Report

这个项目完成的生成一个签名值假装我是中本聪，但是由于一直找不到相应的签名值，因此在本项目中就自己生成一个：

在算法层面，采用的是由优化后的SM2类修改而成的ECDSA算法，采用了SM2类中的一些点乘，点加运算等，同时实现了ECDSA的签名和验签，ECDSA的具体原理过程如下图所示：

- Key Gen: $P = dG$, n is order
- Sign(m)
  - $k \leftarrow Z_n^*, R = kG$
  - $r = R_x \bmod n, r \neq 0$
  - $e = hash(m)$
  - $s = k^{-1}(e + dr) \bmod n$
  - Signature is (r, s)
- Verify (r, s) of m with P
  - $e = hash(m)$
  - $w = s^{-1} \bmod n$
  - $(r', s') = e \cdot wG + r \cdot wP$
  - Check if $r' == r$
  - Holds for correct sig since
  - $es^{-1}G + rs^{-1}P = s^{-1}(eG + rP) =$
  - $k(e + dr)^{-1}(e + dr)G = kG = R$

因此，要假装是别人，就要用不同的明文和签名值，顺利通过验签，伪造的原理如下图所示：

- $\sigma = (r, s)$ is valid signature of m with secret key d
- If only the hash of the signed message is required
- Then anyone can forge signature $\sigma' = (r', s')$ for d
- (Anyone can pretend to be someone else)
- Ecdsa verification is to verify:
- $s^{-1}(eG + rP) = (x', y') = R', r' = x' \bmod n == r$?
- To forge, choose $u, v \in \mathbb{F}_n^*$
- Compute $R' = (x', y') = uG + vP$
- Choose $r' = x' \bmod n$, to pass verification, we need
- $s'^{-1}(e'G + r'P) = uG + vP$
  - $s'^{-1}e' = u \bmod n \rightarrow e' = r'uv^{-1} \bmod n$
  - $s'^{-1}r' = v \bmod n \rightarrow s' = r'v^{-1} \bmod n$
- $\sigma' = (r', s')$ is a valid signature of e' with secret key d

*Project: forge a signature to pretend that you are Satoshi

可以看到最后的执行结果，伪造成功！e是明文的hash值，r和s是签名信息，两次签名的签名信息不同，e也不同，但是都能通过验签。

```
C:\Users\86180>set PYTHONIOENCODING=utf8 & C:\Users\86180\AppData\Local\Programs\Python\Python39\python.exe -u "c:\Users\86180\Desktop\ECDSA-forge sig\ecdsa-forge.py"
初始消息为： I am Satoshi

签名信息为  r: 71855728163943251481295809614414480145425162649231872503490949261369244459518 s: 53057402067833680260057517069482545032775308009500543510207927383795105432505 7 e: ef2cdaa37271e1bea8e95b2b9ec15209f84e5eb3583449b4b4b8e7f2a18d72b9
验签成功

签名信息为  r: 69384537925975776705201541604789655088277497755796146315033184469738311058448 s: 30589254302997446581601338245770366003917011135033408417808448777045091927085 e: ded2925884f8d906948321bc6d88abf2497539f8e21dcda5f3bc2bb9faba952
验签成功
```