

Plaquette 4 – Traiter des données réelles

Algorithmique et Python — Première STMG

Corrections détaillées

Boîte à outils

Une liste de listes. Chaque élément est une ligne de données : *ventes[2]* est la troisième commande, *ventes[2][0]* sa ville, *ventes[2][2]* son montant.

Sélectionner, compter, cumuler.

Objectif	Instruction
garder les commandes web	<code>[v for v in ventes if v[1] == "web"]</code>
les compter	<code>len(...)</code> de la liste obtenue
cumuler leurs montants	<code>sum(v[2] for v in ventes if v[1] == "web")</code>

Tableau croisé. On compte (ou on cumule) pour **chaque couple** de valeurs des deux critères, avec deux boucles imbriquées.

Lire un fichier CSV. Chaque ligne est un texte, que l'on découpe avec `split(";")` ; un montant lu dans un fichier est un **texte** : il faut le convertir avec `float` avant de calculer.

Proportion et fréquence conditionnelle.

$$p = \frac{\text{effectif de la partie}}{\text{effectif total}} \quad f_A(B) = \frac{\text{effectif de } A \cap B}{\text{effectif de } A}$$

Correction de l'exercice 1 - Lire une liste de listes

ventes contient 8 éléments, chacun étant une liste de trois données : `len(ventes)` vaut 8.

ventes[2] est la **troisième** commande : `["Paris", "magasin", 150]`.

ventes[2][2] est le troisième élément de cette commande, son montant : 150.

ventes[0][1] est le deuxième élément de la première commande, son canal : "web".

Le premier indice choisit la **ligne**, le second la **colonne** : on lit la donnée comme dans un tableau à 8 lignes et 3 colonnes, numérotées à partir de 0.

commande $k \rightarrow$ indice $k - 1$

Réponse. 8 ; `["Paris", "magasin", 150]` ; 150 ; "web".

Correction de l'exercice 2 - Sélectionner les commandes en ligne

a) La compréhension garde les commandes dont le canal (*v[1]*) vaut "web". Ce sont les commandes de 120 €, 60 €, 90 € et 200 € : *web* contient 4 éléments.

b) On cumule les montants (*v[2]*) de ces commandes :

```
total_web = sum(v[2] for v in web)
```

$$120 + 60 + 90 + 200 = 470$$

Les commandes en ligne représentent 470 €. Le double signe égal `==` sert à **comparer** ; le simple `=` sert à **affecter**.

Réponse. a) 4 commandes — b) 470 €.

Correction de l'exercice 3 - Compter selon un critère

On parcourt les commandes avec un **compteur** :

```
def nb_commandes(ventes, ville):
    n = 0
    for v in ventes:
        if v[0] == ville:
            n = n + 1
    return n
```

Pour Paris, les commandes concernées sont celles de 120 €, 150 € et 200 € : la fonction renvoie 3.

La proportion des commandes parisiennes est

$$\frac{3}{8} = 0,375$$

soit 37,5 % des commandes.

Réponse. `nb_commandes(ventes, "Paris")` renvoie 3.

Correction de l'exercice 4 - Montant moyen d'une commande

On cumule la troisième colonne, puis on divise par le nombre de commandes :

```
total = sum(v[2] for v in ventes)
moyenne = total / len(ventes)
```

$$\text{total} = 120 + 80 + 150 + 60 + 90 + 200 + 70 + 110 = 880$$

et la moyenne vaut $\frac{880}{8} = 110$ €. Python affiche `110.0`.

Le panier moyen est de 110 €, alors que les commandes vont de 60 € à 200 € : la moyenne résume la série, mais cache cette dispersion.

Réponse. Total 880 €, moyenne 110 €.

Correction de l'exercice 5 - Total par canal de vente

On utilise un **accumulateur** :

```
def total_canal(ventes, canal):
    s = 0
    for v in ventes:
        if v[1] == canal:
            s = s + v[2]
    return s
```

`total_canal(ventes, "web")` renvoie 470 et `total_canal(ventes, "magasin")` renvoie 80 + 150 + 70 + 110 = 410. Contrôle : 470 + 410 = 880, le total général.

La part du chiffre d'affaires en ligne est

$$\frac{470}{880} \approx 0,534$$

soit environ 53,4 %, alors que la moitié seulement des commandes sont passées en ligne : les commandes web sont en moyenne plus importantes.

Réponse. 470 € et 410 € ; part web environ 53,4 %.

Correction de l'exercice 6 - Un tableau croisé des effectifs

Pour chaque couple (ville, canal), on compte les commandes qui vérifient **les deux** critères :

```
n = len([v for v in ventes if v[0] == ville and v[1] == canal])
```

Les deux boucles imbriquées parcourent les $3 \times 2 = 6$ couples. On obtient le tableau croisé :

Ville	web	magasin	Total
Paris	2	1	3
Lyon	1	2	3
Lille	1	1	2
Total	4	4	8

Les totaux se contrôlent : $3 + 3 + 2 = 8$ et $4 + 4 = 8$.

Réponse. Paris : 2 et 1 ; Lyon : 1 et 2 ; Lille : 1 et 1.

Correction de l'exercice 7 - Un tableau croisé des montants

On remplace le comptage par une somme des montants :

```
s = sum(v[2] for v in ventes if v[0] == ville and v[1] == canal)
```

Ville	web	magasin	Total
Paris	320	150	470
Lyon	90	190	280
Lille	60	70	130
Total	470	410	880

Paris réalise plus de la moitié du chiffre d'affaires :

$$\frac{470}{880} \approx 0,534$$

et ses ventes en ligne pèsent à elles seules 320 €, plus que tout le chiffre d'affaires de Lyon en magasin.

Réponse. Paris : 320 et 150 ; Lyon : 90 et 190 ; Lille : 60 et 70.

Correction de l'exercice 8 - Proportions

a) On rapporte le nombre de commandes web au nombre total :

$$\frac{4}{8} = 0,5$$

En Python : `len(web) / len(ventes)`. La moitié des commandes est passée en ligne.

b) On rapporte le montant des commandes web au montant total : $\frac{470}{880} \approx 0,534$, soit environ 53,4 %.

Les deux proportions n'ont pas la même population de référence : la première compte des **commandes**, la seconde des **euros**. Elles diffèrent parce que les commandes en ligne ont un montant moyen plus élevé ($\frac{470}{4} = 117,50$ € contre $\frac{410}{4} = 102,50$ € en

magasin).

Réponse. a) 50 % — b) environ 53,4 %.

Correction de l'exercice 9 - Une fréquence conditionnelle

On se restreint aux commandes parisiennes, puis on compte celles qui sont en ligne :

```
paris = [v for v in ventes if v[0] == "Paris"]
web_paris = [v for v in paris if v[1] == "web"]
f = len(web_paris) / len(paris)
```

Il y a 3 commandes à Paris, dont 2 en ligne :

$$f_{\text{Paris}}(\text{web}) = \frac{2}{3} \approx 0,667$$

Les Parisiens commandent en ligne dans environ 67 % des cas, contre 50 % pour l'ensemble des clients. La population de référence est celle de Paris, et non l'ensemble des commandes.

Réponse. $\frac{2}{3} \approx 0,667$.

Correction de l'exercice 10 - Lire un fichier de données

La boucle lit le fichier ligne par ligne ; chaque ligne est un **texte**, par exemple "Paris;web;120" suivi d'un retour à la ligne.

- *strip()* enlève le retour à la ligne et les espaces en fin de texte ;
- *split(";")* découpe le texte à chaque point-virgule et renvoie la liste ["Paris", "web", "120"] ;
- *float(champs[2])* convertit le texte "120" en nombre 120,0, pour pouvoir faire des calculs.

À la fin, *ventes* est la même liste de listes que celle saisie à la main, et l'on peut calculer par exemple

$$\text{total} = \sum \text{montants} = 880$$

L'intérêt est de traiter des fichiers de milliers de lignes, impossibles à recopier.

Réponse. *strip* nettoie la ligne, *split* la découpe, *float* convertit le montant en nombre.

Correction de l'exercice 11 - Repérer une erreur de type

Sans *float*, les montants restent des **textes** (chaînes de caractères).

- Pour deux textes, + les **met bout à bout** : "120" + "80" donne "12080", et non 200.
- La comparaison de deux textes se fait **caractère par caractère**, comme dans un dictionnaire : "1" vient avant "8", donc "120" < "80", et "120" > "80" renvoie *False*.

Après conversion, *float("120") > float("80")* renvoie *True*, car

$$120 > 80$$

À retenir. Toute donnée numérique lue dans un fichier doit être convertie avant d'être additionnée ou comparée.

Réponse. Les montants sont des textes : + les concatène et la comparaison est alphabétique.

Correction de l'exercice 12 - La plus grosse commande

On garde dans m la commande la plus importante rencontrée, en comparant les montants ($[2]$) :

```
m = ventes[0]
for v in ventes:
    if v[2] > m[2]:
        m = v
print(m)
```

Le programme affiche $["Paris", "web", 200]$: la plus grosse commande est une commande en ligne parisienne de 200 €.

Elle représente

$$\frac{200}{880} \approx 0,227$$

soit près d'un quart du chiffre d'affaires à elle seule.

Réponse. $["Paris", "web", 200]$.

Correction de l'exercice 13 - Le point moyen avec deux listes

La fonction renvoie la somme divisée par le nombre d'éléments :

```
def moyenne(L):
    return sum(L) / len(L)
```

Le point moyen a pour coordonnées $moyenne(x)$ et $moyenne(y)$:

$$\bar{x} = \frac{24}{6} = 4 \quad \bar{y} = \frac{183}{6} = 30,5$$

Python renvoie 4.0 et 30.5 : le point moyen est $G(4; 30,5)$.

La même fonction sert aux deux listes : c'est l'intérêt d'écrire une fonction plutôt que de répéter le calcul.

Réponse. $G(4; 30,5)$.

Correction de l'exercice 14 - Évolution par magasin

a) Pour chaque indice i , on applique la formule du taux d'évolution :

```
[round((a24[i] - a23[i]) / a23[i] * 100, 1) for i in range(3)]
```

On obtient $[10.0, -5.0, 5.0]$: +10 %, −5 % et +5 %.

b) Les totaux valent 750 en 2023 et 782 en 2024 :

$$\frac{782 - 750}{750} \times 100 \approx 4,27$$

Le chiffre d'affaires total progresse d'environ 4,27 %. Ce n'est pas la moyenne des trois taux ($\frac{10 - 5 + 5}{3} \approx 3,33\%$) : les magasins n'ont pas le même poids.

Réponse. a) $[10.0, -5.0, 5.0]$ — b) environ +4,27 %.

Correction de l'exercice 15 - Synthèse : une enquête de satisfaction

a) Deux boucles imbriquées comptent chaque couple :

```
for age in ["-25", "25-50", "50+"]:
```

```
for s in ["oui", "non"]:
    print(age, s, len([r for r in rep if r[0] == age and r[1] == s]))
```

Âge	oui	non	Total
-25	3	1	4
25-50	3	1	4
50+	3	1	4
Total	9	3	12

b) La proportion de satisfaits est $\frac{9}{12} = 0,75$. Dans chaque tranche d'âge, la fréquence des satisfaits vaut aussi

$$\frac{3}{4} = 0,75$$

La satisfaction ne dépend pas de l'âge dans cet échantillon : les fréquences conditionnelles sont toutes égales à la fréquence globale, ce qui correspond à l'indépendance vue en probabilités.

Réponse. b) 75 % de satisfaits, et 75 % dans chaque tranche d'âge.