```
# RSA Encryption & Homomorphic Property Implementation
# Google Colab Compatible

def mod_inverse(e, phi):
    """Find modular inverse using Extended Euclidean Algorithm"""
    for d in range(3, phi):
        if (d * e) % phi == 1:
            return d
    return None

def encrypt(m, e, n):
    """Encrypt message using public key"""
    return pow(m, e, n)

def decrypt(c, d, n):
    """Decrypt ciphertext using private key"""
    return pow(c, d, n)

def main():
    print("=" * 60)
    print("RSA Encryption & Homomorphic Property Demonstration")
    print("=" * 60)

    # Step 1: Generate RSA key pair
    print("\n1. GENERATING RSA KEY PAIR")
    print("-" * 30)

    p = 11
    q = 13
    print(f"p = {p}, q = {q}")

    n = p * q
    phi = (p - 1) * (q - 1)
    print(f"n = {n}, ϕ = {phi}")

    e = 7
    d = mod_inverse(e, phi)

    print(f"Public key (e, n) = ({e}, {n})")
    print(f"Private key (d, n) = ({d}, {n})")

    # Step 2: Encrypt messages
    print("\n2. ENCRYPTING MESSAGES")
    print("-" * 30)

    m1 = 5
    m2 = 9
    print(f"m1 = {m1}, m2 = {m2}")

    c1 = encrypt(m1, e, n)
    c2 = encrypt(m2, e, n)

    print(f"E(m1) = {c1}, E(m2) = {c2}")

    # Step 3: Demonstrate homomorphic property
    print("\n3. HOMOMORPHIC PROPERTY VERIFICATION")
    print("-" * 30)

    ciphertext_product = (c1 * c2) % n
    print(f"E(m1)*E(m2) mod n = {ciphertext_product}")

    decrypted_result = decrypt(ciphertext_product, d, n)
    print(f"Decrypted result = {decrypted_result}")

    expected_result = (m1 * m2) % n
    print(f"Expected (m1*m2 mod n) = {expected_result}")

    # Step 4: Final verification
    print("\n4. FINAL VERIFICATION")
    print("-" * 30)

    if decrypted_result == expected_result:
        print("Homomorphic property verified: E(m1)E(m2) ≡ E(m1*m2) mod n")
    else:
        print("Homomorphic property verification failed")

# Run the demonstration
if __name__ == "__main__":
    main()
```

```
============================================================
RSA Encryption & Homomorphic Property Demonstration
============================================================

1. GENERATING RSA KEY PAIR
------------------------------
p = 11, q = 13
n = 143, ϕ = 120
Public key (e, n) = (7, 143)
Private key (d, n) = (103, 143)

2. ENCRYPTING MESSAGES
------------------------------
m1 = 5, m2 = 9
E(m1) = 47, E(m2) = 48

3. HOMOMORPHIC PROPERTY VERIFICATION
------------------------------
E(m1)*E(m2) mod n = 111
Decrypted result = 45
Expected (m1*m2 mod n) = 45

4. FINAL VERIFICATION
------------------------------
Homomorphic property verified: E(m1)E(m2) ≡ E(m1*m2) mod n
```