*Project11: impl sm2 with RFC6979

说明：

此项目是实现遵循 RFC6979 的 sm2。

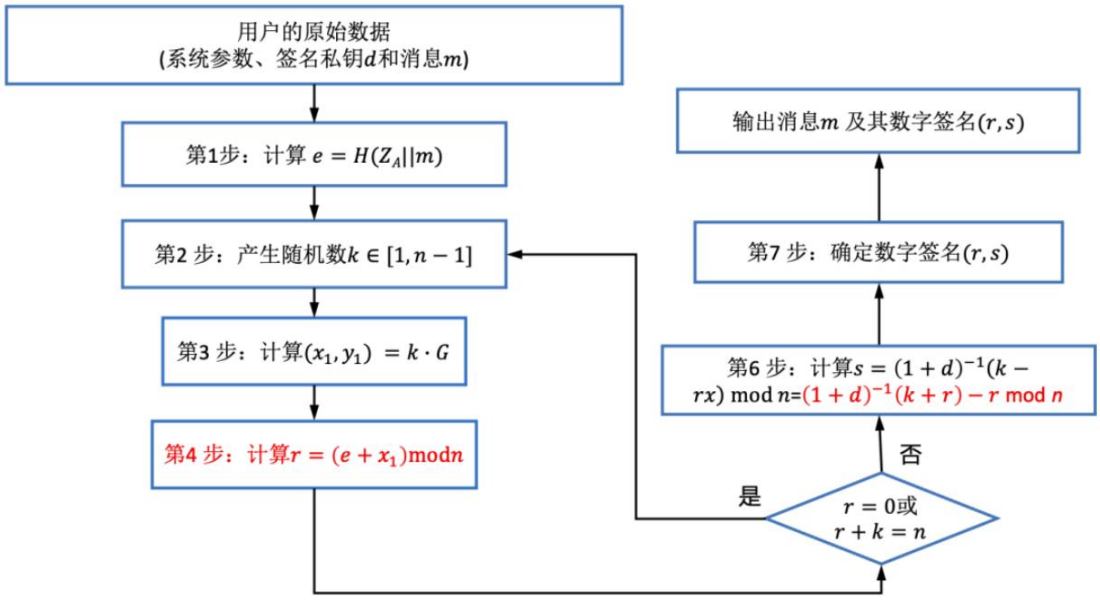
在椭圆曲线里面 k 值(用于签名)是要严格保密的。如果使用相同的值 k 在不同的消息（交易）上生成两个签名，那么任何人都可以计算出签名私钥。在签名算法中重用相同的 k 值的会导致私钥的暴露。k 值要保证两点：保密和唯一。

RFC6979 是一种用“确定性”方式来产生 k 值的方法，保证了“保密”且“唯一”。在本项目中我用以下公式来产生 k 值：

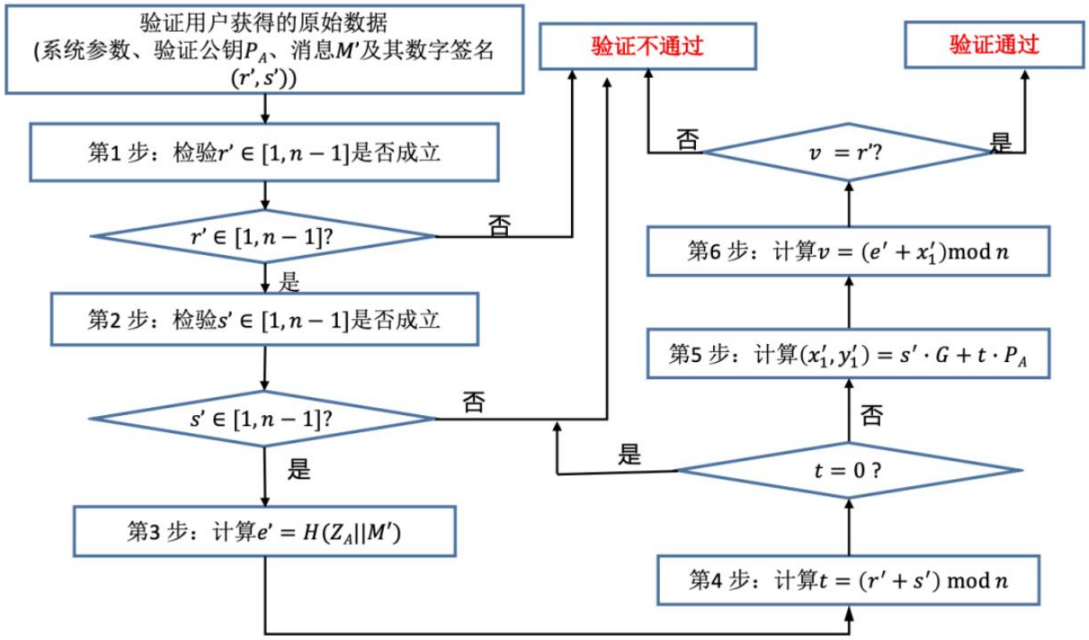
k = SHA256(d + HASH(m))

参数里有私钥 d，就保证了“保密”，再加上消息 m，保证了“唯一”，HASH 用 SM3。只要 SHA256 是安全的，此算法就是安全的。

SM2 数字签名算法签名过程示意图：



SM2 数字签名算法验证过程示意图：



在本项目中我在已封装的 sm2 签名与验签算法 python 实现代码的基础上，改变了 k 值获得的方法，使其符合 RFC6979。sm2 签名与验签算法 python 实现代码的原代码可从 https://blog.csdn.net/weixin_43261410/article/details/126991494 查看，对 k 做出的改变如下：

```
#为了遵循RFC6979改变的地方
sm3 = SM3()
k=SHA256(str(dA)+str(sm3.compression(msg)))
k=int(k,16)
#即k = SHA256(d + HASH(m))
```

实现方式：python

结果：在自己电脑上 CPU：11 代 i7

```
============================ RESTART: E:/project11.py ===============
开始签名：
r:6f27f5b7158cba82ba98d79790ea419c9a05c0c0ed7fc19db19d4d27f4ceda5c
s:59b1ebc9001cdfb7f3fa85a65fb4289effcba562490bc7c031779c102d995f7d
开始验签：
****验证通过****
```

分工：自己独立完成