Name - Vedant Patel

Roll Number - 19BCP138

Experiment – 8

Aim: - Study and Implement of RSA

Introduction: -

RSA is an asymmetric (public-key) algorithm to encrypt plaintext or to decrypt ciphertext. RSA stands for Rivest, Shamir and Adleman who first publicly described it. It is the first algorithm known to be suitable for signing as well as encryption, and was one of the first great advances in public-key cryptography. RSA is widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and using of up-to-date implementations.

RSA is key pair generator. Process of calculating the public key and private key: -

1. Choose two different random random numbers p and q
2. Count n = p q

   n is a module for public key and private keys
3. Count ϕ (n) = (p - 1) (q - 1)
4. Select the whole number k so that 1 <k <ϕ (n) and k are cohesive to ϕ (n): k and ϕ (n) with no elements other than 1; gcd (k, ϕ (n)) = 1.
5. k is issued as a public key advertisement
6. Calculate d to satisfy d k ≡ 1 (mod ϕ (n)) eg .: D k = 1 + x ϕ (n) by a certain number x
7. d is maintained as a private key provider

The public key consists of n and k.

The private key consists of p, q, and the private exponent d.

Encryption: -

- Encrypted text will be pow(message,k) mod n

Decryption: -

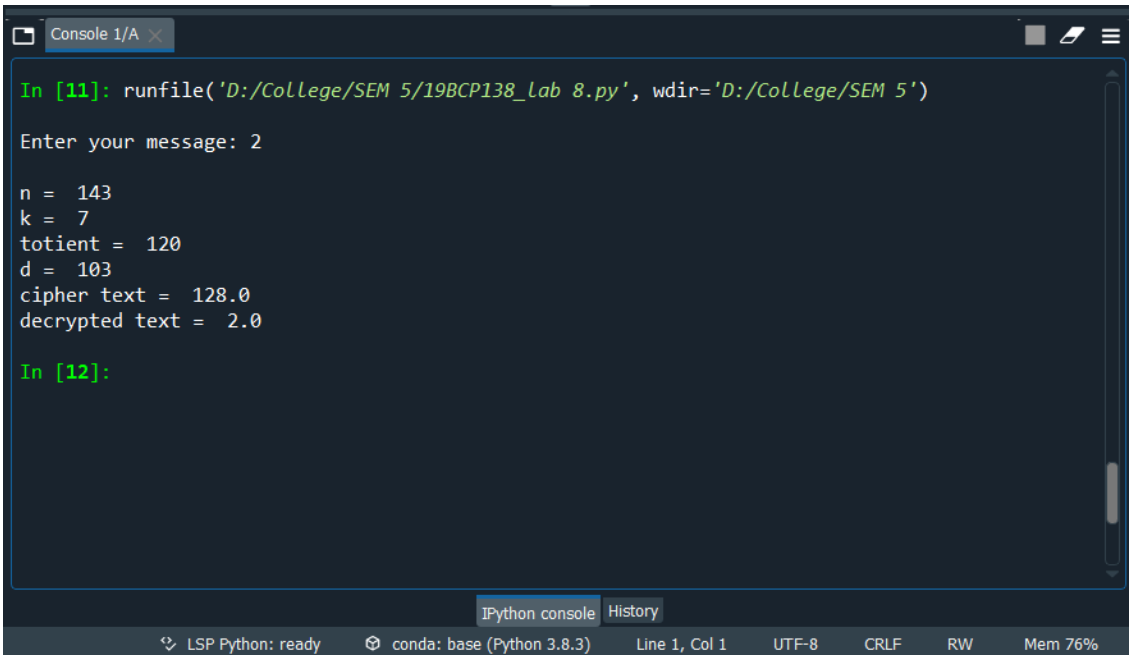
- Decrypted text will be pow(cipher_text,d) mod n

Program:

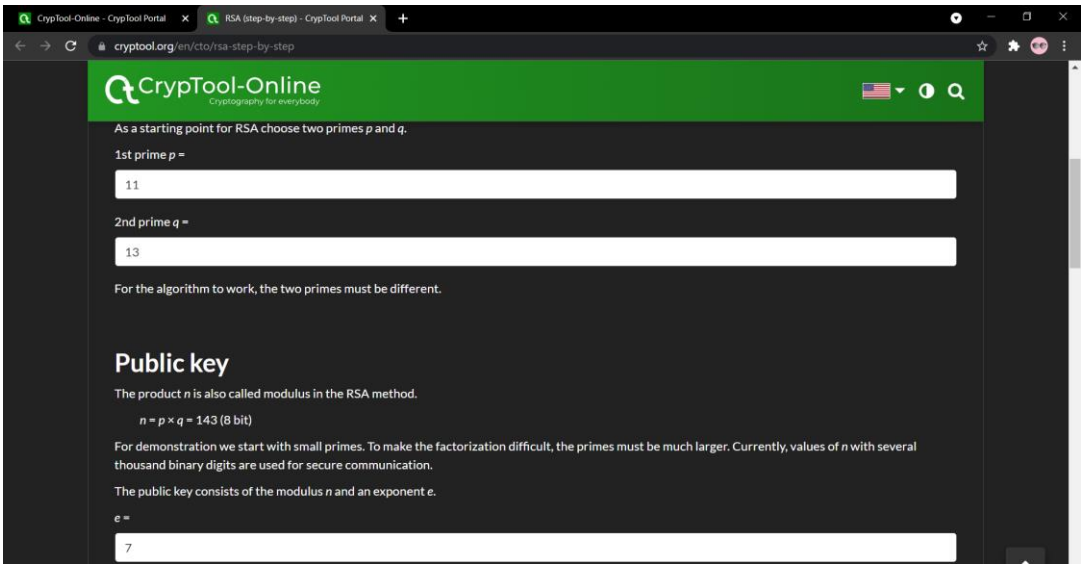
```
from decimal import Decimal

def gcd(m,n):
    if n==0:
        return m
    return gcd(n,m%n)

#input variables
p = 11
q = 13

no = float(input("Enter your message: "))

#calculate n
n = p*q

#calculate totient
totient = (p-1)*(q-1)

#calculate K
for k in range(2,totient):
    if gcd(k,totient)== 1:
        break

for i in range(1,10):
    x = 1 + i*totient
    if x % k == 0:
        d = int(x/k)
        break


local_cipher = Decimal(0)
local_cipher = pow(no,k)
cipher_text = local_cipher % n

decrypt_t = Decimal(0)
decrypt_t= pow(int(cipher_text),d)
decrpyted_text = float(decrypt_t % n)

print('\n'+'n = ',str(n))
```

```
print('k = ',k)
print('totient = ',totient)
print('d = ',d)
print('cipher text = ',cipher_text)
print('decrypted text = ',decrpyted_text)
```

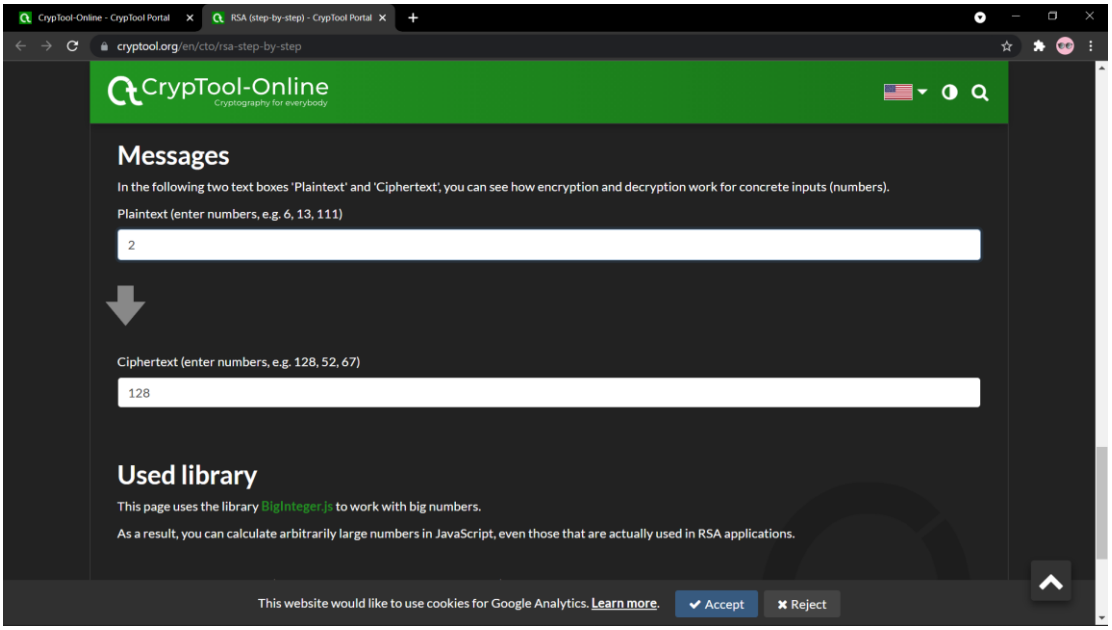
Output (Program): -



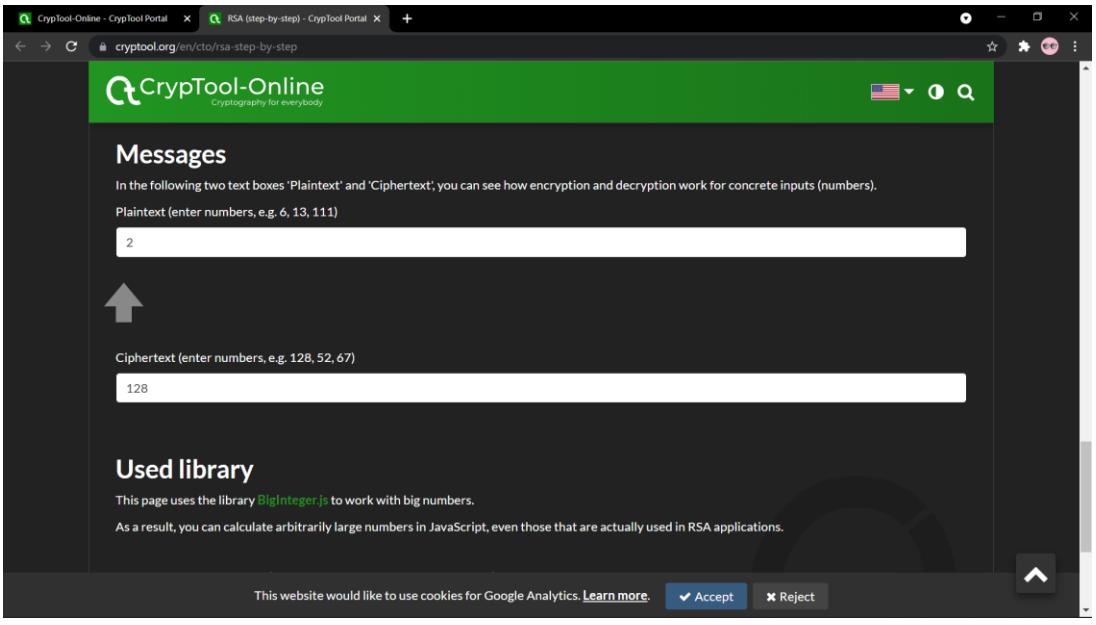
Output (CrypTool) :-



Encryption:



Decryption:



Cryptanalysis: -

We can prove, just using the specification of CBC-MAC, that the messages b || (M(b) ⊕ M(a) ⊕ b) and a || b shares the same tag. This approach is a common method used in cryptanalysis. If you were to use CBC-MAC in a protocol, it provides

information about specific weaknesses and how not to use it. Some other ways to attack RSA are;

Attacks:

- Small factors
- Fermat factorization
- Batch GCD
- Elliptic Curve Method (ECM)
- Weak entropy
- Smooth p-1 or p+1
- Fault injection
- Small private exponent
- Known partial bits
- p/q near a small fraction
- Shared bits
- Weaknesses in signatures
- Side channel attacks
- Number Field Sieve (NFS)
- Shor quantum algorithm

Applications

- Banking - The RSA algorithm is widely used by banks to protect their personal information, such as customer information and transaction records. Other cases are credit cards and office computers.
- Telecommunications - RSA algorithm helps encrypt telephone data such as concerns about privacy issues.
- E-commerce - The RSA algorithm helps to protect transaction user identity.

References

https://speakerdeck.com/rlifchitz/15-ways-to-break-rsa-security

https://www.codespeedy.com/rsa-algorithm-an-asymmetric-key-encryption-in-python/

https://courses.cs.washington.edu/courses/cse484/20au/sections/slides/section_5.pdf