Sm2

*Project: impl sm2 with RFC6979

代码说明：

首先声明函数 gcd 求最大公因子，通过辗转相除法得到最大公因子。再声明函数 findM 求模逆，使用扩展欧几里得算法得到模逆。

```
def gcd(a,b):#求最大公因子
        while a!=0:
            a,b = b%a,a
        return b

def findM(a,m):#扩展欧几里得算法求模逆
    u1,u2,u3 = 1,0,a
    v1,v2,v3 = 0,1,m
    while v3!=0:
        q = u3//v3
        v1,v2,v3,u1,u2,u3 = (u1-q*v1),(u2-q*v2),(u3-q*v3),v1,v2,v3
    return u1%m
```

根据椭圆计算的公式声明 addo,mpoint 函数（如下图）

3.2.3.1 F_p上的椭圆曲线群

椭圆曲线$E(F_p)$上的点按照下面的加法运算规则，构成一个交换群：

a) $O+O=O$；

b) $\forall P=(x,y)\in E(F_p)\backslash\{O\}$，$P+O=O+P=P$；

c) $\forall P=(x,y)\in E(F_p)\backslash\{O\}$，$P$的逆元素$-P=(x,-y)$，$P+(-P)=O$；

d) 两个非互逆的不同点相加的规则：

设$P_1=(x_1,y_1)\in E(F_p)\backslash\{O\}$，$P_2=(x_2,y_2)\in E(F_p)\backslash\{O\}$，且$x_1\neq x_2$，

设$P_3=(x_3,y_3)=P_1+P_2$，则

$$\begin{cases} x_3=\lambda^2-x_1-x_2, \\ y_3=\lambda(x_1-x_3)-y_1, \end{cases}$$

其中

$$\lambda=\frac{y_2-y_1}{x_2-x_1};$$

e) 倍点规则：

设$P_1=(x_1,y_1)\in E(F_p)\backslash\{O\}$，且$y_1\neq 0$，$P_3=(x_3,y_3)=P_1+P_1$，则

$$\begin{cases} x_3=\lambda^2-2x_1, \\ y_3=\lambda(x_1-x_3)-y_1, \end{cases}$$

其中

$$\lambda=\frac{3x_1^2+a}{2y_1}。$$

```
def addo(x1,y1,x2,y2,a,p):#椭圆曲线上的点的加法运算
    if x1==x2 and y1==p-y2:#当两个点互逆时返回False
        return False
    if x1!=x2:#当两个非互逆的不同点相加时的lamda值
        lamda=((y2-y1)*findM(x2-x1,p))%p
    else:
        lamda=(((3*x1*x1+a)%p)*findM(2*y1,p))%p#当两个相同点相加时的lamda值
    x3=(lamda*lamda-x1-x2)%p#根据公式计算x3,y3值
    y3=(lamda*(x1-x3)-y1)%p
    return x3,y3

def mpoint(x,y,k,a,p):#椭圆曲线上的倍点运算
    k=bin(k)[2:]
    m,n=x,y
    for i in range(1,len(k)):#代入addo函数循环两两相加
        m,n=addo(m, n, m, n, a, p)
        if k[i]=='1':
            m,n=addo(m, n, x, y, a, p)
    return m,n
```

声明加密函数 encrypt，循环计算随机数 k([1,n-1])，椭圆曲线点 $C_1=[k]G=(x_1,y_1)$,$S=[h]P_B$,$[k]P_B=(x_2,y_2)$，并计算 t=KDF($x_2||y_2$,klen)。当 t 不为全 0 时跳出循环，计算 C_2,C_3。$C_2=M\oplus t$，$C_3=Hash(x_2||M||y_2)$。

```
def encrypt(m:str):#加密
    plen=len(hex(p)[2:])
    m='0'*((4-(len(bin(int(m.encode().hex(),16))[2:])%4))%4)+bin(int(m.encode().
    klen=len(m)
    while True:
        k=randint(1, n)#产生随机数
        while k==dB:
            k=randint(1, n)
        x2,y2=mpoint(xB, yB, k, a, p)#计算椭圆曲线点
        x2,y2='{:0256b}'.format(x2),'{:0256b}'.format(y2)
        t=kdf(x2+y2, klen)
        if int(t,2)!=0:#当t不是全0时跳出循环
            break
    x1,y1=mpoint(gx, gy, k, a, p)
    x1,y1=(plen-len(hex(x1)[2:]))*'0'+hex(x1)[2:],(plen-len(hex(y1)[2:]))*'0'+he
    c1='04'+x1+y1
    c2=((klen//4)-len(hex(int(m,2)^int(t,2))[2:]))*'0'+hex(int(m,2)^int(t,2))[2:
    c3=sm3h(hex(int(x2+m+y2,2))[2:])
    return c1,c2,c3
```

运行指导：

通过改变 data 的值对不同数据进行 sm2 加密

运行结果：

输出数据值及其加密后的值。

```
====================== RESTART: C:\Users\ano\Desktop\sm2.py ======================
0x656E6372797074696F6E207374616E674616E646172646E207374616E64617264

ciphertext:
04191FC3 893568F8 9416A7D1 159E44BA 417BB4EB 895AAB44 52E223E7 ED1CBCFF E4808499
 A643DAA3 BA0FECF4 D19A423B BB7A36A6 AA623FE2 61477BE1 E00DF335 923076C6 9D9A77E
B 26CC0F8B DB3741D1 722EB564 AE7EEB0D 84B82AC8 CAC4C783 464046C2 EF5B7D50 647C7F
2F 34066A22 A73F7573 90393031 F73C37CC A40D20B8 D02D706A 4511D12A B5F1DB66 1AFBE
485 4359088D B46B7DAC 46BEBA19 AD510113 E6668B0E
```