Chiffrement RSA (3.13)

```
def rsaParameters(p,q,e):
    print 'on calcule n=p*q=',p,'*',q,'=',p*q
    n=p*q
    print
    print 'on calcule φ(n)=(p-1)(q-1)=',p-1,'*',q-1,'=',(p-1)*(q-1)
    print
    print 'on vérifie que e (=',e,') est premier avec φ(n) puisque
gcd(e,φ(n))=',gcd(e,(p-1)*(q-1)),"on calcule aussi l'inverse d=e^(-1)
mod φ(n)."
    print "Pour on utilise l'algorithme d'Euclide étendu, soit
directement d=1/mod(e,φ(n))=",1/mod(e,(p-1)*(q-1))
    d=1/mod(e,(p-1)*(q-1))
    print
    print 'Au final la clef publique est Ke=(e,n)=(',e,',',p*q,'). La
clef privée est Kd=(d,n)=(',d,',',n,').'
    return d
```

```
p=47
q=59
n=p*q
e=17
d=rsaParameters(p,q,e)
```

```
    on calcule n=p*q= 47 * 59 = 2773

    on calcule φ(n)=(p-1)(q-1)= 46 * 58 = 2668

    on vérifie que e (= 17 ) est premier avec φ(n) puisque gcd(e,φ(n))=
    1 on calcule aussi l'inverse d=e^(-1) mod φ(n).
    Pour on utilise l'algorithme d'Euclide étendu, soit directement
    d=1/mod(e,φ(n))= 157

    Au final la clef publique est Ke=(e,n)=( 17 , 2773 ). La clef privée
    est Kd=(d,n)=( 157 , 2773 ).
```

```
def RSAencryption(m,e,n): #
    print 'Cryptage: on calcule',m,'^',e,'mod', n,'=',mod(m^e, n)
    encr=mod(m^e, n)
    return encr
```

```
c=RSAencryption(66,e,n)
```

```
    Cryptage: on calcule 66 ^ 17 mod 2773 = 872
```

```
# On calcule c^d mod n et on vérifie que cela correspond au message
original
```

```
RSAencryption(c,d,n)
```

```
Cryptage: on calcule 872 ^ 157 mod 2773 = 66
66
```