Họ và tên SV: Nguyễn Thị Thanh Bảo

MSSV: 18020201

Mã lớp môn học: 2021II_INT3213_2

KẾT QUẢ VÀ CÁCH LÀM BA HỆ MẬT

1. RSA

- Chọn p, q hai số nguyên tố sao cho n=p*q có 512 bit
  Từ đó ta chọn p có 256 bit, q có 256 bit (với p, q là hai số nguyên tố)
  p=108727583271094938804067468427449177408134313544087916009935531860166325579893
  q=105107824811362760480723104326217167178876041720593089117887439801773691758649
  n=p*q=11428119774621103218341973081665201986950286831892574487534563338906485983694188799959854697680296258504581466766617526443220159720879885337186789905906016
- Kiểm tra số bit của n: được 512 bit

```
p=108727583271094938804067468427449177408134313544087916009935531860166325579893
q=105107824811362760480723104326217167178876041720593089117887439801773691758649

n =p * q
print('Độ dài bit của n là:', len_in_bits(n))
```

- Phi = (p-1)*(q-1)=
  11428119774621103218341973081665201986950286831892574487534563338906485983694188799959854697680296258504581466766617526443220159720879885337186789905906016
- Chọn e sao cho: 0<e<phi và gcd(e,phi)=1
  e do người dùng nhập vào thỏa mãn điều kiện nếu sai thì nhập lại

```
#chọn e thỏa mãn 1<e<phi và gcd(e,phi)=1
e=int(input('Vui lòng nhập số e: '))
while(math.gcd(e,phi)!=1 and e<=1 and e>=phi):
    e= int(input('e không thỏa mãn. Vui lòng nhập lại e: '))
```

  Giả sử t chọn
  e=107020324028697780970257011405130133134438432904802840806508701642073169434969

- Ta kiểm tra được e và phi nguyên tố cùng nhau thông qua hàm math.gcd hoặc có thể xác định thông qua hàm dưới đây:

```
def gcd(a, b):
    if (a == 0):
        return b
    if (b == 0):
        return a
    if (a == b):
        return a
    if (a > b):
        return gcd(a%b, b)
    return gcd(a, b%a)
```

- Tính d= e^-1 mod phi
=34163574813757349866806759732628348855160429384426694200294488679692200869635602854316259266073339353055797141852547549667452772882960386567411606715581129
- Khóa công khai K'=(n,e), khóa bí mật K''=d

Bản rõ x là họ tên đầy đủ

Chuỗi s='nguyenthithanhbao'

Mã ascii chữ cái có mã bé nhất là 97 nên trong vòng lặp for chạy từ i đến độ dài chuỗi s-1 (chạy len(s) vòng), lấy mã ascii bằng câu lệnh ord() Mỗi lần lặp cộng vào giá trị x (ord(s[i])-97)*26^(len(s)-i-1)

```
s="nguyenthithanhbao"
x=0
for i in range(len(s)):
    t= ord(s[i])-97
    x=x+ t*pow(26,len(s)-1-i)
```

Ra được kết quả x=5783273930616242611482 50

- **Mã hóa**:

y=(x^e)modn=27641600688970044768215686863209594699809398304627392291843088945118243866435223009464930347701297604096667574518924348979948156738968850696229173458555 87

- **Giải mã**

z= y^d mod n= 5783273930616242611482 50

- **z= x** => Giải mã thành công
- **Chữ ký:**

Không phải cặp số nào cũng tính được modulo nghịch đảo. Hàm eValid kiểm tra điều kiện này. Nếu đúng trả về giá trị a^-1 mod b.

```python
# Kiểm tra xem số b có là modulo nghịch đảo của a không
def eValid(a,b):
    try:
        d=pow(a,-1,b)
        return d
    except ValueError:
        return False
```

(a,b) phải thỏa mãn a.b =1(mod phi)

Số a do người dùng nhập không thỏa mãn điều kiện tồn tại a^-1 mod phi thì sẽ nhập lại. Giả sử nhập a=17

b= a^-1 mod phi

=100836350952539146044193880132340017531914295575522716066481441225645464562007548234939894391296731692687483530293684056851942585772469576504589322699170 73

chữ ký **sig=x^a mod n**

=779623961036072177697410803572780999111737529776569158867013643710684374987623843371935511130925500921923101374716018598682046437838091897402552904940698 8

Kiểm thử chữ ký:

sig^b mod n = 578327393061624261148250

= x => **chữ ký đúng**

Kết quả thu được khi chạy mã nguồn như hình dưới đây:

```
Số nguyên tố p=  108727583271094938804067468427449177408134313544087916009935531860166325579893
Số nguyên tố q=  105107824811362760480723104326217167178876041720593089117887439801773691758649
Độ dài bit của n là: 512
n= 114281197746211032183419730816652019869502868318925744875345633389064859836944026353679371553795810490773351331112045367984848407260077083088487299232445570
phi=(p-1)*(q-1)= 114281197746211032183419730816652019869502868318925744875345633389064859836944026353679371553795810490773351331112045367984848407260077083088487299232445570
0
Vui lòng nhập số e: 3
e không thỏa mãn. Vui lòng nhập lại e: 107020324028697780970257011405130133134438432904802840806508701642073169434969
Tìm được e với phi nguyên tố cùng nhau
Khóa công khai k=(n,e)=(n= 114281197746211032183419730816652019869502868318925744875345633389064859836944026353679371553795810490773351331112045367984848407260077083088487299232445570 e= 107020324028697780970257011405130133134438432904802840806508701642073169434969 )
Số d (ed=1(mod phi))= 341635748137573498668067597326283488551604293844266942002944867969220086963560285431625926607333935305579714185254754966745277288296038656741160671558112 9
Thông báo cần gửi x= 578327393061624261148250
Kết quả mã hóa y=  276416006889700447682156868632095946998093983046273922918430889451182438664352230094649303477012976040966675745189243489799481567389688506962291734585558 7
Kết quả giải mã z= 578327393061624261148250
x=z => Giải mã thành công
Vui lòng nhập số a: 17
b= 100836350952539146044193880132340017531914295575522716066481441225645464562007548234939894391296731692687483530293684056851942585772469576504589322699170 73
Chữ ký sig(x) là 779623961036072177697410803572780999111737529776569158867013643710684374987623843371935511130925500921923101374716018598682046437838091897402552904940698 8
sig^b mod n = 578327393061624261148250
=> Chữ ký đúng
```

2. Elgamal

2.1. Chọn p là số nguyên tố 256 bít, có thể dùng hàm sẵn có gen số nguyên tố lớn ngẫu nhiên

```
from libnum import generate_prime
p= generate_prime(256)
```

Chọn được:
p=669721480944836992633829689618977017564118039925325447903337501272795990473 21

2.2. Tìm phần tử nguyên thủy

- **Phân tích p-1 ra các thừa số nguyên tố** {2, 42013, 30999980719}

Cách làm: n=p-1

+ khởi tạo một list s rỗng

+ p-1 là một số chẵn lớn nên sẽ chia hết cho 2 (trừ số nguyên tố 2) nên chắc chắn sẽ có thừa số nguyên tố là 2 => s.append(2)

+ giảm n cho tới khi n không còn chia hết cho 2

+ Vòng lặp for cho i từ 3 đến căn(n) (vì phần range chỉ tới trước số đó nên để round(căn (n))+1) với bước nhảy 2 (vì n không còn chia hết cho 2 nên nó chắc chắn không chia hết cho số chẵn nào)

+ Nếu n chia hết cho i dùng vòng while để kiểm tra và giảm n=n/i, thêm i vào list s. Hàm if (i!= s[-1]) để tránh lặp phần tử trong list s

+ Trả về s được một list các số là thừa số nguyên tố của p-1

```
def findPrimefactors(n):
    s =[]
    if(n%2==0): s.append(2)
    while (n%2==0):
        n=n/2
    for i in range(3,round(math.sqrt(n))+1,2):
        while(n%i==0):
            if(i!=s[-1]):
                s.append(i)
            n=n//i
    if(n>2): s.append(n)
    return s
```

- **Thuật toán tìm phần từ nguyên thủy đầu tiên của p là alpha**

+ alpha phải thỏa mãn điều kiện alpha^((p-1)/i) mod p # 1 với mọi i thuộc list thừa số nguyên tố của p-1

+ alpha chạy tử 2 đến p-1. Với mỗi giá trị của alpha

Đặt một biến flag = false

Cho i chạy từ phần tử đầu tới phần tử cuối list các thừa số nguyên tố của p-1:

Nếu alpha^((p-1)/i) mod p =1 thì flag= true ta chuyển luôn alpha tới giá trị tiếp theo. Ngay khi có giá trị alpha thỏa mãn flag =false trả luôn về giá trị đó và kết thúc việc tìm kiếm

```
def findPrimitive(p):
        num =p-1
        s= findPrimefactors(num)
        for n in range (2,p):
                flag= False
                for i in s:
                        k= round(num/i)
                        if pow(n,k,p)==1:
                                flag=True
                                break
                if (flag==False):
                        return n
        return -1
alpha= findPrimitive(p)
```

2.3. Mã hóa và giải mã

- Từ hàm tính phần tử nguyên thủy bên trên tính được phần tử nguyên thủy đầu tiên là alpha=2
- a, k có thể nhập:
- Chọn **a= 1107** – là ngày tháng năm sinh , **k=1107200**
- Tính được **beta= alpha^a** = 2^1107
- Số hóa tên sử dụng kết quả từ RSA **x**=57832739306162426114825O
- **Mật mã của x: (y1,y2)**

+**y1=(alpha^k)modp**=153694393085515386867936754069514522931654353626127026251989811221725508468l9

+ y2=(x*(beta^k)) mod p

t=beta^kmodp=450188676839983947418706024047200551679932568907533404882610607701981391910201

y2=(t*x)modp=255302477992141149849129184014381439907679752340106038877823400786269636009 6

- **Giải mã**:

Res= (y2* (y1^a)^-1) mod p

Tính lần lượt:

res1=y1^amodp=450188676839983947418706024047200551679932568907533404882610607701981391910201

res2=(res1^-1modp)=

226988430404549960753676577790368327216641215363138866760380117748333653010O5

res=res2*y2 mod p=57832739306162426114825O

res= x => giải mã thành công:

2.3 Chữ ký và kiểm thử chữ ký

- Hàm kiểm tra p-1 có là modulo nghịch đảo của k không tương tự vào hàm eValid bên trên RSA
- Khóa k1 do người dùng nhập vào nếu không có modulo đảo ngược thì nhập lại: Giả sử nhập k1=17

```
k1=int(input('Vui lòng nhập khóa k1: '))
while(eValid(k1,p-1)==False):
    k1= int(input('k1 không hợp lệ.Vui lòng nhập lại k1: '))
```

- **Chữ ký (gamma,s)**

gamma= alpha^k1 mod p= 2^17 mod p=131072

s=(x-a*gamma)*k1^-1 mod (p-1)

+ k^-1 mod (p-1)=
630326099712787757773016178464919545942699331694423950967847060021455049857 13

⇨ s=630326099712787757773016178464919545942699331694423951308039644175351588710 98

- **Kiểm thử chữ ký**

Nếu: beta^gamma*gamma^s mod p = alpha^x mod p thì chữ ký đúng

+(beta^gamma)modp=393220339181861103658646383739612052829255374799979623751227854221403382473 48

+(gamma^s)modp=515662039498169960947734307539196768347383054028013799872294466832114679771 09

⇨ beta^gamma* gamma^s mod p=
22716773878548385110189818288122421367020108249998212636661030053734699041 98

+alpha^x=22716773878548385110189818288122421367020108249998212636661030053734699041 98

⇨ Chữ ký đúng

Kết quả sau khi chạy mã nguồn là:

```
C:\Users\BaoBao\Downloads\TongHopBTL18020201>py Elgamal_18020201.py
Số nguyên tố p= 669721480944836992633829689618977017564118039925325447903337501272795990473 21
 p-1 phân tích được ra các thừa số nguyên tố là: [2, 42013, 30999980719.0]
Phần tử nguyên thủy của p là alpha= 2
Vui lòng nhập a: 1107
Vui lòng nhập khóa k: 11072000
x= 578327393061624261148250
a= 1107
k= 11072000
beta=  173862211718321388707500982829986147661247212407263302367810517725160075872542956636229
018665056775581421801765809246508480495686953407103630791018787848047640137772792401796246820 6
610453079133538155156547356566593638199778832634438829476612121994288739297245736985470092145 1
525208468637547709280559595082838355390654451907512531681 28
Mật mã của x là (y1,y2):
y1= 153694393085515386867936754069514522931654353626127026251989811221725508468 19
y2= 255302477992141149849129184014381439907679752340106038877823400786269636009 6
Kết quả giải mã được là: 578327393061624261148250
Giải mã thành công
Vui lòng nhập khóa k1: 17
630326099712787757773016178464919545942699331694423950967847060021455049857 13
Chữ ký sig(x,k)=( 131072 , 630326099712787757773016178464919545942699331694423951308039644175 3
5158871098 )
beta^gamma* gamma^s mod p= 227167738785483851101898182881224213670201082499982126366610300537 3
469904198
alpha^x= 22716773878548385110189818288122421367020108249998212636661030053734699041 98
Chữ ký sig(x,k1) được xác nhận là đúng
```

3. Elliptic

3.1 Các bước tính toán

Chọn p 160 bít

p=1092917513274372122286774856355924354973391200253

(E) y^2= x^3+x+10 mod p

Điểm sinh:

P=(4,1006462020242386297383479675793873753633925500690)

s= 19

B=sP=19P=

(947480247578034037740948639564929572167247053308,

1407461294252468766369454245684618113793332341 15)

Công khai: (E,p,P,B)

Bản tin: chọn M=13P

M=13P=

(978870588477973327035907771061843671039668073718,

2755115626345914199740355635011633403447608466 93)

Chọn k=15

Mã hóa (M1, M2)= (kP, M+kB)

M1= kP = 15P=
(559426316932207925935763372539173988888261533888,
869440360783327024034032343078708814646533075744)

M2=M+kB=
(174538201582295611594780805219446847995575181069,
100499892571436231135995170846493639394282160759 1)

Bản mã (M1,M2)

Giải mã:

sM1= 19M1=
(100290472530789121939743974490604873271493260164 0,
101275443801342862433415713210494002588161340787 6)

M2-sM1=

⇨ (97887058847797332703590777106184367103966807371 8,
27551156263459141997403556350116334034476084669 3)

⇨ Giải mã thành công M2-sM1 = M

Chữ ký: (r,s)

Tính được số điểm trên đường cong là số nguyễn tố:

n=109291751327437212228677628250355612513956207441 1

h=mã hóa tên=578327393061624261148250

Chọn khóa riêng người gửi là d=3

Q=P*d= (100478722353082932088271391730507890220733751522 3,
240119811013494265930943058289752169635393843086)

Chọn k1=15

k1P=k1*P= (559426316932207925935763372539173988888261533888,
869440360783327024034032343078708814646533075744)

r=x(k1P)modn=559426316932207925935763372539173988888261533888

s= (h+d*r)*(k1^-1)mod n=
913358106454314474864121986898935493654948571229

Kiểm thử chữ ký

w= s^-1 mod n= 590720471666905213442405359380281398238311121719

u1= (h*w) mod n=
5455439310002677275784863679384291560483585762 75

u2=(r*w)modn=91106953627428287976061415985741307312344254899 1

z= u1*P +u2*Q

Nếu:v=x(z) mod n=r=> chữ ký đúng

Kết quả chạy mã nguồn

```
C:\Users\BaoBao\Downloads\TongHopBTL18020201>py Elliptic_18020201.py
Elliptic curve: y^2 = x^3 + 1*x + 10 mod p
p= 109291751327437212228677485635592435497339120025 3
P= (4, 100646202024238629738347967579387375363392550069 0)
Công khai (E,p,P,B):
Điểm sinh P= (4, 100646202024238629738347967579387375363392550069 0)
p= 109291751327437212228677485635592435497339120025 3
B= (947480247578034037740948639564929572167247053308, 140746129425246876636945424568461811379333234115)

Bản tin M=13P: M=
(978870588477973327035907771061843671039668073718, 275511562634591419974035563501163340344760846693)
Mã hóa: (M1,M2)
M1= (559426316932207925935763372539173988888261533888, 869440360783327024034032343078708814646533075744
)
M2= (174538201582295611594780805219446847995575181069, 100499892571436231135995170846493639394282160759
1)
Giải mã:
(100290472530789121939743974490604873271493260164 0, 101275443801342862433415713210494002588161340787 6)
(978870588477973327035907771061843671039668073718, 275511562634591419974035563501163340344760846693)
Giải mã thành công
Khóa riêng người gửi là d= 3
Q= (100478722353082932088271391730507890220733751522 3, 240119811013494265930943058289752169635393843086
)
k1= 15
k1P (559426316932207925935763372539173988888261533888, 869440360783327024034032343078708814646533075744
)
Chữ ký là cặp số nguyên (r,s)=( 559426316932207925935763372539173988888261533888 , 913358106454314474864121986898935493654948571229 )
w= 590720471666905213442405359380281398238311121719
u1= 5455439310002677275784863679384291560483585762 75
u2= 911069536274282879760614159857413073123442548991
Chữ ký đúng
```

3.2 Xây dựng thuật toán code

- Tính P+Q (hàm add): Q=P+Q => (x, y) = (x1, y1) + (x2, y2)
  + Xây dựng code theo thuật toán tính lamda
    Nếu P=Q thì lamda = (y2-y1)/(x2-x1)
    Nếu P# Q thì lamda = (3x1^2+a)/(2*y1)
  + x= (lamda^2-x1-x2) % p
  + y= (lamda(x1-x) –y) %p

- Tính nP (hàm multiply):
  Xây dựng thuật toán tính hàm Multiply dựa vào hàm add đã xây dựng:
  + Khởi tạo một điểm Q là điểm rỗng

+ Một biến i chạy vòng lặp for hoặc while từ giá trị 0 đến giá trị n-1
+ Với mỗi vòng lặp cộng một giá trị P vào Q sau n vòng lặp ta sẽ được
=> Q=n*P

```
def multiply(P, n):
    Q = (None, None)
    for i in range(n):
        Q = add(Q, P)
    return Q
```

- Thuật toán với x bất kỳ kiểm tra xem x có thuộc E không và tìm tọa độ y:
+ z= x^3+7x+11 modp
+ Dùng hàm hỗ trợ trong python libnum.sqrtmod để kiểm tra xem có số nào có căn bậc hai (căn(y^2)) mà mod p=z
+ Nếu có thì gán giá trị cho y
+ Nếu y^2%p = (x^3+7x+11)%p thì điểm này thuộc E còn lại thì không thuộc đường cong Elliptic

```
def onCurve(x):
    z=(x**3 +7*x+11)%p
    if(libnum.has_sqrtmod(z,{p:1})):
        y=next(libnum.sqrtmod(z,{p:1}))
    print("P(%d,%d)" % (x,y))
    if ((y**2 % p) == ((x**3+a*x+b) %p)):
        print("Điểm này thuộc E")
    else:
        print("Điểm này không thuộc E")
```

thử với x=4, ta tính được tọa độ
y= 784982772541555539160817507012934597147055058 05

và kiểm tra được điểm đó thuộc P

- **Lập bảng cửu chương cho P (từ 2-nP)**

+ Dùng vòng lặp for cho biến i chạy từ 2 đến n.
+ Trong mỗi vòng lăp: Khởi tạo một điểm T=i*P (dựa vào hàm multiply) Kiểm tra và tính bằng hàm onCurve để biết nó có thuộc E không.

```
for i in range (2,21):   #n=21, có thể thay n tùy ý
    T= ecc.multiply(P,i)
    print(i,'P=')
    onCurve(T[0])
```

+ Kết quả:

Chạy mã nguồn ta thu được bảng cửu chương tới 21P như sau:

```
1 P=
(4,100646202024238629738347967579387375363392550069O)
Điểm này thuộc E
2 P=
(795167549722059204356082988438444963394101930953,25154860343351042484130167097032224230306393636 8)
Điểm này thuộc E
3 P=
(100478722353082932088271391730507890220733751522 3,852797702260877856355831798066172185337997357167)
Điểm này thuộc E
4 P=
(594182434034844715161191756710598549978450297433,711327171637742617620959086087779714320736174 50)
Điểm này thuộc E
5 P=
(565719660318668303378107947755696007740346519573,543086889658798221014508645873678846521221238768)
Điểm này thuộc E
6 P=
(975763367122206321286123851022653399832683217013,598983612956395853024757274275188813432911834252)
Điểm này thuộc E
7 P=
(116879151672824654047376688810632572433975752673,105468176512587593496587374673296066043860142751 8)
Điểm này thuộc E
8 P=
(600117918102396474456895228122716621368150938 29,228903462698566932116804572421170547849337201888)
Điểm này thuộc E
9 P=
(925683488271923476107966517235899250574643596121,903786532490710496753365949437265407185657172052)
Điểm này thuộc E
10 P=
(970442615313314208165876549943768260106239008870,111356755502099115893822651400005721320781149381)
Điểm này thuộc E
11 P=
(662380013302573344961415591690545806743602994682,417701245306207769957779461447636752834880826702)
Điểm này thuộc E
12 P=
(783301568392651082290209371410712247685006118569,983292925182300223201686190936819789879635774630)
Điểm này thuộc E
```

```
13 P=
(97887058847797332703590777106184367103966807371 8,817405950639780702312739292854761014628630353560)
Điểm này thuộc E
14 P=
(33684031022922124869857806570350406767132936132 9,924969122963349010752589729899618295348949908976)
Điểm này thuộc E
15 P=
(55942631693220792593576337253917398888826153388 8,223477152491045098252742513277215540326858124509)
Điểm này thuộc E
16 P=
(38403179345408704911639162813104507268786250782 8,811480258231566262617412949238015798347286999318)
Điểm này thuộc E
17 P=
(69876123384476897776178943877407734638135961884 5,316022361453572453829010103715961846271533736634)
Điểm này thuộc E
18 P=
(91192676167572351338599510615085101335497299258 3,483791586224257098497584668046763725189082550025)
Điểm này thuộc E
19 P=
(94748024757803403774094863956492957216724705330 8,140746129425246876636945424568461811379333234115)
Điểm này thuộc E
20 P=
(99681499059206231191073364602604873405398743053 4,642219007211660442997318445787221848818144064457)
Điểm này thuộc E
```