lettresEtTexte

Documents d'accompagement
Algorithmique et programmation
nouveau programme du lycée 2019

1 Codage d'un message par substitution

1.1 Présentation de l'activité

- **Niveau de classe :** classe de première de la voie générale (spécialité mathématiques).
- **Références au programme :**
- spécialité mathématiques de première générale : *Fréquence d'apparition des lettres d'un texte donné, en français, en anglais*.
- spécialité mathématiques de première générale : *Calcul de factorielle*.
- **Description :** activité présentant une méthode de déchiffrement de messages chiffrés par substitution.

Situation Le chiffrement par substitution simple (ou alphabet désordonné) consiste à remplacer dans un message chacune des lettres de l'alphabet par une autre.

Par exemple, si la lettre m est remplacée par t et la lettre i est remplacée par la lettre o alors le mot **mimi** est remplacé par **toto**.

La substitution est définie par une permutation des lettres de l'alphabet. Nous nous intéressons ici au décodage des messages chiffrés par substitution. Nous nous limiterons aux messages écrits en français sans accent, sans majuscule, sans espace et sans ponctuation. L'alphabet utilisé comporte donc 26 lettres minuscules.

1.2 Nombre de chiffrements par substitution

Chaque permutation des 26 lettres fournit une possibilité de chiffrement. Pour décoder un message chiffré par substitution, on pourrait lister toutes les permutations possibles.

Le nombre de permutations est égal à $26 \times 25 \times 24 \times ... \times 1 = 26!$.

La fonction suivante permet le calcul de la factorielle d'un entier naturel n. Elle prend en paramètre un entier naturel n et renvoie $n!$.

```
def factorielle(n):
    fact=1
    for i in range(1,n+1):
        fact=fact*i
    return fact
print(factorielle(26))
```

```
403291461126605635584000000
```

Suggestions pédagogiques

- **Mathématiques débranchées**

  Dans le cas où on ne présente pas un alphabet comme une permutation de l'ensemble $\{a, b, \ldots, y, z\}$, demander aux élèves de calculer directement le nombre de chiffrements possibles.

- **Expliquer un programme**

  Que fait la ligne 3 ?

- **Compléter un programme**

  Le programme précédent étant fourni en remplaçant les lignes 2 et 4 par `fact = ...` (dans ces deux lignes), demander aux élèves de compléter les lignes 2 et 4.

- **Tester** la fonction `factorielle` pour donner le nombre de permutations possibles pour un alphabet de 26 lettres. Conclure sur la faisabilité de la méthode de déchiffrement basée sur l'examen de tous les chiffrements possibles.

2 Utilisation de la fréquence d'apparition des lettres.

Pour déchiffrer un message codé par substitution, il est plus efficace d'utiliser l'analyse fréquentielle. Celle-ci utilise le fait que, dans une langue donnée, la fréquence d'apparition d'une lettre donnée varie peu d'un texte à l'autre.

2.1 Détermination de la fréquence d'une lettre dans un texte

La fonction `frequenceLettre` prend en paramètres une lettre et un texte (chaines de caractères) en paramètres et renvoie la fréquence d'apparition de la lettre dans le texte.

On la teste ici pour un texte de 206 caractères extrait de Cyrano de Bergerac (sans majuscule, accent, espace ou ponctuation) et la lettre a.

```
texte="ahnoncestunpeucourtjeunehommeonpouvaitdireohdieubiendeschosesensomme

def frequenceLettre(lettre,texte):
    compteur=0
    for l in texte:
        if l==lettre:
            compteur=compteur+1
    return compteur/len(texte)

print(frequenceLettre("a",texte))
```

```
0.08737864077669903
```

Suggestions pédagogiques

- **Expliquer un programme**

  Quel est le type de la variable `l` dans la ligne 5 ?

- **Compléter un programme**

  Le programme précédent étant fourni en remplaçant les lignes 4,6 et 7 par `compteur = ...`, `if l==...` et `compteur = ...`, demander aux élèves de compléter les lignes 4, 6 et 7.

2.2 Fluctuation de la fréquence d'apparition d'une lettre en fonction de la longueur du texte

Pour des textes longs, la fréquence d'apparition observée est proche d'une valeur qui peut s'interpréter comme fréquence d'apparition de la lettre dans la langue française. On étudie ici un texte comprenant 1917 caractères.

2.3 Remarque :

Le code texte[:i] permet d'extraire les i premiers caractères du texte.

Importation de la bibliothèque graphique.

```
%matplotlib inline
import matplotlib.pyplot as plt
```

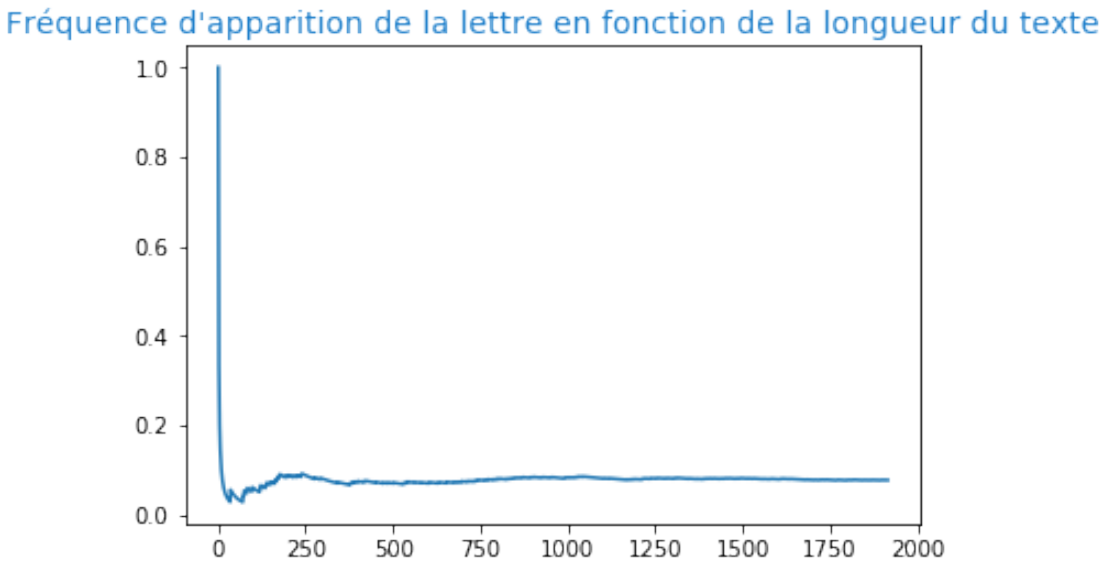
Texte à déchiffrer.

```
texte="ahnoncestunpeucourtjeunehommeonpouvaitdireohdieubiendeschosesensomme
```

Le programme suivant permet de représenter la fréquence d'apparition de la lettre a dans un extrait de ce texte en fonction de la longueur de l'extrait.

```
N=len(texte)
frequences = []
somme = 0
for i in range(1,N):
    frequences.append(frequenceLettre("a",texte[:i]))

plt.title("Fréquence d'apparition de la lettre en fonction de la longueur du
plt.plot(frequences)
plt.show()
print(frequenceLettre("a",texte))
```




```
0.07720396452790819
```

Suggestions pédagogiques

- **Expliquer un programme**

  Expliquer les paramètres du `range` de la ligne 4.

- **Tester** ce programme en modifiant la lettre recherchée.

  Estimer la fréquence d'apparition du e et du s.

- **Compléter un programme**

  Le programme précédent étant fourni en remplaçant la ligne 5 par `frequences.append(...)`, demander aux élèves de compléter la ligne 5.

2.4 Alphabet en Python

La fonction `alphabet` ci-dessous permet de générer une liste contenant les 26 lettres de l'alphabet grâce à la fonction `chr`. Cette fonction renvoie la chaîne représentant un caractère dont le code de caractère Unicode est donné. Par exemple, `chr(97)` renvoie la chaîne de caractères 'a'

```
def alphabet():
    return [chr(i) for i in range(97,123)]

print(alphabet())
```

```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's'
```

Suggestions pédagogiques

- **Tester** la fonction `alphabet` afin de donner les valeurs de `chr(98)` et `chr(97+25)`.
- **Expliquer un programme**

  Que fait la ligne 2 ?

2.5 Liste des fréquences des lettres de l'alphabet

La fonction `listeFrequences` donne la liste des fréquences des 26 lettres de l'alphabet dans un texte. Elle prend en paramètre une chaine de caractères et renvoie une liste contenant la fréquence d'apparition des lettres de l'alphabet (toutes dans l'ordre alphabétique) de la chaine de caractères.

```
def listeFrequences(texte):
    alpha=alphabet()
    liste=[]
    for l in alpha:
        liste.append(frequenceLettre(l,texte))
    return liste
print(listeFrequences(texte))
```

```
[0.07720396452790819, 0.005216484089723527, 0.03547209181011998, 0.026604068857589983, 0.16536
```

Suggestions pédagogiques

- **Compléter un programme**

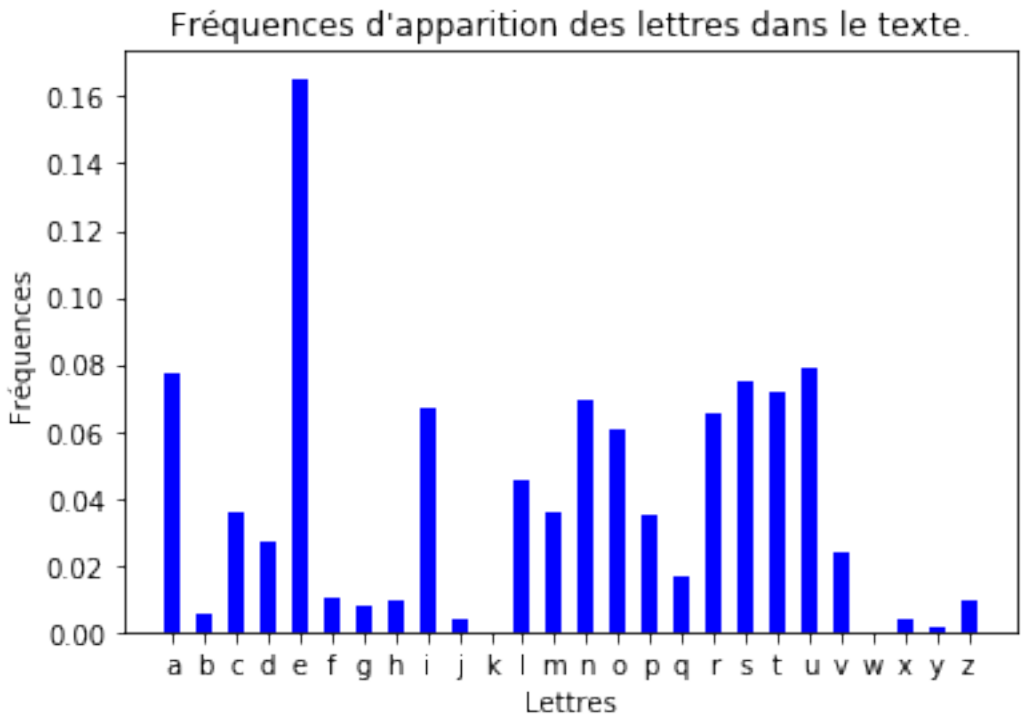
Le programme précédent étant fourni en remplaçant les lignes 4 et 5 par `for ...` et `liste.append(...)`, demander aux élèves de compléter les lignes 4 et 5. - **Écrire un programme**

Écrire la fonction `listeFrequences`.

2.6 Diagramme en bâtons des fréquences d'apparition des lettres de l'alphabet

Le programme suivant permet de représenter les fréquences des lettres apparaissant dans le texte sous la forme d'un diagramme en bâtons.

```
ep = 0.5
lettres = [lettre for lettre in alphabet()]
plt.bar(lettres, listeFrequences(texte), ep, color='b' )
plt.xlabel('Lettres')
plt.ylabel('Fréquences')
plt.title("Fréquences d'apparition des lettres dans le texte.")
plt.show()
```

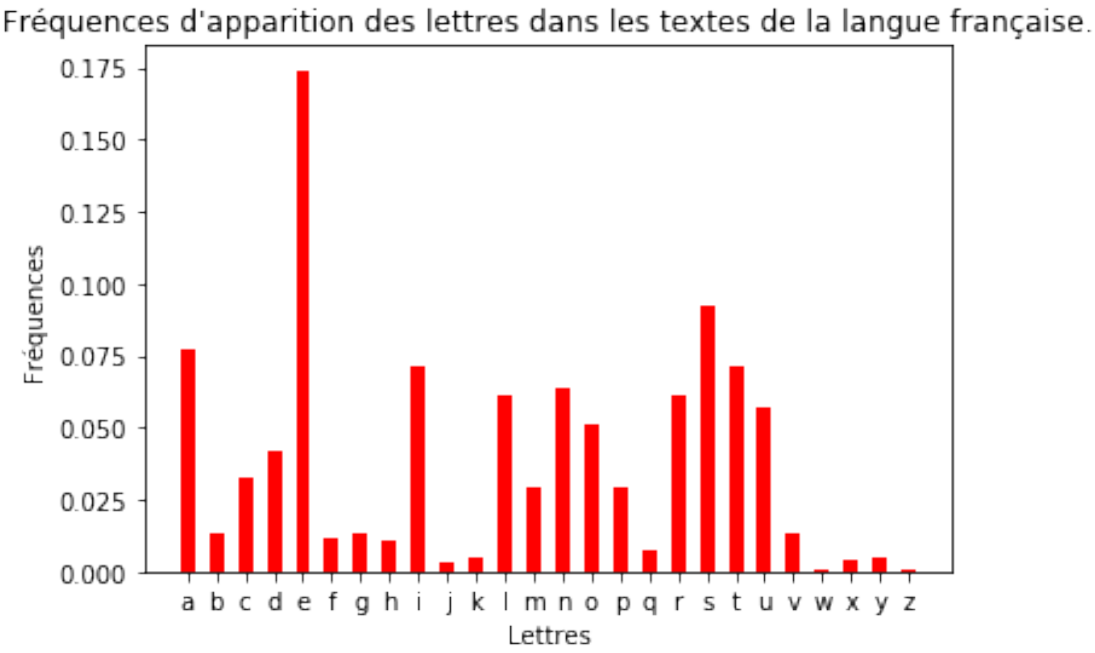


2.7 Fréquences d'apparition des lettres dans la langue française

Le programme suivant représente la fréquence d'apparition des lettres dans les textes en langue française. Il ne s'agit ici que d'estimations. La nature du texte peut en effet faire varier légérement ces fréquences. La liste donnant ces fréquences pourra être donnée ou remplacée par une autre (Wikipedia).

L'objectif est ici de comparer cette représentation à la précédente.

```
frequencesFrance=[0.077,0.013,0.033,0.042,0.174,0.012,0.013,0.011,0.071,0.00
ep = 0.5
lettres = [lettre for lettre in alphabet()]
plt.bar(lettres, frequencesFrance, ep, color='r' )
plt.xlabel('Lettres')
plt.ylabel('Fréquences')
plt.title("Fréquences d'apparition des lettres dans les textes de la langue
plt.show()
```



2.8 Message chiffré

Le message ci-dessous a été chiffré à l'aide d'une substitution simple à partir d'un message écrit en français.

rertepadmneppobewogexaemefopwtvvdpexeaweaxdtxeofxepwerzwobe

Suggestions pédagogiques

- Déchiffrer le message en étudiant les fréquences d'apparition des lettres du message.

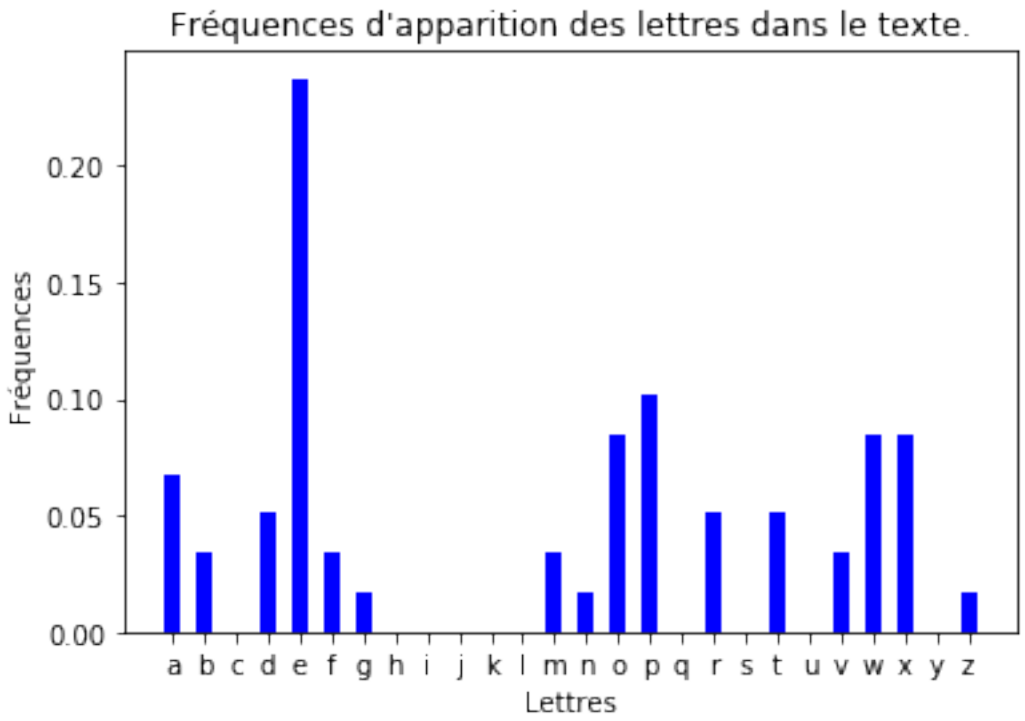
2.9 Fiche enseignant

Correction de la dernière question.

```
message="rertepadmneppobewogexaemefopwtvvdpexeaweaxdtxeofxepwerzwobe"
ep = 0.5
lettres = [lettre for lettre in alphabet()]

plt.bar(lettres, listeFrequences(message), ep, color='b' )
plt.xlabel('Lettres')
plt.ylabel('Fréquences')
plt.title("Fréquences d'apparition des lettres dans le texte.")
```

```
Out[10]: Text(0.5,1,"Fréquences d'apparition des lettres dans le texte.")
```



L'alphabet de substitution est ['o', 'i', 'r', 'w', 'e', 'v', 'b', 'u', 't', 'h', 'y', 'g', 'n', 'm', 'z', 'f', 'l', 'x', 'p', 'a', 'd', 'k', 'j', 'c', 'q', 's'] le message est le suivant :

ceciestunmessagedalertenepasdiffuseretdetruireapresdecodage .

On pourra remarquer que les fréquences aident au déchiffrement mais ne suffisent pas dans le cas d'un message court. En effet, la fluctuation liée à la taille de l'échantillon est importante.

Le programme ci-dessous n'est pas destiné aux élèves. Il permet de faire le chiffrement d'un message par substitution en utilisant un alphabet de substitution aléatoire. L'enseignant pourra l'utiliser pour proposer d'autres messages chiffrés.

```
from random import shuffle
texte="ceciestunmessagedalertenepasdiffuseretdetruireapresdecodage"
texteChiffre=""
alpha=alphabet()
#création de l'alphabet à l'aide de la fonction shuffle effectuant une perm
alphabetSubstitution=alphabet()
shuffle(alphabetSubstitution)
#Affichage de l'alphabet de substitution
print(alphabetSubstitution)
#Création du message chiffré
for x in texte:
    for i in range(26):
        if x==alpha[i]:
            texteChiffre+=alphabetSubstitution[i]
print(texteChiffre)
```

```
['e', 'p', 'x', 'l', 'f', 'm', 'r', 'n', 'j', 'b', 'd', 'y', 'o', 'i', 'w', 'a', 'z', 'h', 'v'
xfxjfvukiofvverfleyfhufifaevljmmkvfhfulfuhkjhfeahfvlfxwlerf
```