Crypto : chiffrement affine et chiffrement de Vigenère

Exercice 1 (0 points)

On donne le code ci-dessous :

```
def vigenere(message, clef):
    n = len(message)
    k = len(clef)
    chiffre = ""
    i = 0
    while i < n:
        z = ord(message[i]) - 65
        d = ord(clef[i%k]) - 65
        z = (z + d)%26
        c = chr(z + 65)
        chiffre += c
        i += 1
    return chiffre

print(vigenere("SECRET", "ZUT"))
```

a) Donner le chiffre qui est affiché à la console après exécution de ce code.

b) Comment peut-on justifier l'utilisation de l'opérateur modulo à la ligne 8 ?

c) Quelle est la fonction de chiffrement utilisée à la ligne 9 ?

a) RYVQYM

b) Cela permet de « recopier en boucle » le mot de passe « en dessous » du texte à chiffrer

0 1 2 0 1 2
ZUTZUT
SECRET
0 1 2 3 4 5

$k = 3$

$\mod k = 3$

0 1 2

c) $f(z) = (z + d) \mod 26$ $d \in \mathbb{Z}_{26}$ est la fonction du chiffre de César.

Exercice 2 (0 points)

On a chiffré un message avec la clef `VALLON` et on a obtenu le chiffre `XOYQWQZNETSY`

a) Déchiffrer ce message.

b) Ecrire la fonction `dech_vigenere(chiffre, clef)` qui permet de retrouver le message automatiquement à partir du chiffre et de la clef.

a) CONFIDENTIEL

b)
```
def dech_vigenere(chiffre, clef):
    k = len(clef)
    n = len(chiffre)
    message = ""
    i = 0
    while i < n:
        z = ord(chiffre[i]) - 65
        d = ord(clef[i % k]) - 65
        z = (z - d) % 26
        c = chr(z) + 65
        message += c
        i += 1
    return message
```

Exercice 3 (0 points)

On a chiffré le message `TRESJOLI` en utilisant le chiffrement de Vigenère et obtenu le chiffre `FRXZBALB`.

a) Trouver la clef de chiffrement.

b) Quel est l'affichage à la console après la ligne ci-dessous ?

```
print(dech_vigenere("FRXZBALB", "TRESJOLI"))
```

A' expliquer.

2) MATHS

b) MATHSMAT

On chiffre comme suit :

M → F

$z' = (z + d) \bmod 26$

$\Leftrightarrow d = (z' - z) \bmod 26$

f déchiffrement

12

MATHMATH

TRESJOLI

19

$z' = (19 + 12) \% 26 = 5$ F

$12 = (z' - 19) \% 26$

$d = (5 - 19) \% 26 = 12$

appliquée au couple :

(chiffre, message)

et qui renvoie la clef qui se répète

en fonction de la longueur du message.

Exercice 4 (0 points)

Donner les éléments de $\mathbb{Z}_{26}^*$. Justifier.

$\mathbb{Z}_{26}^* = \{1; 3; 5; 7; 9; 11; 15; 17; 19; 21; 23; 25\}$

$3 \cdot 9 = 27 \equiv 1 \mod 26$

$5 \cdot 21 = 105 \equiv 1 \mod 26$

$7 \cdot 15 = 105$

$11 \cdot 19 = 209 = 8 \cdot 26 + 1 \equiv 1 \mod 26$

$17 \cdot 23 = 391 = 15 \cdot 26 + 1 \equiv 1 \mod 26$

$25 \cdot 25 = 625 = 24 \cdot 26 + 1 \equiv 1 \mod 26$

Exercice 5 (0 points)

Soit $(a, b) \in \mathbb{Z}_{26}^* \times \mathbb{Z}_{26}$. On considère la fonction de $\mathbb{Z}_{26}$ dans lui-même définie par

$$f(z) = (a \cdot z + b) \mod 26$$

Calculer la réciproque de cette fonction en donnant toutes les étapes.

Soit $a' \in \mathbb{Z}_{26}^*$ tq. $a \cdot a' \equiv 1 \mod 26$

$w = (az + b) \mod 26$

$\Leftrightarrow (w - b) \mod 26 \equiv (a \cdot z) \mod 26$

$\Leftrightarrow$

```
def dech_vigenere(chiffre, clef):
    n = len(chiffre)
    k = len(clef)
    message = ""
    i = 0
    while i < n:
        z = ord(chiffre[i]) - 65
        d = ord(clef[i%k]) - 65
        z = (z - d)%26
        c = chr(z + 65)
        message += c
        i += 1
    return message
```