fortement impactée par les valeurs extrêmes. Si on élimine ses valeurs, on trouvera un corrélation plus forte ce que semble indiqué le début du graphique.

2. **On considère à présent la série statistique double (PIB, Taux d'alphabétisation).**

On trace le nuage de point et on donne le coefficient de corrélation linéaire associés à cette série.



   (a) **Une approximation par régression linéaire est-elle pertinente pour modéliser le taux d'alphabétiser par rapport au PIB?**

Le nuage de point n'a pas une forme allongée qui pourrait être modélisée par une droite. Ainsi, une approximation par régression linéaire n'est pas pertinente.

(b) **On considère les instructions suivantes.**

```
>>> X, Y = Pays['PIB'], Pays['Alphabetisation']
>>> Z = np.log(X)
>>> print(CoeffCorr(Z,Y))
0.7786212347927588
```

Une approximation par régression linéaire entre Z et Y est-elle plus pertinente?
Quelle relation peut-on supposer exister entre X et Y?
Le coefficient de corrélation linéaire entre $Z = ln(X)$ et Y est proche de 1. On peut donc supposer que Y est approximativement une fonction affine de $Z = ln(X)$: $Y \approx aZ + b$ soit $Y \approx a\ln(X) + b$.
Illustration sur le graphique ci-dessous.