HGAME 2022 Week4 writeup by sasasas

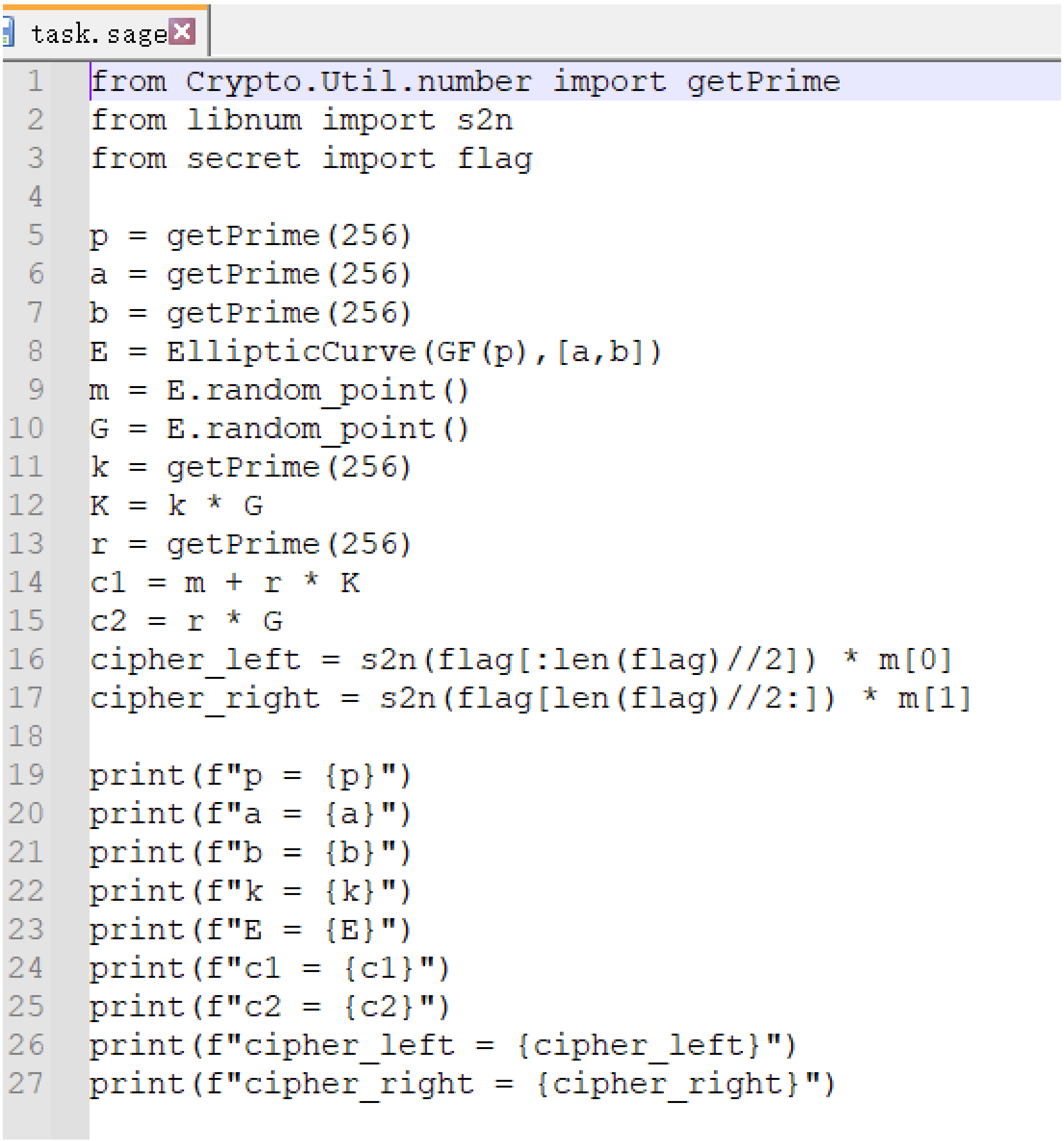
HGAME 2022 Week4 writeup by sasasas
- CRYPTO
  - ECC
  - PRNG

CRYPTO

ECC

1.百度ECC，椭圆曲线加密

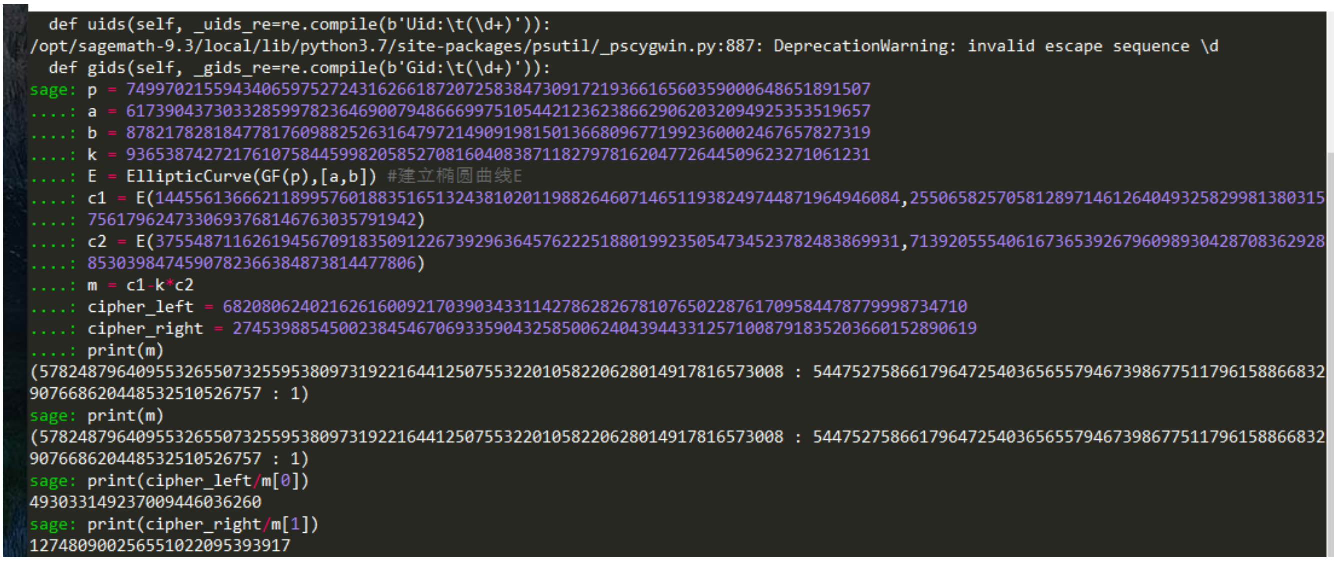
2.欣赏一下，非常标准，信息给的非常充分，大概只要知道m=c1-k*c2就好



```
from Crypto.Util.number import getPrime
from libnum import s2n
from secret import flag

p = getPrime(256)
a = getPrime(256)
b = getPrime(256)
E = EllipticCurve(GF(p),[a,b])
m = E.random_point()
G = E.random_point()
k = getPrime(256)
K = k * G
r = getPrime(256)
c1 = m + r * K
c2 = r * G
cipher_left = s2n(flag[:len(flag)//2]) * m[0]
cipher_right = s2n(flag[len(flag)//2:]) * m[1]

print(f"p = {p}")
print(f"a = {a}")
print(f"b = {b}")
print(f"k = {k}")
print(f"E = {E}")
print(f"c1 = {c1}")
print(f"c2 = {c2}")
print(f"cipher_left = {cipher_left}")
print(f"cipher_right = {cipher_right}")
```

3.但是吧，这里的乘法和除法似乎不太对劲，m,G 也像是奇奇怪怪的数据类型。直接大数乘除结果离谱。

4.最后用SageMath加上网上解密代码，得出flag, 知其然不知其所以然

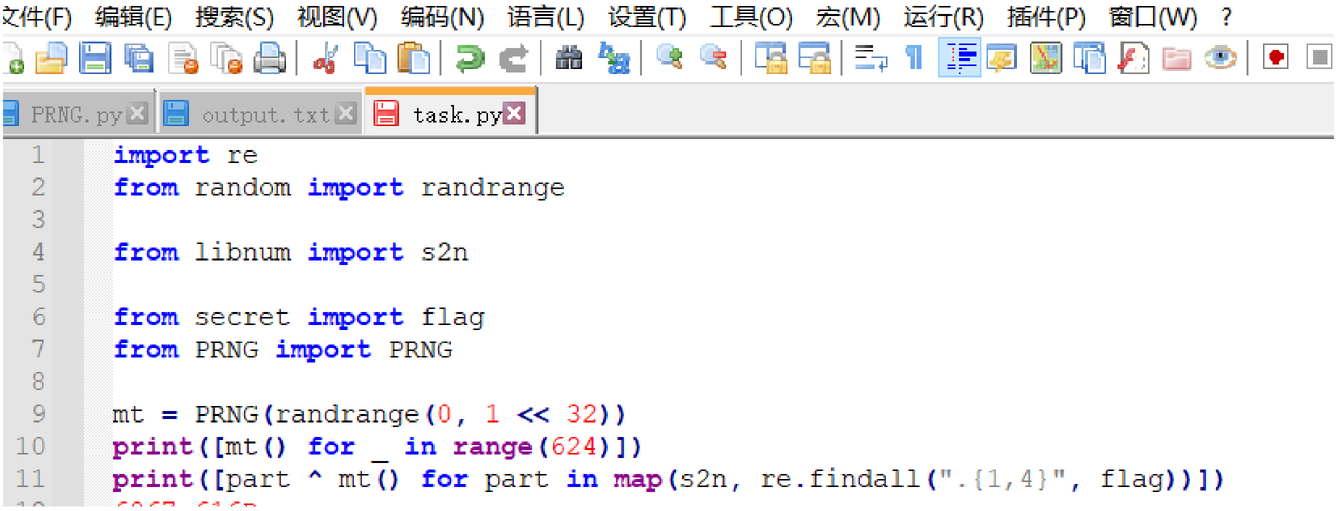


```
  def uids(self, _uids_re=re.compile(b'Uid:\t(\d+)')):
/opt/sagemath-9.3/local/lib/python3.7/site-packages/psutil/_pscygwin.py:887: DeprecationWarning: invalid escape sequence \d
  def gids(self, _gids_re=re.compile(b'Gid:\t(\d+)')):
sage: p = 74997021559434065975272431626618720725838473091721936616560359000648651891507
....: a = 61739043730332859978236469007948666997510544212362386629062032094925353519657
....: b = 87821782818477817609882526316479721490919815013668096771992360002467657827319
....: k = 93653874272176107584459982058527081604083871182797816204772644509623271061231
....: E = EllipticCurve(GF(p),[a,b]) #建立椭圆曲线E
....: c1 = E(14455613666211899576018835165132438102011988264607146511938249744871964946084,25506582570581289714612640493258299813803157
....: 5617962473306937681467630357919942)
....: c2 = E(37554871162619456709183509122673929636457622251880199235054734523782483869931,71392055540616736539267960989304287083629288
....: 8530398474590782366384873814477806)
....: m = c1-k*c2
....: cipher_left = 68208062402162616009217039034331142786282678107650228761709584478779998734710
....: cipher_right = 27453988545002384546706933590432585006240439443312571008791835203660152890619
....: print(m)
(57824879640955326550732559538097319221644125075532201058220628014917816573008 : 54475275866179647254036565579467398677511796158866832
907668620448532510526757 : 1)
sage: print(m)
(57824879640955326550732559538097319221644125075532201058220628014917816573008 : 54475275866179647254036565579467398677511796158866832
907668620448532510526757 : 1)
sage: print(cipher_left/m[0])
493033149237009446036260
sage: print(cipher_right/m[1])
127480900256551022095393917
```

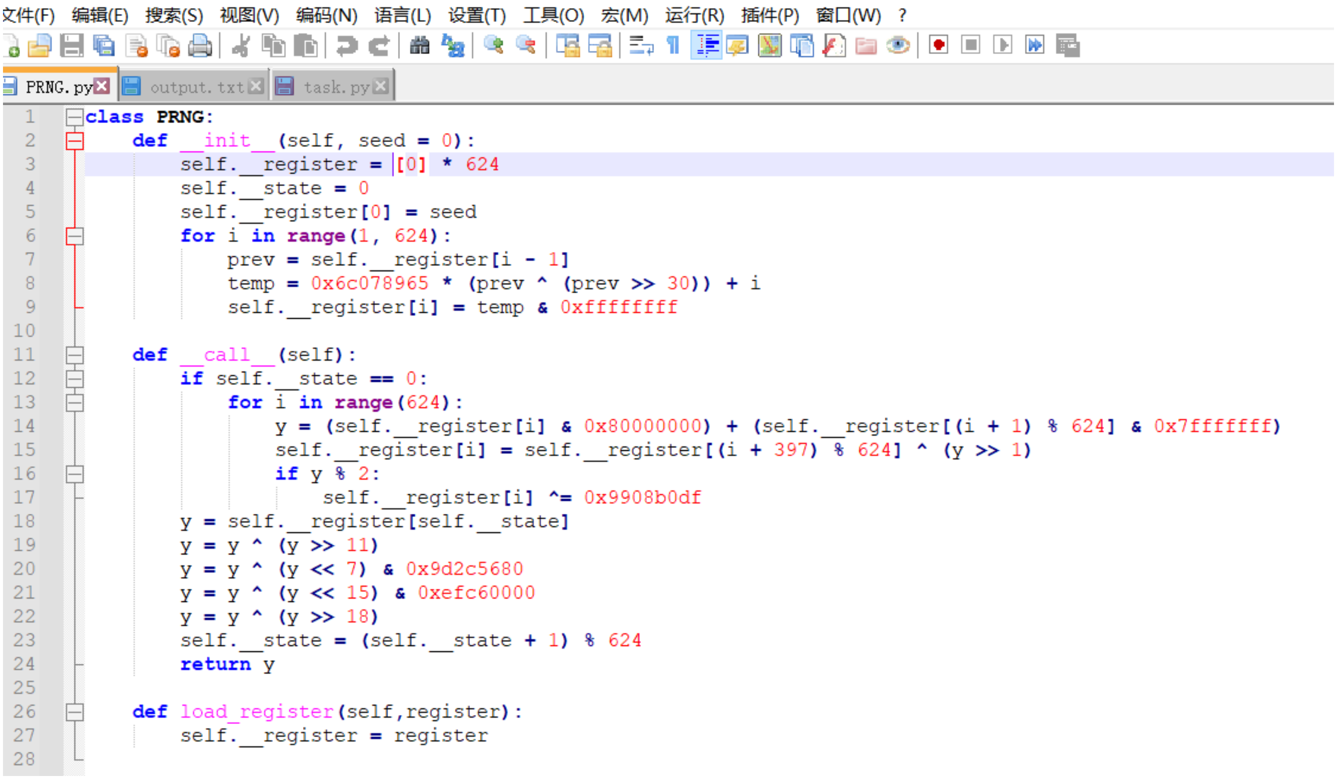
5.最后10进制转16转ascii

PRNG

1.欣赏一下生成代码


文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P) 窗口(W) ?
PRNG.py
output.txt
task.py

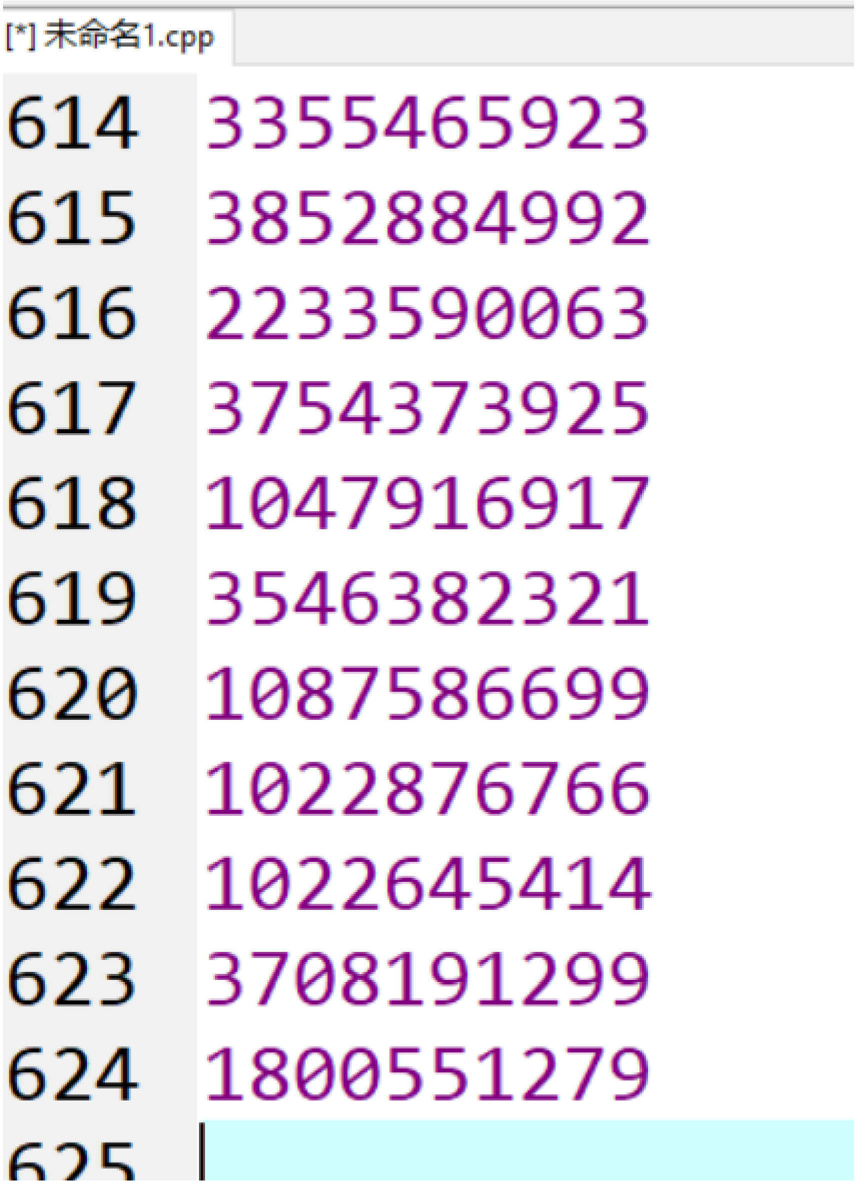
```python
import re
from random import randrange

from libnum import s2n

from secret import flag
from PRNG import PRNG

mt = PRNG(randrange(0, 1 << 32))
print([mt() for _ in range(624)])
print([part ^ mt() for part in map(s2n, re.findall(".{1,4}", flag))])
```


文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P) 窗口(W) ?
PRNG.py
output.txt
task.py

```python
class PRNG:
    def __init__(self, seed = 0):
        self.__register = [0] * 624
        self.__state = 0
        self.__register[0] = seed
        for i in range(1, 624):
            prev = self.__register[i - 1]
            temp = 0x6c078965 * (prev ^ (prev >> 30)) + i
            self.__register[i] = temp & 0xffffffff

    def __call__(self):
        if self.__state == 0:
            for i in range(624):
                y = (self.__register[i] & 0x80000000) + (self.__register[(i + 1) % 624] & 0x7fffffff)
                self.__register[i] = self.__register[(i + 397) % 624] ^ (y >> 1)
                if y % 2:
                    self.__register[i] ^= 0x9908b0df
        y = self.__register[self.__state]
        y = y ^ (y >> 11)
        y = y ^ (y << 7) & 0x9d2c5680
        y = y ^ (y << 15) & 0xefc60000
        y = y ^ (y >> 18)
        self.__state = (self.__state + 1) % 624
        return y

    def load_register(self,register):
        self.__register = register

```

2.我试了一下，发现seed=0 好像不太对，感觉受到了欺骗

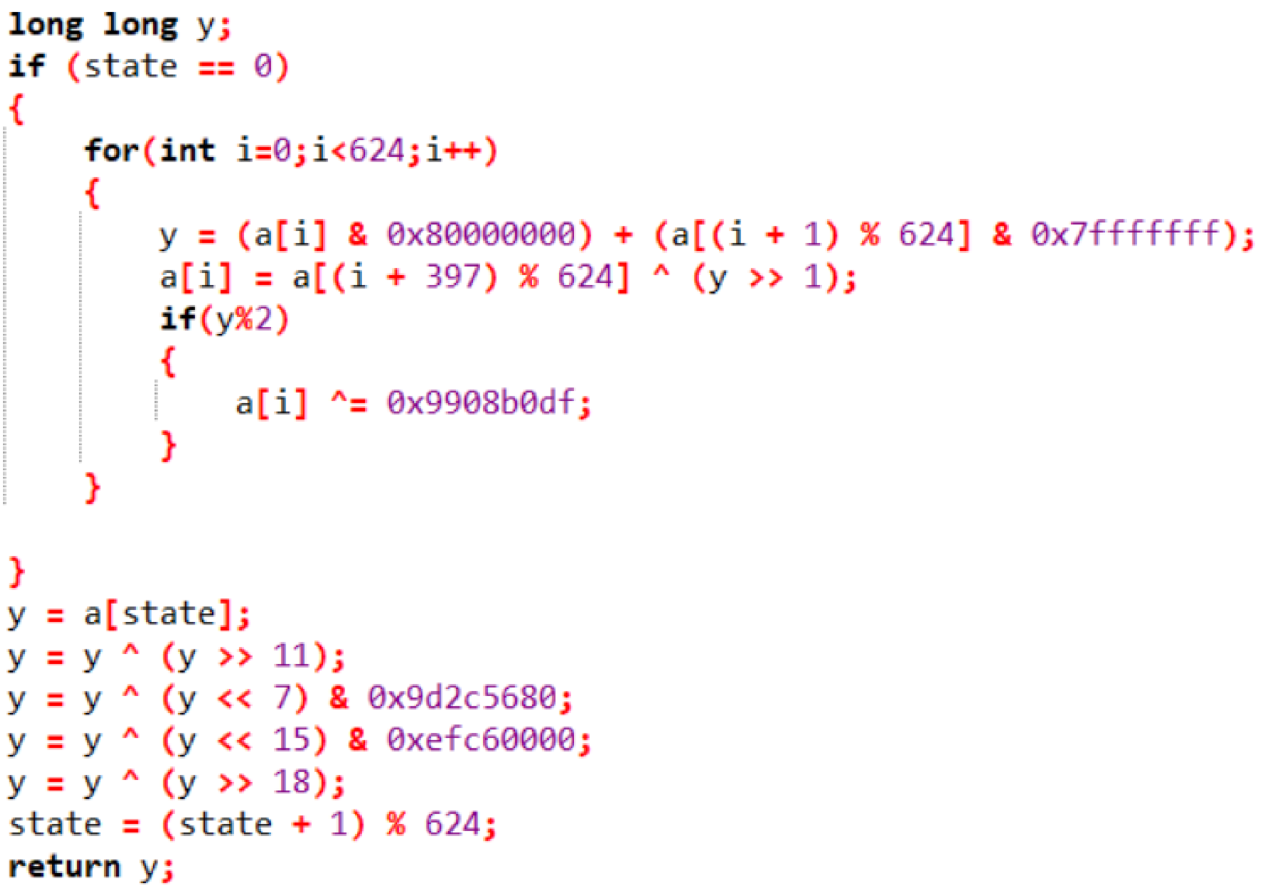
3.然后不是给了624个数吗，我就对于每个数，从0到（1<<32）都当成y暴力算了一遍，去找到624个数对应的初值，大概跑了3，4个小时



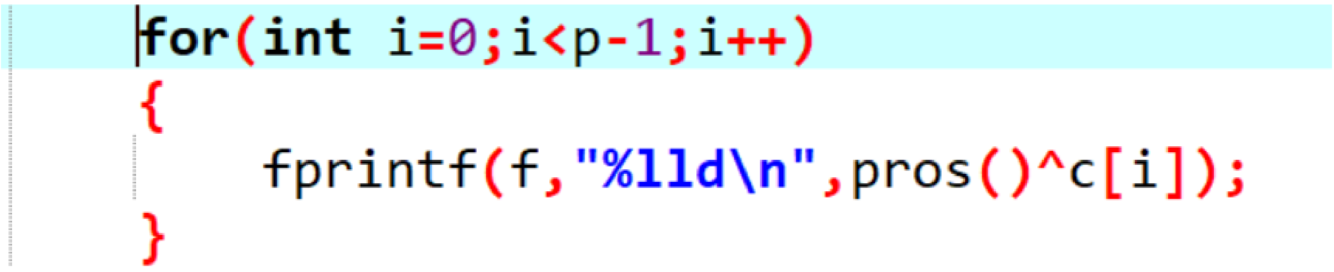
```
3355465923
3852884992
2233590063
3754373925
1047916917
3546382321
1087586699
1022876766
1022645414
3708191299
1800551279
```

4.此时624个数，是执行完print([mt() for _ in range(624)])后，mt里的数，且state=0

5.然后把PRNG的call函数用自己的话翻译一下，写成程序，对密文异或解密，10进制转16，再转ascii，完



```
long long y;
if (state == 0)
{
    for(int i=0;i<624;i++)
    {
        y = (a[i] & 0x80000000) + (a[(i + 1) % 624] & 0x7fffffff);
        a[i] = a[(i + 397) % 624] ^ (y >> 1);
        if(y%2)
        {
            a[i] ^= 0x9908b0df;
        }
    }

}
y = a[state];
y = y ^ (y >> 11);
y = y ^ (y << 7) & 0x9d2c5680;
y = y ^ (y << 15) & 0xefc60000;
y = y ^ (y >> 18);
state = (state + 1) % 624;
return y;
```



```
    for(int i=0;i<p-1;i++)
    {
        fprintf(f,"%lld\n",pros()^c[i]);
    }
```