RSA (Rivest-Shamir-Adleman) is a widely used public-key cryptography algorithm, which is based on the mathematical properties of large prime numbers.

The algorithm involves the following steps:

1. Key Generation:

- Select two large prime numbers, p and q (for example, p = 61 and q = 53)
- Compute n = pq (n = 61 * 53 = 3233)
- Compute φ(n) = (p-1) * (q-1) = (61-1) * (53-1) = 3120
- Select an integer e such that 1 < e < φ(n) and e is coprime to φ(n) (for example, e = 17)
- Compute d such that d ≡ e^{-1} (mod φ(n)) (i.e., d is the modular inverse of e with respect to φ(n)).
- The public key is (n, e) and the private key is (n, d)

2. Encryption:
   To encrypt a message, m, the sender first represents the message as a number between 0 and n-1. Let's say the message is 42.

- The sender then computes the ciphertext, c, as c ≡ m^e (mod n) (for example, c = 42^17 % 3233 = 811)

3. Decryption:
   To decrypt the message, the recipient computes the original message, m, as m ≡ c^d (mod n) (for example, m = 811^2753 % 3233 = 42)

This example demonstrates the basic idea behind RSA encryption and decryption. In practice, much larger prime numbers and more secure methods for key generation are used.

DS

Show that RSA encryption is inverse process of decryption

In RSA, encryption and decryption are inverse operations because they use the same modulus, n, and the encryption exponent, e, is the modular inverse of the decryption exponent, d.

Consider a message, m, that we want to encrypt and decrypt. The encryption process computes the ciphertext, c, as:

c ≡ m^e (mod n)

And the decryption process computes the original message, m', as:

m' ≡ c^d (mod n)

Since e and d are the modular inverse of each other with respect to φ(n), we have:

ed ≡ 1 (mod φ(n))

Therefore, we can rewrite the decryption equation as:

m' ≡ c^d (mod n)
≡ (m^e)^d (mod n) (substituting c = m^e)
≡ m^(ed) (mod n)
≡ m^1 (mod n)
≡ m (mod n)

Thus, we have m' = m, which shows that decryption is the inverse of encryption.