Correction

NSI - 2021 Étranger Jour 2 (21-NSIJ2G11)

Exercice 3 - Codage XOR

1.

Puissances de deux	$2^7 = 64$	$2^6 = 32$	$2^5 = 16$	$2^4 = 8$	$2^3 = 4$	$2^1 = 2$	$2^0 = 1$
Reste	89	25	25	9	1	1	1
Quotient	1	0	1	1	0	0	1

$89_{(10)} = 01011001_{(2)}$ sur 8 bits.

2.

	1	1	0	0	1	1	1	0
$\oplus$	0	1	1	0	1	0	1	1
=	1	0	1	0	0	1	0	1

3.

```
def xor_crypt(message, cle):
    """ Renvoie la liste des entiers correspondant au message codé.
        Le message et la clé ont la même longueur."""
    code = []
    for i in range(len(message)):
        code.append(xor(ord(message[i]), ord(cle[i])))
    return code
```

Remarque : On parle plutôt de « chiffrage » que de « cryptage ». Seule la notion de décryptage a un sens puisque qu'elle consiste à décrypter un code sans en connaître la clé. On parle de « chiffrage » puisque la clé est forcément connue au moment du chiffrage, on ne peut pas crypter sans connaître la clé de chiffrement.

4.

```
def generer_cle(mot, n):
    """ Répète le mot jusqu'à obtenir une chaîne de n caractère."""
    cle = ""
    i = 0
    while len(cle) < n:
        cle += mot[i % len(mot)] # la clé peut être plus longue que n
        i += 1
    return cle
```

5.

E_1	E_2	$E_1 \oplus E_2$	$(E_1 \oplus E_2) \oplus E_2$
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

Pour décoder un message codé par cette méthode, il suffit d'appliquer la même opération que celle réalisée lors du codage, c'est-à-dire l'opération XOR bit à bit pour chaque lettre du message codé et la clé.