**西安电子科技大学网信院**

**信息安全基础与密码学**

**综合实验**

**实 验 报 告（五）**

**SM2椭圆曲线公钥密码算法**

**班级：**

**姓名：**

**学号：**

**日期：2024-11-16**

一、实验目的（包括实验环境、实现目标等等）

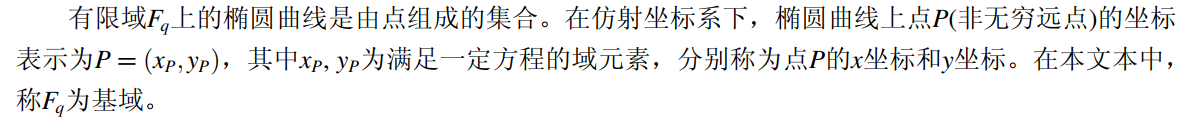
Python环境

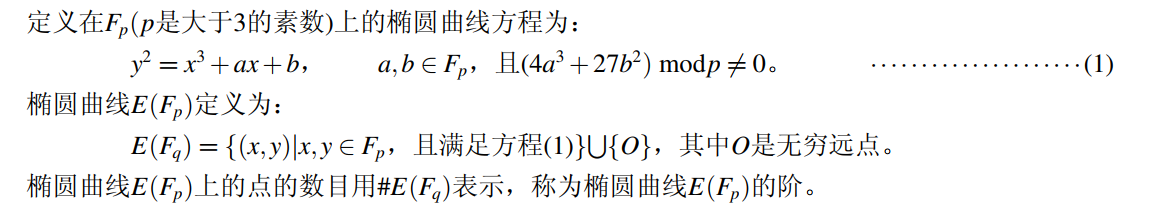
实现SM2椭圆曲线上加解密实现。

运行后屏幕显示出必要的椭圆曲线参数，以及明文、对应密文、解密后的明文、是否解密正确等。

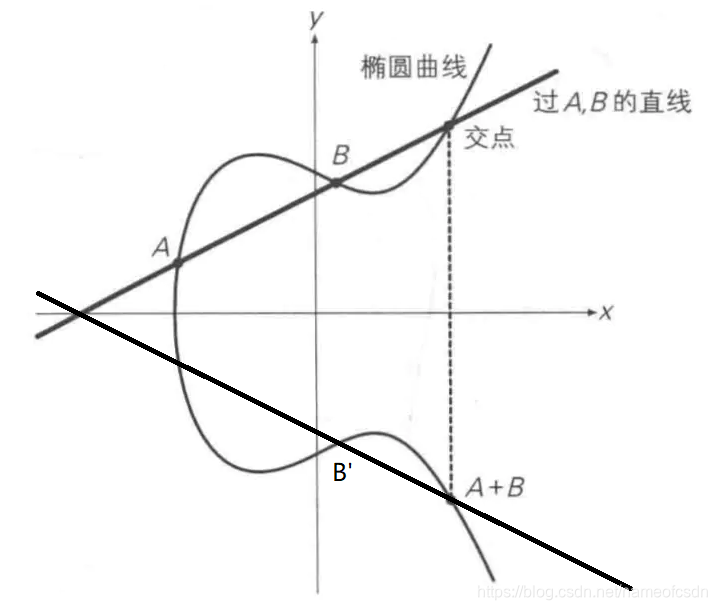
二、方案设计

（包括背景、原理、必要的公式、图表、算法步骤等等）

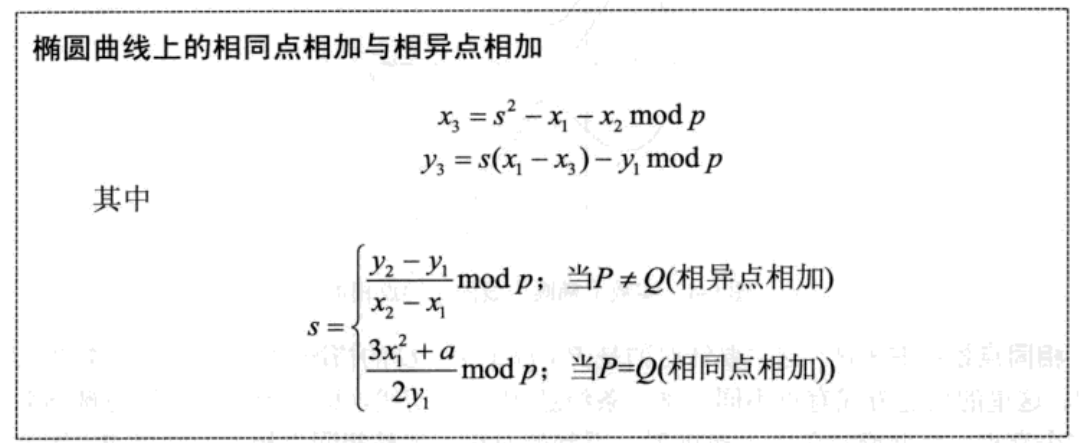




椭圆曲线上点的加法：几何意义如下



也可以直接把点坐标代入公式



有限域上的椭圆曲线上的所有点（包括一个无穷远点）构成一个加法群。

单位元是无穷远点

每个元素都有一个逆元

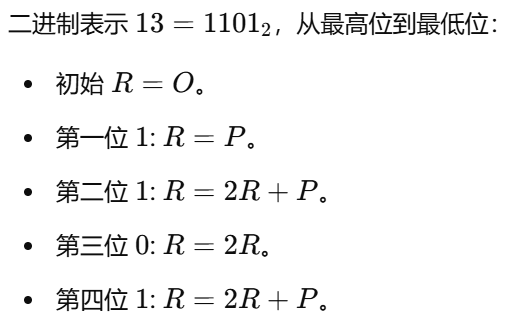
群运算规则为点的加法，定义依赖于椭圆曲线方程，并在该曲线的有限域上进行。

倍点运算：kP=P+P+⋯+P (重复 k 次)

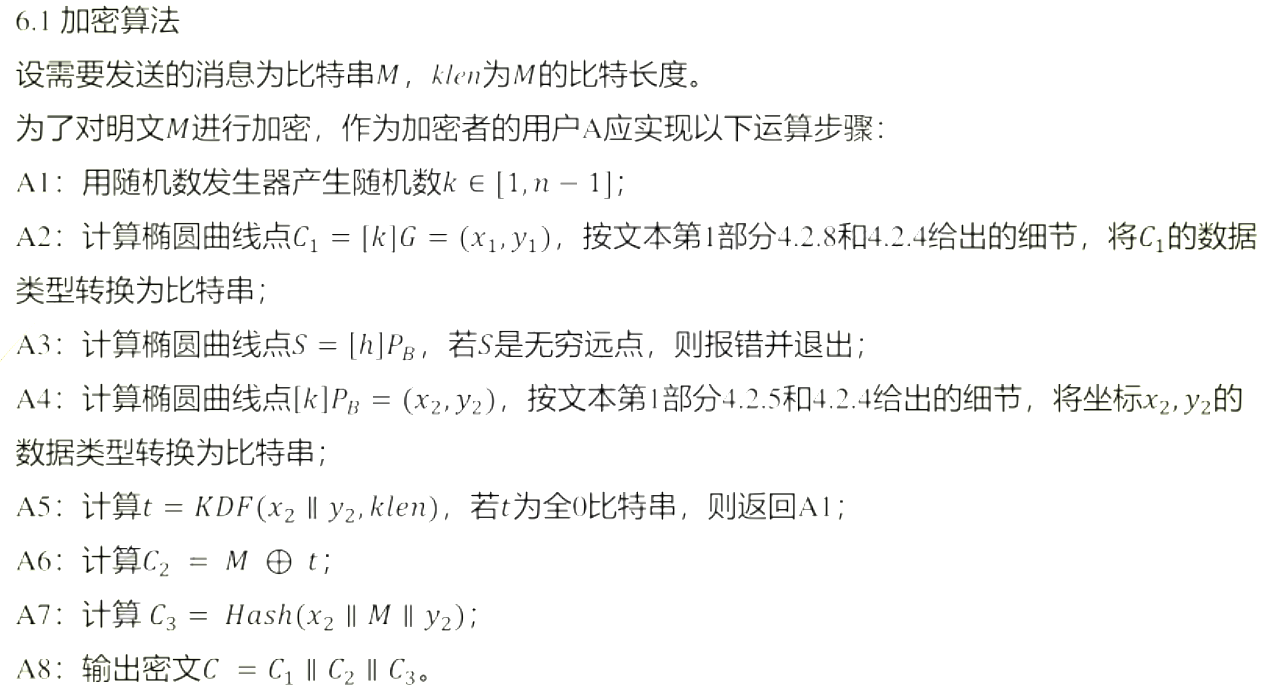
如果k不在域内，则通过 模运算 将其映射回域内：k mod n

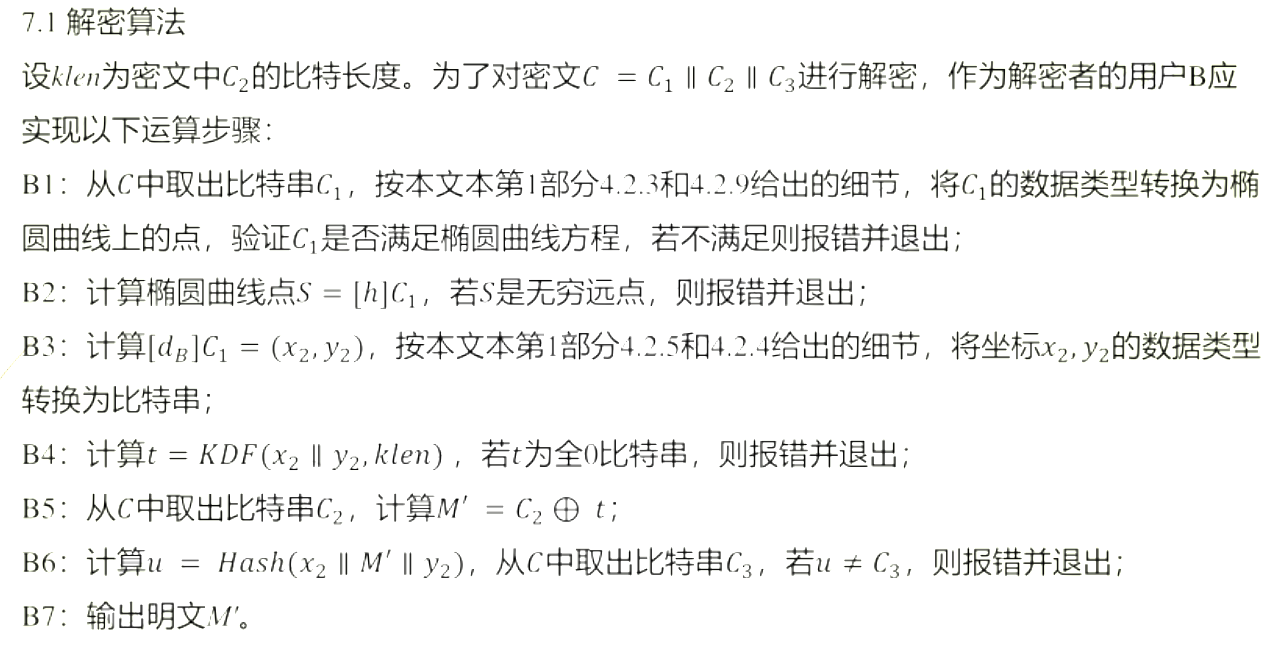
“倍加法”可以优化倍点运算。

示例：计算13P



SM2加解密算法：



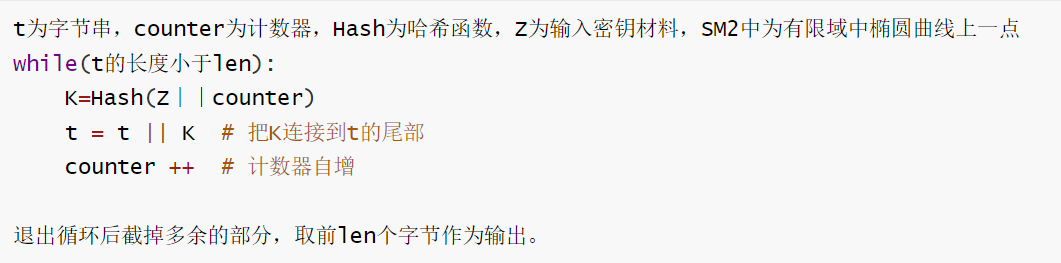


为了实现SM2算法，我们需要一个有限域上的一条椭圆曲线，和一个基点

算法中提到的h为余因子，定义为，其中为椭圆曲线在有限群上点的个数，包括无穷远点，为所用基点的阶

算法中提到的KDF为密钥派生函数，通过一个哈希函数，将输入密钥材料（共享秘密 Z）处理成输出密钥材料，从而生成随机性强且长度合适的密钥。

KDF依赖于一个哈希函数，算法流程大致如下：



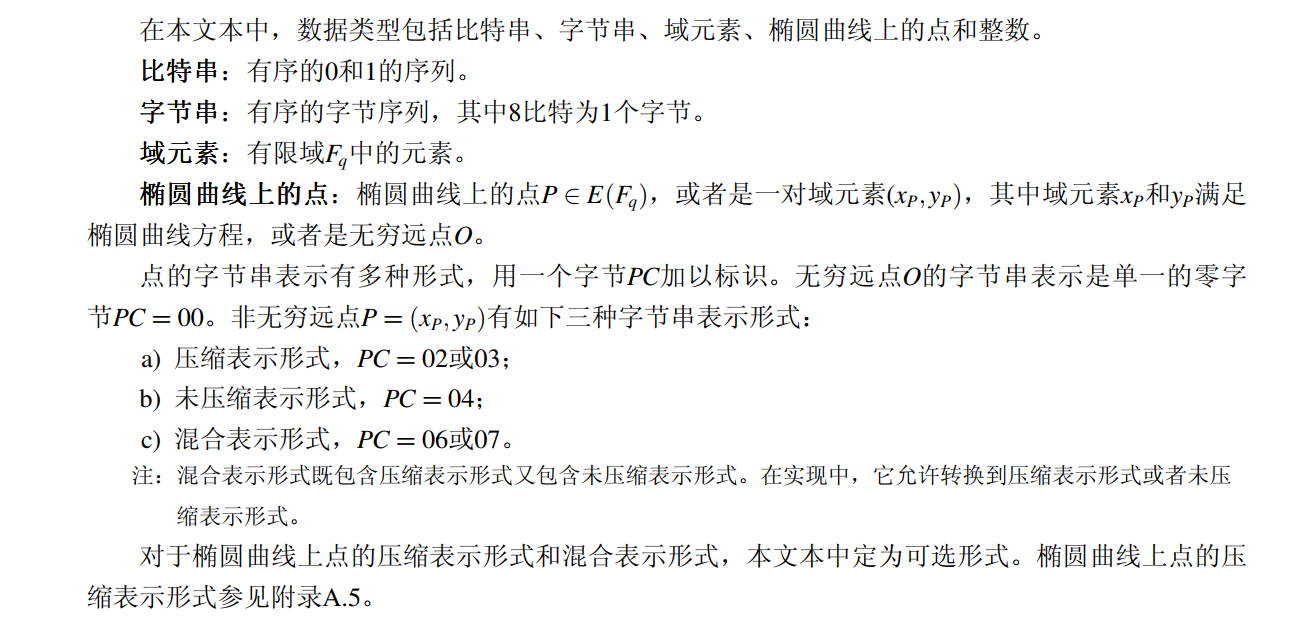
算法中提到的数据类型转换为比特串，本实验使用未压缩表示法：  
x, y 坐标都是域元素

将x，y坐标直接转换成固定长度的字节。长度取决于椭圆曲线参数 p 的大小。

在本实验中，x，y各自转换成32长度的字节。总长度为1+32+32=65字节

字节序列的格式：<0x04> <x 坐标的 32 字节> <y 坐标的 32 字节>

头部的0x04用于表示使用非压缩的方式。



三、方案实现

（包括算法流程图、主要函数的介绍、算法实现的主要代码等等）

在 utils.py中定义了两个工具函数：

xor\_bytes ： 用于两个字节流的按位异或

split\_bytes : 用于从密文中提取出三段密文。

在EllipticCurve.py中，定义了类EllipticCurvePoint(EllipticCurve)

这个类表示有限域椭圆曲线上的点，主要通过重载运算符，实现其功能.

主要方法有如下几个：

def \_\_eq\_\_(self, other)： 重载 == 运算符，比较两个点是不是同一个

def \_\_neg\_\_(self)： 重载 – 运算符，用于对点取反。

def \_\_add\_\_(self, other): 重载 + 运算符，根据数论，实现有限域椭圆曲线上两个点的相加

def \_\_mul\_\_(self, scalar)： 重载 \* 运算符，根据数论，通过二次倍加法，实现标量乘一个点的结果。

def to\_bytes(self): 把点转换为字节形式，由于模数p是256bits的，所以点坐标最多都是256bits，即32字节。这个函数将返回64字节的bytes，两半表示是x, y的32位字节表示

def from\_bytes(data: bytes): 上述函数的逆函数，从64字节的bytes中读取一个点。

SM2.py是本程序的顶层模块。

在其中定义了SM2Cipher类，用于实现SM2加解密

主要方法有如下几个：

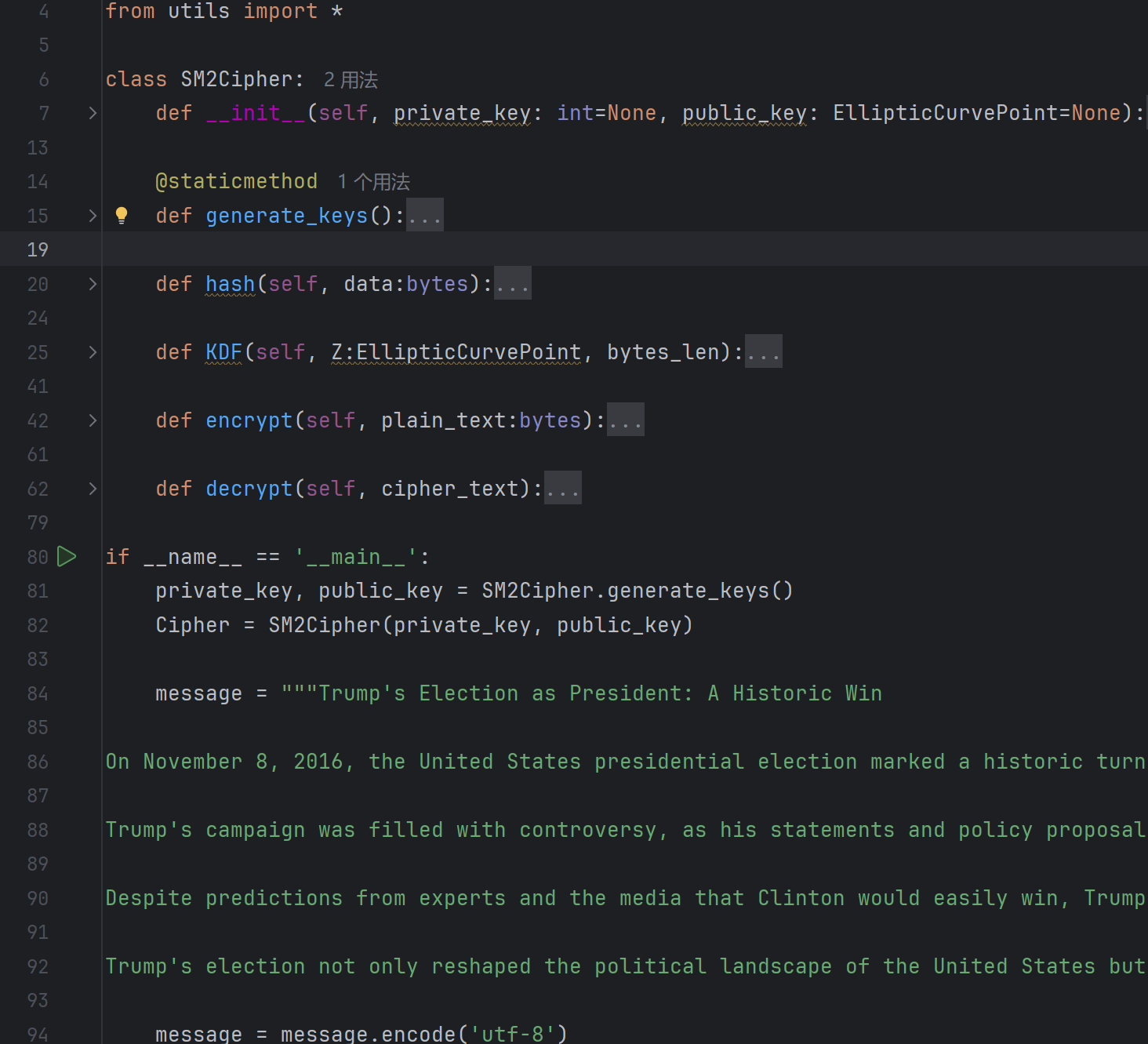
def generate\_keys()：生成密钥。返回私钥和公钥（一个点）

def hash(self, data:bytes): 本算法用的哈希函数，我采用的是SHA256

def KDF(self, Z:EllipticCurvePoint, bytes\_len)：KDF函数

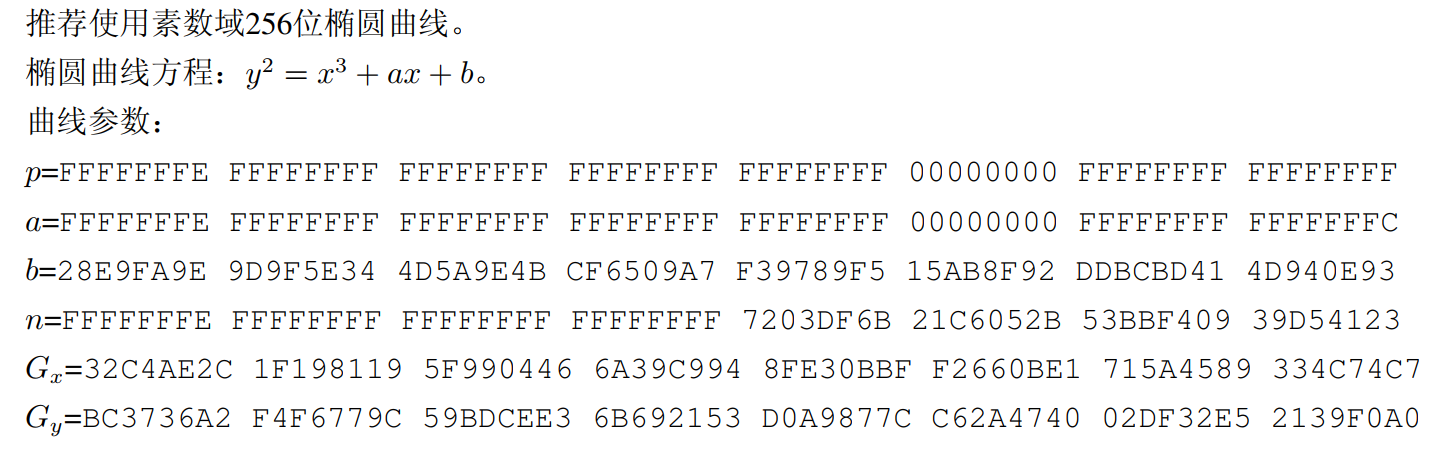
def encrypt(self, plain\_text:bytes): 加密

def decrypt(self, cipher\_text):解密

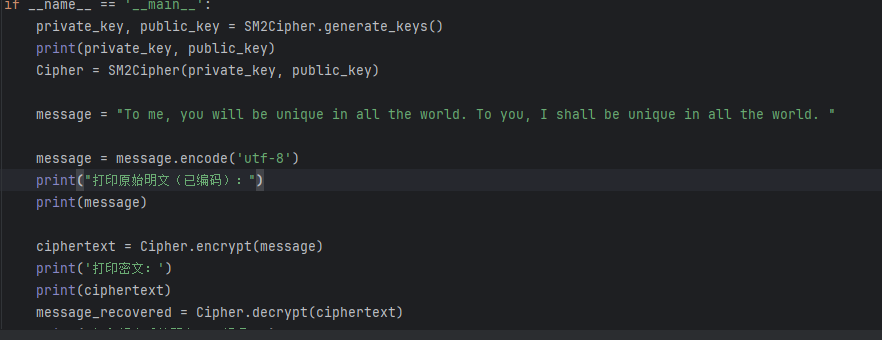


四、数据分析(包括算法测试数据的分析，运行结果截图等等)

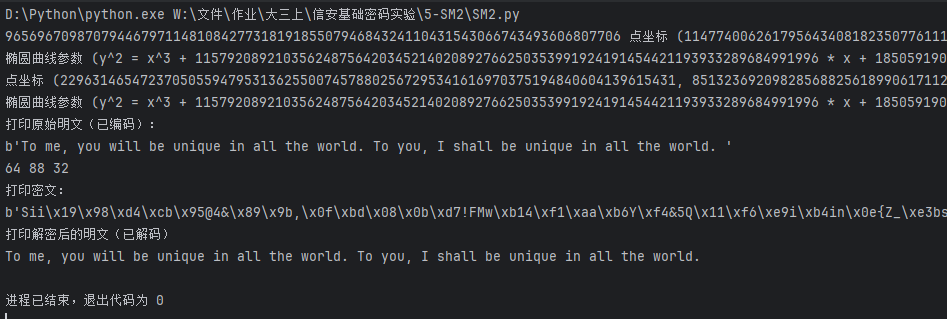
算法的参数使用如下数据：



程序中明文如下：



控制台打印的信息：



**生成的私钥为：**

96569670987079446797114810842773181918550794684324110431543066743493606807706

**生成的公钥为：**

点坐标 (114774006261795643408182350776111259653881178518569495101164143150238318542843, 59384830326766862515912112935723255354607127564983162681010741987190360594347)

椭圆曲线参数 (y^2 = x^3 + 115792089210356248756420345214020892766250353991924191454421193933289684991996 \* x + 18505919022281880113072981827955639221458448578012075254857346196103069175443)

**明文为：**To me, you will be unique in all the world. To you, I shall be unique in all the world.

**明文加密后的结果为：**

b'Sii\x19\x98\xd4\xcb\x95@4&\x89\x9b,\x0f\xbd\x08\x0b\xd7!FMw\xb14\xf1\xaa\xb6Y\xf4&5Q\x11\xf6\xe9i\xb4in\x0e{Z\_\xe3bsd\x10L\xdd\xb3\x88c\xf7x\xbdz!n&\xea\xa9\xa8\xc1\xd1\xaf\xd8P\x0c%\x89\xee\xb9\x86\xdeq\xf4\xfb\x00\x99\xda\xd1;\xf4\x1a\x94\x0c\xffe\xdc\x9eE\xb4gD\xf0M\xba\x83]\xf7\xd4A\xc5.\x87\n\x87\xd1\xd7\x8c\xc1\x9c%\xa5 \xa5k\x90\xda\xf9\xa36\xa2\x9a\xba\xe2<d\xfe\x1a\xa9\xb4\x9c\xc4\xad\xac\x01GF\xa2\x85\xc9\xa5\xfb\x82\x85\xceS\xc9\x1a\xf2v\x83!Bo\xdd\x12\x06\xb6I\x82\xa9J\xef\xffc\xf2s3\xafF\x97\x12\x99\xd1\t\xae\xd2\x86|\r'

**恢复出的明文：**

To me, you will be unique in all the world. To you, I shall be unique in all the world.

**恢复出的明文与原明文一致。**

五、思考与总结

1. 实验过程中遇到了什么问题，如何解决的？

在实验中，由于对密码学的基础理论和数论知识掌握不足，对于 SM2 算法的核心参数和算法（如 h 和 KDF）的作用和计算方法感到困惑。例如：

* 余因子 h 的数学意义和其对安全性的影响。
* 密钥派生函数 KDF 的实现流程及其与密钥生成的关系。

**解决方法：**

通过查阅密码学相关书籍和论文，深入理解了椭圆曲线的群结构、点加法计算原理以及 SM2 算法的具体步骤。同时，学习了以下内容：

* h 作为曲线总点数与子群阶之间的比例，通常等于 1，验证了其对曲线安全性的重要性。
* KDF 的作用是确保密钥派生的随机性和不可预测性，其实现依赖于哈希函数。

1. 通过该实验有何收获和感想？

通过本次实验，我深入理解了椭圆曲线密码学（ECC）的基础理论，特别是点加法、点倍乘等运算背后的数学原理。学习到了了 SM2 算法的完整流程，包括密钥生成、加密和解密的核心步骤。掌握了余因子 h 和密钥派生函数 KDF 的概念及其在算法中的具体作用，明确了它们对安全性的保障。

此外，我还深刻认识到密码学的核心在于数学的严谨性，算法设计需要兼顾效率与安全性。

**实验报告提交说明**：

1. 实验报告同时提交word文档与源代码（.c或者.py）。
2. 实验报告与源代码命名规则：实验5-学号-姓名，例如：实验5-20009200400-张三。
3. 请于截止日期前在西电智课平台（学在西电）提交相关文档，逾期未提交，该部分成绩记为0分，周知。