TP4 – statistiques

A/ Première partie : statistique univariée (1 h maximum)

Situation

Voici deux séries statistiques, sous forme de listes Python (non triées), qui concernent les diamètres en millimètres de tomates :

D1NT=[52,58,58,58,55,53,49,56,57,54,51,59,60,50,51,54,49,59,55,56,53,49,56,58,
57,54,53,56,55,54,55,58,52,59]

D2NT=[58,50,47,51,56,54,61,53,52,50,61,49,59,55,51,59,51,56,52,51,52,48,52,62,
54,50,50,51,50,52,59,49,60,53,61]

On cherche à déterminer quel échantillon de tomates provient d'une culture biologique et quel échantillon provient d'une culture plus « industrielle».

1°) Avec une série statistique non triée

a) Écrire une fonction Python d'entête ***moyenne(L)*** qui prend en argument d'entrée une liste de nombres (entiers ou flottants) L et qui renvoie la moyenne de cette série statistique.
En testant cette fonction avec la liste D1NT, on doit obtenir 54.794117647058826.
En testant cette fonction avec la liste D2NT, on doit obtenir 53.68571428571428.

b) Écrire une fonction Python d'entête ***v_e_t(L)*** (pour **v**ariance et **é**cart-**t**ype) qui prend en argument d'entrée une liste de nombres (entiers ou flottants) L et qui renvoie le tuple (variance, écart-type) de cette série statistique.
En testant cette fonction avec la liste D1NT, on doit obtenir (9.634083044982617, 3.1038819315467876).
En testant cette fonction avec la liste D2NT, on doit obtenir (17.87265306122481, 4.227606067412716).

2°) Avec une série statistique triée

On considère les mêmes listes, mais cette fois triées :

D1T=[49,49,49,50,51,51,52,52,53,53,53,54,54,54,54,55,55,55,55,56,56,56,56,57,57,
58,58,58,58,58,59,59,59,60]

D2T=[47,48,49,49,50,50,50,50,50,51,51,51,51,51,52,52,52,52,52,53,53,54,54,55,56,
56,58,59,59,59,60,61,61,61,62]

a) Écrire une fonction Python d'entête ***mediane(L)*** qui prend en argument d'entrée une liste triée de nombres (entiers ou flottants) L et qui renvoie la médiane de cette série statistique.
En testant cette fonction avec la liste D1T, on doit obtenir 55.0 .
En testant cette fonction avec la liste D2T, on doit obtenir 52 .

b) Écrire une fonction Python d'entête ***quartiles(L)*** qui prend en argument d'entrée une liste triée de nombres (entiers ou flottants) L et qui renvoie le tuple (Q_1, Q_2) des premier et troisième quartiles de cette série statistique.
En testant cette fonction avec la liste D1T, on doit obtenir (53, 58) .
En testant cette fonction avec la liste D2T, on doit obtenir (50, 58) .

3°) Si on a le temps, avant la fin de la première heure

On considère encore les mêmes échantillons mais cette fois avec les valeurs données avec leurs effectifs (sans répétitions) :

D1TSR=[[49,3],[50,1],[51,2],[52,2],[53,3],[54,4],[55,4],[56,4],[57,2],[58,5],[59,3],[60,1]]

D2TSR=[[47,1],[48,1],[49,2],[50,5],[51,5],[52,5],[53,2],[54,2],[55,1],[56,2],[58,1],[59,3],[60,1],[61,3],[62,1]]

a) Écrire une fonction Python d'entête ***diagbat(L)*** (pour **diag**ramme en **bât**ons) qui prend en argument d'entrée une liste L de listes à deux nombres (le premier étant la valeur du caractère, le deuxième étant l'effectif correspondant) et qui trace le diagramme en bâtons de cette série statistique.
On importera pour cela le module *matplotlib.pyplot* et on pourra utiliser une instruction du style *bar(X,Y,0.5,color='r')* dans laquelle X désigne la liste des valeurs du caractère, Y la liste des effectifs correspondants et le flottant 0.5 la largeur des bâtons.

b) Écrire une fonction Python d'entête ***moyenne_bis(L)*** qui prend en argument d'entrée une liste L de listes à deux nombres (le premier étant la valeur du caractère, le deuxième étant l'effectif correspondant) et qui renvoie la moyenne de cette série statistique.
(On doit obtenir avec les listes D1TSR et D2TSR les mêmes résultats que dans la question 1°)a).)

c) Écrire une fonction Python d'entête ***variance_bis(L)*** qui prend en argument d'entrée une liste L de listes à deux nombres (le premier étant la valeur du caractère, le deuxième étant l'effectif correspondant) et qui renvoie la variance de cette série statistique.
(On doit obtenir avec les listes D1TSR et D2TSR les mêmes résultats que dans la question 1°)b).)

d) Écrire une fonction Python d'entête ***etendue(L)*** qui prend en argument d'entrée une liste (a priori non triée) de nombres (entiers ou flottants) L et qui renvoie l'étendue de cette série statistique.
En testant cette fonction avec la liste D1NT, on doit obtenir 11 .
En testant cette fonction avec la liste D2NT, on doit obtenir 15 .

e) Écrire une fonction Python d'entête ***deciles(L,p)*** qui prend en arguments d'entrée une liste triée de nombres (entiers ou flottants) L ainsi qu'un entier naturel p avec $1 \leq p \leq 9$, et qui renvoie le $p^{ème}$ décile de cette série statistique.
En testant cette fonction avec la liste D1T par exemple, le 7ème décile est 57 .

f) Écrire une fonction Python d'entête ***un_mode(L)*** qui prend en argument d'entrée une liste L de listes à deux nombres (le premier étant la valeur du caractère, le deuxième étant l'effectif correspondant) et qui renvoie un mode de cette série statistique.

g) Écrire une fonction Python d'entête ***modes(L)*** qui prend en argument d'entrée une liste L de listes à deux nombres (le premier étant la valeur du caractère, le deuxième étant l'effectif correspondant) et qui renvoie la liste du/des mode(s) de cette série statistique.
En testant cette fonction avec la liste D1TSR, on doit obtenir [58] .
En testant cette fonction avec la liste D2TSR, on doit obtenir [50, 51, 52] .

B/ Statistique bivariée

Situation

Une enquête du service commercial d'une chaîne d'hôtels « quatre étoiles » a permis de connaître l'évolution de la demande y_i de nuitées, par rapport au prix x_i proposé en période « creuse » :

Prix TTC en euros (x_i)	80	100	120	140	160	180	200
Demande mensuelle en nuitées (y_i)	540	452	335	188	120	68	18

Le but de l'exercice est de donner une estimation de la demande mensuelle en nuitées pour un prix de 130 euros.

On utilisera la liste de listes à deux éléments Python suivante :

H=[[80,540],[100,452],[120,335],[140,188],[160,120],[180,68],[200,18]]

1°) Écrire une fonction Python d'entête ***pm(L)*** (pour **p**oint **m**oyen) qui prend en argument d'entrée une liste L de listes à deux nombres (entiers ou flottants) et qui renvoie le tuple des coordonnées du point moyen du nuage de points correspondant.
En testant cette fonction avec la liste H, on doit obtenir (140.0, 245.85714285714286) .

2°) Écrire une fonction Python d'entête ***covariance(L)*** qui prend en argument d'entrée une liste L de listes à deux nombres (entiers ou flottants) et qui renvoie la covariance du couple des deux caractères considérés.
En testant cette fonction avec la liste H, on doit obtenir -7282.857142857141 .

3°) Écrire une fonction Python d'entête ***ccl(L)*** (pour **c**oefficient de **c**orrélation **l**inéaire) qui prend en argument d'entrée une liste L de listes à deux nombres (entiers ou flottants) et qui renvoie le coefficient de corrélation linéaire du couple des deux caractères considérés.
En testant cette fonction avec la liste H, on doit obtenir -0.9839114765143085 .

4°) Écrire une fonction Python d'entête ***caa(L)*** (pour **c**oefficients de l'**a**justement **a**ffine) qui prend en argument d'entrée une liste L de listes à deux nombres (entiers ou flottants) et qui renvoie le tuple (a, b) tel que la droite d'ajustement affine du nuage de points considéré est d'équation $y = ax + b$.
En testant cette fonction avec la liste H, on doit obtenir (-4.551785714285713, 883.1071428571427) .

5°) (Si on a le temps) Écrire une fonction Python d'entête ***illustration(L)*** qui prend en argument d'entrée une liste L de listes à deux nombres (entiers ou flottants) et qui trace sur un même graphique le nuage de points correspondant ainsi que la droite d'ajustement affine.