projet *Car Model Names*

Inspiration:
https://xkcd.com/1571/
https://www.explainxkcd.com/wiki/index.php/1571:_Car_Model_Names

1. Pour une chaîne comportant des noms de modèle de voiture, faire un décompte de toutes les lettres stocké dans un tableau de 26 compteurs.
2. Appliquer la même méthode sur une autre chaîne de caractères de texte général (wikipedia?) pour avoir un point de comparaison.
3. En appliquant les comptages bruts, produire la liste des scores de chaque lettre obtenus par la formule Score(x) = Frequency_in_cars(x) - Frequency_in_French(x). *Essayer d'avoir des chaînes de longueurs comparables pour que les comptages soient eux-mêmes comparables.*
4. Calculer le score de quelques noms de marque existants.
5. Prendre des mots au hasard et calculer leur score.
6. Générer des mots possibles avec des scores élevés ou faibles.

TP Bonus : rendre le code pour chacune des questions, accompagné d'un document texte donnant les résultats produits par le programme et explicitant les choix effectués. Le texte ne doit pas faire plus de deux pages. Le tout doit être envoyé ou déposé sur iCampus avant le 28 novembre 2025 à 16h.

Proposition de correction

Question 1 : décompte des lettres d'une chaîne de caractères

[9]:
```
modeles = "Peugeot Yaris Honda xls berlingot C3 aircross holidays captur Civic␣
 ↪Jazz prelude ix3 carrera ami8 aronde"
compteurs_marques = [0] * 26
for l in modeles:
    indice = ord(l.lower()) - ord('a')
    if indice >= 0 and indice <= 25:
        compteurs_marques[indice] += 1
print(compteurs_marques)
```

```
[10, 1, 6, 4, 7, 0, 2, 2, 8, 1, 0, 4, 1, 3, 6, 3, 0, 10, 5, 3, 3, 1, 0, 2, 2, 2]
```

Question 2 : décompte des lettres d'un texte quelconque *Remarque:* Le code précédent peut être réutilisé quasiment à l'identique

[10]:
```
texte = "Ce projet vise à offrir un contenu librement réutilisable, objectif et␣
 ↪vérifiable, que chacun peut modifier et améliorer."
compteurs_texte = [0] * 26
for l in texte:
    indice = ord(l.lower()) - ord('a')
    if indice >= 0 and indice <= 25:
        compteurs_texte[indice] += 1
print(compteurs_texte)
```

```
[4, 4, 5, 1, 15, 5, 0, 1, 11, 2, 0, 5, 3, 5, 6, 2, 1, 9, 2, 8, 6, 2, 0, 0, 0, 0]
```

Discussion: On a noté que le code qui a été utilisé dans les deux cas précédent est le même *à quelques exceptions près*: ces exceptions constituent ce qu'on appelle les *paramètres* du (sous-)programme. Dans le cas présent, les paramètres sont (1) le texte (`modeles` ou `texte`) et (2) les compteurs. Ci dessous on propose une *fonction* qui travaille précisément avec ces deux paramètres. La liste des compteurs est un résultat du calcul, elle est renvoyée par la fonction.

[11]:
```
def decompte(txt):
    compteurs = [0] * 26
    for l in txt:
        indice = ord(l.lower()) - ord('a')
        if indice >= 0 and indice <= 25:
            compteurs[indice] += 1
    return compteurs
```

On peut alors répondre aux questions 1 et 2 en utilisant la même fonction, avec des paramètres différents.

[12]:
```
modeles = "Peugeot Yaris Honda xls berlingot C3 aircross holidays captur Civic␣
 ↪Jazz prelude ix3 carrera ami8 aronde"
compteurs_marques = decompte(modeles)
texte = "Ce projet vise à offrir un contenu librement réutilisable, objectif et␣
 ↪vérifiable, que chacun peut modifier et améliorer."
compteurs_texte = decompte(texte)
```

Question 3: Scores pour chaque lettre

[13]:
```
scores = []
for i in range(26):
    scores.append(compteurs_marques[i] - compteurs_texte[i])
print(scores)
```

```
[6, -3, 1, 3, -8, -5, 2, 1, -3, -1, 0, -1, -2, -2, 0, 1, -1, 1, 3, -5, -3, -1,
0, 2, 2, 2]
```

Question 4: calculer le score de quelques noms de marque Il y a plusieurs manières d'envisager le score. On peut faire la somme des scores individuels de chaque lettre, par exemple le mot "abc" aurait alors le score $ 6+(-3)+1 $, mais les scores vont varier beaucoup en fonction de la taille des mots, il semble préférable de diviser le score par le nombre de lettres, ce qui revient à calculer le score moyen.

[14]:
```
def calcule_score(mot):
    total = 0
    for l in mot:
        indice = ord(l)-ord('a')
        if indice >= 0 and indice < 26:
            total = total + scores[indice]
    return total / len(mot)
```

[15]:
```
# Quelques essais
print(calcule_score("peugeot"))
print(calcule_score("yaris"))
print(calcule_score("kow"))
```

```
-3.0
1.8
0.0
```

Question 5 : scores de mots au hasard

[16]:
```
lmots = ["pain", "maison", "xeres", "plastic", "xyz"]
for m in lmots:
    print("Score de %s : %.3f" % (m,calcule_score(m)))
```

```
Score de pain : 0.500
Score de maison : 0.333
Score de xeres : -2.000
Score de plastic : 0.286
Score de xyz : 2.000
```

Question 6 : générer des mots ayant des scores élevés ou faibles. Première méthode: on prend des mots au hasard, et on sélectionne le mot qui obtient le score le plus bas (on peut faire la même chose avec le plus haut). Deuxième méthode: on peut chercher les lettres qui ont les scores les plus bas (ou haut), et essayer de former des mots avec elles. Cette méthode n'est pas implémentée dans ce corrigé.

[17]:
```
long_texte = """Wikipédia est définie par des principes fondateurs.
Son contenu est sous licence Creative Commons BY-SA.
Il peut être copié et réutilisé sous la même licence, sous réserve
d'en respecter les conditions. Wikipédia fournit tous ses contenus
gratuitement, sans publicité, et sans recourir à l'exploitation
des données personnelles de ses utilisateurs."""

smin = calcule_score('maison')
mmin = 'maison'
for m in long_texte.split():
    s = calcule_score(m.lower())
    if s < smin:
        smin = s
        mmin = m
print('Le mot ayant le score le plus bas est "%s" (score %.3f)' % (mmin, smin))
```

```
Le mot ayant le score le plus bas est "et" (score -6.500)
```