Codage

Ave Cesar (zud bdrzq)

On cherche à chiffrer un texte t de longueur n composé de caractères en minuscules (soit 26 lettres différentes) représentés par des entiers compris entre 0 et 25 (0 ↔ **a**, 1 ↔ **b**,...25 ↔ **z**). Nous ne tenons pas compte des éventuels espaces.

Ainsi, le texte **ecolepolytechnique** est représenté par le tableau suivant où la première ligne représente le texte, la seconde les entiers correspondants, et la troisième les indices dans le tableau t.

e	**c**	**o**	**l**	**e**	**p**	**o**	**l**	**y**	**t**	**e**	**c**	**h**	**n**	**i**	**q**	**u**	**e**
4	**2**	**14**	**11**	**4**	**15**	**14**	**11**	**24**	**19**	**4**	**2**	**7**	**13**	**8**	**16**	**20**	**4**
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**

I. Codage de César

Ce codage est le plus rudimentaire que l'on puisse imaginer. Il a été utilisé par Jules César (et même auparavant) pour certaines de ses correspondances. Le principe est de décaler les lettres de l'alphabet vers la gauche de 1 ou plusieurs positions. Par exemple, en décalant les lettres de 1 position, le caractère **a** se transforme en **z**, le **b** en **a**, etc. le **z** en **y**. Le texte **avecesar** devient donc **zudbdrzq**.

Question 1. Que donne le codage du texte **maitrecorbeau** en utilisant un décalage de 5 ?

Question 2. Écrire la fonction **codageCesar**(t,d) qui prend en arguments le tableau t et un entier d et qui retourne un tableau de même taille que t contenant le texte t décalé de d positions.

Question 3. Écrire de même la fonction **decodageCesar**(t,d) prenant les mêmes arguments mais qui réalise le décalage dans l'autre sens.

Pour réaliser ce décodage, il faut connaître la valeur du décalage. Une manière de la déterminer automatiquement est d'essayer de deviner cette valeur. L'approche la plus couramment employée est de regarder la fréquence d'apparition de chaque lettre dans le texte chiffré. En effet, la lettre la plus fréquente dans un texte suffisamment long en français est la lettre **e**.

Question 4. Écrire la fonction **frequences**(t) qui prend en argument un tableau t représentant le texte chiffré et qui retourne un tableau de taille 26 dont la case d'indice i contient le nombre d'apparitions du nombre i dans t ($0 \leqslant i < 26$).

Question 5. Écrire la fonction **decodageAuto**(u) qui prend en argument le tableau u représentant le texte chiffré et qui retourne le texte t d'origine (en calculant la clé pour que la lettre **e** soit la plus fréquente dans le texte déchiffré).

II. Codage de Vigenère

Au XVIème siècle, Blaise de Vigenère a modernisé le codage de César très peu résistant de la manière suivante. Au lieu de décaler toutes les lettres du texte de la même manière, on utilise un texte clé qui donne une suite de décalages.

Prenons par exemple la clé **concours**. Pour chiffrer un texte, on code la première lettre en utilisant le décalage qui envoie le **a** sur le **c** (la première lettre de la clé). Pour la deuxième lettre, on prend le décalage qui envoie le **a** sur le **o** (la seconde lettre de la clé) et ainsi de suite. Pour la huitième lettre, on utilise le décalage **a** vers **s**, puis, pour la neuvième, on reprend la clé à partir de sa première lettre. Sur l'exemple **ecolepolytechnique** avec la clé **concours**, on obtient : (la première ligne donne le texte, la seconde le texte chiffré et la troisième la lettre de la clé utilisée pour le décalage)

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

Question 6. Donner le codage du texte **becunfromage** en utilisant la clé de codage **jean**.

Question 7. Écrire la fonction **codageVigenere**(t, c) qui prend comme arguments un tableau t représentant le texte à chiffrer et un tableau d'entiers c donnant la clé servant au codage et qui retourne un tableau de même taille que t contenant le texte chiffré.

Maintenant, on suppose disposer d'un texte u assez long chiffré par la méthode de Vigenère et on veut retrouver le texte t d'origine. Pour cela, on doit trouver la clé c ayant servi au codage. On procède en deux temps : 1) détermination de la longueur k de la clé c, 2) détermination des lettres composant c.

La première étape est la plus difficile. On remarque que deux lettres identiques dans t espacées de $\ell \times k$ caractères (où ℓ est un entier et k la taille de la clé) sont codées par la même lettre dans u. Mais cette condition n'est pas suffisante pour déterminer la longueur k de la clé c puisque des répétitions peuvent apparaître dans u sans qu'elles existent dans t. Par exemple, les lettres **t** et **n** sont toutes deux codées par la lettre **h** dans le texte chiffré à partir de **ecolepolytechnique** avec **concours** comme clé. Pour éviter ce problème, on recherche les répétitions non pas d'une lettre mais de séquences de lettres dans u puisque

deux séquences de lettres répétées dans t, dont les premières lettres sont espacées par $\ell \times k$ caractères, sont aussi chiffrées par deux mêmes séquences dans t'.

Dans la suite de l'énoncé, on ne considère que des séquences de taille 3 en supposant que toute répétition d'une séquence de 3 lettres dans u provient exclusivement d'une séquence de 3 lettres répétée dans t. Ainsi, la distance séparant ces répétitions donne des multiples de k.

La valeur de k est obtenue en prenant le PGCD de tous ces multiples. Si le nombre de répétitions est suffisant, on a de bonnes chances d'obtenir la valeur de k. On suppose donc que cette assertion est vraie.

Question 8. Écrire la fonction **`pgcd`**(a, b) qui calcule le **`PGCD`** des deux entiers strictement positifs a et b.

Question 9. Écrire la fonction **`pgcdDesDistancesEntreRepetitions`**(u, i) qui prend en argument le texte chiffré u et un entier i ($0 \leqslant i < n-2$ avec n la longueur de u) qui est l'indice d'une lettre dans u et qui retourne le **`pgcd`** de toutes les distances entre les répétitions de la séquence de 3 lettres $\langle u[i], u[i+1], u[i+2] \rangle$ dans la suite du texte $\langle u[i+3], u[i+4], \ldots u[n-1] \rangle$. Cette fonction retourne 0 s'il n'y a pas de répétition.

Question 10. Écrire la fonction **`longueurDeLaCle`**(u) qui prend en argument le texte chiffré u et qui retourne la longueur k de la clé de codage.

Question 11. Donner un ordre de grandeur du nombre d'opérations réalisées par la fonction **`longueurDeLaCle`** en fonction de la longueur n du texte chiffré (on ne comptera que le nombre d'appels à la fonction **`pgcd`**).

Question 12. Une fois la longueur de la clé connue, donner une idée d'algorithme permettant de retrouver chacune des lettres de la clé. (Il s'agit de décrire assez précisément l'algorithme plutôt que d'écrire le programme).

Question 13. Écrire la fonction **`decodageVigenereAuto`**(u) qui prend en argument le tableau u représentant le texte chiffré et qui retourne le texte t d'origine. (On n'hésitera pas à recopier des parties de texte dans des tableaux intermédiaires).

Question 14. Décoder le message :

```
KQOWEFVJPUJUUNUKGLMEKJINMWUXFQMKJBGWRLFNFGHUDWUUMBSVLPS
NCMUEKQCTESWREEKOYSSIWCTUAXYOTAPXPLWPNTCGOJBGFQHTDWXIZA
YGFFNSXCSEYNCTSSPNTUJNYTGGWZGRWUUNEJUUQEAPYMEKQHUIDUXFP
GUYTSMTFFSHNUOCZGMRUWEYTRGKMEEDCTVRECFBDJQCUSWVBPNLGOYL
SKMTEFVJJTWWMFMWPNMEMTMHRSPXFSSKFFSTNUOCZGMDOEOYEEKCPJR
GPMURSKHFRSEIUEVGOYCWXIZAYGOSAANYDOEOYJLWUNHAMEBFELXYVL
WNOJNSIOFRWUCCESWKVIDGMUCGOCRUWGNMAAFFVNSIUDEKQHCEUCPFC
MPVSUDGAVEMNYMAMVLFMAOYFNTQCUAFVFJNXKLNEIWCWODCCULWRIFT
WGMUSWOVMATNYBUHTCOCWFYTNMGYTQMKBBNLGFBTWOJFTWGNTEJKNEE
DCLDHWTVBUVGFBIJG
```

Corrections

Q 2. Sachant que t est une liste de nombres, le principe est construire une nouvelle liste à partir des éléments de t auquels on aura retranché d. Cependant, il faut tenir compte du cas où l'on obtient un résultat négatif et dans ce cas lui ajouter 26 (ou alors travailler modulo 26).

```
def codageCesar(t,d):
    n = len(t)
    u = []
    for i in range(n):
        x = t[i]-d
        if x<0:
            x = x+26
        u.append(x)
    return u
```

Ou plus simplement :

```
def codageCesar(t,d):
    n = len(t)
    u = []
    for i in range(n):
        u.append((t[i]-d)%26)
    return u
```

et encore plus simplement :

```
def codageCesar(t,d):
    u = []
    for x in t:
        u.append((x-d)%26)
    return u
```

On définit également :

```
def tableau(s):
    """
    Construit le tableau de nombres associé à une chaine
    Exemple : tableau('abc') renvoie [0,1,2]
    """
    t = []
    for c in s:
        t.append(ord(c)-ord('a'))
    return t

def chaine(t):
    """
    Construit la chaine associée à un tableau de nombres
    Exemple : chaine([0,1,2]) renvoie 'abc'
    """
    alphabet = 'abcdefghijklmnopqrstuvwxyz'
    s = ''
    for x in t:
        s = s+alphabet[x]
    return s
```

```
print(tableau("ecolepolytechnique"))
```

```
[4, 2, 14, 11, 4, 15, 14, 11, 24, 19, 4, 2, 7, 13, 8, 16, 20, 4]
```

```
print(chaine(codageCesar(tableau("avecesar"),1)))
```

```
zudbdrzq
```

```
print(codageCesar(tableau("maitrecorbeau"),5))
```

```
[7, 21, 3, 14, 12, 25, 23, 9, 12, 22, 25, 21, 15]
```

Q 3. Il faut procéder de même mais en ajoutant d et en tenant compte du fait qu'il ne faut pas dépasser 25.

```
def decodageCesar(t,d):
    n = len(t)
    u = []
    for i in range(n):
        x = t[i]+d
        if x>25:
            x = x-26
        u.append(x)
    return u
```

```
print(chaine(decodageCesar(codageCesar(tableau('maitrecorbeau'),5),5)))
```

```
maitrecorbeau
```

Q 4.

```
def frequences(t):
    f = [0]*26
    for x in t:
        f[x] = f[x]+1
    return f
```

```
print(frequences(tableau('ecolepolytechnique')))
```

```
[0, 0, 2, 0, 4, 0, 0, 1, 1, 0, 0, 2, 0, 1, 2, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1,
0]
```

Q 5.

```
def decodageAuto(u):
    f = frequences(u)
    # On recherche l'indice du maximum de f, noté imax
    imax = 0
    for i in range(1,26):
        if f[i]>f[imax]:
            imax = i
    # En fait imax est l'indice du maximum le plus à gauche de f
    # On recherche quel décalage d envoie 4 ('e') sur imax
    # On doit avoir 4-d = imax
    d = (4-imax)%26
    return decodageCesar(u,d)
```

```
u = codageCesar(tableau('ecolepolytechnique'),12)
print(chaine(decodageAuto(u)))
```

```
ecolepolytechnique
```

```
u = codageCesar(tableau('ecolepolytechnique'),5)
print(chaine(decodageAuto(u)))
```

```
ecolepolytechnique
```

```
u = codageCesar(tableau('maitrecorbeau'),12)
print(chaine(decodageAuto(u)))
```

```
znvgerpbeornh
```

Q 7.

```
def codageVigenere(t,c):
    u = []
    k = len(c)
    j = 0 # Position dans la clé
    for x in t:
        u.append((x+c[j])%26)
        j = j+1
        if j==len(c):
            j = 0
    return u
```

```
print(chaine(codageVigenere(tableau('ecolepolytechnique'),tableau('concours'))
```

```
gqbnsjfdahrevhziws
```

```
print(chaine(codageVigenere(tableau('becunfromage'),tableau('jean'))))
```

```
kichwjrbvegr
```

Q 8. On utilise simplement l'algorithme d'Euclide.

```
def pgcd(a,b):
    while b!=0:
        r = a%b
        a = b
        b = r
    return a
```

```
print(pgcd(28,98))
```

```
14
```

Q 9.

```
def pgcdDesDistancesEntreRepetitions(u,i):
    n = len(u)
    d = 0 # Le pgcd des distances trouvées
    for j in range(3,n-i-2):
        if u[i]==u[i+j] and u[i+1]==u[i+j+1] and u[i+2]==u[i+j+2]:
            # On a trouvé une répétition de u[i],u[i+1],u[i+2]
            # qui est distante de j
            if d==0:
                d = j
            else:
                d = pgcd(d,j)
    return d
```

Q 10.

```
def longueurDeLaCle(u):
    k = 0
    n = len(u)
    for i in range(0,n-2):
        d = pgcdDesDistancesEntreRepetitions(u,i)
        if d>0:
            if k==0:
                k = d
            else:
                k = pgcd(k,d)
    return k
```

```
print(longueurDeLaCle(codageVigenere(tableau('ecolepolytechnique'),tableau('co
```

```
0
```

```
import string

f = open('/home/ba/Enseignement/PC/Python/Data/IleMysterieuseChap1.txt')
s = ""
for c in f.read().lower():
    if c in string.ascii_lowercase:
        s = s+c
f.close()
print(s)
```

```
chapitreiremontonsnousnonaucontrairenousdescendonspisquecelamonsieurcyrusnoustombons
```

```
print(longueurDeLaCle(codageVigenere(tableau('messagertresmesquinmesopotamien'
```

```
12
```

```
s = """
KQOWEFVJPUJUUNUKGLMEKJINMWUXFQMKJBGWRLFNFGHUDWUUMBSVLPS
NCMUEKQCTESWREEKOYSSIWCTUAXYOTAPXPLWPNTCGOJBGFQHTDWXIZA
YGFFNSXCSEYNCTSSPNTUJNYTGGWZGRWUUNEJUUQEAPYMEKQHUIDUXFP
GUYTSMTFFSHNUOCZGMRUWEYTRGKMEEDCTVRECFBDJQCUSWVBPNLGOYL
SKMTEFVJJTWWMFMWPNMEMTMHRSPXFSSKFFSTNUOCZGMDOEOYEEKCPJR
GPMURSKHFRSEIUEVGOYCWXIZAYGOSAANYDOEOYJLWUNHAMEBFELXYVL
WNOJNSIOFRWUCCESWKVIDGMUCGOCRUWGNMAAFFVNSIUDEKQHCEUCPFC
MPVSUDGAVEMNYMAMVLFMAOYFNTQCUAFVFJNXKLNEIWCWODCCULWRIFT
WGMUSWOVMATNYBUHTCOCWFYTNMGYTQMKBBNLGFBTWOJFTWGNTEJKNEE
DCLDHWTVBUVGFBIJG
"""
s = s.lower()
s = s.replace('\n','')

print(longueurDeLaCle(tableau(s)))
```

```
5
```

Q 11. La fonction **pcgdDesDistancesEntreRepetitions** appelle O(n) fois la fonction **pgcd** et la fonction **longueurDeLaCle** appelle O(n) fois la fonction **pcgdDesDistancesEntreRepetitions**. On obtient donc O(n^2) appels à la fonction **pgcd** pour un texte chiffré de taille n.

Q 13.

```
def decodageVigenereAuto(u):
    k = longueurDeLaCle(u)
    n = len(u)
    t = [0]*n
    for j in range(k):
        # On commence par extraire de u les éléments u[j],u[j+k],u[j+2k],...
        # On crée ainsi un tableau uj
        uj = []
        i = j
        while i<n:
            uj.append(u[i])
            i = i+k
        # Toutes les lettres de uj ont été codées avec le même décalage
        # C'est donc un codage de César, on utilise le décodage automatique
        tj = decodageAuto(uj)
        # On recopie tj à la bonne place dans le tableau t
        for i in range(len(tj)):
            t[j+k*i] = tj[i]
    return t
```

```
s = chaine(decodageVigenereAuto(tableau(s)))
while len(s)>60:
    print(s[:60])
    s = s[60:]
print(s)
```

```
souventpoursamuserleshommesdequipageprennentdesalbatrosvaste
soiseauxdesmersquisuiventindolentscompagnonsdevoyagelenavire
glissantsurlesgouffresamersapeinelesontilsdeposessurlesplanc
hesquecesroisdelazurmaladroitsethonteuxlaissentpiteusementle
ursgrandesailesblanchescommedesavironstraineracotedeuxcevoya
geurailecommeilestgaucheetveuleluinagueresibeauquilestcomiqu
eetlaidlunagacesonbecavecunbrulegueulelautremimeenboitantlin
firmequivolaitlepoeteestsemblableauprincedesnueesquihantelat
empeteetseritdelarcherbaudelaire
```