TD : La simulation numérique
Cryptographie – Corrigé

Méthode de césar :

```
import string
alphabet = string.ascii_uppercase

def cesar(m,d):
   assert type(m)==str
   assert type(d)==int
   mc="" #message codé
   n = len(alphabet)
   for c in m:
       if c in alphabet:
           indice = (alphabet.index(c) +d) % n
           mc+= alphabet[indice]
       else:
           mc+=c
   return mc

#test
m="LE TRESOR EST CACHE DANS LE CHATEAU DE: SUITE AU PROCHAIN MESSAGE!"
d=3
mc=cesar(m,d)
print(mc)

def decesar(mc,d):
   return cesar(mc,-d)

#test
m=decesar(mc,d)
print(m)

def cassecesar(m):
   assert type(m)==str
   L=[0]*26
   for c in alphabet:
   L[alphabet.index(c)]=mc.count(c)
   d = L.index(max(L)) - alphabet.index('E')
   return decesar(m,d)
#test
print(cassecesar(mc))
```

Méthode de vigenere:

```
import string

alphabet = string.ascii_uppercase

def rang(c):
    assert c.isalpha() and c in alphabet
    return alphabet.index(c)

def lettre(r):
    assert type(r)==int
    return alphabet[r%len(alphabet)]

def vigenere(m,cle,k=1):
    assert type(m)==str
    assert type(cle)==str

    mc=""
    i=0
    for c in m :
        if c in alphabet :
            mc += lettre(rang(c) + k * rang(cle[i%len(cle)]))
            i+=1
        else:
            mc += c

    return mc

vigenere(m,cle)

def devigenere(m,cle):
    return vigenere(m,cle,k=-1)

devigenere(m,cle)
```