Name: Gaurang Vaghela
Rollno: TEAD-22561
Mini Project Lab
Practical 4
Problem Statement: Implementation of ECC Algorithm

Code:

```
from Crypto.PublicKey import ECC
from Crypto.Util.number import bytes_to_long, long_to_bytes

# Generate a private key
private_key = ECC.generate(curve='P-256')
public_key = private_key.public_key()

# Encrypt a message
message = b'Hello, world!'
plaintext = bytes_to_long(message)

# Generate a random k
k = ECC.generate(curve='P-256').d

# Compute C1 = k * G (base point of the curve)
C1 = k * ECC._curves["P-256"].G  # Use base point G instead of private key's
pointQ

# Compute the shared secret using the recipient's public key
shared_secret = int((public_key.pointQ * k).x)  # Ensure it’s an integer

# Encrypt message
C2 = plaintext ^ shared_secret  # XOR with shared secret

# Decryption
decrypted_secret = int((C1 * private_key.d).x)  # Correct shared secret calculation
decrypted_plaintext = C2 ^ decrypted_secret  # Perform XOR

# Convert back to bytes
decrypted_message = long_to_bytes(decrypted_plaintext)

print('Original message:', message)
print('Decrypted message:', decrypted_message)
```

Output:

```
Original message: b'Hello, world!'
Decrypted message: b'Hello, world!'
```