IS Assignment # 1

Roll no: 20K-0409

Screen shots:

IS Assignment # 1_20K-0409_Mukand

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

+ Code

```python
import hashlib
import os

# Compute the greatest common divisor of two numbers using Euclid's algorithm.
def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a

# Compute the modular multiplicative inverse of 'a' under modulo 'm'.
def modinv(a, m):
    m0, x0, x1 = m, 0, 1
    while a > 1:
        q = a // m
        m, a = a % m, m
        x0, x1 = x1 - q * x0, x0
    if x1 < 0:
        x1 += m0
    return x1
```

RSA Signature:

```python

# Sign a message using the RSA private key.
# The signature ensures message authenticity and integrity.
def sign(private_key, message):
    # Convert the message into a SHA-256 hash.
    message_hash = int(hashlib.sha256(message.encode()).hexdigest(), 16)
    d, n = private_key
    signature = pow(message_hash % n, d, n)
    return signature

# Verify the authenticity of a message using its signature and the RSA public key.
def verify(public_key, message, signature):
    # Convert the message into a SHA-256 hash.
    message_hash = int(hashlib.sha256(message.encode()).hexdigest(), 16)
    e, n = public_key
    decrypted_hash = pow(signature, e, n)
    return message_hash % n == decrypted_hash

```

Encrypt and Decrypt:

```python

# Encrypt a plaintext using the RSA public key.
def encrypt(pk, plaintext):
    e, n = pk
    return [pow(ord(char), e, n) for char in plaintext]

# Decrypt a ciphertext using the RSA private key.
def decrypt(pk, ciphertext):
    d, n = pk
    return ''.join([chr(pow(char, d, n)) for char in ciphertext])

```

Output:

```
RSA Encryption/Decryption
Enter a prime number (e.g., 17, 19, 23): 109
Enter another prime number (different from the first): 103

Public key: (5, 11227)
Private key: (8813, 11227)

Want to input the message directly or from a .txt file? (m/f): m
Enter a message to encrypt (small data): hi, 102 , # mukand

Encrypted message: 875669332821815639297203768281562821815619698156654018127925100432181100 57
Decrypted message: hi, 102 , # mukand

Digital Signature: 3107
The signature is valid!
```

Input from file:

```python
    is_verified = verify(public, message, signature)
    if is_verified:
        print("The signature is valid!")
    else:
        print("The signature is invalid!")

```

```
RSA Encryption/Decryption
Enter a prime number (e.g., 17, 19, 23): 103
Enter another prime number (different from the first): 97

Public key: (5, 9991)
Private key: (3917, 9991)

Want to input the message directly or from a .txt file? (m/f): f
Enter the path to the .txt file: /content/I_S.txt

Encrypted message: 671687562504396465425043964654963016416425820764081004654164216525043968756764058204654219316416539616425047640360546541313896
Decrypted message: This is mukand krishna pursuing CS.

Digital Signature: 3652
The signature is valid!
```