Họ và tên: Phùng Đức Dũng

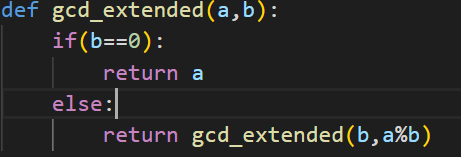
Báo cáo thực hành Cryptohack

Bài 1: Extended GCD

**Giải thuật Euclid**: là giải thuật đi tìm ước chung lớn nhất của 2 số a,b nguyên. Tìm ước chung lớn nhất của a,b bằng cách phân tích a,b thành tích các số nguyên tố và từ đó tìm ra tích các số nguyên tố chung có ở cả 2 số a,b chia hết cho tích đó

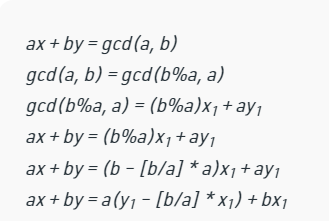
Để tìm UCLN của a,b ta tiến hành đệ quy chia lấy phần dư a cho b, và ngược lại đến khi nào phần còn lại bằng 0

Minh hoạ thuật toán:

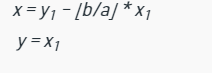


**Giải thuật Euclid mở rộng**:

Đây là bài toàn dạng ax + by = gcd(a,b) .Trong đó a,b là hệ số nguyên, x,y nhận giá trị nguyên . Thuật toán Euclid mở rộng là cập nhật kết quả gcd(a,b) bằng cách đệ quy gcd(b%a,a), tương tư giống như thuật toán Euclid cơ bản. Các giá trị x,y được tính bằng cách đệ quy x1, y1 và công thức tổng quát x,y được chứng minh như sau:

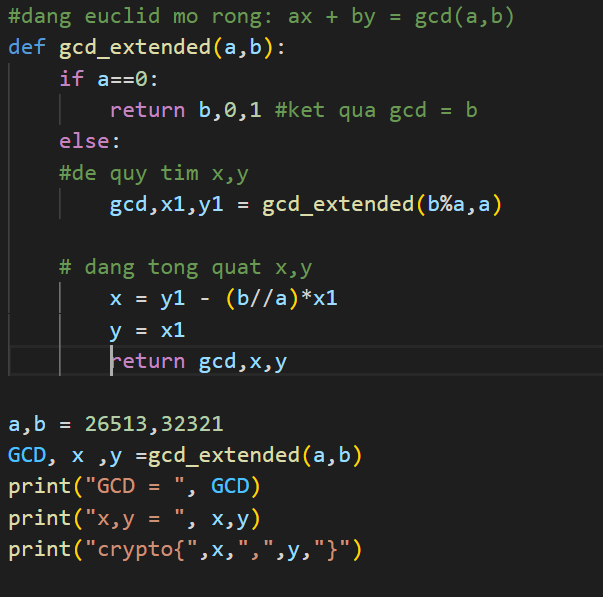


So sánh 2 phương trình cuối rút ra công thức tổng quát

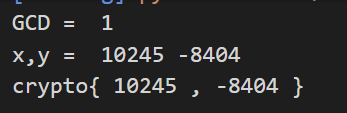


Trong bài toán Cryptohack cần tìm u,v khi đã có p,q . Như vậy ta cần tìm x,y từ công thức tổng quát trên

Cài đặt thuật toán theo công thức trên



Kết quả x,y ta nhận được:



Bài 2: Modular arithmetic 1

Đây là công thức đồng dư module dạng a b mod m hay a%m = b

Đề bài cho tính b của phương trình bài toán đã cho và so sánh đưa ra kết quả nhỏ nhất

Cài đặt thuật toán như sau



Kết quả bài toán



Bài 3: modular arithmetic 2

Đây là dạng bài luỹ thừa module có dạng ab mod m (a,b,m nguyên)

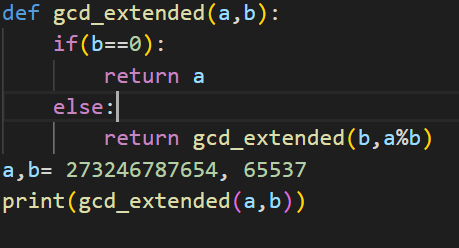
Tuy nhiên bài toán có đề cập đến thuyết đồng dư Fermat là dạng đặc biệt đồng dư luỹ thừa

Tính chất đồng dư Fermat được thể hiện như sau:

+ Nếu p là số nguyên tố, với mọi số nguyên a, thì ap a (mod p) hay ap % p = a

+ Đặc biệt nếu a không chia hết cho p (p<a), thì ap-1 1 (mod p) hay ap-1 % p =1

Cài đặt thuật toán như sau:



Kết quả như sau:



Do định lý fermat ap-1 % p =1 với mọi a không chia hết cho p (p<a) thì công thức này luôn đúng

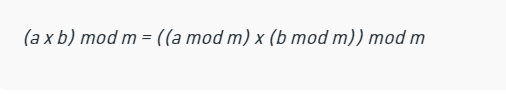
Bài 4: Modular inverting

Đây là dạng bài phổ biến tìm nghịch đảo module của a mod m là b với điều kiện : a, m là nguyên tố cùng nhau tức là gcd(a,m) =1, do đó (a\*b) mod m =1 thì b là nghịch đảo module của a theo m

Điều kiện cho b là phải nằm trong khoảng nguyên {1,2,….m-1} tức là trong khoảng nguyên module m (b<m). Nếu bài toán phức tạp hơn khi người ta sẽ cho m lớn để không thể sử dụng phương p pháp thử thủ công 1 đến m -1 được

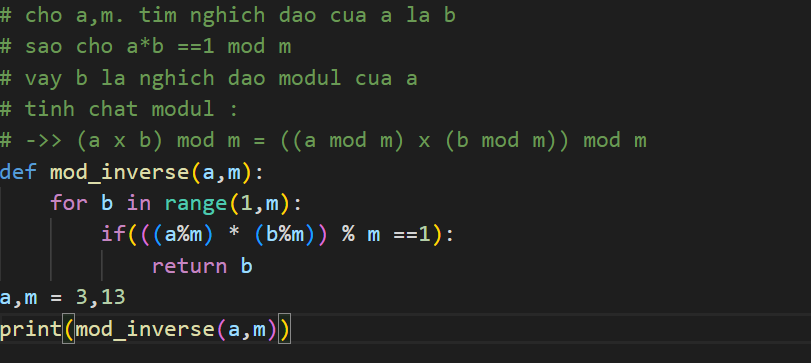
Bài toán cần tìm ở đây cho a,m nguyên tố cùng nhau và ta phải tìm nghịch đảo module của a là b

Áp dụng tính chất nhân module ta biến đổi công thức như sau



Để a\*b 1 mod m thì ((a%m) \* (b%m)) % m =1

Cài đặt thuật toán như sau:

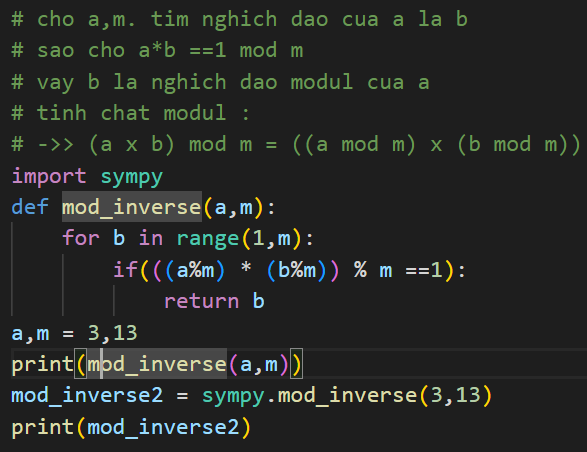


Kết quả đưa ra bài toán:

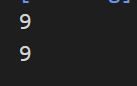


Trong classroom có yêu cầu thêm là sử dụng hàm inverse xây dựng sẵn trong python thì có thể sử dụng tới hàm **sympy.mod\_inverse(a,b)**

Cài đặt như sau:



Kết quả thu được

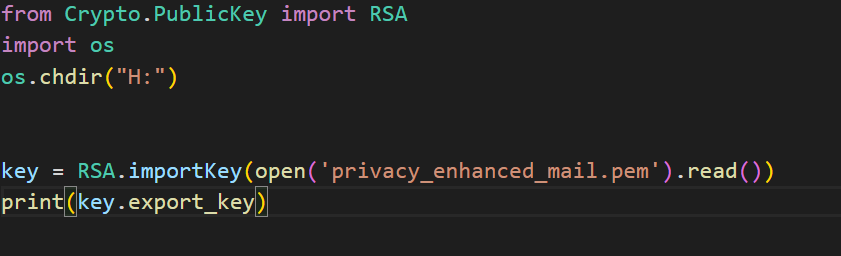


Bài 5: privacy enhanced mail

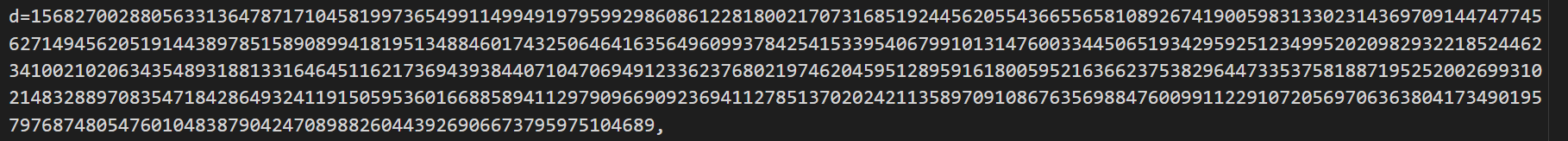
Tệp văn bản PEM được mã hoá bằng base64 ascii , ta có thể nhìn thấy dòng mã hoá ascii từ bắt đầu đoạn -----BEGIN CERTIFICATE------ và kết thúc trong -------END CERTIFICATE-----

Để trích xuất dữ liệu từ tập tin này , đề bài đưa ra là tìm khoá d trong hệ mã RSA của tệp PEM, và có cho ta gợi ý là sử dụng thư viện **PyCryptodome** và dùng hàm **RSA.importKey().** Thêm 1 gợi ý từ classroom là: key được mã hoá dưới dạng PEM format. Tuy nhiên có nhiều cách để trích xuất được data ra như là sử dụng OpenSSL, online decoder, python,.. Cá nhân tôi thích viết code thì sẽ trực quan hiểu được bản chất hơn, cộng với sử dụng tối đa được gợi ý bài toán.Như vậy ý tưởng đưa ra là ta cố gắng đọc được tệp này và tìm ra khoá **d** đã được mã hoá trong file pem theo như bài toán yêu cầu:

Mô tả đoạn code viết bằng Python như sau:



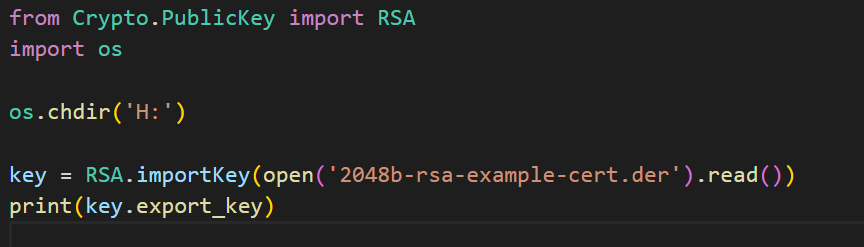
Kết quả bài toán:



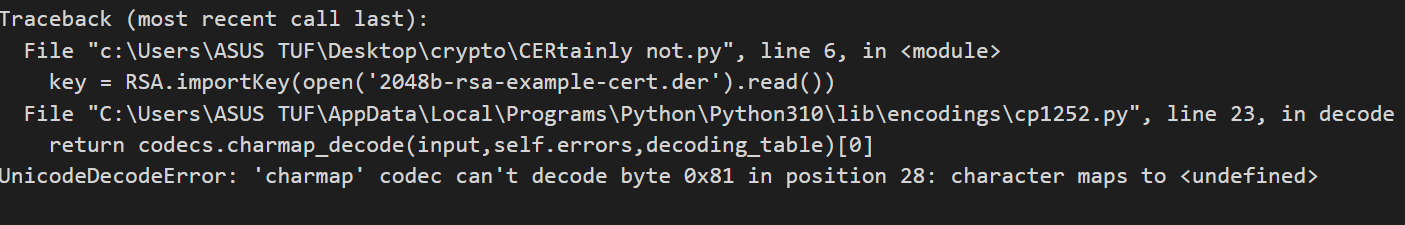
Bài 6: CERtainly not

Đây là 1 dạng khác của chứng chỉ khác so với PEM format mã hoá bằng base64 ascii thì bài này sẽ cho ta 1 chứng chỉ dạng DER format mã hoá nhị phân. Ta có gợi ý từ classroom là tiếp tục sử dụng hàm importKey(). Hàm này khá là hữu dụng khi có thể hỗ trợ RSA key encoded dạng nhị phân hoặc là PEM format

Ta thử làm theo cách giống như bài số 5



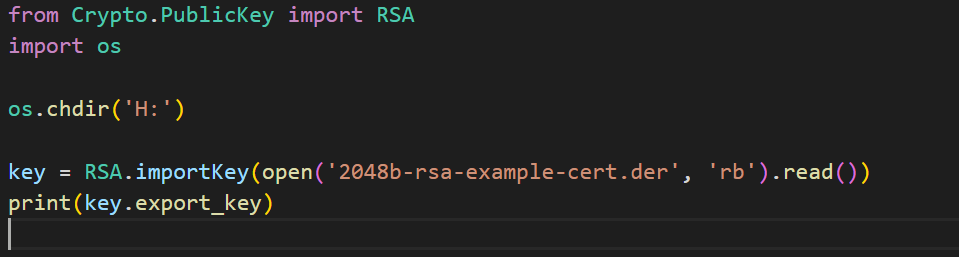
Cho ra kết quả như sau:



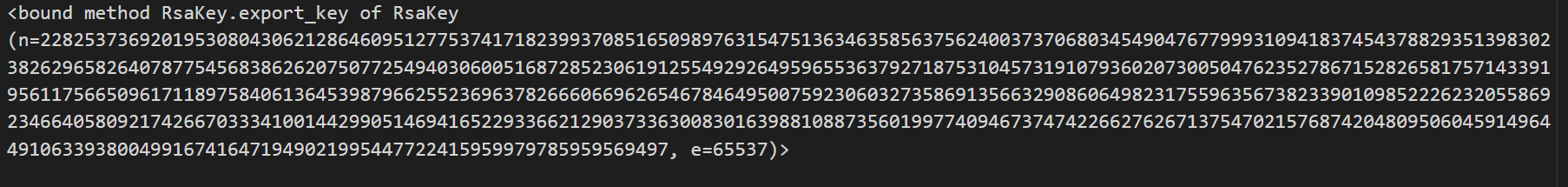
Lỗi ở đây là không thể decode 1 đoạn byte nào đó trong certificate được

Để ý rằng là đây là certificate định dạng DER có data encoded binary như vậy thì ta muốn importKey được thì phải đọc certificate ở dạng binary

Ta sửa lại đoạn code sau: đọc file **.der** dạng binary **rb**

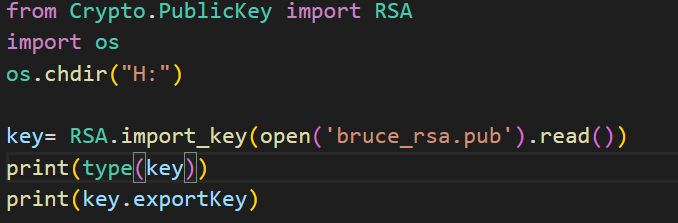


Kết quả thu được như sau:



Bài 7: SSH Keys

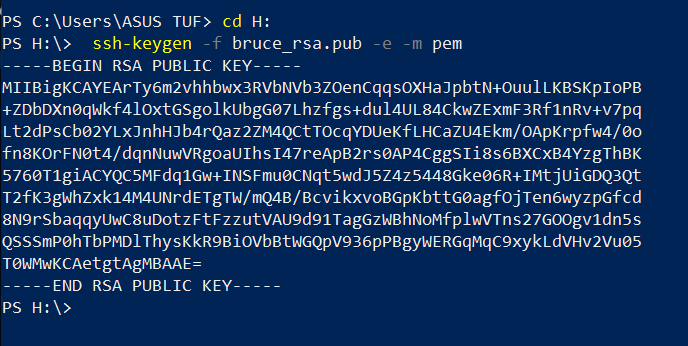
Mở file bruce-rsa.pub đây là tệp chứa khoá công khai public key, để ý hint từ classroom là convert the ssh public key to PEM format, ta lại tiếp tục sử dụng hàm importKey() xem thu được lợi nhuận gì không



Rất là hay importKey() cho phép file .pub được định dạng về PEM format nên là vẫn xuất ra được key

Tiếp cận được bài này thì có thể sử dụng openssl ta convert định dạng file .pub về file .pem và sử dụng keygen sinh ra cặp khoá, từ đó xuất ra thông tin public key cần tìm

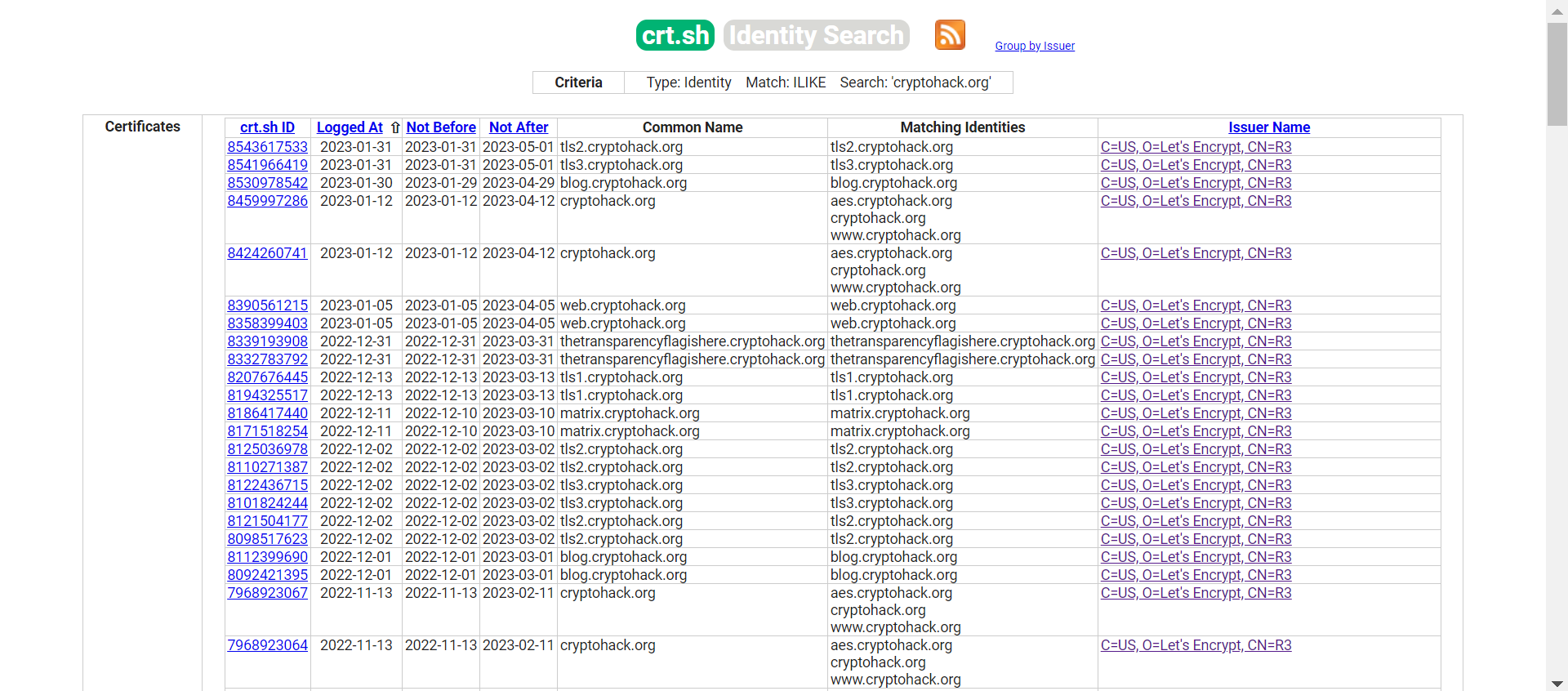
Mở powershell lên:



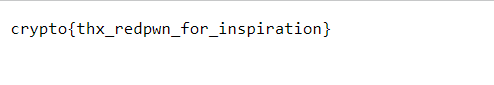
Bài 8: Transparency

Bài này gợi ý rằng hãy tìm subdomain của cryptohack và truy cập vào dường link subdomain để tìm ra đáp án flag bên trong link, kết hợp với hint “search for the flag subdomain and curl it” , là thoả mãn được sử dụng hết các hint đã cho

Để tìm được subdomain của cryptohack ta có thể truy cập trang web <https://crt.sh/> để khai thác thông tin của trang web này, sử dụng keyword cryptohack.org rồi Search xem và cho ra kết quả như sau:



Ta có thể thấy rất nhiều subdomain phụ và để ý có 1 domain phụ là thetransparencyflagishere.cryptohack.org ,kiểm tra truy cập theo tên miền đó và có ngay kết quả



Như vậy ta đã biết được flag của bài này