



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

CRYPTOGRAPHY LAB

ASSESSMENT

NAME: SHIVANK PANDEY

REG NUMBER: 21BKT0021

1) RSA ALGO IN PYTHON

CODE-

```
import math
p = eval(input());
q = eval(input());
n = p*q
print("n =", n)
phi = (p-1)*(q-1)
e = 2
while(e<phi):
    if (math.gcd(e, phi) == 1):
        break
    else:
        e += 1

print("e =", e)
k = 2
d = ((k*phi)+1)/e
print("d =", d)
print(f'Public key: {e, n}')
print(f'Private key: {d, n}')
msg = 11
print(f'Original message:{msg}')
C = pow(msg, e)
C = math.fmod(C, n)
print(f'Encrypted message: {C}')
M = pow(C, d)
M = math.fmod(M, n)
print(f'Decrypted message: {M}')
```

OUTPUT-

```
PROBLEMS 20 OUTPUT TERMINAL PORTS GITLENS SQL CONSOLE DEBUG CONSOLE Code + - [ ] [ ] ... ^ X

c:\Users\Shivank\Desktop\CryptoGraphy>python -u "c:\Users\Shivank\Desktop\CryptoGraphy\RSA.py"
3
5
n = 15
e = 3
d = 5.666666666666667
Public key: (3, 15)
Private key: (5.666666666666667, 15)
Original message:11
Encrypted message: 11.0
Decrypted message: 12.328822647104971

c:\Users\Shivank\Desktop\CryptoGraphy>python -u "c:\Users\Shivank\Desktop\CryptoGraphy\RSA.py"
7
11
n = 77
e = 7
d = 17.285714285714285
Public key: (7, 77)
Private key: (17.285714285714285, 77)
Original message:11
Encrypted message: 11.0
Decrypted message: 49.0

c:\Users\Shivank\Desktop\CryptoGraphy>
```

Connect Java: Ready Ln 27, Col 36 (506 selected) Spaces: 4 UTF-8 CRLF Python 3.9.6 64-bit Go Live Spell Prettier

10:16 PM
14-04-2024

2)ECC ALGO IN PYTHON

```
from tinyec import registry
from Crypto.Cipher import AES
import hashlib, secrets, binascii
```

```

def encrypt_AES_GCM(msg, secretKey):
    aesCipher = AES.new(secretKey, AES.MODE_GCM)
    ciphertext, authTag = aesCipher.encrypt_and_digest(msg)
    return (ciphertext, aesCipher.nonce, authTag)

def decrypt_AES_GCM(ciphertext, nonce, authTag, secretKey):
    aesCipher = AES.new(secretKey, AES.MODE_GCM, nonce)
    plaintext = aesCipher.decrypt_and_verify(ciphertext, authTag)
    return plaintext

def ecc_point_to_256_bit_key(point):
    sha = hashlib.sha256(int.to_bytes(point.x, 32, 'big'))
    sha.update(int.to_bytes(point.y, 32, 'big'))
    return sha.digest()

curve = registry.get_curve('brainpoolP256r1')

def encrypt_ECC(msg, pubKey):
    ciphertextPrivKey = secrets.randbelow(curve.field.n)
    sharedECKey = ciphertextPrivKey * pubKey
    secretKey = ecc_point_to_256_bit_key(sharedECKey)
    ciphertext, nonce, authTag = encrypt_AES_GCM(msg, secretKey)
    ciphertextPubKey = ciphertextPrivKey * curve.g
    return (ciphertext, nonce, authTag, ciphertextPubKey)

def decrypt_ECC(encryptedMsg, privKey):
    (ciphertext, nonce, authTag, ciphertextPubKey) = encryptedMsg
    sharedECKey = privKey * ciphertextPubKey
    secretKey = ecc_point_to_256_bit_key(sharedECKey)
    plaintext = decrypt_AES_GCM(ciphertext, nonce, authTag, secretKey)
    return plaintext

msg = b'Text to be encrypted by ECC public key and ' \
      b'decrypted by its corresponding ECC private key'
print("original msg:", msg)
privKey = secrets.randbelow(curve.field.n)
pubKey = privKey * curve.g

encryptedMsg = encrypt_ECC(msg, pubKey)
encryptedMsgObj = {
    'ciphertext': binascii.hexlify(encryptedMsg[0]),
    'nonce': binascii.hexlify(encryptedMsg[1]),
    'authTag': binascii.hexlify(encryptedMsg[2]),
    'ciphertextPubKey': hex(encryptedMsg[3].x) + hex(encryptedMsg[3].y %
2)[2:]
}
print("encrypted msg:", encryptedMsgObj)

decryptedMsg = decrypt_ECC(encryptedMsg, privKey)

```

```
print("decrypted msg:", decryptedMsg)
```

OUTPUT-

```
c:\Users\Shivank\Desktop\CryptoGraphy>python -u "c:\Users\Shivank\Desktop\CryptoGraphy\RSA.py"
original msg: b'Text to be encrypted by ECC public key and decrypted by its corresponding ECC private key'
encrypted msg: {'ciphertext': b'fbae5e90bf9f02870c3cae598c7ff0f0d6aaa6336904b27178782df5f5deb3651771bb535bd9651a93e8f2c7c1174652b22ad803fe46e5df50cbd4247efec041b75dec3ca9f63e15dde297d1d5782b381a45cadd37552ace34', 'nonce': b'b875a39de3a58fd161aaff613072836b', 'authTag': b'017d32eabd8e45b3ba028a42cc87d077', 'ciphertextPubKey': '0x2704cbb3f493c39aca26d57b688869518dd37a9c07fce2bcla68fa2a153dbf360'}
decrypted msg: b'Text to be encrypted by ECC public key and decrypted by its corresponding ECC private key'
```

```
c:\Users\Shivank\Desktop\CryptoGraphy>
```

Connect Java: Ready

Ln 5, Col 37 Spaces: 4 UTF-8 CRLF Python 3.9.6 64-bit Go Live 10 Spell Prettier

