Eldis YMERAJ → #22015179

Zied JERBI → #22013944

RAPPORT TP2

LE CODE RSA

Dans ce rapport nous allons expliquer comment on a fait les functions de code RSA et leurs jeu d'essais.

Dans un fichier.py on a crié tout les functions que nous avons besoin pour les jeu d'essais.

1. Fonction pgcd(a, m)

```
def pgcd(a, b):
    while b!=0:
        a, b= b, a % b
    return a
```

Le function pgcd trouve le plus grand diviseur entre deux nombres. Le function a des paramètres deux entiers a et b, tant que b est different de 0 alors, a = b et b = a[b] et a la fin on va retourner a(On a déjà ecpliqué ca en Tp1).

2. Fonction inverse(a, m)

```
def inverse(a,m):
    if pgcd(a,m)==1:
        u1,u2,u3,=1,0,a
        v1,v2,v3 = 0,1, m
        while v3!=0:
            q=u3//v3
            u1, u2, u3, v1, v2, v3=v1, v2, v3, (u1-q*v1), (u2-q*v2), (u3-q*v3)

        return (u1%m)
```

Le function Inverse permet de donner l'inverse multiplicatif modulo m d'un nombre et pour réaliser ca on a utilisé algo d'Euclide Étendu.

Notre fonction prends pour parametre le nombre qu'on va trouver l'inverse et le modulo.

Si pgcd de deux nombres qu'on va tester est egal a 1, ca veut dire que les nombres sont premiers entre eux alors, on passe dans la methode d'Euclide Étendu qu'on a fait en cours. U1 prends la valeur 1, u2=0 et u3 =a, v1=0, v2=1 et v3=m. Tant que notre nombre qu'on va tester est different de 0 alors, q=u3/v3 et u1=v1, u2=v2, u3=v3, v1=u1-q*v1,v2=u2-q*v2, v3=u3-q*v3

Eldis YMERAJ → #22015179

Zied JERBI → #22013944

Et a la fin on retourne u1[m] qui est l'inverse multiplicatif de a.

On est base dans ce tableau qu'on a fait en cours(Euclide Étendu)>

r	u	v
$r_0 = a$	$u_0 = 1$	$v_0 = 0$
$r_1 = b$	$u_1 = 0$	$v_1 = 1$
…	…	…
r_{i-1}	u_{i-1}	v_{i-1}
r_i	u_i	v_i
$r_{i-1} - (r_{i-1} \div r_i) r_i$	$u_{i-1} - (r_{i-1} \div r_i) u_i$	$v_{i-1} - (r_{i-1} \div r_i) v_i$
…	…	…
$r_n = pgcd(a, b)$	$u_n = u$	$v_n = v$
0	u_{n+1}	v_{n+1}

3. EstPremier(Nb)

Ce function permet de montrer si un nombre est premier ou non, si le nombre est premier, le function va retourner True.

```
def estPremier(Nb):
    if Nb < 2:
        return False
    i=2
    while i<int(math.sqrt(Nb)+1) and Nb%i !=0:
        i=i+1
        if i == int(math.sqrt(Nb)):
            return True
```

Notre fonction prends un argument Nb qui est le nombre qu'on va tester s'il est premier. Si Nb est inferieur a 2 il peut pas être premier donc le function va retourner faux. On prends i=2 et on entre dans le boucle while, si i est inferieur a la racine carrée de Nb + 1 et Nb mod i est different de 0 alors on decale i→i+1, après si est égal a la racine carrée de Nb le function renverra True ca veut dire que le nombre qu'on a testé est premier.

4. LPremiers(Nb)

Ce function permet de nous trouver la liste de tous les nombres premiers inferieur ou égal à Nb.

```
def LPremiers(Nb):
    liste=[int]*Nb
    j=0
    for i in range(2, int(math.sqrt(Nb)+1)):
        if estPremier(i):
            liste[j]=i
            j=j+1
    return (liste)
```

Eldis YMERAJ → #22015179

Zied JERBI → #22013944

Notre fonction prend pour paramètre un entier Nb, liste va être une tableau d'entiers avec Nb cases. On a initialisé j=0 et pour i de 2(car 0 et 1 ne sont pas prèmiers) à racine de Nb +1 alors, on utilise le fonction estPremier pour vérifier les nombres premiers <= Nb a partir de 2ème case de liste jusqu'à la derrnière. Donc si les nombres qui testent dans fonction estPremier sont premiers alors, on les mettre dans un autre liste qui sont juste les nombres premiers et a la fin on renvoie la liste avec les nombres premiers.

P.s : On a essayé de faire cette fonction avec une autre manière mais il y a des erreurs qu'on a pas arrivé de les trouver et on a fait avec la manière que nous avons expliqué précédament.

```
def LPremiers(Nb):
    Tab=[True]*Nb
    Tab[0]= False
    Tab[1]= False
    for i in range(2, Nb + 1):
        j = 2 *i
        while j< Nb+i:
            Tab[j] = False
            j = j + 1

        liste = ''
        for i in range(Nb):
            if Nb[i] == True:
                liste = liste + i
```

5. Trouve(Nb)

Ce function permet de nous trouver les nombres p et q.

```
def trouve(Nb):
    liste= LPremiers(Nb)
    i=0
    while i<numpy.size(liste):
        if Nb%liste[i]==0 and estPremier(Nb/liste[i]):
            return(liste[i],Nb/liste[i])
        i=i+1
```

Pour faire cette fonction on a cherché en internet et on avait besoin pour un autre bibliothèque qu'il s'appelle numpy.size() qui compte le nombre d'éléments le long d'un axe donné.

Notre liste contient tout les nombres premiers <= Nb. Tant que i est infèrieur à nombre d'éléments qui sont dans notre liste alors, si Nb mod liste est égal à 0 et si les nombres(Nb/liste[i) sont premiers alors on va retourner la liste et la division de Nb/liste[i] et avec cette manière on a trouvé p et q.

Eldis YMERAJ → #22015179

Zied JERBI → #22013944

6. trouveN(p, q), trouvePhi(p, q)

Le function trouveN permet de trouver N qui est égal à p*q et le function trouvePhi permet de trouver phi qui est égale avec (p-1)(q-1).

```
def trouveN(p, q):
    return p*q

def trouvePhi(p, q):
    return (p-1) * (q-1)
```

Ici on a juste calcule N et phi qu'on a donné les formules dans le sujet.

7. Chiffrer(M, E, N)

Ce function permet de chiffrer un message M et on vérifie si M est infèrieur à N et à la fin il calcule le cryptogramme $C=M^E\%N$

```
def chiffrer(M, E, N):
    if(M < N):
        return pow(M, E, N)
    else:
        print("M est superieur a N donc on peut pas chiffrer le message!")
```

Ici notre fonction prend pour parametttre le message M, clé publique E et N. Si M est inferieur à N le function va retourner M^E mod N et sinon le function va imprimer : M est superieur a N donc on peut pas chiffrer le message!

8. Dechiffrer(C, N, D)

Ce function va déchiffrer le cryptogramme C.

```
def dechiffrer(C, N, D):
        return pow(C, D, N)
```

Ici le function prend pour paramèttre le cryptogramme C qu'on a chiffré, N et clé privé D. Il va retourner $C^D\%N$ comme dans les instructions dans la feuille de sujet.

9. cryptMess(message)

Eldis YMERAJ → #22015179

Zied JERBI → #22013944

Ce function permet de faire la correspondance entre les lettres d' alphabet et nombres.

```python
def cryptMes(message):
  alphabet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
  message = 'RSA'
  charCodee = []
  for char in message:
    charCodee.append(alphabet.index(char.upper()))
  return charCodee
```

Ici on a mis notre alphabet dans un tableau et chaque lettre de l'alphabet correspond à indexe de tableau.

Jeux d'essais

Exercice 1 et 2 :

Pour les jeux d'essais on a fait dans un nouveau qu'on a appelé TP2 fichier et on a les importé dans n

```python
from ficOutil import inverse
from ficOutil import trouve
from ficOutil import trouveN
from ficOutil import trouvePhi
from ficOutil import testerE
from ficOutil import chiffrer
from ficOutil import dechiffrer
```

Les réponses d'exo 1 et 2 sont :

```
============================EXERCICE 1============================
1.1 Alice va obtenir nombre:  204
1.2 Les nombres p et q sont:  17 23.0
1.3 phi de N:  352.0
============================EXERCICE 2============================
2.1(a) Le message chiffree est C =  122
2.1(b) Le cryptogramme C dechiffree est M' =  12
2.2(a)       N =  3763
         phi(N) =  3640
2.2(b) Est-ce que E est acceptable?  True
       Le nombre correspondant est D =  83
2.2(c) Les elements qui constituent le cle publique sont E et N, on a deja trouve.
       Les elements qui constituent le cle privee sont D et N
```

Jeu d'essai pour un petite piece d'exo 3 :

Eldis YMERAJ → #22015179

Zied JERBI → #22013944

```
==========================EXERCICE 3==========================
Chiffrage de message RSA est:  [17, 18, 0]

In [16]:
```

Affichage de tout jeux d'essais :

```
**************************TP2 CODE RSA**************************
*            TRAVAILLE PAR:                                    *
*                          Eldis YMERAJ --> #22015179          *
*                          Zied JERBI   --> #22013944          *
****************************************************************



==========================EXERCICE 1==========================
1.1 Alice va obtenir nombre:  204
1.2 Les nombres p et q sont:  17 23.0
1.3 phi de N:  352.0
==========================EXERCICE 2==========================
2.1(a) Le message chiffree est C =  122
2.1(b) Le cryptogramme C dechiffree est M' =   12
2.2(a)       N =  3763
         phi(N) =  3640
2.2(b) Est-ce que E est acceptable?  True
       Le nombre correspondant est D =  83
2.2(c) Les elements qui constituent le cle publique sont E et N = , on a deja trouve.
E =  307
N =  221
       Les elements qui constituent le cle privee sont D et N
D =  83
==========================EXERCICE 3==========================
Chiffrage de message RSA est:  [17, 18, 0]
```