1. Cryptographic Operations

- generate_prime_candidate(length): Generates an odd integer of a specified bit length by setting the most significant bit (MSB) and the least significant bit (LSB) to 1 to ensure it's odd and has the desired length.
- is_prime(n, k): Implements the Miller-Rabin primality test to check if a number n is prime, with k iterations for accuracy. It's used to find prime numbers that are secure for cryptography.
- generate_prime_number(length): Generates a prime number of a given bit length using the is_prime function to ensure it's suitable for cryptographic use.
- gcd(a, b): Computes the greatest common divisor of a and b, used in the key generation process to ensure that the chosen encryption exponent e is coprime with φ(n) (phi of n).
- multiplicative_inverse(e, phi): Finds the multiplicative inverse of e modulo phi, necessary for generating the private decryption key d in RSA cryptography.
- generate_keys(): Generates a public-private key pair for RSA encryption. It chooses two prime numbers p and q, calculates n = p*q and φ(n) = (p-1)*(q-1), selects an encryption exponent e, and computes the decryption exponent d.

2. Utility Functions for Encoding and Decoding

- Attach(objects): Combines various objects (strings, integers, tuples) into a single string with specified delimiters, facilitating the encryption and transmission of complex data structures.
- Detach(s): Reverses the process of attach, breaking down a string into its constituent components based on the delimiters, aiding in the decryption and interpretation of received messages.

3. Encryption and Decryption

- Encrypt(pk, plaintext): Encrypts a plaintext message with a public key pk, converting each character to an integer and raising it to the power of the encryption key modulo n.
- Decrypt(pk, ciphertext): Decrypts a ciphertext with a private key pk, converting each integer back to its character representation in the plaintext.

4. PKDA Server and Client Classes

PKDA Class:

Maintains a record of client public keys and provides a method to register clients.
Offers a service to encrypt client public keys for secure distribution to other clients.

Client Class:

Each client has a unique ID, a pair of RSA keys, and methods to interact with the PKDA and other clients.
Clients can request and store public keys of other clients securely from the PKDA.
Supports sending and receiving encrypted messages, utilizing nonce values for added security and to prevent replay attacks.

5. Simulation Functions

- simulate(pkda, initiator, responder): A function to demonstrate the process of key exchange and initial communication between two clients via the PKDA.
- simulate_msg(client_a, client_b): Shows how clients can exchange messages securely after obtaining each other's public keys, highlighting the encryption and decryption process in communication.

6. Main Flow

- Initializes the PKDA and registers clients.
- Demonstrates secure communication between clients "A" and "B", including key exchange and message exchange processes.
- Introduces a third client "C" to showcase the flexibility of the system in adding new clients and facilitating secure communication.
- This implementation showcases the essentials of a Public Key Infrastructure (PKI), focusing on key distribution, secure message exchange, and cryptographic principles underlying secure digital communications.