During the exam you must solve the following 5 problems.

El Gamal E-Signature €3.00 | ElGamal Encryption €3.00 | Schnorr E-Signature €3.00 | Schnorr Identification €3.00 | MINI-HTTPS €5.00

Compatibility relations of modular arithmetic:
(a + b)mod p = (a mod p + b mod p) mod p.
(a * b) mod p =((a mod p) * (b mod p)) mod p.
a^p mod p = (a mod p)^p mod p.

Fermat little theorem: If p is prime, then for any integer a holds **$a^p = a$ mod p**.

1. We may assume that a is in the range $0 \le a \le p - 1$.
This is a simple consequence of the laws of modular arithmetic; we are simply saying that we may first reduce a modulo p since
a^p mod p = ((a)mod p)^p mod p.
1. It suffices to prove that for a in the range $1 \le a \le p - 1$.17

$$a^p = a \bmod p \quad | \; a^{-1} \bmod p$$
$$a^p \cdot a^{-1} = a \cdot a^{-1} \bmod p$$
$$\boxed{a^{p-1} = 1 \bmod p} \qquad 0 < a < p.$$

Indeed, if the previous assertion holds for such a, multiplying both sides by a yields the original form of the theorem.

Computation of exponents mod (p-1):

$$s = xh + r \longrightarrow g^{s \bmod (p-1)} \bmod p$$
$$g^{p-1} = 1 \bmod p \;\&\; g^0 = 1 \bmod p \qquad 0 \equiv p-1$$

a^z mod p = $a^{z \bmod (p-1)}$ mod p.

>> p=genstrongprime(28)   // strong prime number g eneration
p = 215393099   // p - is strong prime iff p=2*q+1, when q is also prime: q=(p-1)/2

```
>> q=(p-1)/2
q = 107696549
```

// finding a generator g: g - is a generator in $Z_p^* = \{1, 2, 3, \ldots, p-1\}$ *mod p
// iff g^q =/= 1 & g^2 =/= 1

```
>> g=2
g = 2
>> mod_exp(g,q,p)
ans = 215393098
>> mod_exp(g,2,p)
ans = 4
```
// g=2 is a generator

$s = (xh + r) \bmod (p-1)$; $g^s \bmod p = g^{s \bmod (p-1)} \bmod p$.

```
>> x=int64(randi(p-1))
x = 169506978
>> a=mod_exp(g,x,p)
a = 99428799
>> con=concat('hello bob',a)
con = hello bob99428799
>> h=hd28(con)
h = 252819993
>> r=int64(randi(p-1))
r = 96513791
```

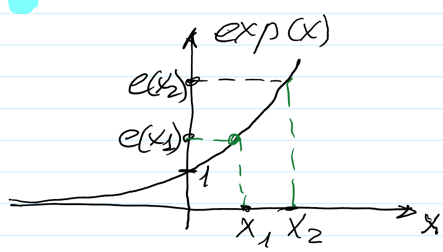
```
>> s=int64(x*h+r)
s = 42854753087924945
>> smodpm1=mod(s,p-1)
smodpm1 = 150400265

>> g_s=mod_exp(g,s,p)
g_s = 162111969
>> g_smodpm1=mod_exp(g,smodpm1,p)
g_smodpm1 = 162111969
```

DEF homomorphism

$exp(x) = e^x$; $exp: R \to R$; $e = 2{,}71\ldots$
$x \in R \to e^x \in R$.

$exp(x_1 + x_2) = e^{(x_1+x_2)} = e^{x_1} \cdot e^{x_2} = exp(x_1) \cdot exp(x_2)$

Additively - multiplicative homomorphism.
Since it is 1-to-1, then it is isomorphism.



$DEF(x) = g^x \bmod p$; p - strong prime
g - generator in $Z_p^* = \{1, 2, 3, \ldots, p-1\}$

$x \in Z_{p-1} = \{0, 1, 2, 3, \ldots, p-2\}$; $+ \bmod (p-1)$, $* \bmod (p-1)$, $- \bmod (p-1)$
$/ \bmod (p-1)$.
$|Z_{p-1}| = p-1$

$DEF(x) = a \in Z_p^* = \{1, 2, 3, \ldots, p-1\}$; $* \bmod p$, $/ \bmod p$.
$|Z_p^*| = p-1 = |Z_{p-1}|$

DEF realizes the following mapping   DEF: $Z_{p-1} \dashrightarrow Z_p^*$

$$DEF(x_1+x_2) = g^{(x_1+x_2)\bmod (p-1)} \bmod p = g^{x_1}\cdot g^{x_2} \bmod p =$$

(a * b) mod p =((a mod p) * (b mod p)) mod p.

$$=((g^{x_1} \bmod p)\cdot(g^{x_2} \bmod p)) \bmod p = DEF(x_1)\cdot DEF(x_2)$$

Additively - multiplicative homomorphism.
Since it is 1-to-1, then it is isomorphism.

ElGamal Encryption-Decryption

Public Parameters generation **PP** = (p, g).

Asymmetric Signing - Verification

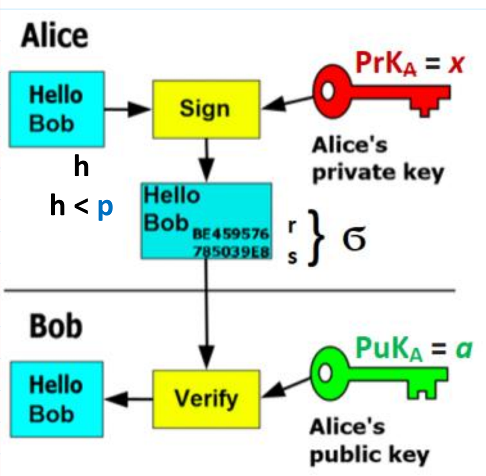
Sign(PrK_A, h) = σ = (r, s)

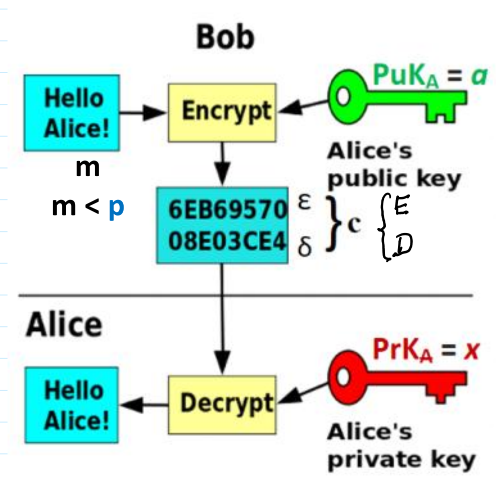
V=Ver(PuK_A, h, σ), V$\in$ {True, False} $\equiv$ {1, 0}

Asymmetric Encryption - Decryption

c=Enc(PuK_A, m)

m=Dec(PrK_A, c)





ElGamal Cryptosystem

1.Public Parameters generation **PP** = (p, g).

Generate strong prime number **p**:   >> p=genstrongprime(28) % strong prime of 28 bit length

Find a generator **g** in Z_p^*= {1, 2, 3, ..., **p**-1} using condition.

Strong prime p=2q+1, where q is prime, then g is a generator of Z_p^* iff

$g^q \neq$ 1 mod p and $g^2 \neq$ 1 mod p.

Declare **Public Parameters** to the network   **PP** = (**p**, **g**);

p= 268435019; **g=2;**

2^28-1= 268,435,455

>> 2^28-1
ans = 2.6844e+08
>> int64(2^28-1)
ans = 268435455

PrK = x <-- randi(Z_p^*) ==> PuK = a = g^x mod p

Asymmetric Encryption-Decryption: El-Gamal Encryption-Decryption

p=268435019; g=2;

Let message ***m~*** needs to be encrypted, then it must be encoded in decimal number ***m***: 1< ***m*** < ***p***.
E.g. ***m*** = 111222. Then ***m*** **mod** ***p*** = ***m***.

$\mathcal{Z}_p^* = \{1, 2, 3, \ldots, p-1\}$; $*$ mod p

$\mathcal{A}$: $\xrightarrow{PuK_A = a}$ $\mathcal{B}$: is able to encrypt m to $\mathcal{A}$: $m < p$

$\mathcal{B}$: $i \leftarrow randi(\mathcal{Z}_p^*)$

$$\left.\begin{aligned} E &= m \cdot a^i \bmod p \\ D &= g^i \bmod p \end{aligned}\right\} c = (E, D) \longrightarrow$$

$\mathcal{A}$: is able to decrypt $c = (E, D)$ using her $PrK_A = x$.

1. $D^{-x \bmod (p-1)} \bmod p$
2. $E \cdot D^{-x} \bmod p = m$

$(-x) \bmod (p-1) = (0 - x) \bmod (p-1) =$
$= (p-1-x) \bmod (p-1)$

$(p-1) \bmod (p-1) = 0$ since

$$\begin{array}{r|l} p-1 & p-1 \\ \hline -\,p-1 & 1 \\ \hline 0 & \end{array}$$

$(-x) \bmod (p-1) = (p-1-x)$

$D^{-x} \bmod (p-1) = D^{p-1-x} \bmod (p-1)$

$>> D\_mx = mod\_exp(D, p-1-x, p)$

```
> x=123
x = 123
>> pp=127
pp = 127
>> isprime(pp)
ans = 1
>> mx=mod(-x,pp-1)
mx = 3
>> mod(x+mx,pp-1)
ans = 0
```

D^{-x} mod ***p*** computation using Fermat theorem:
If ***p*** is prime, then for any integer ***a*** in Z_p^* holds $\boldsymbol{a^{p-1} = 1 \bmod p}$.

$D^{p-1} = 1 \bmod p \quad / \cdot D^{-x \bmod (p-1)} \bmod p$

$D^{p-1} \cdot D^{-x} = 1 \cdot D^{-x} \bmod p \Rightarrow D^{p-1-x} = D^{-x} \bmod p$

$$\boxed{D^{-x} \bmod p = D^{p-1-x} \bmod p}$$

Correctness

$Enc(PuK_A = a, i, m) = c = (E, D) = (E = m \cdot a^i \bmod p; D = g^i \bmod p)$

$Enc(PuK_A = a, i, m) = c = (E, D) = (E = m \cdot a^i \bmod p ; D = g^i \bmod p)$

$Dec(PrK_A = x, c) = E \cdot D^{-x} \bmod p = m \cdot a^i \cdot (g^i)^{-x} \bmod p =$

$= m \cdot (\underbrace{g^x}_{a})^i \cdot g^{-ix} = m \cdot g^{xi} \cdot g^{-ix} = m \cdot g^{xi-ix} \bmod p = m \cdot g^0 \bmod p =$

$= m \cdot 1 \bmod p = m \bmod p = m = 111222$

since $m < p$

$27 \mid 5$
$25 \quad 5$
2

If $m > p \rightarrow m \bmod p \neq m$; $27 \bmod 5 = 2 \neq 27$.

If $m < p \rightarrow m \bmod p = m$; $19 \bmod 31 = 19$.

Decryption is correct if $m < p$.

ASCII: 8 bits per char.

$\frac{2048}{8} = 256$ char.

Homomorphic Encryption

Let m_1 and m_2 have to be encrypted.

$Enc(PuK_A = a, i_1, m_1) = c_1 = (E_1, D_1) = (E_1 = m_1 a^{i_1} \bmod p, D_1 = g^{i_1} \bmod p)$

$Enc(PuK_A = a, i_2, m_2) = c_2 = (E_2, D_2) = (E_2 = m_2 a^{i_2} \bmod p, D_2 = g^{i_2} \bmod p)$

$c_{12} = c_1 \cdot c_2 = (E_1, D_1) \cdot (E_2, D_2) = (E_1 \cdot E_2, D_1 \cdot D_2) = (E_{12}, D_{12})$

$E_{12} = m_1 \cdot m_2 \cdot a^{i_1} a^{i_2} \bmod p = m_1 \cdot m_2 \, a^{(i_1+i_2) \bmod (p-1)} \bmod p =$

$= m_1 \cdot m_2 \, a^{i_{12}} \bmod p = m_{12} \, a^{i_{12}} \bmod p$

$D_{12} = D_1 \cdot D_2 = g^{i_1} \cdot g^{i_2} \bmod p = g^{i_1+i_2} \bmod p = g^{i_{12}} \bmod p$

$$\begin{cases} m_{12} = m_1 \cdot m_2 \bmod p \\ i_{12} = (i_1 + i_2) \bmod (p-1) \end{cases}$$

$Enc(PuK_A = a, i_{12}, m_1 \cdot m_2) = c_{12} = c_1 \cdot c_2 = (E_1 \cdot E_2, D_1 \cdot D_2) = (E_{12}, D_{12})$

Multiplicatively homomorphic encryption.

We need an additively-multiplicative encryption.

$n_1 = g^{m_1} \bmod p \Rightarrow Enc(PuK_A = a, i_1, n_1) = (E_1, D_1) =$

$= (E_1 = n_1 \cdot a^{i_1} \bmod p, D_1 = g^{i_1} \bmod p)$.

$n_2 = g^{m_2} \bmod p \Rightarrow Enc(PuK_A = a, i_2, n_2) = (E_2, D_2) =$

$= (E_2 = n_2 \cdot a^{i_2} \bmod p, D_2 = g^{i_2} \bmod p.$

By transforming m_1, m_2 to n_1, n_2 we obtain additively homomorphic encryption with respect to m_1, m_2, by encryption n_1, n_2 in a standard way.
Since
$n_1 * n_2 \bmod p = g^{m1} * g^{m2} \bmod p = g^{m1+m2 \bmod (p-1)} \bmod p$,
Then
Enc(a, $i_1+i_2 \bmod(p-1)$, $n_1 * n_2 \bmod p$) = Enc(a, $i_1+i_2 \bmod(p-1)$, $g^{m1+m2 \bmod (p-1)} \bmod p$) = $c_{12} = c_1 * c_2 \bmod p$.

Till this place