TP 5 cryptographie

La cryptographie est indispensable pour échanger des informations confidentielles. Autrefois réservée aux utilisations militaires, la cryptographie s'étend maintenant aux échanges bancaires, aux informations médicales, aux transferts de données industrielles,...Ce TP propose de réaliser des programmes en Python, de cryptage et de décryptage de texte suivant diverses méthodes en analysant les faiblesses des codes les plus simples.

I Cryptage par le code de (Jules) César

I.1 Codage

Le cryptage de César (empereur romain) est le tout premier code de cryptage qui ait existé.
La méthode est simple: il suffit de décaler toutes les lettres de l'alphabet du même nombre de lettres. Par exemple, en choisissant un décalage de 3, le **a** devient le **d**, le **b** devient le **e**, ...,le **w** devient **z** Pour la fin de l'alphabet, il suffit de revenir au début : le **x** devient **a**, le **y** devient **b** et le **x** devient **c**. Tous les caractères distincts des minuscules de l'alphabèt restent identiques à eux mêmes dans ce codage. Ainsi un message comme "zoé aime l'informatique" devient par décalage d'une lettre "apé bjnf m'jogpsnbujrvf". (les caractères espace, aprostrophe et e accentué sont inchangés).

Dans la suite, on définira dans les fonctions qui le nécessitent la chaîne de caractères 'abcdefghijklmnopqrstuvwxyz' qu'on appellera `alphabet`.

Q1.1 Ecrire une fonction `position(x)` d'argument un caractère `x` qui renvoie la position de x dans la chaîne de caractères `alphabet`. Cette fonction renverra
-1 si x n'est pas élément de alphabet.
Par exemple `position('a')` doit renvoyer 0, `position('d')` doit renvoyer 3, `position('é')` doit renvoyer -1

Q1.2 On veut écrire une fonction `decalage(x,n)` qui renvoie le caractère obtenu du caractère x en le décalant de n dans la chaîne alphabet. On voudrait par exemple que `decalage('d',3)` renvoie 'g', `decalage('z',3)` renvoie 'c'. On voudrait aussi prendre en compte des décalages négatif (correspondant à un décalage vers la gauche) de sorte que par exemple `decalage('g',-3)` renvoie 'd' et `decalage('c',-3)` renvoie 'z'. Si i est la position du caractère x à décaler de n, la valeur i+n n'appartient pas necessairement à l'intervalle $[[0, 25]]$ des positions possibles des caractères dans la chaîne `alphabet`.
Quelle opération arithmétique permet de ramener cette valeur i+n dans l'intervalle $[[0, 25]]$?
Ecrire une fonction `decalage(x,n).`

Q1.3 Ecrire une fonction `codage(texte,n)` d'arguments un entier `n` et une chaîne de caractères `texte` qui renvoie la chaîne obtenue de `texte` par codage de César avec un décalage de `n` lettres (seuls les caractères minuscules de l'alphabet latin subissent une modification).

Tester la fonction codage sur un texte de votre choix.

I.2 Décodage du code de César

Q2.1 Expliquer pourquoi il suffit de connaître le codage d'un unique caractère pour décoder un texte codé avec le code César.

Q2.2 Ecrire une fonction `nombre_occurences(texte,x)` qui renvoie le nombre de caractère x dans la chaîne de caractères texte.
Ecrire une fonction `plus_frequent(texte)` qui renvoie le caractère de l'alphabet le plus fréquent dans texte (ou l'un des plus fréquents en cas d'ex-aequos).

La lettre la plus fréquente de la langue française est le e. Nous allons supposer dans la suite que le e est le caractère le plus fréquent du message qui a été codé. En cherchant la lettre la plus fréquente dans le message codé, on peut revenir au message initial

Q2.3 Ecrire une fonction `decryptage(code)` qui, en supposant que `code` est le résultat du codage de César, renvoie la chaîne de caractère décodée. Cette fonction ne pourra s'appliquer avec un résultat correct que lorsque, dans le message à coder, le caractère e est le plus fréquent, ce qui est le cas dès que le message est suffisamment long (et n'a pas été écrit avec l'intention d'échapper à cette règle comme le roman "la.disparition" de Georges Perec).

Q2.4 Copier coller un texte assez long pour définir une chaîne de caractères dans votre éditeur python. Effectuer un codage de ce texte avec la fonction codage avec un décalage tiré au hasard. Utiliser la fonction decryptage pour retrouver le texte initial. Pour le tirage au hasard:

```
import random
n=random.randint(0,25)# n est un entier tire au hasard entre 0 et 25
```

II Cryptage par substitution mono-alphabétique

Le codage par substitution mono-alphabétique consiste à remplacer chaque lettre par une lettre différente. Par exemple, en utilisant la table de substitution suivante :

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
w	x	e	h	y	z	t	k	c	p	j	i	u	a	d	g	l	q	m	n	r	s	f	v	b	o

le message 'il fait beau' sera codé par 'ci zwcn xywr'.
Pour effectuer un tel codage, on a besoin de la chaîne de caractères 'wxehyztkcpjiuadglqmnrsfvbo' (qui correspond à l'argument substitution de la fonction de Q3.1) qui indique comment se fait la substitution. On convient de ne coder que les lettres du message minuscules de l'alphabet latin non accentuées. Les autres lettres ou caractères seront conservés.

Q3.1 Ecrire une fonction `codage_substitution(texte, substitution)` qui renvoie le résultat du codage par subsitution de la chaîne de caractères texte où l'argument `substitution` est la chaîne de caractères donnant dans le même ordre les valeurs à subsituer au caractères de `alphabet`.

Q3.2 La personne qui a reçu le texte codé connait la chaîne `substitution` qui a permis de coder le texte. Aidons la à retrouver le texte original. Ecrire une fonction `decryptage_substitution(texte-cod substitution)` qui renvoie le texte original si texte_code est le résultat du codage par substitution fait avec la substitution alpha de la chaîne de caractères `alphabet`. Indication: on utilisera la fonction précédente.

III Le chiffre de Vigenère

Pour palier à la faiblesse du code de César, Le diplomate Français du $16^{ème}$ siècle Blaise de Vigenère eu l'idée d'utiliser un chiffre de César, mais avec un décalage qui change de lettre en lettre grâce à une clé (un mot de longueur arbitraire). Pour coder un message, on décale chaque lettre du message par le même décalage qui fait passer la lettre a à la lettre correspondante de la clé écrite sous le message (et répétée autant de fois que nécessaire).

Exemple 1 *La table suivante illustre le codage "lechiffrementestutile" à l'aide de la clé "azerty"*

message	*l*	*e*	*c*	*h*	*i*	*f*	*f*	*r*	*e*	*m*	*e*	*n*
clé	*a*	*z*	*e*	*r*	*t*	*y*	*a*	*z*	*e*	*r*	*t*	*y*
décalage	*+0*	*+25 (-1)*	*+4*	*-9*	*-7*	*-2*	*0*	*-1*	*+4*	*-9*	*-7*	*-2*
code	*l*	*d*	*g*	*y*	*b*	*d*	*f*	*q*	*i*	*d*	*x*	*l*

(à finir ligne suivante)

message	*t*	*e*	*s*	*t*	*u*	*t*	*i*	*l*	*e*
clé	*a*	*z*	*e*	*r*	*t*	*y*	*a*	*z*	*e*
-décalage	*0*	*-1*	*+4*	*-9*	*-7*	*-2*	*0*	*-1*	*+4*
code									

. Programmer cette méthode.

Remarque *Des algorithmes permettent de déchiffrer le codage de Vigenère lorsque la clé n'est pas trop longue. Avec une clé très longue, il est impossible à déchiffrer. La faiblesse du chiffrage réside alors dans le fait que la clé, si elle est partagée par de multiple utilisateurs, risque de tomber dans de mauvaises mains. Ce genre de problème a été résolu par les méthodes de chiffrement modernes.*

Correction

Q1.1 Ecrire une fonction **position(x)** d'argument un caractère **x** qui renvoie la position de x dans la liste alphabet. Cette fonction renverra
-1 si x n'est pas élément de alphabet.

```
def position(x):
    alphabet='abcdefghijklmnopqrstuvwxyz'
    res=-1
    for i in range(26):
        if x==alphabet[i]:
            res=i
    return res
```

Soit x un caractère de position i dans la liste alphabet. Quelle est la position obtenu du caratère x par décalage de n?

Q1.2 Ecrire une fonction **decalage(x,n)** qui renvoie le caractère obtenu de x en le décalant de n dans la chaîne alphabet

```
def decalage(x,n):
    alphabet='abcdefghijklmnopqrstuvwxyz'
    i=(position(x)+n)%26 #position de la lettre decalee
    return alphabet[i]
```

Q1.3 Ecrire une fonction **codage(texte,n)** d'arguments un entier **n** et une chaîne de caractères **texte** qui renvoie la chaîne obtenue de **texte** par codage de César avec un décalage de n lettres (seuls les caractères minuscules de l'alphabet latin subissent une modification).

```
def codage(texte,n):
    code=""# chaine vide
    for i in range(len(texte)):
        if position(texte[i])==-1: pas dans l'alphabet
            code=code+texte[i] # caractere non code
        else:
            code=code+decalage(texte[i],n)
    return code
```

Q2.1 Si on sait par exemple que a est codé en f, alors on sait que le décalage est +5. Pour décoder il sufffira de coder avec un décalage -5.

Q2.2 Ecrire une fonction **nombre_occurences(texte,x)** qui renvoie le nombre de caractère x dans la chaîne dans la chaîne de caractères texte.

```
def nombre_occurences(texte,x):
    compteur=0
    for i in range(len(texte)):
        if texte[i]==x:
            compteur=compteur+1
    return compteur
```

Ecrire une fonction **plus_frequent(texte)** qui renvoie le caractère de l'alphabet le plus fréquent dans texte.

```
def plus_frequent(texte):
    alphabet='abcdefghijklmnopqrstuvwxyz'
    Cmax,max='a',nombre_occurences(texte,'a')
    for i in range(len(alphabet)): # on ne s'interesse qu'aux caracteres de l'alphabet
        if nombre_occurences(texte,alphabet[i])>max:
            Cmax,max=alphabet[i],nombre_occurences(texte,alphabet[i])
    return alphabet[i]
```

La lettre la plus fréquente de la langue française est le e. Quelle devrait être le caractère le plus fréquent dans le texte codé?
Cela devrait être le caractère obtenu en décalant le 'e'.

Q2.3 Ecrire une fonction **decryptage(code)** qui, en supposant que **code** est le résultat du codage de César, renvoie la chaîne de caractère décodée.

```
def decryptage(code):
    D=position(plus_frequent(texte))-position('e')# decalage probable
    return codage(code,-D) #on decale dans le sens invers.
```

Q2.4 Copier coller un texte assez long pour définir une chaîne de caractères dans votre éditeur python (ne pas choisir un extrait de "la.disparition" de Georges Perec). Effectuer un codage de ce texte avec la fonction codage. Entre temps, vous avez complètement oublié votre texte et le décalage utilisé. Utiliser la fonction decryptage pour retrouver le texte initial.

III.1 Cryptage par substitution mono-alphabétique

Q3.1 Ecrire une fonction **codage_substitution(texte, substitution)** qui renvoie le résultat du codage par subsitution de la chaîne de caractères texte avec une substitution alpha de la chaîne de caractères `alphabet`.

```
def codage_substitution(texte,substitution):
    code=""# chaine vide
    for i in range(len(texte)):
        j=position(texte[i])
        if j==-1:
             code=code+texte[i]
        else:
            code=code+substitution[j]
    return code
```

Q3.2 La personne qui a reçu le texte codé connait la liste **substitution** qui a permis de coder le texte. Aidons la à retrouver le texte original.
Ecrire une fonction **decryptage_substitution(code, substitution)** qui renvoie le texte original si code est le résultat du codage fait avec la chaîne de caratères **substitution.**

```
def decryptage(code,substitution):
    S=substitution_inverse(substitution)
    return codage_substitution(code,S)

def substitution_inverse(substitution):
    alphabet='abcdefghijklmnopqrstuvwxyz'
    S=""
    for i in range(len(alphabet)):
        j=0
        while substitution[j]!=alphabet[i]:
            j=j+1 # j est la place de alphabet[i] dans substitution
        S=S+alphabet[j]
    return S
```

III.2 Le chiffre de Vigenère

Pour palier à la faiblesse du code de César, Le diplomate Français du $16^{ème}$ siècle Blaise de Vigenère eu l'idée d'utiliser un chiffre de César, mais avec un décalage qui change de lettre en lettre grâce à une clé (un mot de longueur arbitraire). Pour coder un message, on décale chaque lettre du message par le même décalage qui fait passer la lettre a à la lettre correspondante de la clé écrite sous le message (et répétée autant de fois que nécessaire).

Exemple 2 *La table suivante illustre le codage "lechiffrementestutile" à l'aide de la clé "azerty"*

message	*l*	*e*	*c*	*h*	*i*	*f*	*f*	*r*	*e*	*m*	*e*	*n*
clé	*a*	*z*	*e*	*r*	*t*	*y*	*a*	*z*	*e*	*r*	*t*	*y*
décalage	*+0*	*+25 (-1)*	*+4*	*-9*	*-7*	*-2*	*0*	*-1*	*+4*	*-9*	*-7*	*-2*
code	*l*	*d*	*g*	*y*	*b*	*d*	*f*	*q*	*i*	*d*	*x*	*l*

(à finir ligne suivante)

message	*t*	*e*	*s*	*t*	*u*	*t*	*i*	*l*	*e*
clé	*a*	*z*	*e*	*r*	*t*	*y*	*a*	*z*	*e*
-décalage	*0*	*-1*	*+4*	*-9*	*-7*	*-2*	*0*	*-1*	*+4*
code									

. Programmer cette méthode.

```
def Vigenere(texte,cle):
    p=len(cle)
    code=''
    j=0 # position dans la cle
    for i in range(len(texte)):
        k=position(cle[j])
        if k==-1:
            code=code+texte[i]
        else:
            code=code+decalage(texte[i],k)
        j=(j+1)%p # decalage de 1 dans la cle et on revient au debut en fin de cle
    return code
```