REPORT

## Q1

After I get C,N and e values we can use another ciphertext which is related to inital ciphertext and with the responce on that c_ value m_ will be relevant to m. I selected a random int = 10 and took it power to e with respect to N and multiplied it with C. My C_ value is = C * (10^e) Mod N and C is m ^e. While decrpytion it will take it's power to d and e*d is 1. Thus M_ will be m * 10 i.e. m=m_ /10. You can check RSA_Oracle_client.py script and my output is:

Bravo! You found it. Your secret code is 87912

Congrats

## Q2

R is a 8 bit value and PIN is between 1000 and 9999. These numbers are not cryptographically big numbers. So what we can do is that test for a given PIN including each R values whether is equal to C or not. In other words, I created 2 loops, first for the PIN's and second for the random 8 bit values each iteration I checked if my candidate is equal to C. And the secret pin code is 1308. Please check q2.py. Output:

Secret code is: 1308

## Q3

When I checked ElGamal implementation this row shows that k is a random but it is too small so that it can be reached by exhaustive search.

```
    k = random.randint(1, 2**16-1)
```

Keeping k value secret is very important for security, we know k value and t = h^k * m i.e. m = t * (h^k) ^-1 . Taking inverse of k'th power of h with respect to mod p and multiplying with t will give the message. The output of q3.py is :

k= 31659

b'Be yourself, everyone else is already taken.'

## Q4

The r values of two encrypted results are the same, which means that same k values are used in these encrpytions. This is a serious issue that results in a leak that is shown in lectures :

$t1/m1 \equiv \beta^k \equiv t2/m2 \bmod p \rightarrow m2 \equiv (t2m1)/t1$

I translated byte array m into integer, took inverse of t1 and applied the above equation. The output of q4.py function:

b'A person can change, at the moment when the person wishes to change.'