Le DM doit être traité par un groupe de 2 élèves.
Envoyer par mail les codes sources et rendre un document écrit avec les réponses aux questions et en annexe les listings des codes sources et les résultats des tests effectués.

Cryptographie : chiffre de César et chiffrement affine

1 Vocabulaire et principes de cryptologie

L'homme a toujours eu besoin de transmettre un message en le protégeant de toute tentative d'interception par un intrus. Parmi les techniques possibles on distingue :

- la **stéganographie** (du grec *steganos* : étanche, et **graphein** : écrire)qui consiste à dissimuler le message (principe de l'encre sympathique). L'interception est empêchée par la dissimulation.
- la **cryptographie** (du grec *kruptos* : caché, et **graphein** : écrire) qui est l'art de coder le message d'une façon connue uniquement de l'émetteur et du récepteur. L'interception est empêchée par l'incapacité de l'intrus à interpréter le message sous sa forme cryptée. En toute rigueur, un **chiffre** est une transformation caractère par caractère alors qu'un **code** remplace un mot par un autre mot ou par un symbole (comme les hiéroglyphes). Mais on peut parler indistinctement de chiffre, de code, de **cryptage**.

  Le message à chiffrer s'appelle **le texte en clair** et on lui fait correspondre **un texte chiffré**.

Si la **cryptographie** est **l'art de chiffrer**, la **cryptanalyse** est l'art de **déchiffrer**. Ce sont les deux pendants de la science des codes secrets appelée **cryptologie**.

En pratique, pour chiffrer un message on utilise un **procédé de chiffrement** et **une clef de chiffrement**. Les militaires ont par exemple besoin de chiffrer de grandes quantités de messages et rapidement : il serait trop lourd de changer de procédé, pour modifier le chiffrement on joue donc sur un paramètre secret appelé clef.
Par exemple, le procédé du chiffre de César consiste à appliquer le même décalage à chaque lettre du message en clair et sa clef de chiffrement est 3 : A est codée par D, D par E, Z par C . . .

Le **principe de Kerckhoff** est la base de toute méthode cryptographique moderne :

> *Tous les procédés de chiffrement doivent être publics, seules les clefs doivent rester secrètes.*

Autrement dit, la fiabilité d'un chiffre doit reposer entièrement sur sa clef. En effet, il est impossible de tenir longtemps secret un procédé de chiffrement,il vaut mieux le rendre public, ainsi les spécialistes en cryptographie du mode entier pourront tester sa solidité.

2 Le chiffre de César

D'après la légende, César aurait chiffré sa correspondance avecun **chiffre par substitution monoalphabétique** : chaque lettre de l'alphabet est remplacée dans le texte chiffré par une autre lettre, toujours la même.
Il existe aussi des **chiffres par substitution polyalphabétique** comme le chiffre de Vigenère : chaque lettre de l'alphabet est remplacée dans le texte chiffré par une autre lettre, mais qui varie selon la position dans le message.
Dans le chiffre de César, chaque lettre du texte chiffré s'obtient par un décalage de la lettre du texte en clair. Ce décalage est la clef du chiffre, pour le chiffre de César cette clef est 3 : A est chiffré par D, B par E,W par Z et X par A.
Notre alphabet comptant 26 lettres, on peut repérer A par 0, B par 1 . . . Z par 25.

Le chiffre de César peut alors se modéliser sous la forme d'une fonction mathématique qui à une lettre en clair repérée par x avec $0 \leqslant x \leqslant 25$ associe une lettre chiffrée repérée par $y \equiv x+3 \mod 26$.
Cette notation se lit y congru à $x+3$ modulo 26 et signifie que y est égal au reste de la division euclidienne de $x+3$ par 26.

1. Compléter le tableau ci-dessous avec le chiffe de César :

Lettre en clair	A	B	...	W	X	Y	Z
x	0	1	...	22	23	24	25
$y \equiv x+3 \mod 26$	3	4	...	...	...	...	...
Lettre chiffrée	D	E	...	...	...	...	...

2. Ecrire sous la forme d'un produit le nombre de chiffres par substitution monoalphabétique distincts pour un alphabet de 26 lettres.
Ce nombre peut se noter 26!, qui se lit factorielle 26. Pour en obtenir une estimation, il suffit de taper 26! sous Google.
A raison d'une clef testée par nanoseconde, combien d'années faudrait-il à un ordinateur pour tester les 26! clefs possibles pour un chiffre par substitution monoalphabétique ?
Mais si le texte est assez long, un cryptanalyste peut facilement contourner cette explosion combinatoire en analysant les fréquences des lettres du texte chiffré et en les comparant aux fréquences des lettres mesurées sur l'ensemble des mots de la langue française (si on sait que le message est en Français). Pour la distribution des fréquences des lettres dans la langue française, on pourra consulter le site `http://www.lexique.org/`.

3. Pour le cas particulier des chiffre de substitution par décalage comme le chiffre de César, combien de clefs (décalages) sont-elles possibles ?

4. En Python, la fonction `ord()` retourne le code ASCII d'un caractère. La fonction `chr()` retourne le caractère associé à un code ASCII.

Définir en quelques lignes le codage ASCII des caractères.

Tester les codes suivants :

```
>>> alphabet = 'ABCDEFGHIJ'
>>> for c in alphabet:
...     print(ord(c))
```

```
>>> for i in range(65,91):
...     print(chr(i))
```

5. Compléter le code du programme `cesar_chiffreDM.py` ci-dessous pour qu'il réalise successivement les actions suivantes :

- prendre en entrée une chaîne de caractères stockant le texte en clair
- remplacer les symboles de ponctuation par des espaces, les minuscules par des majuscules et supprimer les accents
- coder le texte sous la forme d'une liste d'entiers compris entre -1 et 25 : le rang alphabétique de 0 à 25 pour les lettres et la valeur -1 pour l'espace ⇒ *on peut s'en passer*
- constituer puis retourner une chaîne de caractères représentant le texte initial chiffré avec le chiffre de César en codant les espaces par le caractère '@' ou un espace ' ' ou tout autre caractère.

cesar_chiffreDM.py

```
chaine = input('Entrez le texte en clair : \n')
ponctuation = [', ','! ','? ','_','-',' : ',; ',"\n","\t","'",'... ','...
   ','. ','.']
#on remplace tous les caractères spéciaux ou de ponctuation de chaine par
   un espace
for c in ponctuation:
    chaine = chaine.replace(c,' ')
#on transforme chaine en majuscules
chaine = chaine.upper()
#on élimine tous les accents
chaine = chaine.replace('É','E')
........
.......
chiffre = ''
........
print('texte chiffré : \n',chiffre)
```

6. Modifier le programme précédent en `cesar_dechiffreDM.py` pour qu'il permettre de déchiffrer un texte chiffré avec le chiffre de César.

3 Chiffrement affine

Parmi les chiffres de substitution monoalphabétique, le chiffre de César est un cas particulier de **chiffrement affine**.
Si on code chaque lettre de notre alphabet latin de 26 lettres, par son rang alphabétique, un chiffrement affine peut se modéliser par une fonction mathématique qui au rang x de la lettre en clair compris entre 0 et 25 associe le rang $y \equiv ax + b \bmod 26$ de la lettre chiffrée.
Ainsi y est une fonction affine de x de coefficients a et b, le calcul étant réalisé modulo 26 c'est-à-dire que y est le reste de la division euclidienne de $ax + b$ par 26. Le couple $(a\,;\,b)$ constitue la **clef** du chiffrement affine.
Par exemple si on choisit $a = 11$ et $b = 3$:

- la lettre A de rang $x = 0$ est chiffrée par la lettre de rang $y \equiv 11 \times 0 + 3 \equiv 3 \bmod 26$ donc par D
- la lettre J de rang $x = 9$ est chiffrée par la lettre de rang $y \equiv 11 \times 9 + 3 \equiv 24 \bmod 26$ donc par Y.

Avec $a = 1$ et $b = 3$, on retrouve le chiffre de César.
Les calculs modulo 26 ne s'effectuent que sur des entiers. On admet les règles suivantes sur des entiers x, y, a, b :

- si $x \equiv a \bmod 26$ et $y \equiv b \bmod 26$ alors $x + y \equiv a + b \bmod 26$
- si $x \equiv a \bmod 26$ et $y \equiv b \bmod 26$ alors $xy \equiv ab \bmod 26$

1. Si on choisit la clef $(a\,;\,b) = (10\,;\,0)$ pour un chiffrement affine, que peut-on dire des lettres chiffrant D et Q ? Est-ce acceptable ?

On admettra qu'un couple d'entiers $(a\,;\,b)$ est une clef de chiffrement affine si et seulement si a et 26 sont premiers entre eux (s'ils n'ont pas de diviseur commun autre que 1).
Sinon on peut montrer qu'il y a au moins deux lettres en clair qui sont chiffrées par la même lettre.

2. Soit le chiffrement affine de clef $(a\,;\,b) = (11\,;\,3)$.

Vérifier que le texte en clair **AVE CESAR** est chiffré par **DAV ZVTDI**.

3. Ecrire un programme Python `chiffre_affineDM.py` qui réalise le chiffrement affine de clef $(a\,;\,b)$ d'un texte en clair. On fera le même prétraitement du texte que pour le chiffre de César (remplacement des symboles de ponctuation par des espaces, des minuscules par des majuscules, suppression des accents).

4 Déchiffrement affine, partie facultative

Soit un chiffrement affine de clef $(a\,;\,b) = (11\,;\,3)$. $a = 11$ et 26 n'ont pas de diviseur commun donc cette clef est possible d'après un résultat admis.
Pour le vérifier, il nous suffit de montrer que toute lettre chiffrée correspond à une unique lettre en clair (s'il y a deux solutions ce n'est pas un chiffre acceptable).
Si on connaît y le rang de la lettre chiffrée, existe-t-il toujours un rang $0 \leqslant x \leqslant 25$ de lettre en clair tel que $y \equiv 11x + 3 \bmod 26$?
De $y \equiv 11x + 3 \bmod 26$ on déduit que $y - 3 \equiv 11x \bmod 26$.
Celà peut paraître étrange mais pour la multiplication modulo 26, un entier peut avoir un inverse qui est un autre entier : ainsi $5 \times 21 = 105$ et $105 = 4 \times 26 + 1$ donc $5 \times 21 \equiv 1 \bmod 26$ et 21 est l'inverse de 5 modulo 26. $\frac{1}{5}$ est l'inverse de 5 dans l'ensemble des réels mais pas dans l'ensemble des entiers modulo 26 (les entiers compris entre 0 et 25).
Attention tous ces entiers n'ont pas un inverse modulo 26, 10 par exemple n'est pas inversible ...
Mais si 11 a un inverse d modulo 26 alors x existe car : $d(y-3) \equiv d \times 11 \times x \bmod 26$ et donc $d(y-3) \equiv x \bmod 26$.

1. Créer une feuille de calcul avec le tableur `Calc` pour calculer tous les produits $a \times n \bmod 26$ pour a entier quelconque et n entier compris entre 0 et 25.

On utilisera la fonction `MOD()` qui calcule le reste de la division euclidienne du contenu d'une cellule par un entier avec la syntaxe `=MOD(cellule;entier)`.

	A	B	C	D	E	F	G	H	I
1	a	11							
2									
3	n	0	1	2	3	4	5	6	7
4	a*n mod 26	0	11	22	7	18	3	14	25

2. Quel est l'inverse de 11 modulo 26 ?

3. Ecrire un programme Python `dechiffre_affineDM.py` qui réalise le déchiffrement affine d'un texte chiffré avec une clef $(a\,;\,b)$ (l'inverse de a modulo 26 étant saisi par l'utilisateur).

Tester le programme pour déchiffrer **DAV ZVTDI** avec la clef $(11\,;\,3)$.