Cryptographie affine

A) Le codage affine

Pour transmettre un message secret, on peut utiliser la procédure :

- A toute lettre de l'alphabet, on associe le nombre lu dans le tableau ci-dessous :

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
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

- Soit x le nombre obtenu à partir de la lettre de départ. On calcule $y = ax + b$.
  Le couple d'entiers $(a;b)$ où $a \neq 0$, s'appelle la *clé* du codage.
- On calcule ensuite $c(x)$, reste de la division de y par 26.
- Enfin, on associe à $c(x)$ la lettre correspondante par lecture inverse du tableau.

1) Cas $a = 1$

Dans ce cas, le codage se résume à un décalage. Pendant la guerre des Gaules, Jules César utilisait un tel procédé ($y = x + 3$) pour envoyer des messages chiffrés à Cicéron qui était resté en poste au Sénat à Rome.

Quel mathématicien se cache derrière **YXKETM** avec la clé (1,19) ?

2) Quelques exemples de clé

a) Coder votre prénom avec la clé (7,17).

b) Vérifier que si $a \equiv a'$ [26] et $b \equiv b'$ [26], les codes obtenus avec les clés (a,b) et (a',b') sont identiques.

c) De combien de clés dispose-t-on en prenant $1 \leqslant a \leqslant b$ et $0 \leqslant b \leqslant 25$?

d) On prend pour clé (2,13). Recopier et compléter le tableau suivant :

Mot Initial	**E**	**N**	**T**	**I**	**E**	**R**
Code x	4					
$2x + 13$	21					
$c(x)$	21					
Mot codé	v					

Le petit nombre de clés explique que cette méthode ne soit plus employée depuis longtemps.

Quel problème apparaît dans le codage ci-dessus ?

3) Ecrire un programme permettant de :

- Demander à l'utilisateur un texte constitué de lettres en majuscules.
- Parcourir chaque caractère du texte précédent, si c'est une lettre parmi les 26 lettres de l'alphabet, pratiquer le chiffrement affine sinon, laisser le caractère inchangé.
- Afficher le texte codé.

Sur TI, vous pourrez utiliser les instructions suivantes, présentes dans le menu Math puis String.
length(x) donnant la longueur de la chaîne c, mid(x,k,l) donnant la sous-chaîne contenue dans x définie entre les caractères de position k et l dans x, char(65) donne "A" et char(66) donne "B" sont les instructions permettant d'obtenir les codages mémoires des deux premières majuscules, les suivants étant consécutives, ord("A") donne 65 et ord("B") donne 66 : de même aux codes consécutifs partants de 65, on peut retrouver les lettres majuscules consécutives, "A"&"B" donne "AB" ; c'est la concaténation des chaînes de caractères.

B) Décodage

1) Dans le cas d'un codage affine de clé (7,17), chercher une lettre dont le codage final soit **B**.

a) A l'aide de l'algorithme d'Euclide, trouver deux entiers u et v tels que $7u - 26v = 1$. Justifier $7u \equiv 1$ [26].

b) Soit x le code initial de la lettre cherchée.
Démontrer que x vérifie (E) : $7x \equiv -16$ [26].
En déduire que $x \equiv -16u$ [26].

c) En déduire l'entier x compris entre 0 et 25 solution de (E)

puis la lettre cherchée.

2) Expliquer pourquoi la méthode ci-dessus assure le décodage de n'importe quelle lettre dès qu'on choisit une clé (*a*,*b*) telle que *a* soit premier avec 26.

3) Avec la clé (5,13), quel mot se cache derrière **SUNOF** ?

4) Programme de calcul des coefficients dans la relation de Bezout.
a et *b* sont des entiers naturels, $a > b > 0$ et *g* est leur PGCD. Alors il existe *u* et *v* entiers relatifs tels que $au + bv = g$.
L'exemple ci-après et le programme permettent de trouver *u* et *v*.

On met en œuvre sur l'exemple ci-après l'algorithme d'Euclide et on calcule les restes successifs en fonction de *a* et *b*. On sait que le dernier reste non nul est le PGCD. On développe alors les calculs de façon à faire apparaître à chaque étape une écriture du reste de la forme $au + bv$.

Prenons $a = 47$ et $b = 35$.

$47 = 35\times1 + 12$ soit $a = b + 12$	donc	$12 = a - b$
$35 = 12\times2 + 11$	$b = (a - b)\times2 + 11$	$11 = -2a + 3b$
$12 = 11\times1 + 1$	$a - b = (-2a + 3b)\times1 + 1$	$1 = 3a - 4b$
$11 = 11\times1 + 0$		

Nous avons bien obtenu le PGCD comme combinaison linéaire de *a* et *b*.

Les programmes suivants, pour Casio et TI, permettent d'afficher les valeurs de *U* et *V* après avoir demandé à l'utilisateur les valeurs de *A* et *B*.

Famille Casio	**Famille TI**
"A=" : ? → R "B=" : ? → Y I → U : 0 → W : 0 → V : I → X While Y ≠ 0 Int(R÷Y) → Q U → Z : W → U : Z-Q*W → W V → Z : X → V : Z-Q*X → X R → Z : Y → R : Z-Q*Y → Y WhileEnd "U=" : U◢ : "V=" : V◢ "PGCD=" : R◢	Input "A=", R Input "B=", Y I→ U : 0 → W : 0 → V : I → X While Y ≠ 0 Int(R÷Y) → Q U → Z : W → U : Z-Q*W → W V → Z : X → V : Z-Q*X → X R → Z : Y → R : Z-Q*Y → Y End Disp "U=", U, "V=3, V Disp "PGCD=", R

Reprendre le programme précédent afin de :
- Demander à l'utilisateur un texte constitué de lettres en majuscules et la clé de codage.
- Déterminer la clé de décodage associée.
- Parcourir chaque caractère du texte précédent, si c'est une lettre parmi les 26 lettres de l'alphabet, pratiquer le déchiffrement affine sinon, laisser le caractère inchangé.
- Afficher le texte décodé.