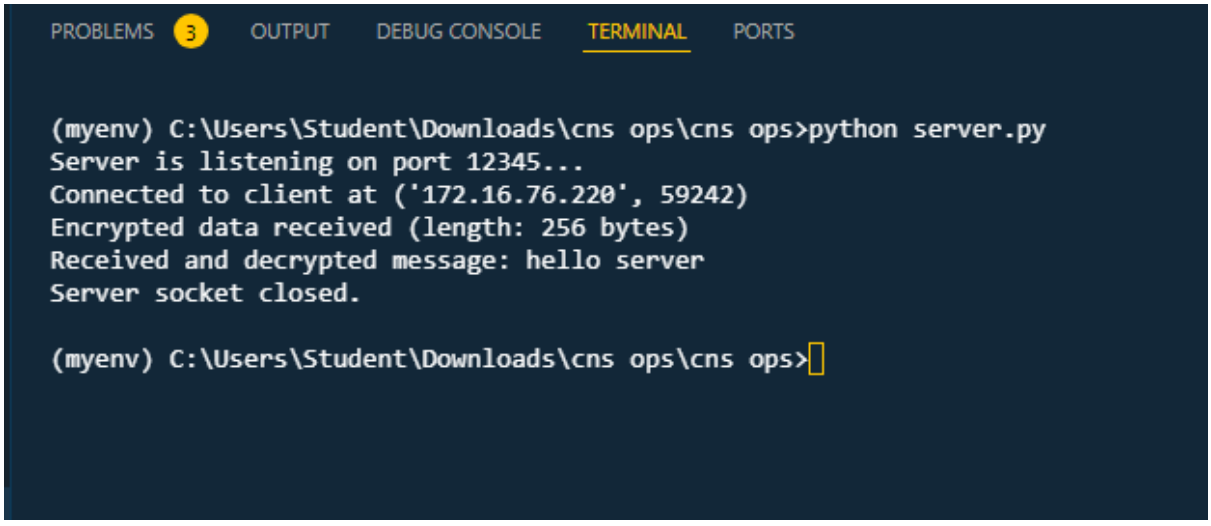
Generate_keys.py

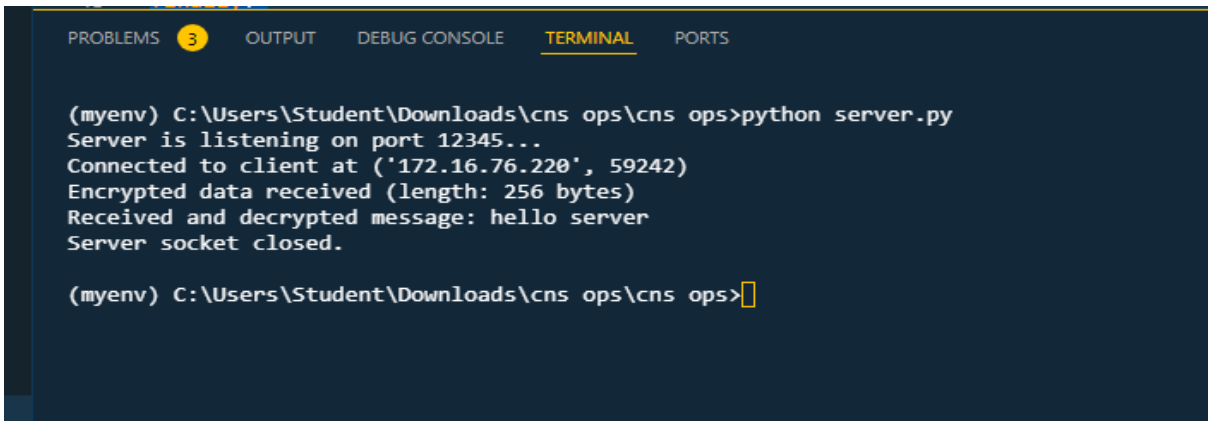
```python
import rsa

# Generate RSA key pair
(public_key, private_key) = rsa.newkeys(2048)

# Save the public key to a PEM file
with open("server_public.pem", "wb") as pub_file:
    pub_file.write(public_key.save_pkcs1("PEM"))

# Save the private key to a PEM file
with open("server_private.pem", "wb") as priv_file:
    priv_file.write(private_key.save_pkcs1("PEM"))

print("Keys generated and saved.")
```



Server.py

```python
import socket
import rsa

# Constants
PORT = 12345

# Load RSA private key
try:
    with open("server_private.pem", "rb") as priv_file:
        private_key = rsa.PrivateKey.load_pkcs1(priv_file.read())
except FileNotFoundError:
    print("Error: server_private.pem not found.")
    exit(1)
```

```python
# Setup server socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', PORT))
server_socket.listen(1)
print(f"Server is listening on port {PORT}...")

try:
    # Accept client connection
    client_socket, client_addr = server_socket.accept()
    print(f"Connected to client at {client_addr}")

    # Receive encrypted data from client
    data = client_socket.recv(4096)

    if data:
        print(f"Encrypted data received (length: {len(data)} bytes)")

        try:
            # Attempt to decrypt
            decrypted_message = rsa.decrypt(data, private_key).decode('utf-8')
            print(f"Received and decrypted message: {decrypted_message}")
        except rsa.DecryptionError:
            print("❌ Decryption failed: Possibly wrong key or corrupted
data.")
        except Exception as e:
            print(f"❌ Unexpected error during decryption: {e}")
    else:
        print("⚠ No data received from client.")

finally:
    # Close client and server sockets
    client_socket.close()
    server_socket.close()
    print("Server socket closed.")
```



Client.py

```
import socket
import rsa

# Constants
PORT = 12345
MAX_MESSAGE_LENGTH = 245  # Max for 2048-bit RSA

# Load server's public key
try:
    with open("server_public.pem", "rb") as pub_file:
        public_key = rsa.PublicKey.load_pkcs1(pub_file.read())
except FileNotFoundError:
    print("Error: server_public.pem not found.")
    exit(1)

# Create client socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Get server IP address from user
server_ip = input("Enter server IP address: ").strip()

try:
    # Connect to the server
    client_socket.connect((server_ip, PORT))
except ConnectionRefusedError:
    print(f"Could not connect to {server_ip}:{PORT}. Is the server running?")
    exit(1)

# Get message from user
message = input("Enter message to send (max 245 characters): ")

if len(message.encode('utf-8')) > MAX_MESSAGE_LENGTH:
    print("Error: Message too long for RSA encryption (max 245 bytes).")
    client_socket.close()
    exit(1)

# Encrypt the message using the server's public key
try:
    encrypted_message = rsa.encrypt(message.encode('utf-8'), public_key)
except Exception as e:
    print("Encryption failed:", str(e))
    client_socket.close()
    exit(1)

# Send the encrypted message
client_socket.sendall(encrypted_message)
```

```python
print("Encrypted message sent successfully.")

# Close the socket
client_socket.close()
```