1. Exercice 1 :

Cryptage affine Chacune des 26 lettres est associée à l'un des entiers de 0 à 25, selon le tableau de correspondance suivant.

A B C D E F G 0 1 2 3 4 5 H I J K L M 6 N O P Q R S T 7 U 8 V 9 10 11 12 13 W 14 15 16 17 18 X Y Z 19 20 21 22 23 24 25
Le cryptage affine se fait à l'aide d'une clé, qui est un nombre entier k fixé, compris entre 1 et 25.
Pour crypter une lettre donnée on suit le processus $\wp$ suivant :
- on repère le nombre x associé à la lettre dans le tableau de correspondance précédent
- on multiplie ce nombre x par la clé k
- on calcule le reste r de la division euclidienne du nombre obtenu par 26 - on repère la lettre associée au nombre r dans le tableau de correspondance, qui devient la lettre cryptée.

Par exemple, pour crypter la lettre P avec la clé k = 11 :
- le nombre x associé à la lettre P est le nombre 15 - on multiplie 15 par la clé k, ce qui donne 11 × 15 = 165 - on calcule le reste de la division euclidienne par 26 : on obtient 165 % 26 = 9 - on repère finalement la lettre associée à 9 dans le tableau, c'est-à-dire J.

Ainsi avec la clé k = 11, la lettre P est cryptée en la lettre J. On crypte un mot en cryptant chacune des lettres de ce mot.
En Python, on crée une liste L qui contient les 26 lettres de l'alphabet rangées dans l'ordre alphabétique à l'aide de la commande ci-dessous :

L = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

!!! exo "Question 1." Que penser du cryptage obtenu lorsque la clé k est égale à 1 ?

Corrigé

Lorsque la clé est égale à 1, pour tout $x \in [0,25]$, $k \times x = x$ donc le reste de la division euclidienne de $k \times x$ par 26 est x. Chaque lettre est cryptée en elle-même, le cryptage n'a donc pas d'effet.

Question 2.

En quoi la lettre A constitue-t-elle un cas particulier dans le processus de cryptage ?

Corrigé

Lorsque la lettre est A, d'indice x = 0, pour tout $k \in [1,25]$, $k \times x = 0$ et le reste de la division euclidienne de $k \times x$ par 26 est 0. La lettre A est cryptée par elle-même, quelle que soit la clé de cryptage utilisée. Elle est invariante.

Question 3.

Dans le cas où la clé est égale à 11, crypter le mot MIRO.

Corrigé

Pour la lettre M d'indice 12, on a 12 × 11 = 132 et 132 % 26 = 2, indice correspondant à la lettre C
Pour la lettre I d'indice 8, on a 8 × 11 = 88 et 88 % 26 = 10, indice correspondant à la lettre K
-Pour la lettre R d'indice 17, on a 17 × 11 = 187 et 187 % 26 = 5, indice correspondant à la lettre F
-Pour la lettre O d'indice 14, on a 14 × 11 = 154 et 154 % 26 = 24, indice correspondant à la lettre Y
Le mot MIRO est donc crypté par le mot CKFY.

Question 4.

Ecrire une fonction `indice` qui prend en paramètre une lettre de l'alphabet et qui renvoie son indice dans la liste L (L étant supposée définie comme variable globale).

Corrigé

```
def indice (lettre) :
    return L.index(lettre)
```

Ou bien (sans la fonction index) :

```
def indice (lettre) :
    i = 0
        while L[i] != lettre :
            i += 1
    return i
```

Question 5.

Ecrire une fonction `crypterLettre` qui prend en paramètre une chaîne de caractères constituées d'une lettre majuscule de l'alphabet et une clé et qui renvoie la lettre cryptée. Cette fonction utilisera la fonction `indice` précédente. En supposant qu'un appel à la fonction indice compte pour une opération élémentaire, quel est le nombre d'opérations élémentaires effectuées à chaque appel de la fonction crypterLettre ?

Corrigé

```
def crypterLettre (lettre, k) :
    c = (indice(lettre)*k)%26
    return L[c]
```

Le nombre d'opérations élémentaires effectuées à chaque appel de la fonction crypterLettre est 4 : un appel à indice, une multiplication, un modulo, et une affectation.

Question 6.

(a) Ecrire une fonction `crypterTexte` qui prend en paramètre une chaîne de caractères dont les éléments sont soit des lettres majuscules soit des espaces (qui ne seront pas modifiés), et une clé. La fonction renvoie la chaîne de caractères cryptés.

(b) Donner l'appel à effectuer pour répondre à la question 3

Corrigé

(a)

```
def crypterTexte (texte, k) :
    T = ''
    for lettre in texte :
        if lettre in L :
            T += crypterLettre(lettre,k)
        else :
            # tout ce qui n'est pas une majuscule
            T += lettre
    return T
```

(b) `crypterTexte('MIRO',11)`

Question 7.

On dit qu'une clé est une bonne clé de cryptage si elle possède une clé associée k', qui est un nombre entier compris entre 1 et 25 tel qu'en appliquant le processus $\wp$ avec la clé k' à une lettre cryptée (avec la clé k) on obtient la lettre initiale. k' est alors appelée clé de décryptage associée à k. On admet qu'une clé k est une bonne clé de cryptage si et seulement si k est différent de 1 et si le seul diviseur commun dans $\mathbb{N}$ à k et à 26 est 1.

(a) On suppose que 19 est une clé de décryptage associée à la clé k = 11. Avec la clé k = 11, un mot a été crypté. On a obtenu le mot HARK. Retrouver le mot initial à l'aide de la clé de décryptage « à la main », puis donner l'appel à la fonction `crypterTexte` donnant le même résultat.

(b) Soit F la liste qui contient les termes de la suite de Fibonaci strictement inférieurs à 26 rangés par ordre croissant. On rappelle que la suite de Fibonacci est définie par ses deux premiers termes 0 et 1 et par le fait qu'à partir de son troisième terme, chaque terme est égal à la somme des deux précédents. Expliciter F.

(c) Déterminer la liste G des éléments de F qui sont de bonnes clés de cryptage.

(d) On admet que la clé de décryptage k' associée à une bonne clé de cryptage k est unique et que c'est le nombre entier strictement compris entre 0 et 26 qui est tel que le reste de la division euclidienne par 26 de k \times k'$ est 1. Vérifier que 19 est la clé de décryptage associée à la clé k = 11.

Corrigé

(a)
Le nombre associé à la lettre H est 7. On a 7 × 19 = 133 et 133 % 26 = 3 donc décrypté en D
La lettre A est invariante.
Le nombre associé à la lettre R est 17. On a 17 × 19 = 323 et 323 % 26 = 11 donc décrypté en L
Le nombre associé à la lettre K est 10. On a 10 × 19 = 190 et 190 % 26 = 8 donc décrypté en I
Le mot initial est donc DALI.
En utilisant la fonction, on écrit l'instruction :
`crypterTexte('HARK',19)`

(b)
F = `[0,1,1,2,3,5,8,13,21]`

(c)
Par définition, 0 n'est pas une clé de cryptage et 1 n'est pas une bonne clé de cryptage.

- 2 ∧ 26 = 2 ≠ 1
- 3 ∧ 26 = 1
- 5 ∧ 26 = 1
- 8 ∧ 26 = 2 ≠ 1
- 13 ∧ 26 = 13 ≠ 1
- 21 ∧ 26 = 1

Donc G = [3,5,21]

(d)
Le reste de la division euclidienne par 26 de 11 × 19 = 209 est 1, donc 19 est la clé de cryptage associée à la clé k = 11.

Question 8.

Ecrire une fonction `cleDecryptage` qui prend en paramètre un entier k premier avec 26 et qui renvoie la clé de décryptage associée k'. On dispose de bonnes clés de cryptage avec les éléments de G, la fonction `cleDecryptage` nous permet ainsi de calculer aussi les clés de décryptage associées.

Corrigé

```
def cleDecryptage (k) :
    # k est premier avec 26
    for i in range(26) :
        if (i*k) % 26 == 1 :
            return i
```

Question 9.

Un petit futé cherche à décrypter un long message crypté écrit en français sans connaître ni la clé de cryptage, ni la clé de décryptage. Pour cela, il repère dans la chaîne de caractères que constitue le message la lettre la plus fréquente et en déduit que c'est la lettre cryptée qui correspond à E (la lettre la plus utilisée en français).

(a) Que renvoie la fonction suivante ? On ne justifiera pas la réponse.

```
import numpy as np
def mystere (C):
    """ C est une chaîne de caractères dont les
éléments sont soit des lettres majuscules, soit des
espaces """
    V = np.zeros(26)max
    max, indexMax = 0 , 0
    for j in range(26):
        for j in range(len(C)):
            if C[i] == L[j]:
                V[j] += 1
        if V[j] > max :
            max, indexMax = V[j], j
    return L[indexMax]
```

(b) Donner une preuve rapide de la terminaison de cette fonction.

(c) *Question bonus*. En utilisant les fonctions `count`, `max` et `index` de Python, réécrire la fonction précédente avec le moins de lignes possible, et donner lui aussi un nom plus explicite.

Corrigé

(a)
La fonction mystere renvoie la lettre la plus fréquente dans la chaîne de caractères C passé en paramètre

(b)
La boucle portant sur i est exécutée len(C) fois et le corps de cette boucle ne contient qu'une comparaison et une addition/affectation. Elle se termine donc. La boucle portant sur j est exécutée 26 fois et le corps de cette boucle contient la boucle portant sur i, une comparaison et deux affectations. La fonction se termine donc bien.

(c)

```
def maxOccurence (C) :
    V = [C.count(L[j]) for j in range(26)]
    return L[V.index(max(V))]
```

Question 10.

Connaissant E et la lettre cryptée correspondante, le petit futé peut en déduire la clé de cryptage et ainsi « cracker » le code. Ecrire une fonction `cracker` qui prend en paramètre la lettre cryptée correspondant à E et qui renvoie la liste des clés de cryptage possibles (liste pouvant être vide) permettant de crypter E en cette lettre.

Corrigé

On parcourt toutes les lettres possibles, et on conserve celles qui coderaient la lettre E (d'indice 4) en la lettre en paramètre.

```
def cracker (lettre) :
    N = []
    for i in range(26) :
        if (4*i) % 26 == indice(lettre) :
            N += [i]
    return N
```

Question 11.

Que se passe-t-il s'il essaie de « cracker » un message crypté où la lettre E n'est pas la lettre la plus fréquente du message ?

Corrigé

Il ne parviendra pas à « cracker » le message puisque la clé de décryptage calculée ne sera pas la bonne (indice de la lettre E de 4 utilisé pour décrypter une lettre qui n'est pas E). Un message « décrypté » sera obtenu mais pas le bon.