Dreams are not what you see in sleep, they are the things that don't let you sleep - Dr APJ Abdul Kalam

Seminar On Internship

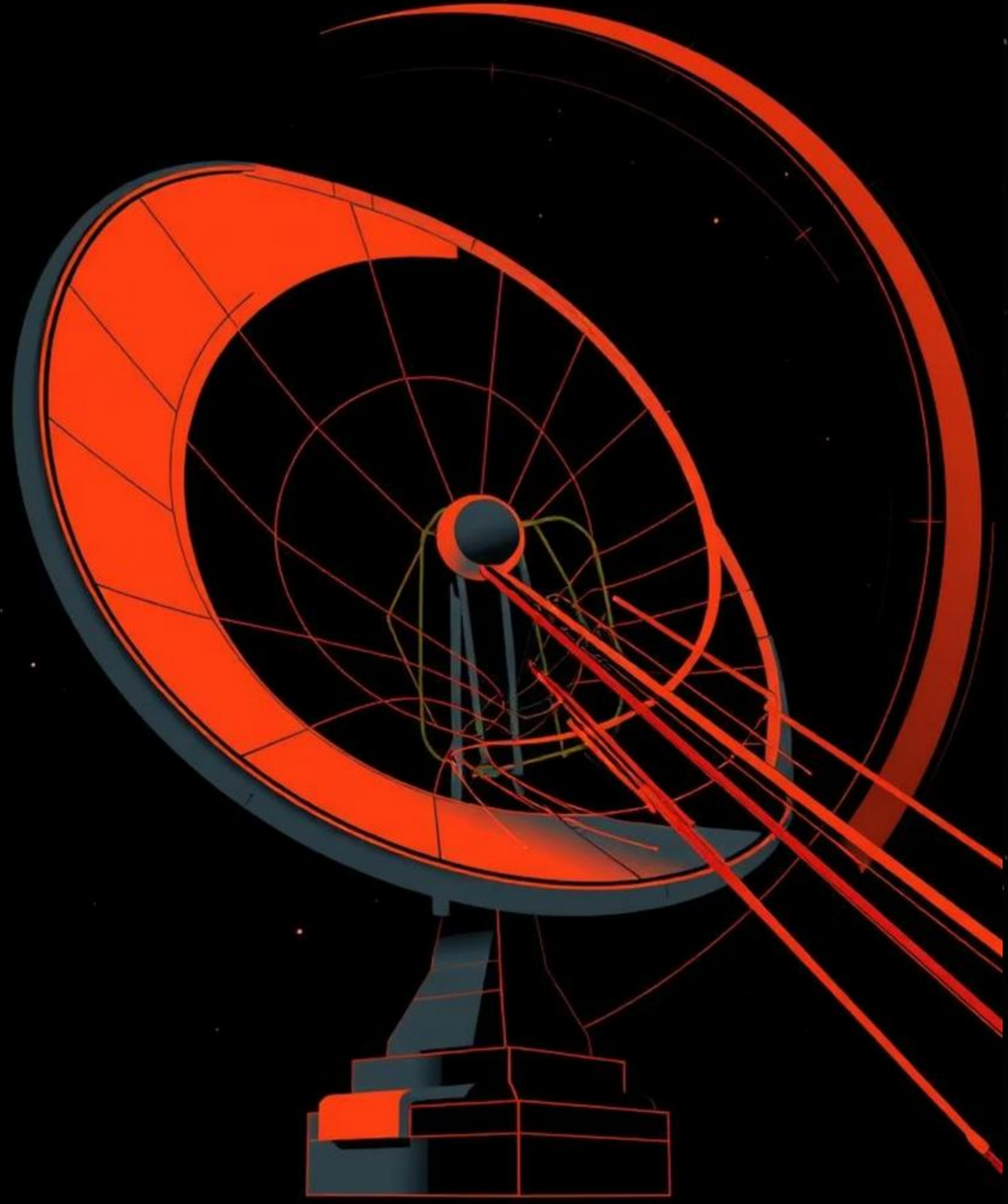
VEER SURENDRA SAI UNIVERSITY OF TECHNOLOGY, ODISHA, BURLA

Cryptography and its application in Range EOTS

Presented by :-

Rohan Mahapatra

Regd No. - 2202070091



Contents

1. ITR-DRDO
2. Electro Optical Tracking System
3. Range EOTS at ITR
4. Role of Cryptography in Test Range Scenario
5. What is Cryptography ?
6. Types of Cryptography
7. Symmetric Cryptography
8. Asymmetric Cryptography
9. RSA Algorithm
10. Process (Key Generation, Encryption, Decryption)
11. RSA Algorithm
12. Example of Encryption & Decryption
13. Symmetric Vs Asymmetric Cryptography
14. Conclusion
15. References
16. Certificate

ITR - DRDO

•DRDO was established in 1958, with the motto: "Balasya Mulam Vigyanam" – science is the source of strength.

•Its mission was to make India self reliant in key defense Technologies.

•It has developed missiles like Agni, Akash, Prithvi, Pinaka and BrahMos and have played a major role in integrating them with aircrafts like Russian Sukhoi 30 MKI, French Rafale and indigenous Tejas.

•Today DRDO is a network of around 41 laboratories across the nation.

•Among them ITR acts as one of the most important labs of DRDO that ensures safe and precise evaluation of rockets, missiles, and airborne systems.

Electro-Optical Tracking Systems (EOTS)

EOTS plays a major role in the **detection, tracking, and recording** of high-speed objects like missiles and aircraft during flight trials. To Achieve this, they use high resolution optical cameras including daylight and infrared sensors. The system is coordinated with other systems like radar and telemetry to provide full data.

High-Resolution Optics

- Daylight and thermal/IR cameras.
- Long-range zoom lenses for detailed observation.

Precision Tracking

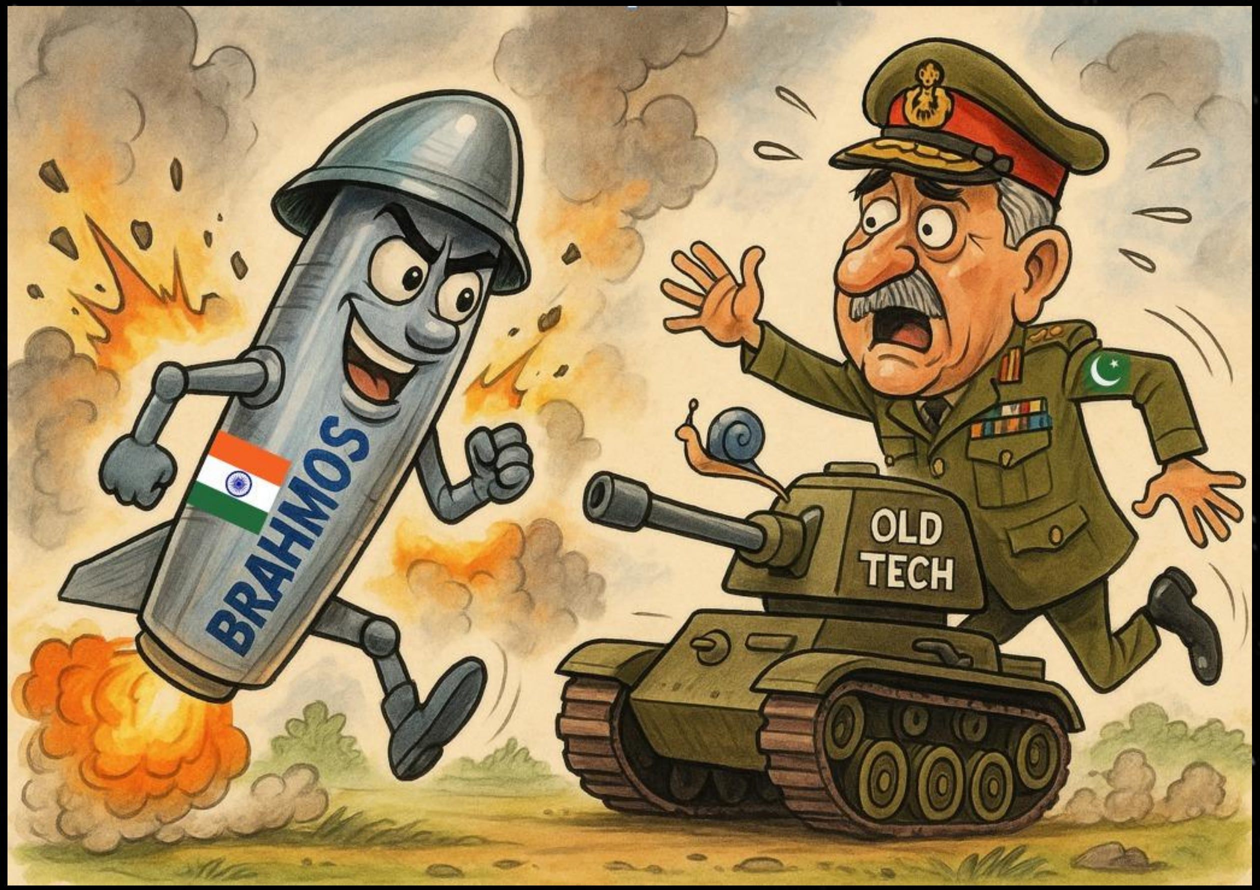
- Automated Tracking Mechanisms
- Real Time Data Acquisition and recording.

System Integration

- Seamless Integration with radar and telemetry systems.
- Enhanced Tracking even in low visibility.

Range EOTS at ITR

At facilities like the **Integrated Test Range (ITR),** Range EOTS form the backbone of instrumentation systems. They are setup at specific locations to track missiles and aircrafts



- **Tracking Flight Vehicles:** They track missiles and aircrafts using real time data.
- **Flight Analysis:** During the flight, velocity, acceleration, and orientation are recorded.
- **Safety Monitoring:** Ensures that missiles follow the predefined trajectory.
- **Post-Test Evaluation:** After the test is over , the recorded footage and data is evaluated for performance analysis.

The Role of Cryptography in Test Range Scenario

In test ranges like ITR, cryptography ensures the security and integrity of sensitive data such as command signals and video feeds.

Data Confidentiality

Encrypts real-time transmissions and stored information

Integrity Assurance

Keeps the data real by preventing data tampering.

Authentication

Prevents unauthorized personnel from getting the data.

What is Cryptography ?

Derived from Greek word *kryptos* (hidden) and *graphein* (to write) together form "hidden writing", cryptography is the science of transforming information to secure it from adversaries.

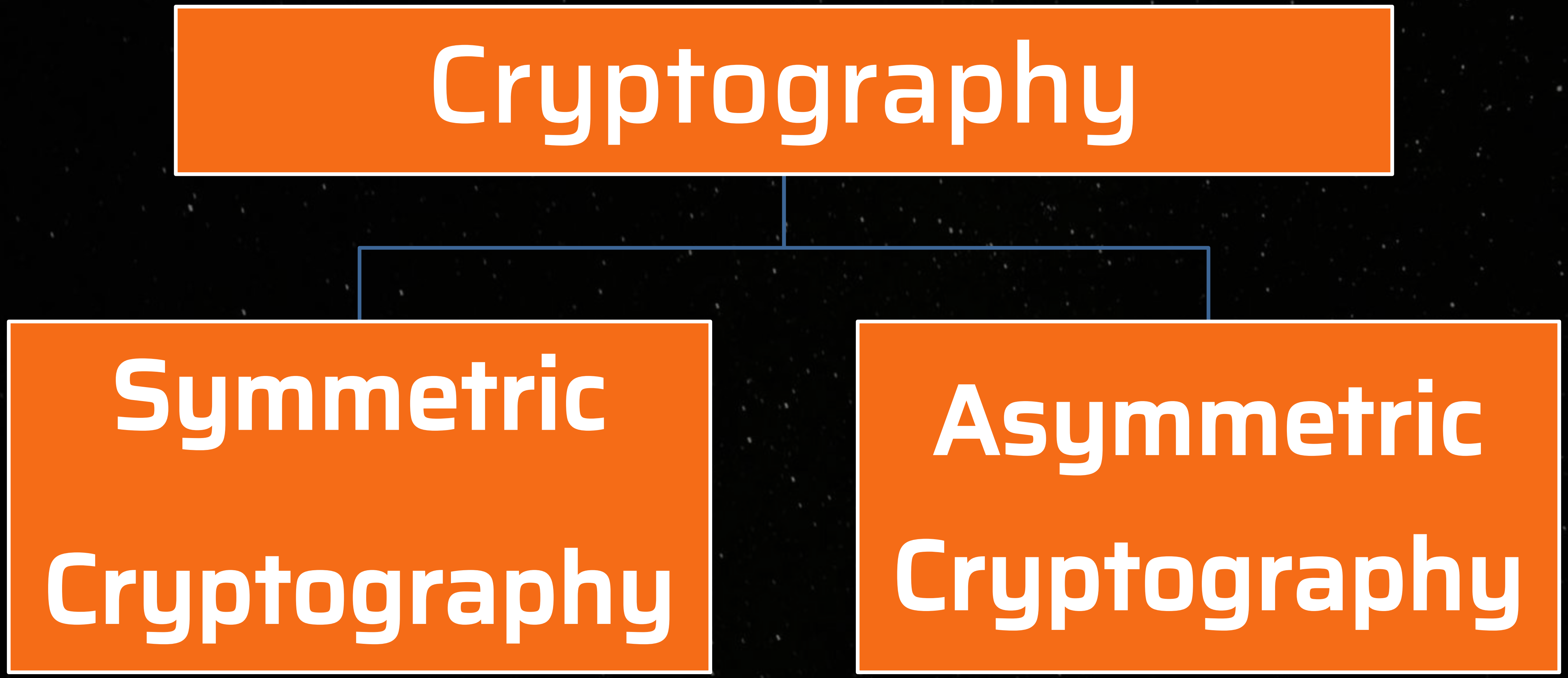
> "Cryptography is the science of securing information by converting it into a form that unauthorized users cannot understand."

Key Terms

- **Cryptography:** The science of encrypting data

- **Cryptanalysis:** The science of breaking down the encrypted data to readable format.

Types of Cryptography



Cryptography

Symmetric Cryptography

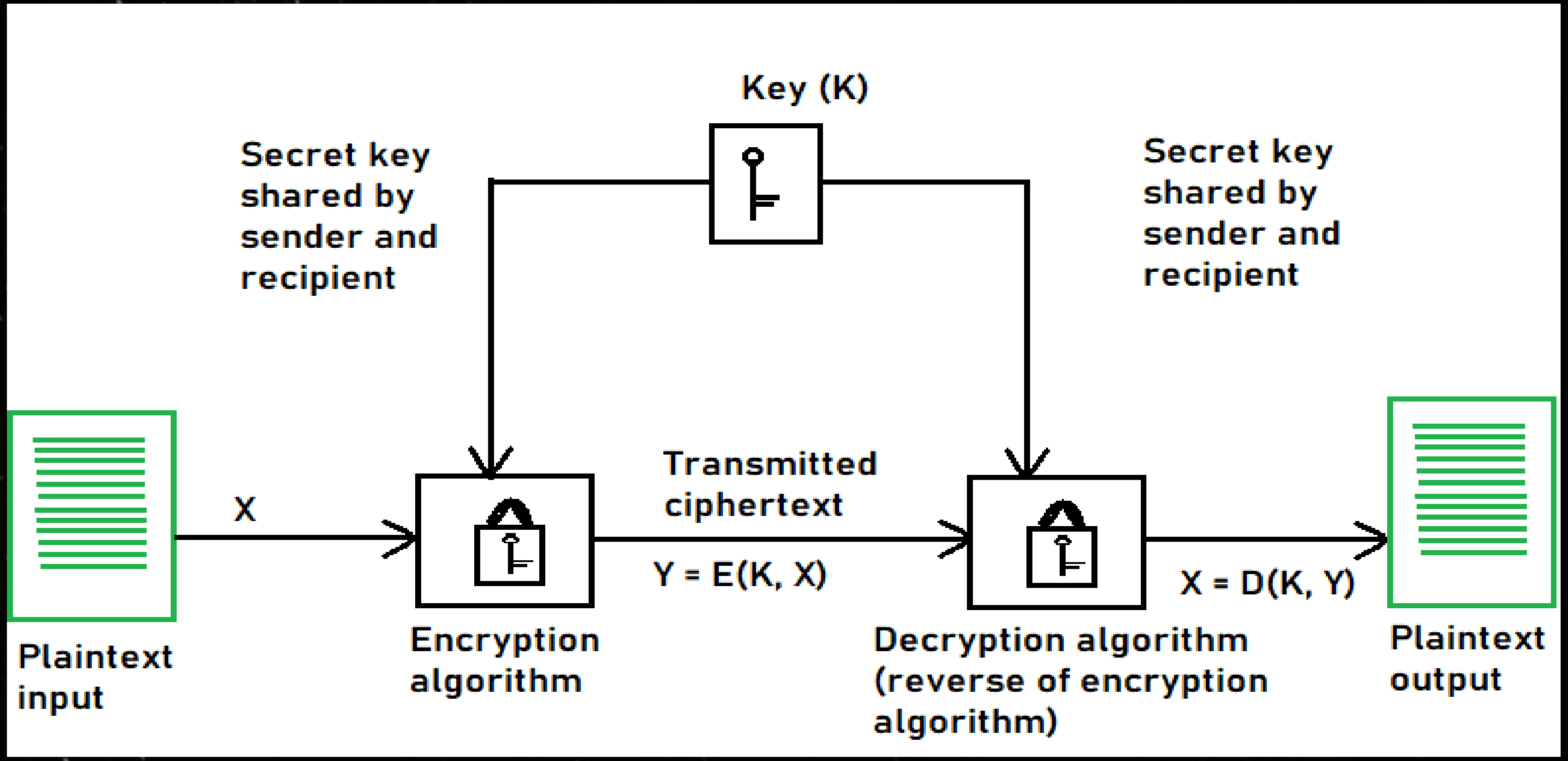
Asymmetric Cryptography

- Same key for encryption & decryption.
- Fast, but key sharing is a challenge.

- Uses public & private key pairs.
- More secure for key exchange, but slower.

Symmetric Cryptography

- Sender and receiver have same single secret key
- Sender uses it to encrypt plain text to cyphertext, receiver uses the same key to decrypt.
- Fast and efficient for encrypting **large amounts of data like video streams and image feeds.**
- Vulnerable if the key is **leaked or compromised.**
- Suitable for **confidentiality in closed systems.**
- Examples: AES, DES **etc.**


Key (K)
Secret key shared by sender and recipient
Secret key shared by sender and recipient
X
Transmitted ciphertext
Y = E(K, X)
X = D(K, Y)
Plaintext input
Encryption algorithm
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Asymmetric Cryptography

Definition:

Asymmetric cryptography (Public-Key Cryptography) uses **two keys**:

1. **Public key** (shared with everyone)
2. **Private key** (kept secret)

Used for **encryption, decryption, digital signatures**, and secure key exchange.

Uses:

Asymmetric cryptography is widely used in

- Secure email
- Digital signatures
- Online transactions

Common algorithms like RSA (Rivest–Shamir–Adleman) and ECC (Elliptic Curve Cryptography) are used for Asymmetric Cryptography

RSA Algorithm

- In 1977, Ronald Rivest, Adi Shamir and Leonard Adleman proposed a scheme which became the most widely used asymmetric cryptographic scheme called RSA.
- It is based on the mathematical difficulty of factoring product of large prime numbers — a problem that is easy to verify but extremely hard to reverse.
- Easy to multiply 2 primes, difficult to reverse the process and find the original primes.

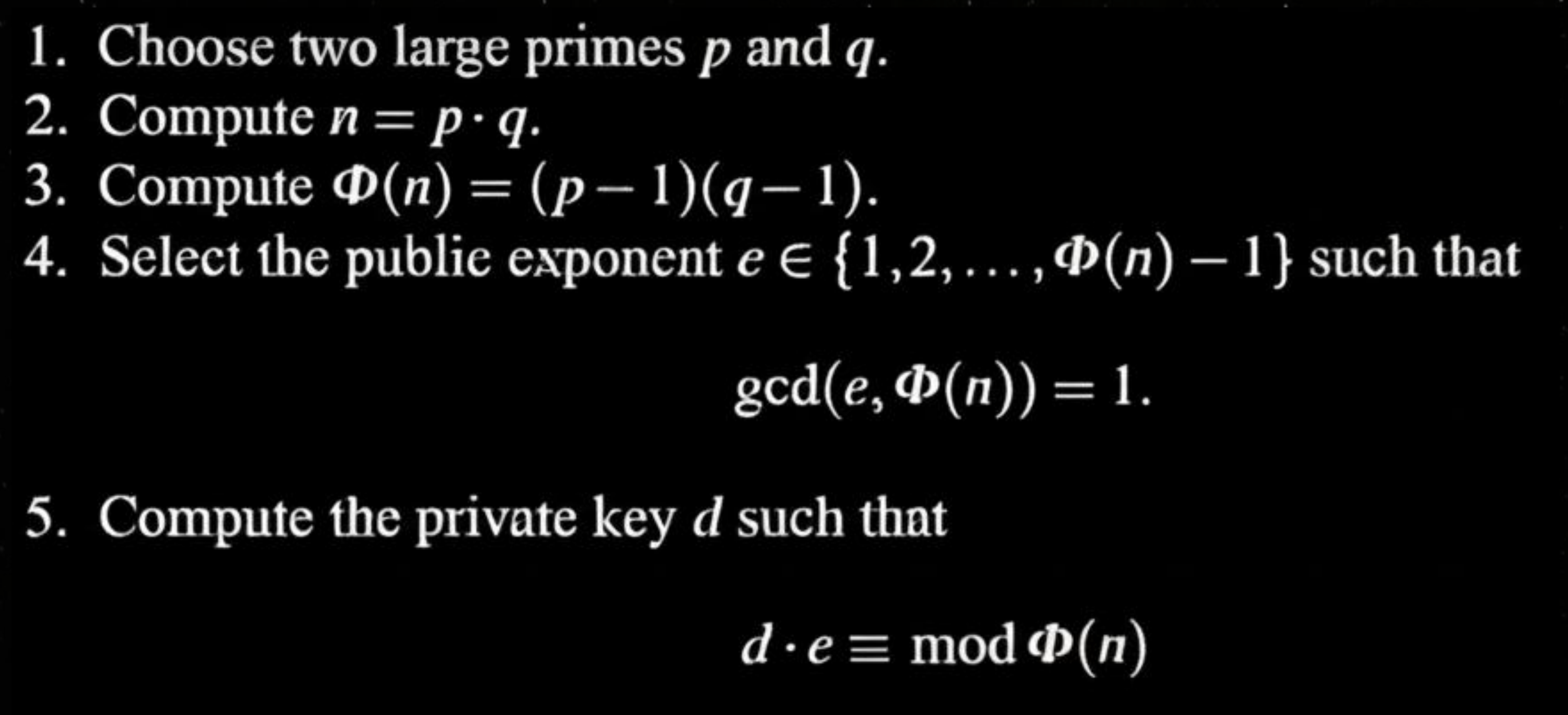
RSA uses **two keys**:

1. A **public key** for encryption (shared openly)
2. A **private key** for decryption (kept secret)

Anyone can use the public key to encrypt data, but only the holder of the private key can decrypt it.

Key Generation:

The process of creating a matching pair of public and private keys for secure communication.



1. Choose two large primes p and q.
2. Compute $n = p \cdot q$.
3. Compute $\Phi(n) = (p-1)(q-1)$.
4. Select the public exponent $e \in \{1, 2, \ldots, \Phi(n)-1\}$ such that

$$\gcd(e, \Phi(n)) = 1.$$

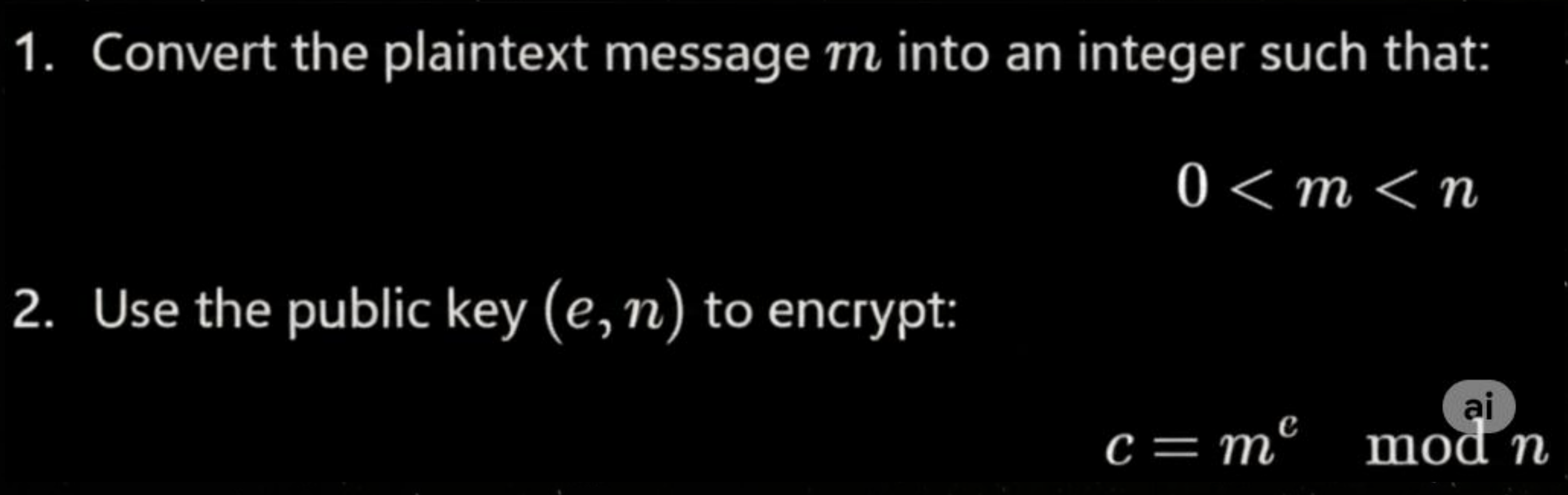
5. Compute the private key d such that

$$d \cdot e \equiv \bmod \Phi(n)$$

🔓 **Public Key: (e, n) – shared with everyone**
🔒 **Private Key: (d, n) – kept secret**

Encryption:

The process of converting a plaintext message into an unreadable ciphertext using the recipient's public key.



1. Convert the plaintext message m into an integer such that:

$$0 < m < n$$

2. Use the public key (e, n) to encrypt:

$$c = m^e \mod n$$

Alisha

where C is the cypher text

Decryption:

The process of converting the ciphertext back into the original plaintext using the private key.

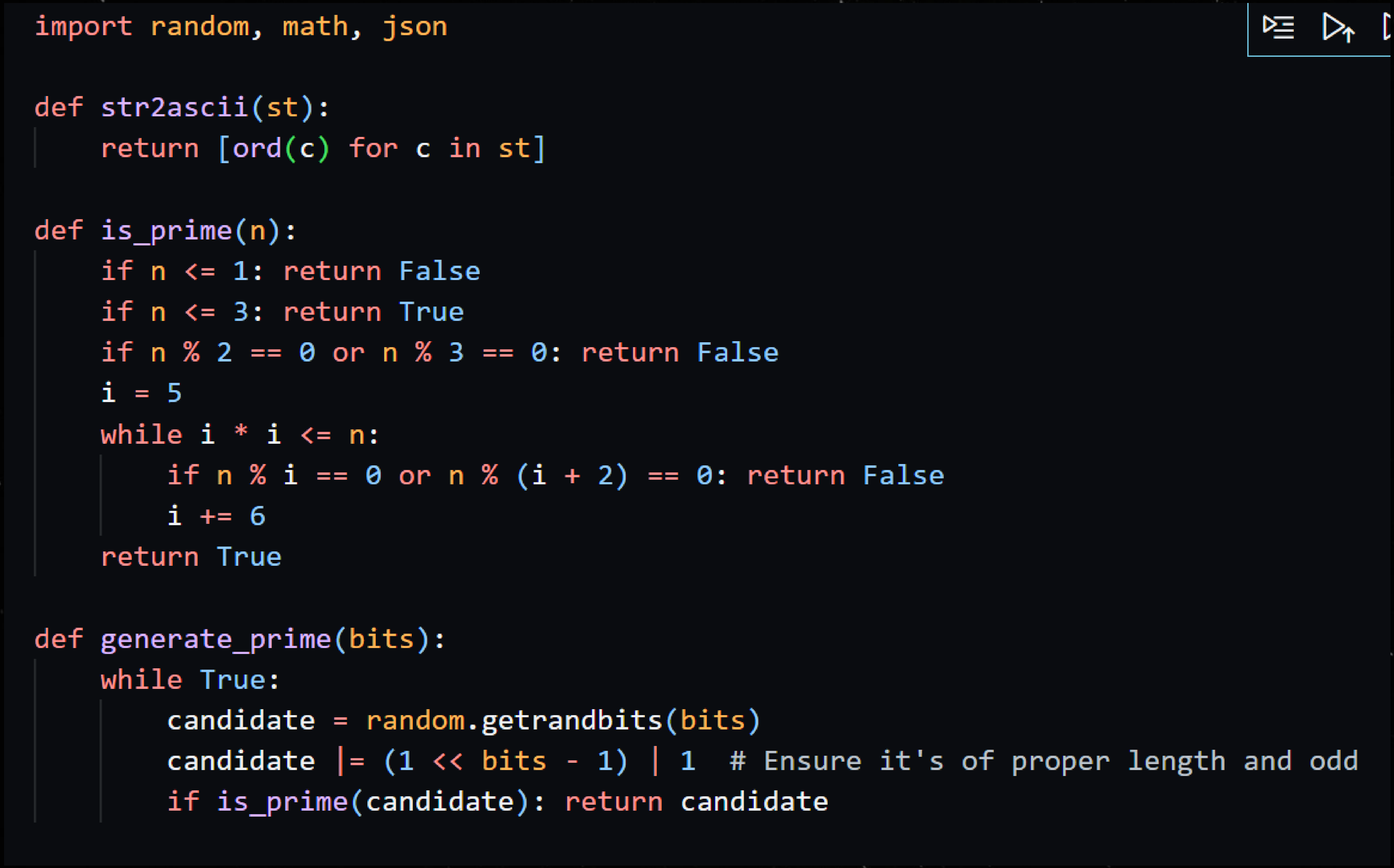
$$m = c^d \mod n$$

Bobby

Where:
c = the encrypted message (ciphertext)
d = private exponent
n = modulus
m = original plaintext message

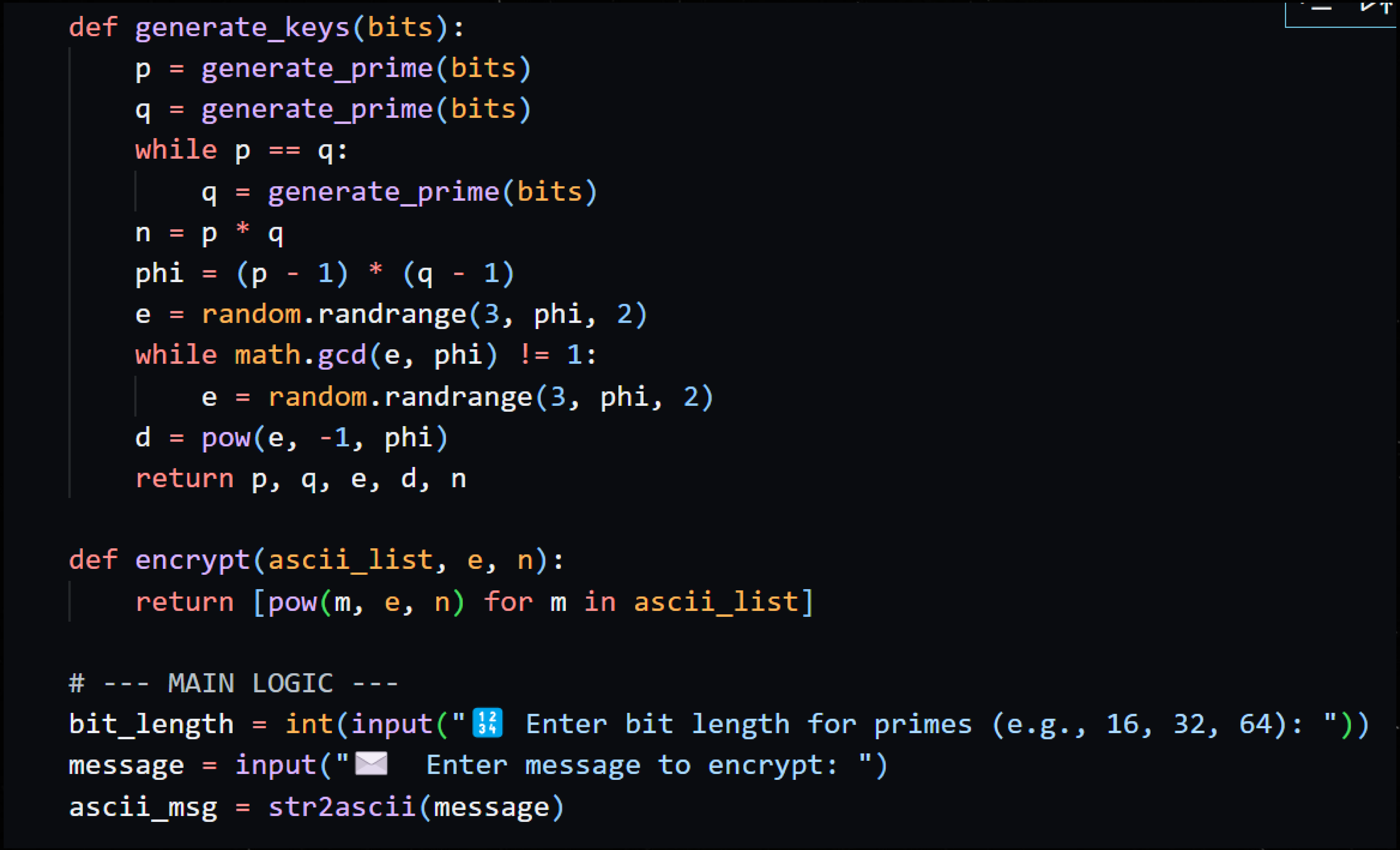
RSA Algorithm

Encryption:



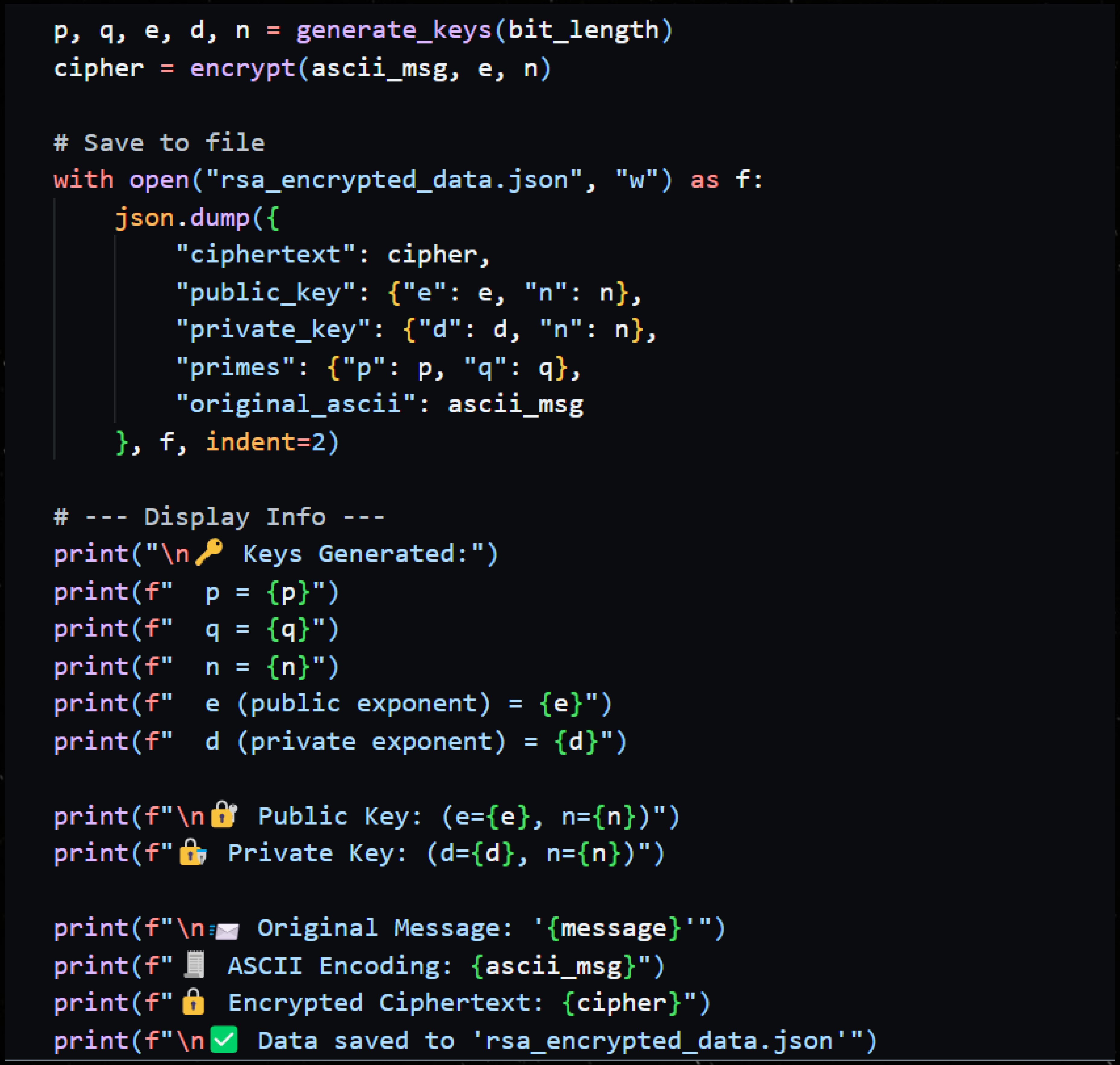
```python
import random, math, json

def str2ascii(st):
    return [ord(c) for c in st]

def is_prime(n):
    if n <= 1: return False
    if n <= 3: return True
    if n % 2 == 0 or n % 3 == 0: return False
    i = 5
    while i * i <= n:
        if n % i == 0 or n % (i + 2) == 0: return False
        i += 6
    return True

def generate_prime(bits):
    while True:
        candidate = random.getrandbits(bits)
        candidate |= (1 << bits - 1) | 1  # Ensure it's of proper length and odd
        if is_prime(candidate): return candidate
```

RSA Algorithm



```python
def generate_keys(bits):
    p = generate_prime(bits)
    q = generate_prime(bits)
    while p == q:
        q = generate_prime(bits)
    n = p * q
    phi = (p - 1) * (q - 1)
    e = random.randrange(3, phi, 2)
    while math.gcd(e, phi) != 1:
        e = random.randrange(3, phi, 2)
    d = pow(e, -1, phi)
    return p, q, e, d, n

def encrypt(ascii_list, e, n):
    return [pow(m, e, n) for m in ascii_list]

# --- MAIN LOGIC ---
bit_length = int(input("🔢 Enter bit length for primes (e.g., 16, 32, 64): "))
message = input("✉️  Enter message to encrypt: ")
ascii_msg = str2ascii(message)
```

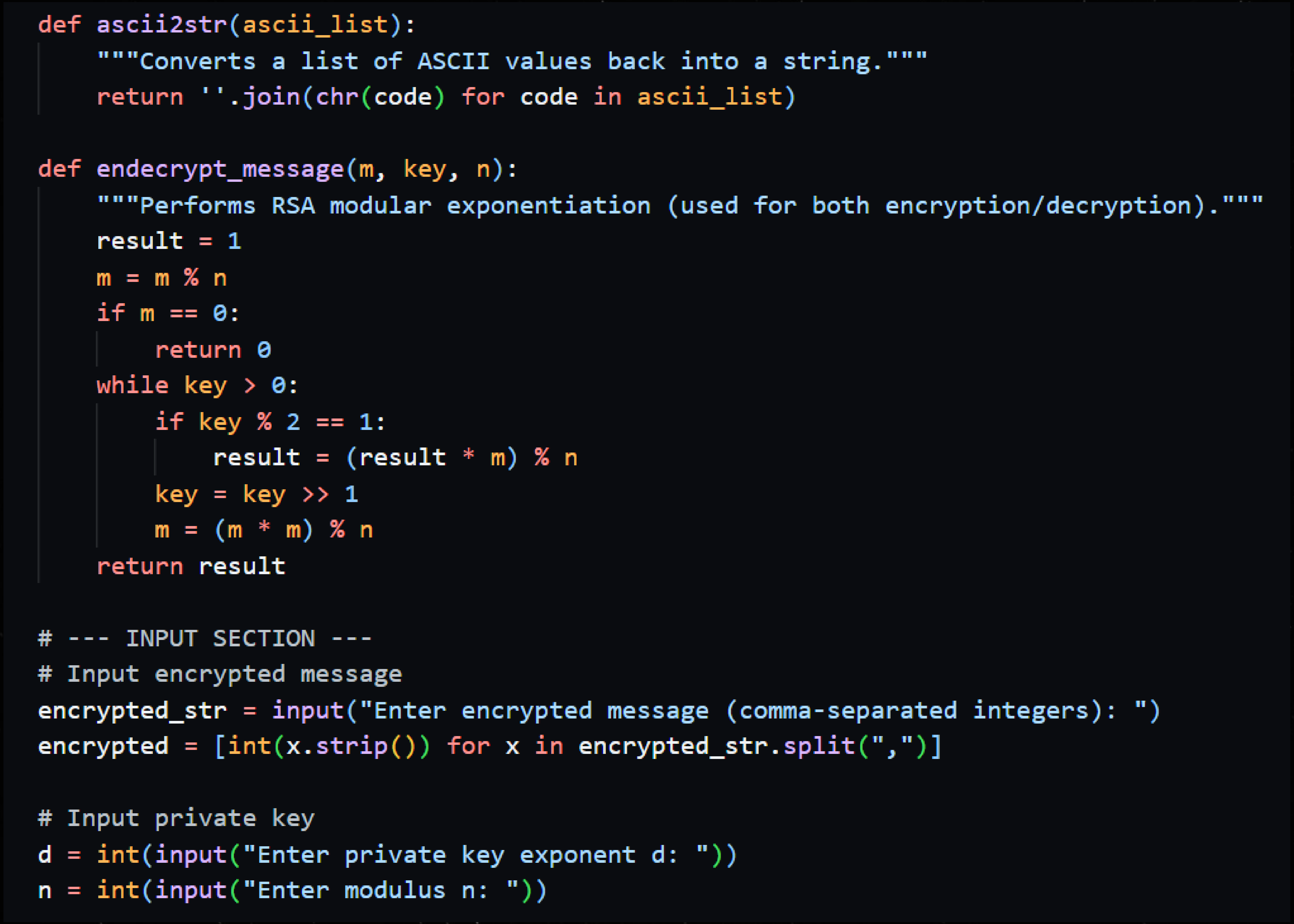
RSA Algorithm



```python
p, q, e, d, n = generate_keys(bit_length)
cipher = encrypt(ascii_msg, e, n)

# Save to file
with open("rsa_encrypted_data.json", "w") as f:
    json.dump({
        "ciphertext": cipher,
        "public_key": {"e": e, "n": n},
        "private_key": {"d": d, "n": n},
        "primes": {"p": p, "q": q},
        "original_ascii": ascii_msg
    }, f, indent=2)

# --- Display Info ---
print("\n🔑 Keys Generated:")
print(f"  p = {p}")
print(f"  q = {q}")
print(f"  n = {n}")
print(f"  e (public exponent) = {e}")
print(f"  d (private exponent) = {d}")

print(f"\n🔐 Public Key: (e={e}, n={n})")
print(f"🔏 Private Key: (d={d}, n={n})")

print(f"\n📨 Original Message: '{message}'")
print(f"🧾 ASCII Encoding: {ascii_msg}")
print(f"🔒 Encrypted Ciphertext: {cipher}")
print(f"\n✅ Data saved to 'rsa_encrypted_data.json'")
```

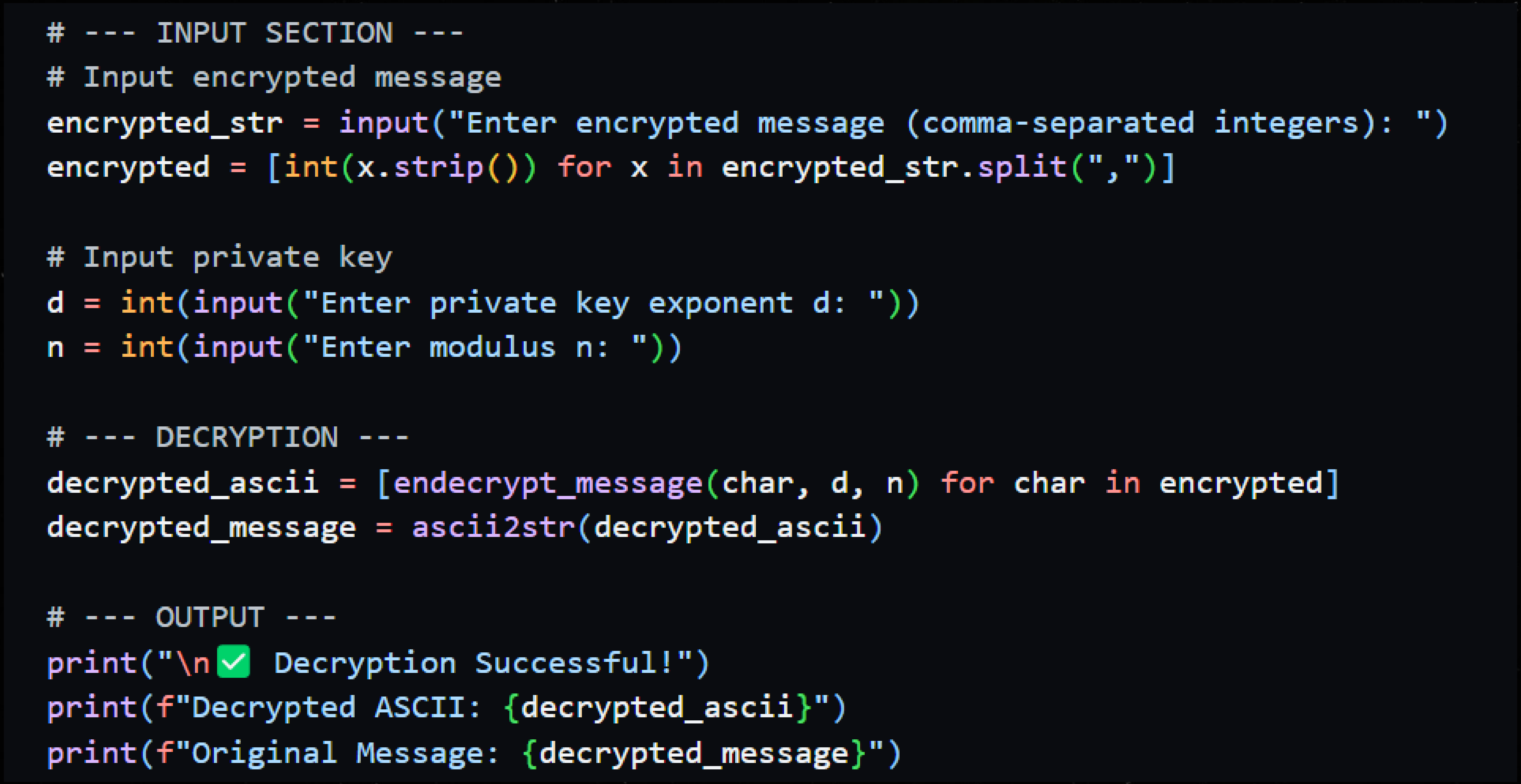
RSA Algorithm

Decryption:



```
def ascii2str(ascii_list):
    """Converts a list of ASCII values back into a string."""
    return ''.join(chr(code) for code in ascii_list)

def endecrypt_message(m, key, n):
    """Performs RSA modular exponentiation (used for both encryption/decryption)."""
    result = 1
    m = m % n
    if m == 0:
        return 0
    while key > 0:
        if key % 2 == 1:
            result = (result * m) % n
        key = key >> 1
        m = (m * m) % n
    return result

# --- INPUT SECTION ---
# Input encrypted message
encrypted_str = input("Enter encrypted message (comma-separated integers): ")
encrypted = [int(x.strip()) for x in encrypted_str.split(",")]

# Input private key
d = int(input("Enter private key exponent d: "))
n = int(input("Enter modulus n: "))
```

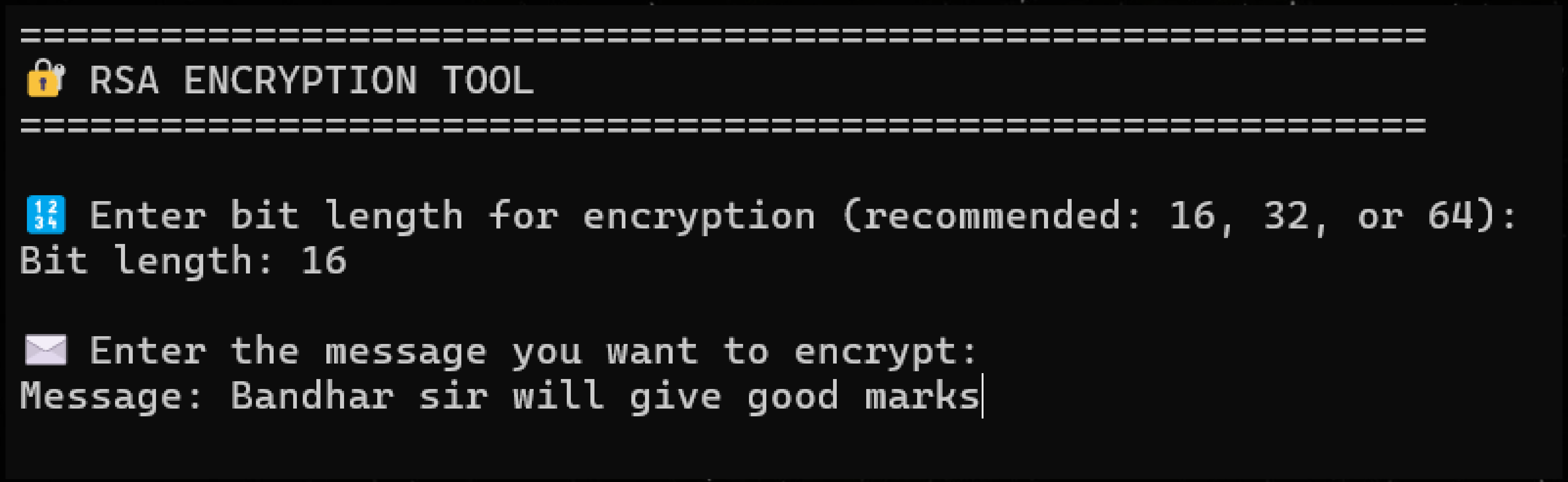
RSA Algorithm



```
# --- INPUT SECTION ---
# Input encrypted message
encrypted_str = input("Enter encrypted message (comma-separated integers): ")
encrypted = [int(x.strip()) for x in encrypted_str.split(",")]

# Input private key
d = int(input("Enter private key exponent d: "))
n = int(input("Enter modulus n: "))

# --- DECRYPTION ---
decrypted_ascii = [endecrypt_message(char, d, n) for char in encrypted]
decrypted_message = ascii2str(decrypted_ascii)

# --- OUTPUT ---
print("\n✅ Decryption Successful!")
print(f"Decrypted ASCII: {decrypted_ascii}")
print(f"Original Message: {decrypted_message}")
```

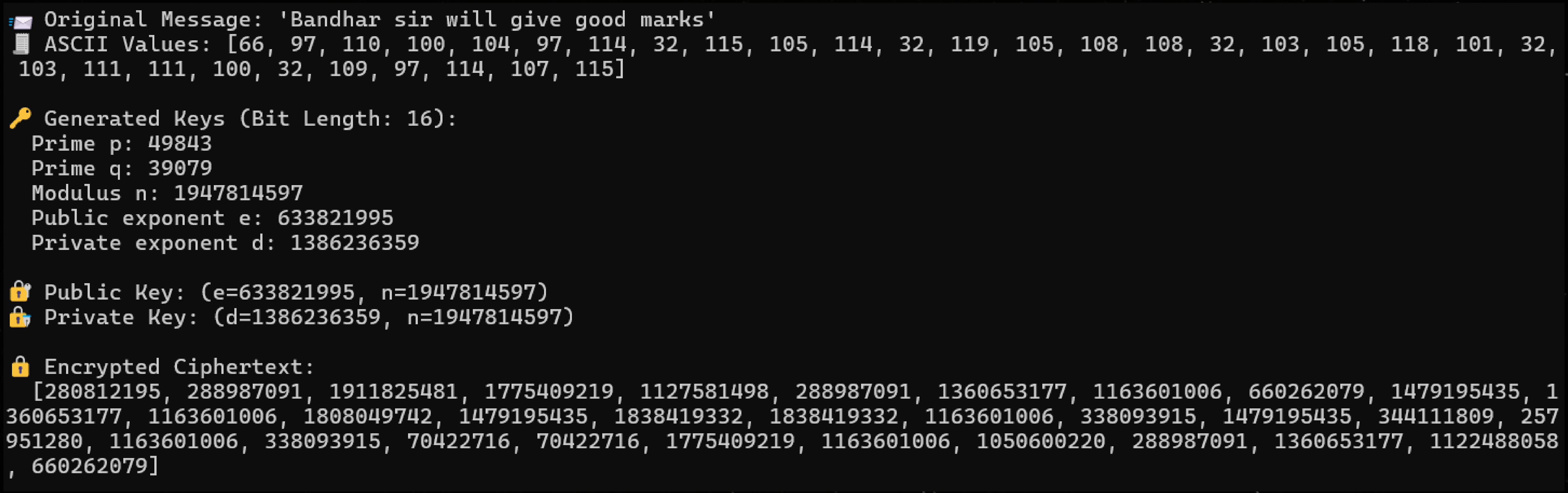
Example of Encryption

Input:



```
============================================================
🔐 RSA ENCRYPTION TOOL
============================================================

🔢 Enter bit length for encryption (recommended: 16, 32, or 64):
Bit length: 16

✉️ Enter the message you want to encrypt:
Message: Bandhar sir will give good marks
```

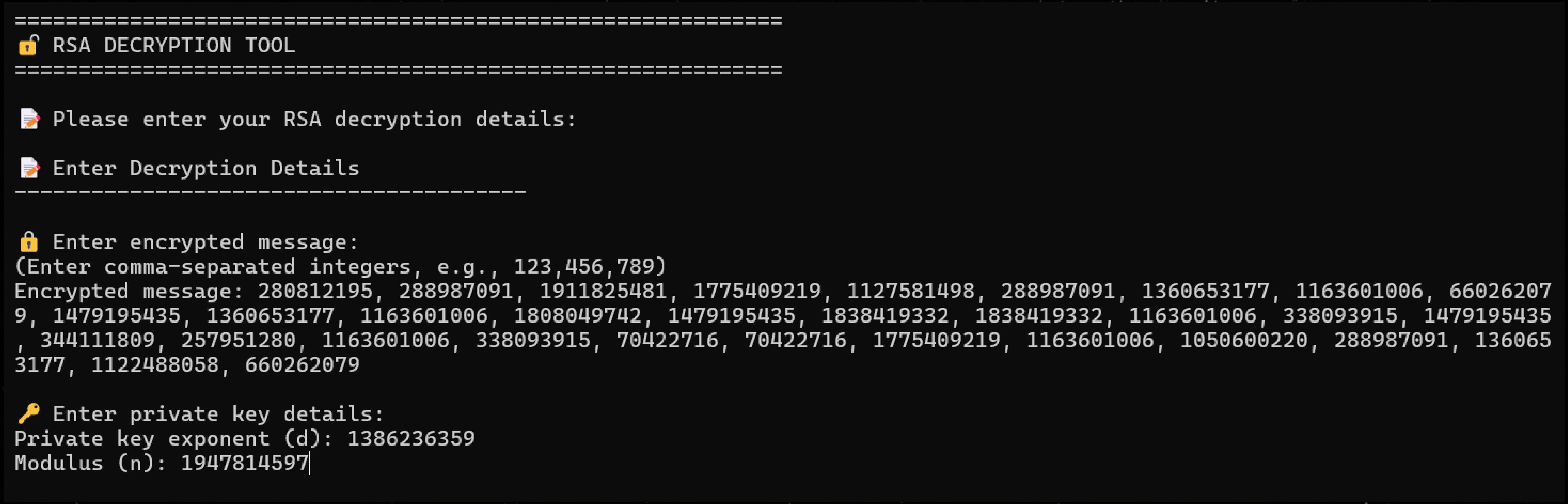
Output:



```
📨 Original Message: 'Bandhar sir will give good marks'
🧾 ASCII Values: [66, 97, 110, 100, 104, 97, 114, 32, 115, 105, 114, 32, 119, 105, 108, 108, 32, 103, 105, 118, 101, 32,
 103, 111, 111, 100, 32, 109, 97, 114, 107, 115]

🔑 Generated Keys (Bit Length: 16):
  Prime p: 49843
  Prime q: 39079
  Modulus n: 1947814597
  Public exponent e: 633821995
  Private exponent d: 1386236359

🔐 Public Key: (e=633821995, n=1947814597)
🔓 Private Key: (d=1386236359, n=1947814597)

🔒 Encrypted Ciphertext:
  [280812195, 288987091, 1911825481, 1775409219, 1127581498, 288987091, 1360653177, 1163601006, 660262079, 1479195435, 1
360653177, 1163601006, 1808049742, 1479195435, 1838419332, 1838419332, 1163601006, 338093915, 1479195435, 344111809, 257
951280, 1163601006, 338093915, 70422716, 70422716, 1775409219, 1163601006, 1050600220, 288987091, 1360653177, 1122488058
, 660262079]
```

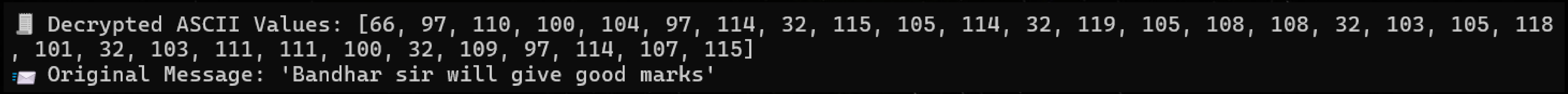
Example of Decryption

Input:



```
============================================================
🔓 RSA DECRYPTION TOOL
============================================================

📝 Please enter your RSA decryption details:

📝 Enter Decryption Details
----------------------------------------

🔒 Enter encrypted message:
(Enter comma-separated integers, e.g., 123,456,789)
Encrypted message: 280812195, 288987091, 1911825481, 1775409219, 1127581498, 288987091, 1360653177, 1163601006, 66026207
9, 1479195435, 1360653177, 1163601006, 1808049742, 1479195435, 1838419332, 1838419332, 1163601006, 338093915, 1479195435
, 344111809, 257951280, 1163601006, 338093915, 70422716, 70422716, 1775409219, 1163601006, 1050600220, 288987091, 136065
3177, 1122488058, 660262079

🔑 Enter private key details:
Private key exponent (d): 1386236359
Modulus (n): 1947814597
```

Output:



```
🗒 Decrypted ASCII Values: [66, 97, 110, 100, 104, 97, 114, 32, 115, 105, 114, 32, 119, 105, 108, 108, 32, 103, 105, 118
, 101, 32, 103, 111, 111, 100, 32, 109, 97, 114, 107, 115]
📨 Original Message: 'Bandhar sir will give good marks'
```

Symmetric Vs Asymmetric Cryptography

Feature	Symmetric Cryptography	Asymmetric Cryptography
🔑 Key Used	Same secret key for encryption and decryption	Public key for encryption, private key for decryption
🔄 Speed	Very fast and efficient for **large data**	Slower, due to complex mathematical operations
🧠 Complexity	Simple mathematical operations	Involves complex mathematical operations
📈 Uses	Ideal for Large continuous data like video streams or data transmissions	Ideal for key exchange or digital signatures etc.
🛡️ Security	Low Security due to risk of single key	High security due to 2 separate keys.
📚 Examples	AES, DES, Blowfish etc.	RSA, ECC etc.

Conclusion

- **Protects sensitive data:** Ensures security of telemetry, video feeds, and control signals.
- **Symmetric encryption:** Fast & efficient for large, real-time data like video and sensor streams.
- **Asymmetric encryption:** Used for secure key exchange, authentication, and access control.

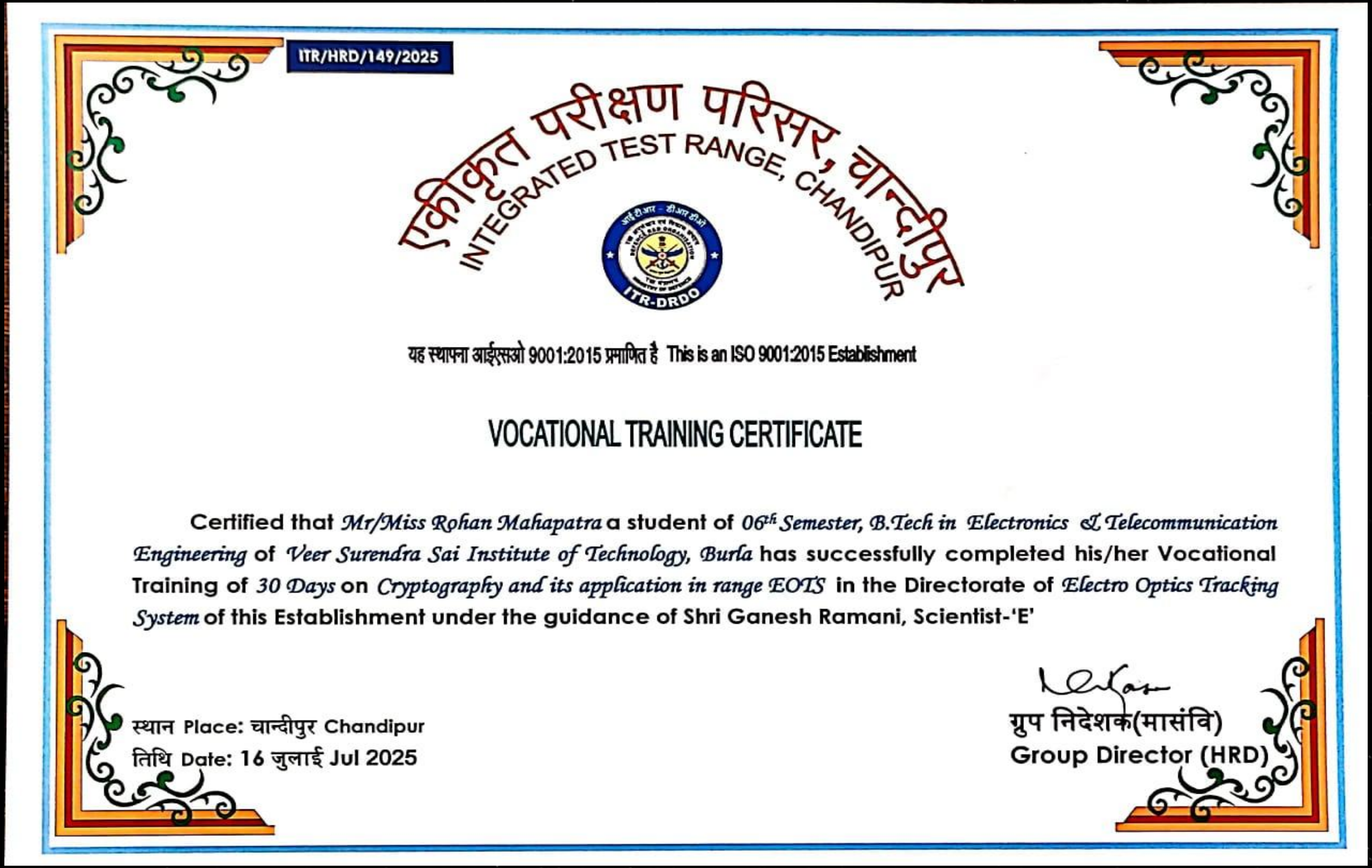
Hybrid approach:

- Asymmetric → for exchanging session key
- Symmetric → for efficient data encryption
- **Ensures**: Confidentiality, integrity, authenticity of range data.

References

- Christof Paar and Jan Pelzl, *Understanding Cryptography* – A Textbook for Students and Practitioners, First Edition, Springer, pp. 55-75, 2010
- William Stallings, *Cryptography and Network Security*
- Ganesh Ramani, Niraj Kumar, B K Das, *Comprehensive Multimedia Encryption System Generalization and Comparison*
- Alfred Menezes, Paul van Oorschot, and Scott Vanstone, *Handbook of Applied Cryptography*
- Bruce Schneier, *Applied Cryptography - Protocols, Algorithms, and Source Code in C*, 2nd Edition, Wiley, pp. 120–145, 1996
- Niels Ferguson, Bruce Schneier, and Tadayoshi Kohno, *Cryptography Engineering – Design Principles and Practical Applications*, 1st Edition, Wiley, pp. 60–95, 2010
- Jean-Philippe Aumasson, Serious Cryptography – *A Practical Introduction to Modern Encryption*, 1st Edition, No Starch Press, pp. 200–235, 2017
- *Symmetric Key Cryptography* - GeeksforGeeks

Certificate



ITR/HRD/149/2025

एकीकृत परीक्षण परिसर, चान्दीपुर

INTEGRATED TEST RANGE, CHANDIPUR

ITR-DRDO

यह स्थापना आईएसओ 9001:2015 प्रमाणित है This is an ISO 9001:2015 Establishment

VOCATIONAL TRAINING CERTIFICATE

Certified that *Mr/Miss Rohan Mahapatra* a student of *06th Semester, B.Tech in Electronics & Telecommunication Engineering* of *Veer Surendra Sai Institute of Technology, Burla* has successfully completed his/her Vocational Training of *30 Days* on *Cryptography and its application in range EOTS* in the Directorate of *Electro Optics Tracking System* of this Establishment under the guidance of Shri Ganesh Ramani, Scientist-'E'

स्थान Place: चान्दीपुर Chandipur
तिथि Date: 16 जुलाई Jul 2025

ग्रुप निदेशक(मासंवि)
Group Director (HRD)

Thank You