TD Maple n°4. Cryptographie.

Exercice 1 *Chiffrement/Déchiffrement*

1. Implémenter, comprendre et tester la procédure suivante sur quelques exemples.

```
> to_number := proc(L :: list)
  local long,chiffre,car,ii,num ;
  num :=table(['a'=0,'b'=1,'c'=2,'d'=3,'e'=4,'f'=5,'g'=6,
  'h'=7,'i'=8,'j'=9,'k'=10,'l'=11,'m'=12,'n'=13,'o'=14,
  'p'=15,'q'=16,'r'=17,'s'=18,'t'=19,'u'=20,'v'=21,
  'w'=22,'x'=23,'y'=24,'z'=25,'_'=26]) ;
  long :=nops(L) :
  chiffre :=[] :
  for ii from 1 to long
            do  chiffre :=[op(chiffre),num[L[ii]]] :
            od :
  chiffre ;
  end ;
```

2. Construire une procédure `to_letter` qui effectue l'opération inverse.

****

Exercice 2 *Cryptage de César.*

1. Construire une procédure `cesarcode` d'arguments (L, k) qui code une liste de caractères L par un cryptage de César de clé k. (On pourra crypter le message suivant : "*le tresor est dans la cave*").
2. Construire une procédure `cesardecode` d'arguments (L, k) qui décode une liste de caractères cryptée par un cryptage de César de clé k.

****

Exercice 3 *Codage affine*

1. Construire une procédure `affinecode` d'arguments (L, a, b) qui code une liste de caractères par un cryptage affine de clé (a, b). (On pourra ajouter un test `if` qui renvoie un message d'erreur si (a, b) n'est pas une bonne clé).
   (a) Construire une procédure `InvModn` d'arguments (a, n) qui renvoie l'inverse de a modulo n si a est inversible modulo n, et 0 sinon.

(b) Construire une procédure `affinedecode` d'arguments (L, a, b) qui décode une liste de caractères par un cryptage affine de clé (a, b).

⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆

Exercice 4 *Piratage*

En espionnant Alice et Bob, on voit passer le message suivant :

```
[m, r, a, h, x, _, c, q, r, h, o, c, h, w, c, e, r, q, q, h, c, d, q, q, h, h]
```

On sait qu'Alice et Bob communiquent à l'aide d'un cryptage de César. Quel est le message initial ? (On pourra tenter une attaque exhaustive ou une attaque statistique).

⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆⋆