Partie 3

Corrections

3.1 Chiffrement symetrique par la fonction XOR

```
def bin2text(bits):
    i = 0
    m = ''
    while i + 8 <= len(bits):
        oct = bits[i:i+8]
        m += chr(bin2dec(oct))
        #print(oct)
        i = i+8
    return m

>>> bin2text('0100100001000101010011000100110001001111')
>>> 'HELLO'

def xor(a,b):
    return int(a!=b)

def chiffre_xor(bits,cle):
    # bits: str constitué de bytes
    # cle: str
    m = ''
    for b in range(len(bits)):
        bit1 = int(bits[b])
        bit2 = int(cle[b%len(cle)])
        m += str(xor(bit1,bit2))
    return m
```

en console :

```
>>> m = text2bin('HELLO')
>>> num_chiffre = chiffre_xor(m,'101')
>>> print('code:' + num_chiffre)
>>> m_chiffre = bin2text(num_chiffre)
>>> print(m_chiffre)
>>> num_de = chiffre_xor(num_chiffre,'101')
>>> m_de = bin2text(num_de)
>>> print(m_de)
code:1111111010011110001000011111101010010100
þ !ú
HELLO
```