Bài tập lớn môn Nhập môn An toàn thông tin

Hoàng Đức Minh - 18020890

## 1. Hệ mật RSA với p,q - 512 bit

### 1.1. Quy trình

#### 1.1.1. A chuẩn bị và gửi B khóa công khai

* **Bước 1**: Chọn 2 số nguyên tố p, q có độ lớn 512 bit bằng hàm generate\_prime\_number(512) thu được
  + p=9607148119662215966953860291134935552021529970948227821414276310259699224328420089251860001734469581369286662661929489456209928679379851155917574132015383
  + q=7942814997448628785890615271902035560839675652138142370694028632742919644785493862232366756328919273314953243481427167839097110118579835974997476662633073
  + Từ đó, tính được n = pq và Φ(n) = (p-1)(q-1)
* **Bước 2**: Chọn giá trị b (1<b<Φ(n)) sao cho gcd(b,Φ(n)) = 1. Để đơn giản, chọn b là số nguyên tố bằng hàm generate\_prime\_number(10) thu được: b=673
* **Bước 3**: Tính được a = b-1 mod Φ(n) bằng hàm modinv(b, Φ(n)). Khi đó ta thu được, khóa công khai (n, b) và khóa bí mật (p, q, a)

#### 1.1.2. B mã hóa dựa vào khóa công khai rồi gửi cho A bản mã

* **Bước 4**: Yêu cầu B nhập nội dung cần mã hóa và thực hiện mã hóa bằng hàm rsa\_encrypt(x, n, b) rồi gửi kết quả này cho A

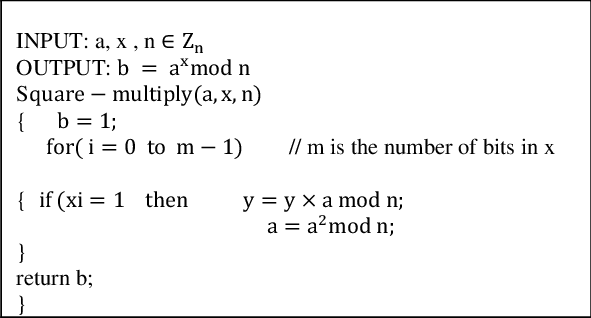
#### 1.1.3. A giải mã thu được bản rõ

* **Bước 5**: Giải mã bằng hàm rsa\_decrypt(encrypt, a, n)

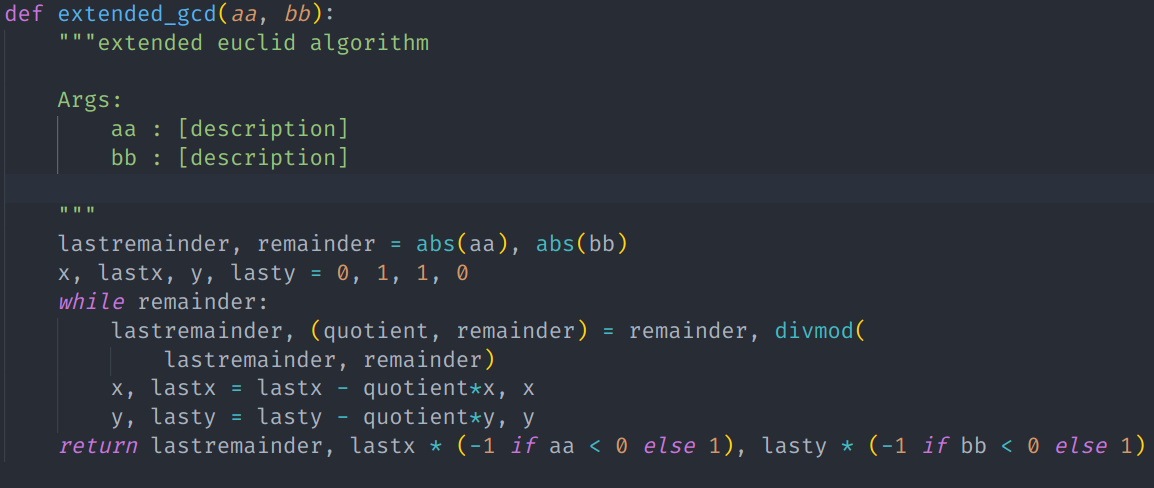
### 1.2. Thiết kế thuật toán:

#### 1.2.1. Thuật toán tính nghịch đảo modulo

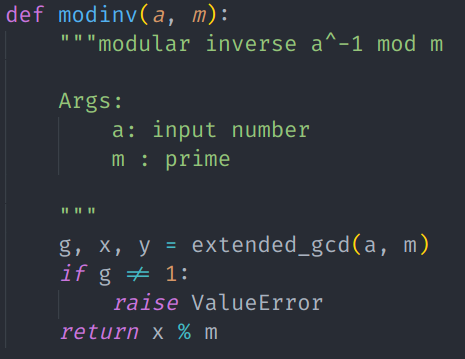
Thuật toán tính nghịch đảo modulo được xây dựng dựa trên thuật toán euclid mở rộng. Với phương trình a\*x+m\*y=1, từ thuật toán euclid mở rộng ta có thể xác định được x và y. Trong toán học modulo với mod m, ta có thể bỏ được m\*y đi, từ đó x chính là a-1



*Thuật toán euclid mở rộng*

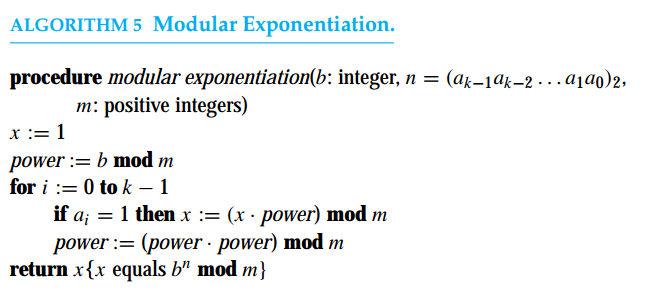


*Cài đặt thuật toán euclid mở rộng*

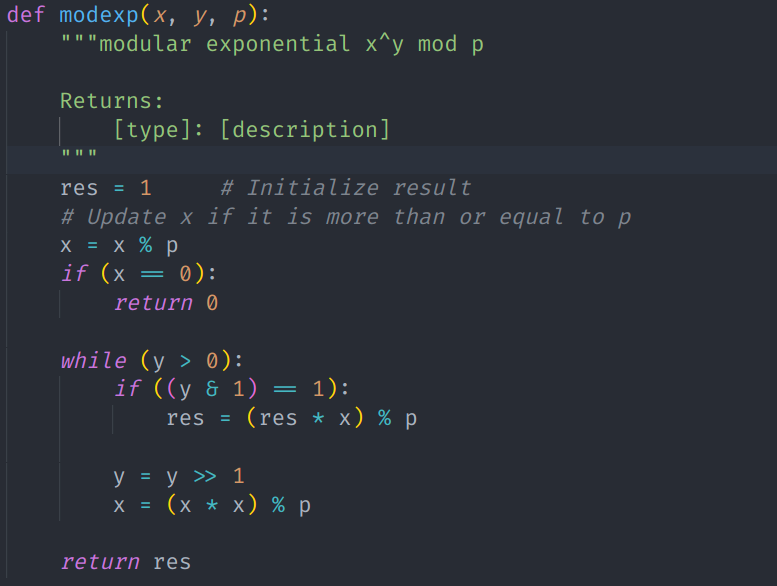


*Thuật toán nghịch đảo modulo*

#### 1.2.2. Thuật toán lũy thừa modulo



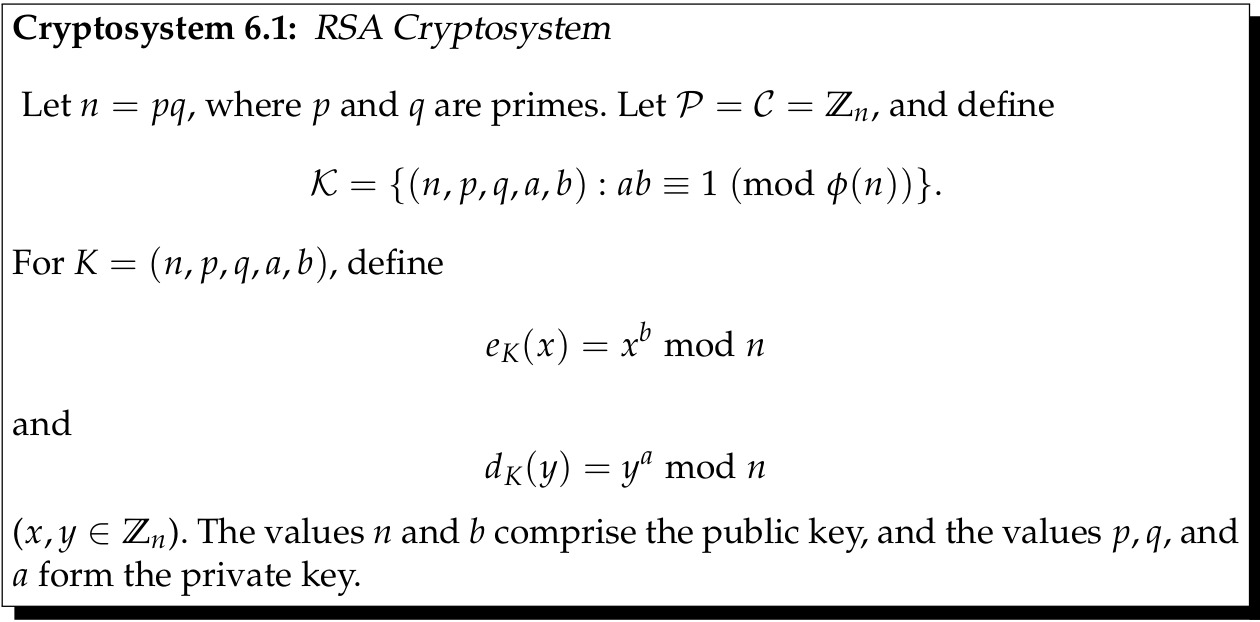
*Thuật toán tính lũy thừa modulo*

**

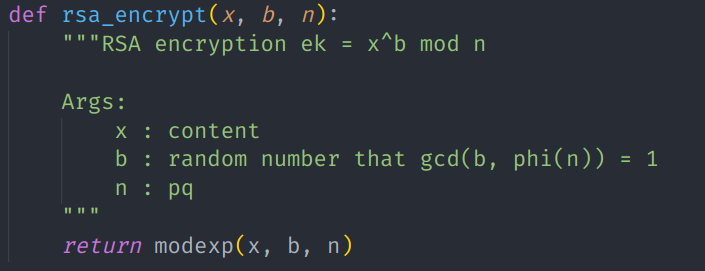
*Cài đạt thuật toán tính lũy thừa modulo*

#### 1.2.3. Cài đặt hệ mật RSA

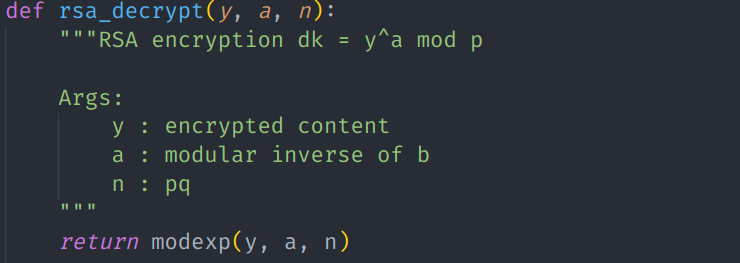
RSA gồm hai phần mã hóa và giải mã sử dụng thuật toán tính lũy thừa modulo



*Mã hóa và giải mã hệ mật RSA*

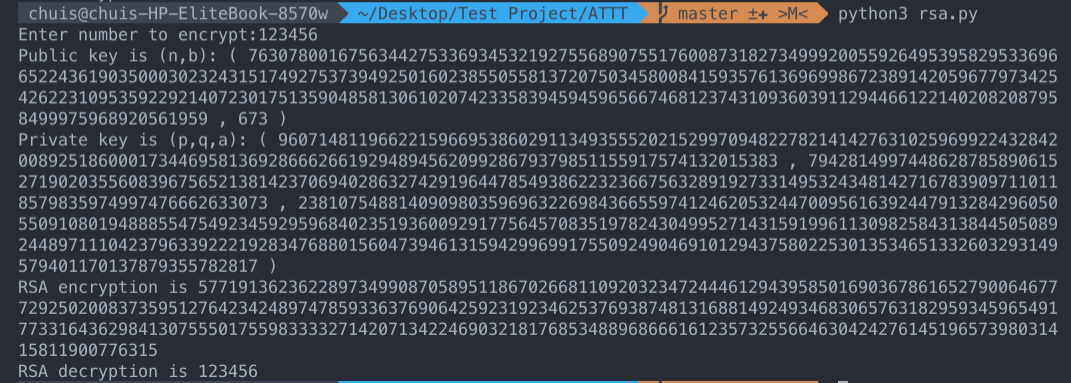


*Cài đặt thuật toán mã hóa RSA*

**

*Cài đặt thuật toán giải mã RSA*

### 1.3. Kết quả:

****

## 

## **2. Hệ mật ElGama với p - 256 bit**

### 2.1. Quy trình

#### 2.1.1. A chuẩn bị và gửi B khóa công khai

* **Bước 1**: Chọn số nguyên tố p có độ lớn 256 bit bằng hàm generate\_prime\_number(256) thu được p=101772665954685200507098931443389343082543279947190588868406755488643105839567
* **Bước 2**: Tìm phần tử nguyên thủy α bằng hàm primitive(p) thu được α = 5
* **Bước 3**: A nhập khóa bí mật a. Từ đó tính được β = αa (mod p). Khi đó ta thu được khóa công khai (p, α, β) và khóa mật (a)

#### 2.1.2. B mã hóa dựa vào khóa công khai rồi gửi cho A bản mã

* **Bước 4:** Sinh tự động số k bất kì (bí mật) bằng hàm k = randrange(0, 10000000, 1)
* **Bước 5**: Yêu cầu B nhập nội dung cần mã hóa và thực hiện mã hóa bằng hàm elgama\_encrypt(x, α, β, p) thu được (y1, y2) rồi gửi cho A

#### 2.1.3. A giải mã thu được bản rõ

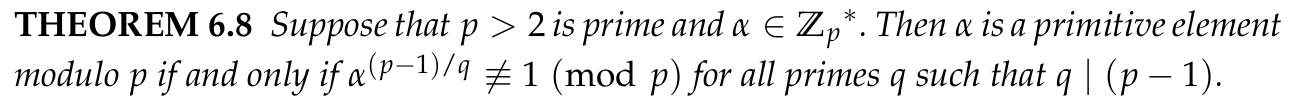
* **Bước 5**: Giải mã bằng hàm elgama\_decrypt(y1, y2, a, p)

### 2.2. Thiết kế thuật toán:

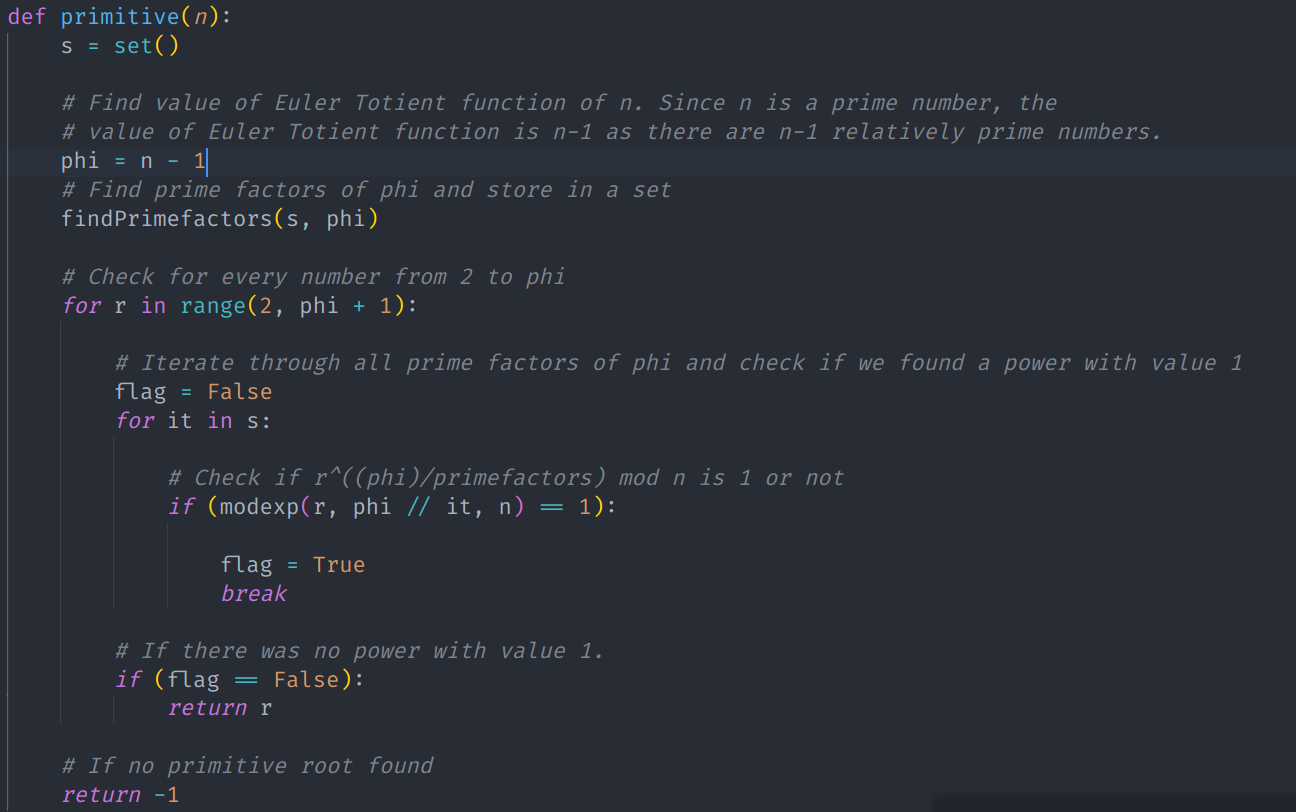
#### 2.2.1. Thuật toán tính lũy thừa modulo

Đã trình bày ở Hệ mật RSA

#### 2.2.2. Thuật toán xác định phần tử nguyên thủy modulo n

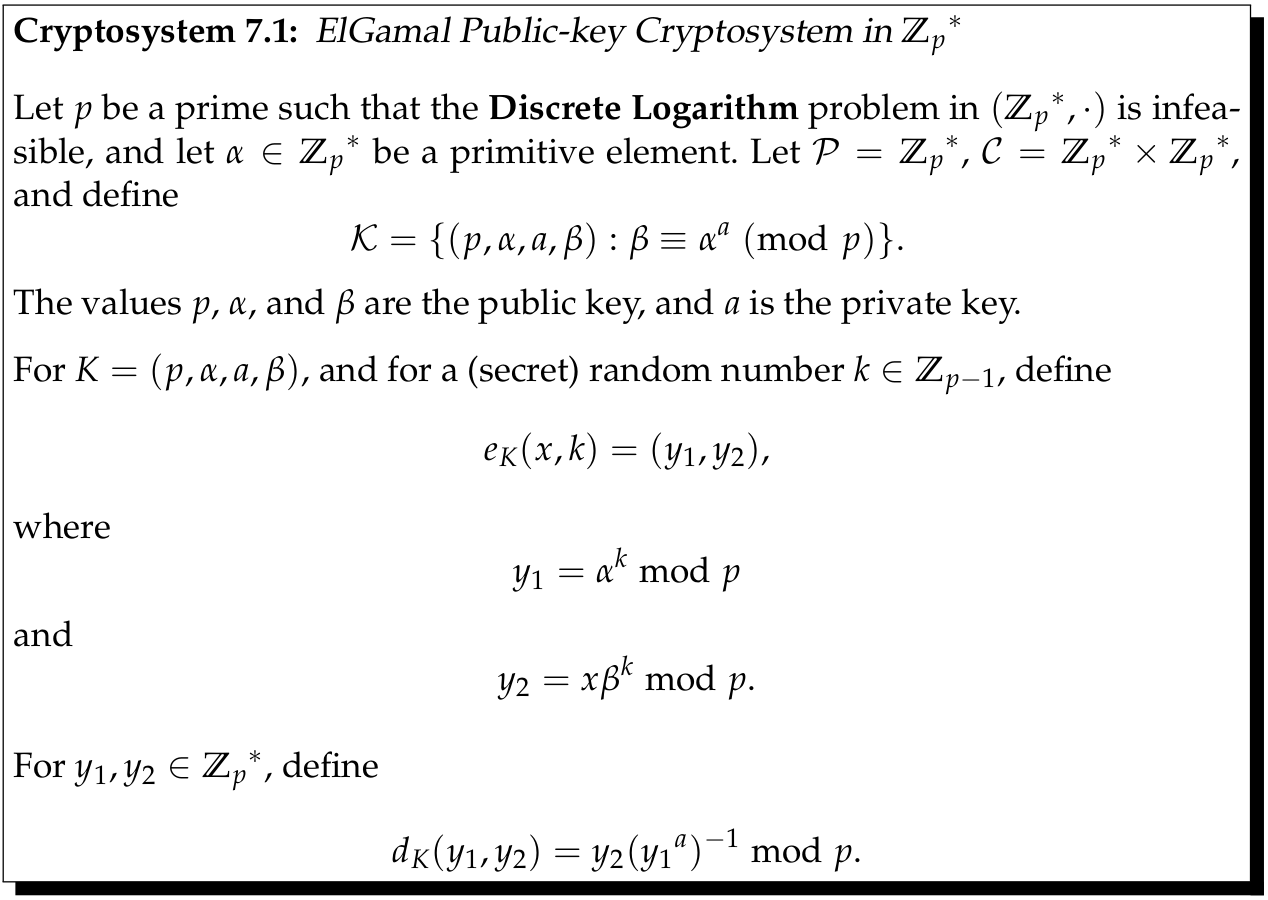


*Thuật toán xác định phần tử nguyên thủy modulo*

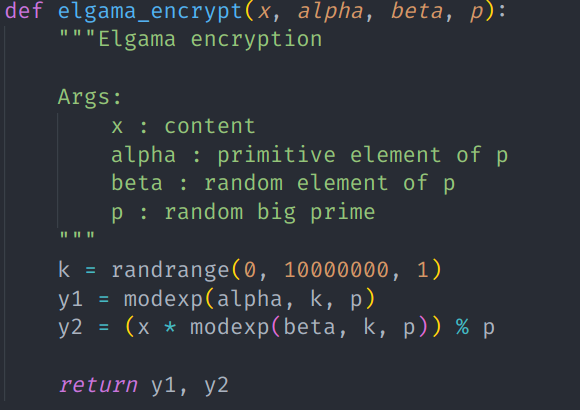
**

*Cài đặt thuật toán xác định phần tử nguyên thủy modulo*

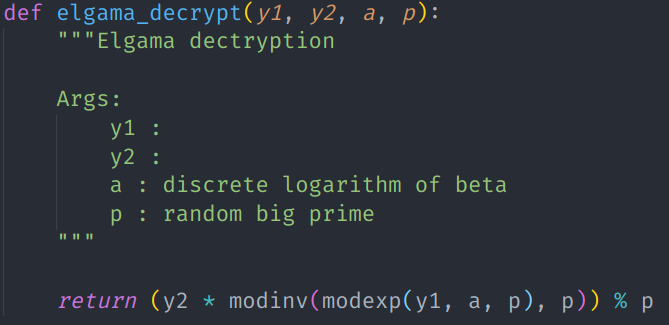
#### 2.2.3. Cài đặt hệ mật ElGama

ElGama gồm hai phần mã hóa và giải mã 

*Hệ mật ElGama*

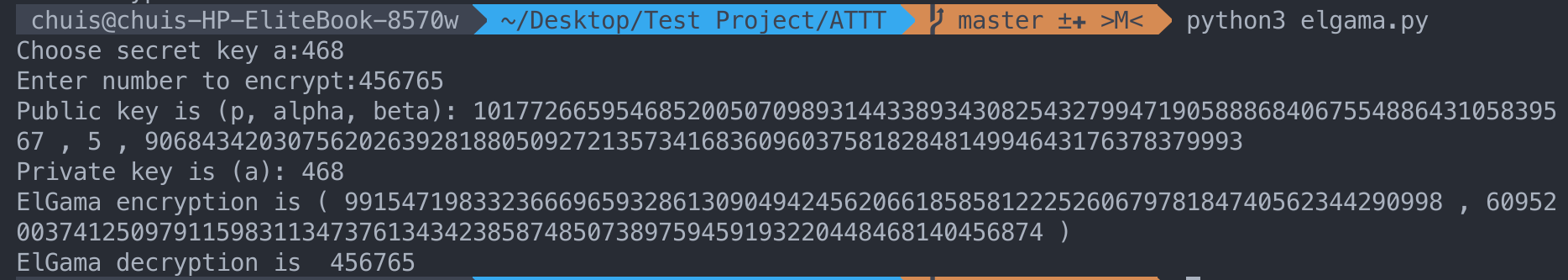
**

*Mã hóa ElGama*

**

*Giải mã ElGama*

### 1.3. Kết quả:

****

## **3. Mã hoá trên đường cong Elliptic với p = 160 bit**

### 3.1. Quy trình

#### 3.1.1. A chuẩn bị và gửi B khóa công khai

* **Bước 1**: Chọn a=53, b=7 và p=1114597119506223026265579259036275469126397408411. Việc chọn a, b, p thông qua việc chọn p là số nguyên tố 160 bit qua hàm generate\_prime\_number(160), sau đó random a, b cho đến khi bậc của đường cong Elliptic là số nguyên tố. Bậc được tính bằng thuật toán schoof, điều này nhằm đảm bảo tất cả các điểm đều là điểm sinh.
* **Bước 2**: Chọn điểm đầu tiên P của đường Elliptic làm điểm sinh qua hàm findFirstPoint(a, b, p)
* **Bước 3**: A nhập khóa bí mật s. Từ đó tính được B=sP. Khi đó ta thu được khóa công khai (E, p, P, B) và khóa mật (s)

#### 3.1.2. B mã hóa dựa vào khóa công khai rồi gửi cho A bản mã

* **Bước 4:** Yêu cầu B nhập điểm M cần mã hóa và thực hiện mã hóa bằng hàm ecc\_encrypt(M, a, b, p, P, B) thu được (M1, M2) rồi gửi cho A

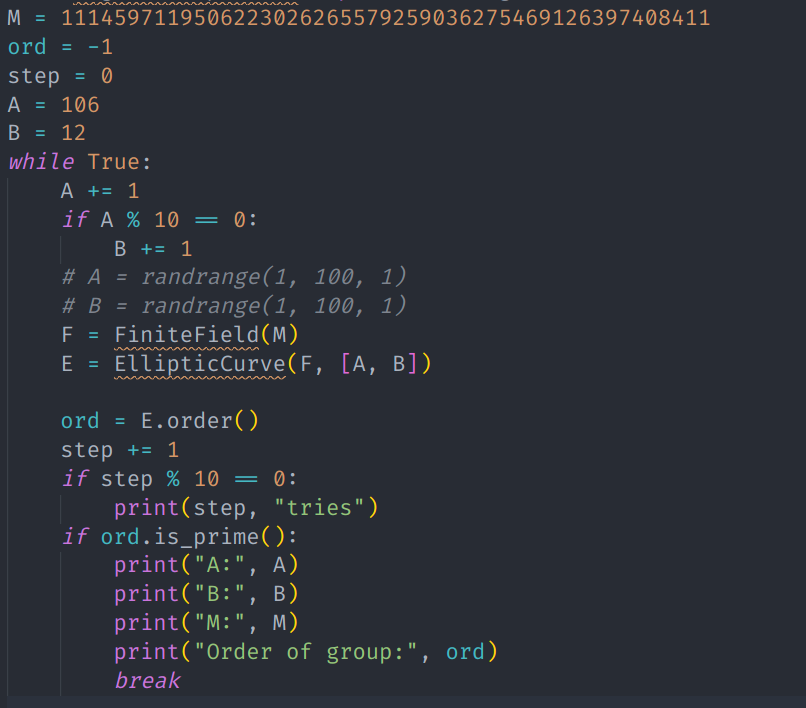
#### 3.1.3. A giải mã thu được bản rõ

* **Bước 5**: Giải mã bằng hàm ecc\_decrypt(M1, M2, s, p, a)

### 3.2. Thiết kế thuật toán:

#### 3.2.1. Thuật toán chọn đường Elliptic thỏa mãn

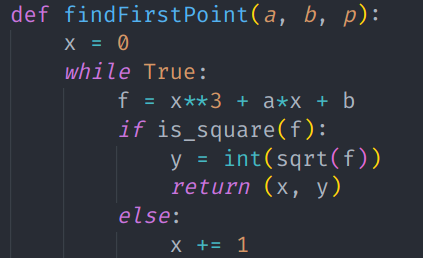
Với p đã chọn, thực hiện random a và b để tìm được đường Elliptic có bậc là số nguyên tố, từ đó tất cả các phần tử đều là phần tử sinh.



*Thuật toán xác định đường cong Elliptic*

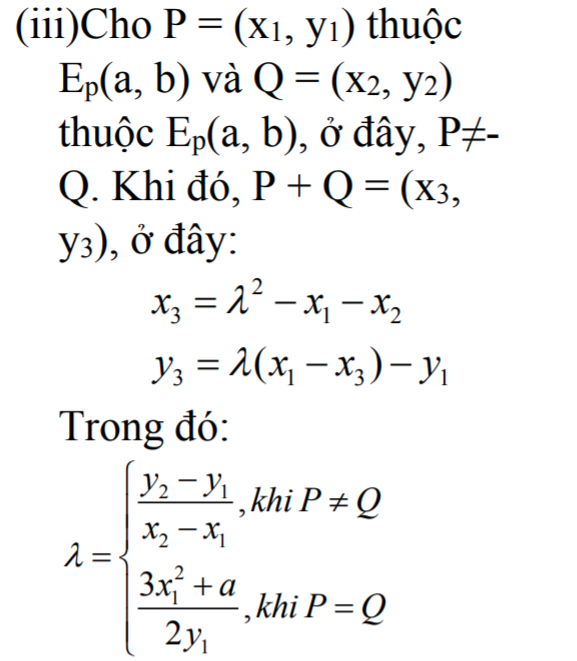
#### 3.2.1. Thuật toán tìm điểm thuộc đường cong Elliptic

Việc xác định điểm đầu tiên thuộc đường cong Elliptic được thực hiện bằng cách chạy x rồi tính y cho đến khi y có giá trị là phần tử thặng dư bậc 2 của p

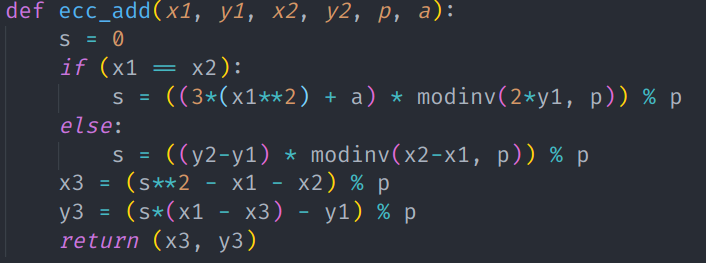


*Thuật toán tìm điểm thuộc đường cong Elliptic*

#### 3.2.2. Thuật toán cộng trên đường cong Elliptic

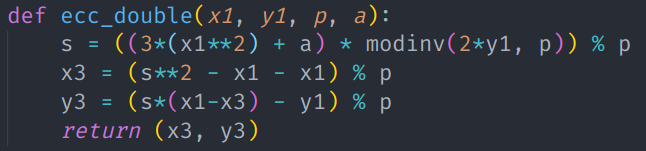


*Thuật toán cộng trên đường cong Elliptic*

**

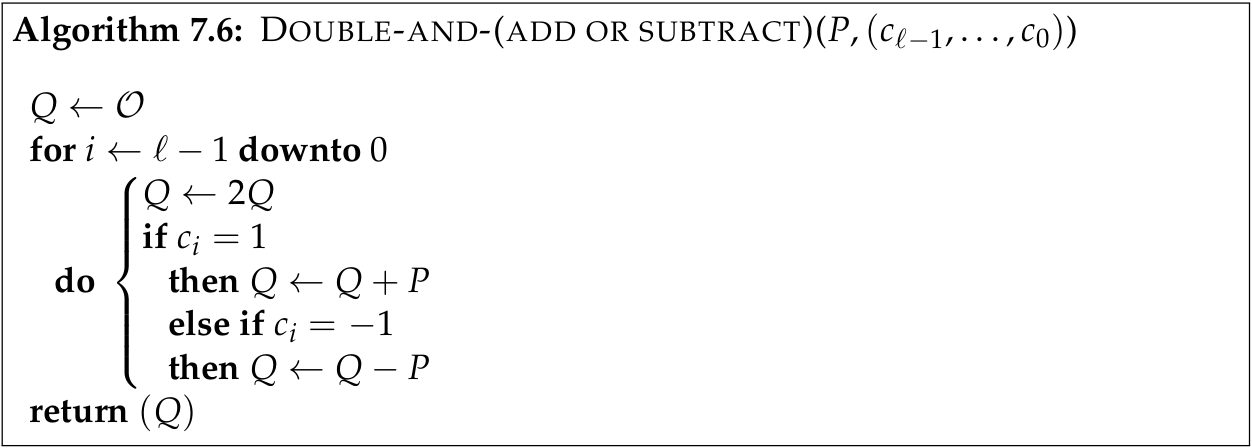
*Cài đặt thuật toán cộng trên đường cong Elliptic*

#### 3.2.3. Thuật toán nhân đôi trên đường cong Elliptic

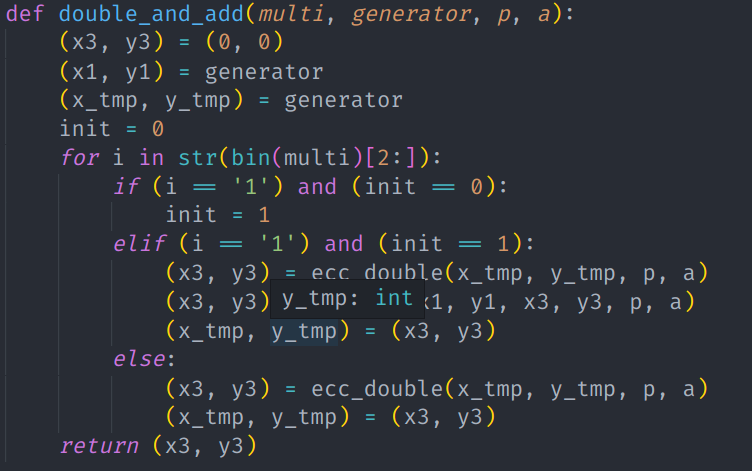


*Cài đặt thuật toán cộng trên đường cong Elliptic*

#### 3.2.4. Thuật toán nhân và cộng trên đường cong Elliptic

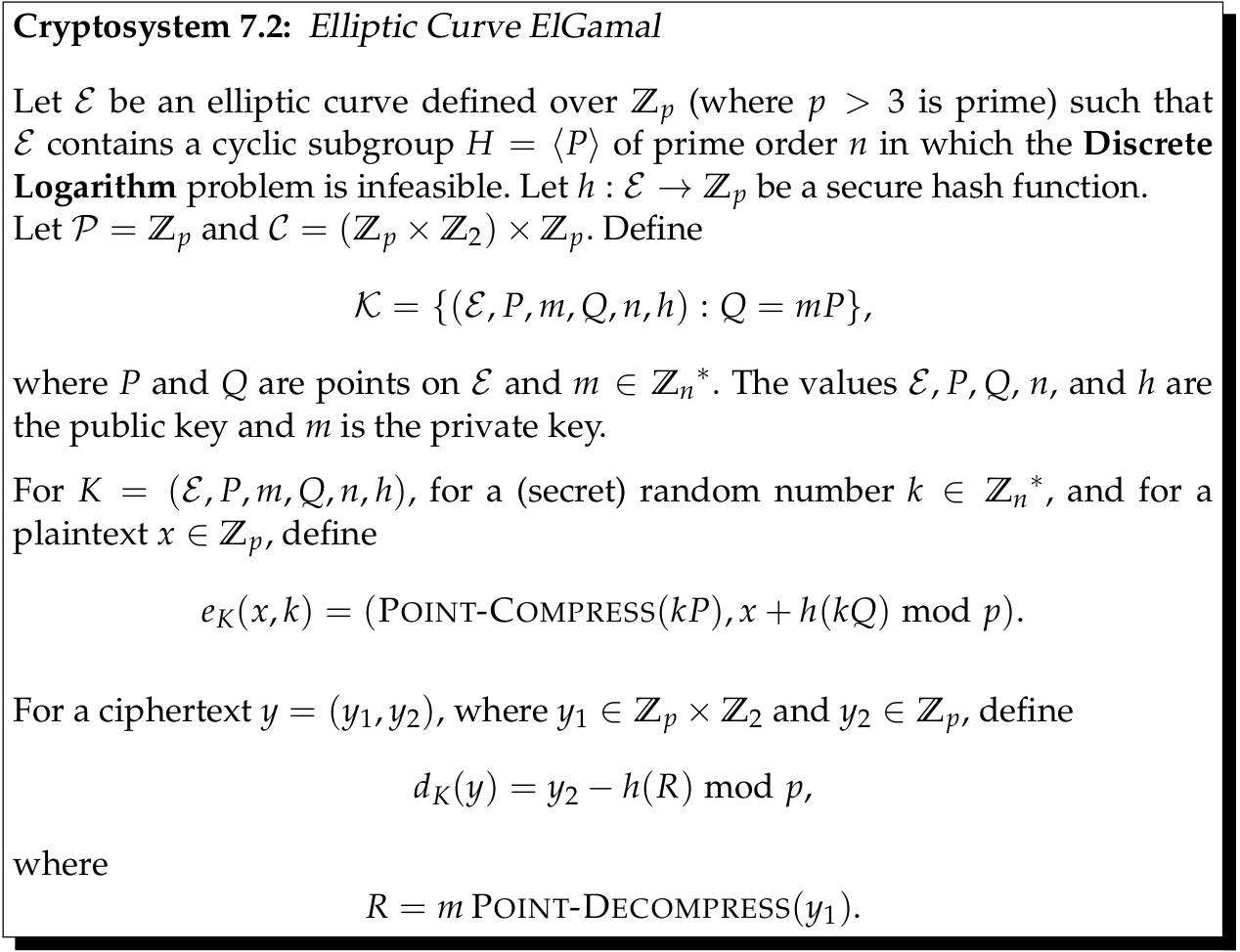


*Thuật toán nhân và cộng trên đường công Elliptic*

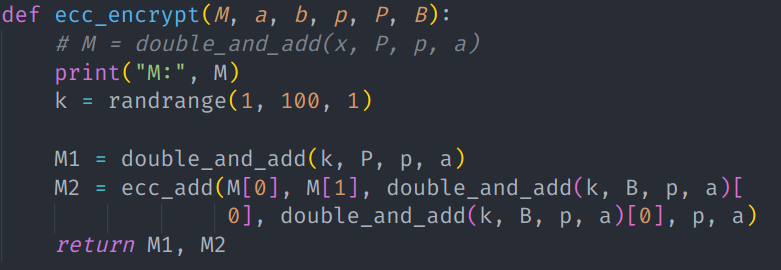
**

*Cài đặt thuật toán nhân và cộng trên đường công Elliptic*

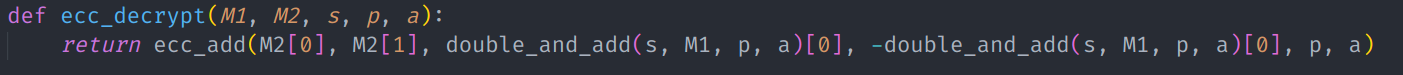
#### 3.2.5. Cài đặt hệ mật ElGama Elliptic Curve



*Hệ mật EEC*

**

*Mã hóa hệ mật EEC*

**

*Giải mã hệ mật EEC*

### 3.3. Kết quả:

