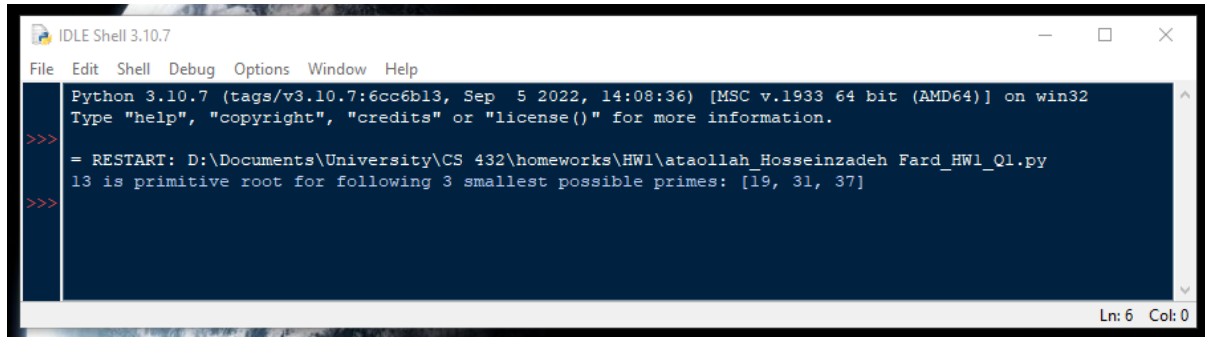


## Question 1

For this question I wrote a python program that searches for first 3 smallest possible primes that 13 is their primitive roots.



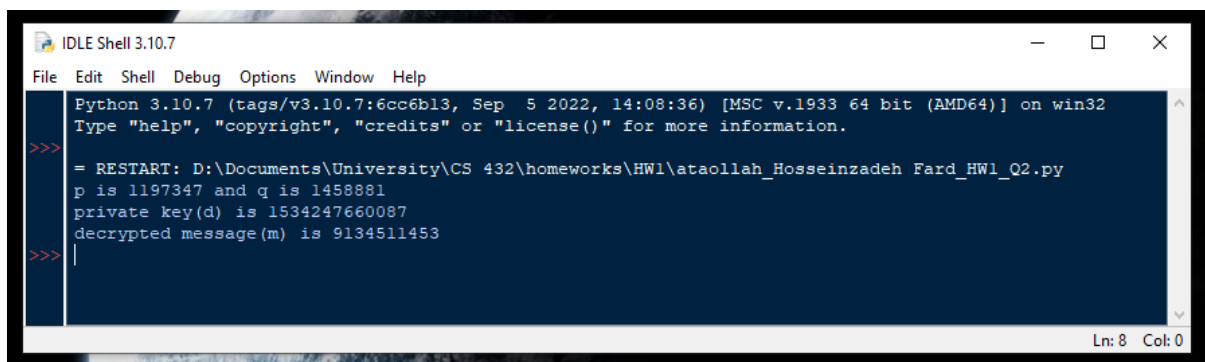
```
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep 5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>> = RESTART: D:\Documents\University\CS 432\homeworks\HW1\ataollah_Hosseinzadeh Fard_HW1_Q1.py
13 is primitive root for following 3 smallest possible primes: [19, 31, 37]
>>>
```

Therefore, 3 smallest possible prime numbers that 13 is their primitive root are 19, 31, 37.

## Question 2

In this question first I had to factor out N to find p and q to calculate phi. Then I had to calculate private key using the phi, which calculated in previous step, for finding private key I had to use a modular multiplicative inverse algorithm with “gcd(e, phi) = 1” approach. Lastly, I have all information to be able to decrypt message or find out the original message, also to decrypt I had to find a fast modular exponentiation algorithm.



```
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep 5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

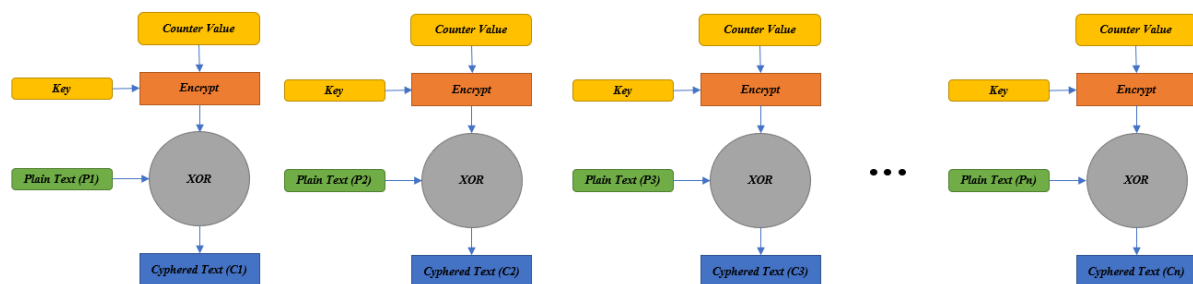
>>> = RESTART: D:\Documents\University\CS 432\homeworks\HW1\ataollah_Hosseinzadeh Fard_HW1_Q2.py
p is 1197347 and q is 1458881
private key(d) is 1534247660087
decrypted message(m) is 9134511453
>>> |
```

After execution, results are as following:

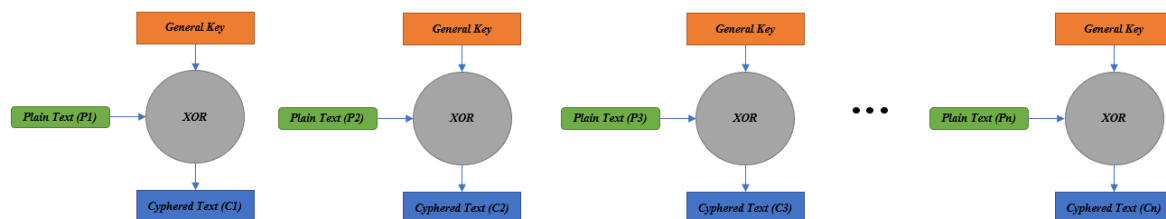
$p = 1197347$  and  $q = 1458881$  and  $d = 1534247660087$  and  $m = 9134511453$

## Question 3

Since the question tells that every key and counter value are the same, it is easy to find encryption key and then decrypt cyphered text. It is needed to find the encryption key, which is the encrypt block, which after this I will call it “General Key”, in following diagram.



since all the counter values and keys are same, we can generate easier diagram as follows:

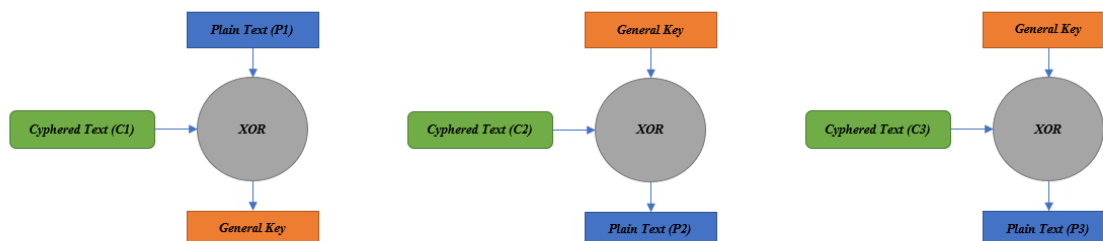


and now this diagram can be used both for encryption and decryption.

$$C_i = P_i \oplus \text{General\_Key}$$

$$P_i = C_i \oplus \text{General\_Key}$$

So, with using these equations first I found the General Key with doing xor on  $c_1$  and  $p_1$ , then to find  $p_2$  and  $p_3$  I did xor General Key with  $c_2$  and General Key with  $c_3$ . All the operation I did are as following diagram:



```

Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep 5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:\Documents\University\CS 432\homeworks\HW1\ataollah Hosseinzadeh Fard_HW1_Q3.py
General Key is (0xa7144867b7a64144) that is obtained by 'c1 XOR p1'
p2 is (0x89aab50fc37aaf6b) that is obtained by 'c2 XOR General Key'
p3 is (0x6590a8bbf8900955) that is obtained by 'c3 XOR General Key'
>>>
  
```

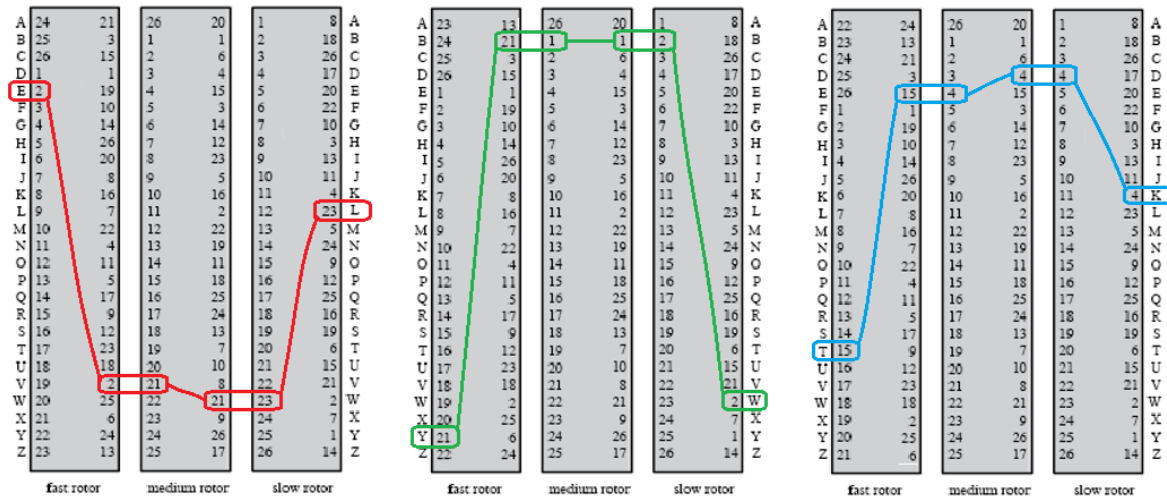
After execution, the results are as following:

$$P_2 = 89aab50fc37aaf6b$$

$$P_3 = 6590a8bbf8900955$$

## Question 4

To encrypt “EYT” with 3 rotors, we should trace the indexes in each rotor to find position of next index. Moreover, after encryption of each letter, a stroke happens, that means left-most rotor shifts down, since we have only 3 letters, strokes will be limited to “fast rotor”, so for 1<sup>st</sup> letter no strokes, for 2<sup>nd</sup> letter 1 stroke, and for 3<sup>rd</sup> letter 2 strokes needed on the initial setting form.



Therefore, encryption of “EYT” with 3 rotors, with given initial settings, is “LWK”.