实验报告

【实验目的】

以面向对象编程的方式搭建一个密码库，同时有一定的差错检测，兼容多种算法和数学运算。

【实验环境】

·python 3.10.9 64-bit

【实验内容】

一.密码库的搭建

1.基础数学运算库

（1）欧几里得算法、扩展的欧几里得算法

（2）整数求模逆运算

（3）快速模幂算法

（4）米勒拉宾素性检验算法

（5）大素数生成算法

（6）字节串转整数、整数转字节串算法

（7）十六进制串转二进制串、二进制串转十六进制串算法

（8）字节串转二进制串、二进制串转十六进制串算法

2.分组密码算法

采用SM4分组密码算法，含有CTR和OFB两种工作模式，可以对文件实现加解密操作，并具有对密钥长度、初始向量长度、输入类型等的差错检测。

3.公钥加密算法

采用SM2公钥加密算法，算法中包含的哈希函数采用SM3哈希算法，并具有对输入类型、输入数值等的差错检测。

4.数字签名算法

采用ElGamal数字签名算法，算法中包含的哈希函数采用SM3哈希算法，并具有对输入类型、输入数值等的差错检测。

5.哈希算法

采用SM3哈希算法，支持对文件进行哈希运算。

特别说明：以上所述的支持对文件进行操作首先需要将文件进行读入，以字节串的方式进行存储。

6.CLI交互设计

命令行输入的结构为**python CLI.py [-h] Operation source_path target_path**，其中Operation代表需要执行的操作，包含“BC_CTR/OFB_enc/dec”, “PK_enc/dec”, “Hash”,“Sign/Verify”几种，source_path代表需要执行操作的文件的路径，target_path表示执行操作后的文件内容的存储路径。

```
usage: CLI.py [-h] Operation source_path target_path

This is a cryptographic library that contains some common mathematical operations, SM4 block cipher algorithm, SM2
public key encryption algorithm, ElGamal digital signature algorithm, SM3 hash algorithm and some other algorithms.

positional arguments:
  Operation    the operation you want to perform on the file, including 'BC_CTR/OFB_enc/dec', 'PK_enc/dec',
               'Hash','Sign/Verify'
  source_path  the file_path which you want to operate
  target_path  the file_path which you want to output

options:
  -h, --help   show this help message and exit
```

二.差错检测机制

1.变量类型错误

该密码库对所有函数的每一个变量的类型都有明确的规定，因此一旦变量类型出错，程序会立马报错。

```
def SM4_OFB(self, n: int, IV: str, byte: bytes):
```

```
def encrypt(self, PBx:int, PBy:int, k: int, byte: bytes):
```

2.字符串长度错误

该密码库对部分字符串变量的长度有明确的规定，例如在SM4算法中，明确要求密钥key和初始向量IV的长度均为32，否则程序将会报错。

```
if len(key) != 32:
    raise ValueError("The length of the key must be 32!")
```

```
if len(IV) != 32:
    raise ValueError("The length of IV must be 32!")
```

3.数值不符合要求

该密码库对部分变量的数值有明确的规定，例如在整数求模逆运算中，要求整数和模数必须互素；在SM2算法中，需要保证椭圆曲线的参数p为素数，且基点和公钥点在椭圆曲线上，满足椭圆曲线方程；在ElGamal数字签名算法中，需要检测p为素数；在SM4算法中，规定初始化反馈缓存器每次左移的字节位数n需在1到16之间。

```
if not isPrime(p):
    raise ValueError("p is not a prime!")
```

```
if Gy**2 % p != (Gx**3 + a * Gx + b) % p:
    raise ValueError("G is not on the elliptic curve!")
```

```
if n > 16 or n < 1:
    raise ValueError("Invalid n!")
```

4.文件路径错误

即FileNotFoundError，对路径错误的文件进行报错。

5.命令行指令输入无效

例如，如果命令行输入的Operation在上述规定的所有可行的指令之外，将会提示"Error: Unknown operation"。

6.其余错误

例如OSError、Exception等等。

三.密码库正确性与可用性

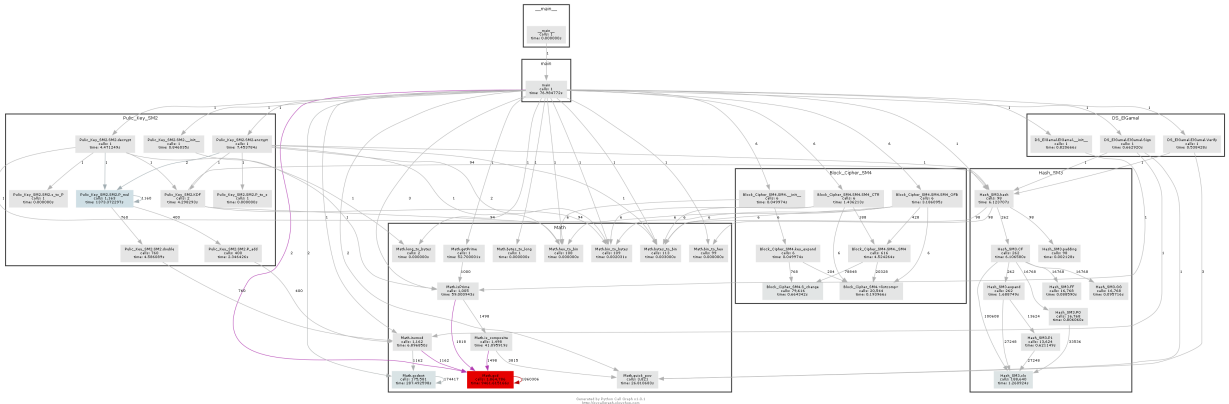
通过引入此前密码学实验课的部分样例数据，运行test.py和Collection.py文件，将生成结果与样例进行一一比对，得到如下结果：

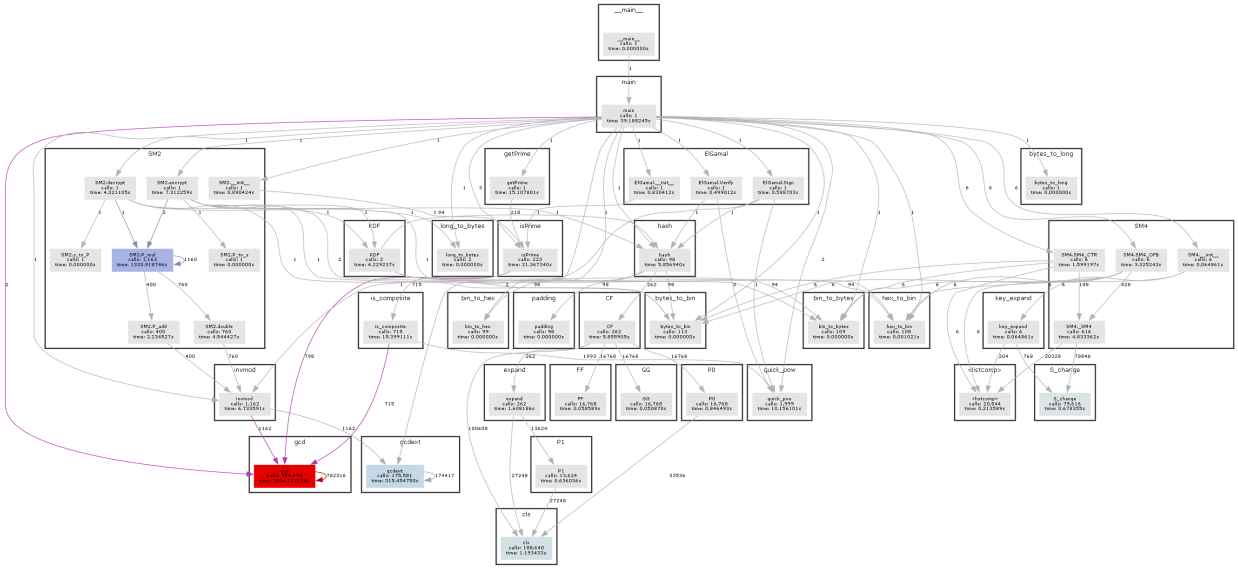
```
--------------------Part I: Math--------------------
gcd is OK!
gcdext is OK!
invmod is OK!
quick_pow is OK!
isPrime is OK!
getPrime is OK!
bytes_to_long is OK!
long_to_bytes is OK!
hex_to_bin is OK!
bin_to_hex is OK!
bytes_to_bin is OK!
bin_to_bytes is OK!
--------------------Part I is OK!--------------------

--------------------Part II: Block_Cipher_SM4--------------------
SM4_CTR is OK!
SM4_OFB is OK!
--------------------Part II is OK!--------------------

--------------------Part III: Public_Key_SM2--------------------
--------------------Part III is OK!--------------------

--------------------Part IV: DS_ElGamal--------------------
--------------------Part IV is OK!--------------------
...
--------------------Part V: Hash_SM3--------------------
--------------------Part V is OK!--------------------

The password vault has been verified and can be put into use!
```

可见，所有结果均执行正确。

四.密码库函数全集

以下两图分别为运行test.py和Collection.py文件时生成，包含了该密码库中所有函数的调用关系。




__main__
main
SM2
SM2.decrypt
SM2.encrypt
SM2.__init__
SM2.P
SM2.mul
SM2.N
SM2.add
SM2.double
getPrime
ElGamal
ElGamal.__init__
ElGamal.Verify
ElGamal.Sign
bytes_to_long
KDF
long_to_bytes
isPrime
hash
SM4
SM4.SM4_CTR
SM4.SM4_CFB
SM4.__init__
is_composite
bin_to_hex
padding
CF
bytes_to_bin
bin_to_bytes
hex_to_bin
key_expand
SM4._SM4
invmod
expand
FF
GG
P0
quick_pow
distcomp
S_change
gcd
gcdext
P1
cls