Computer Networks Lab Assessment 3

Suryakumar P 21MIS1146

(a) Write a program to implement a simple message transfer from client to server using UDP sockets.

Problem Definition:

To write a Java Code for implementation of simple Message Transfer using UDP Sockets.

Method:

Client Side:

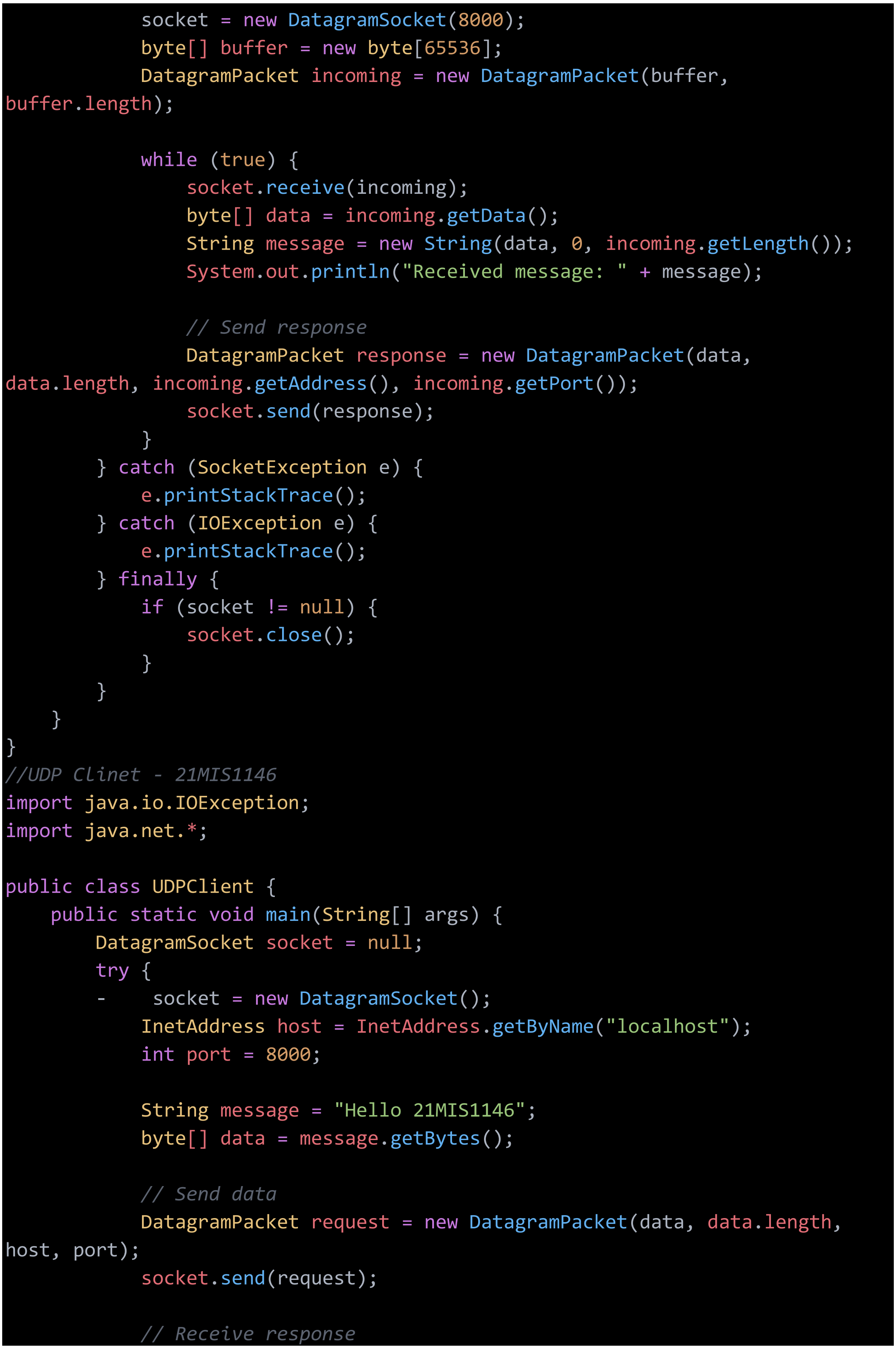
1. The client program prompts the user to enter a message to be sent to the server.
2. The client creates a UDP socket for communication.
3. The client converts the message to bytes and creates a UDP packet containing the message and the server's address and port.
4. The client sends the UDP packet to the server using the UDP socket.

Server Side:

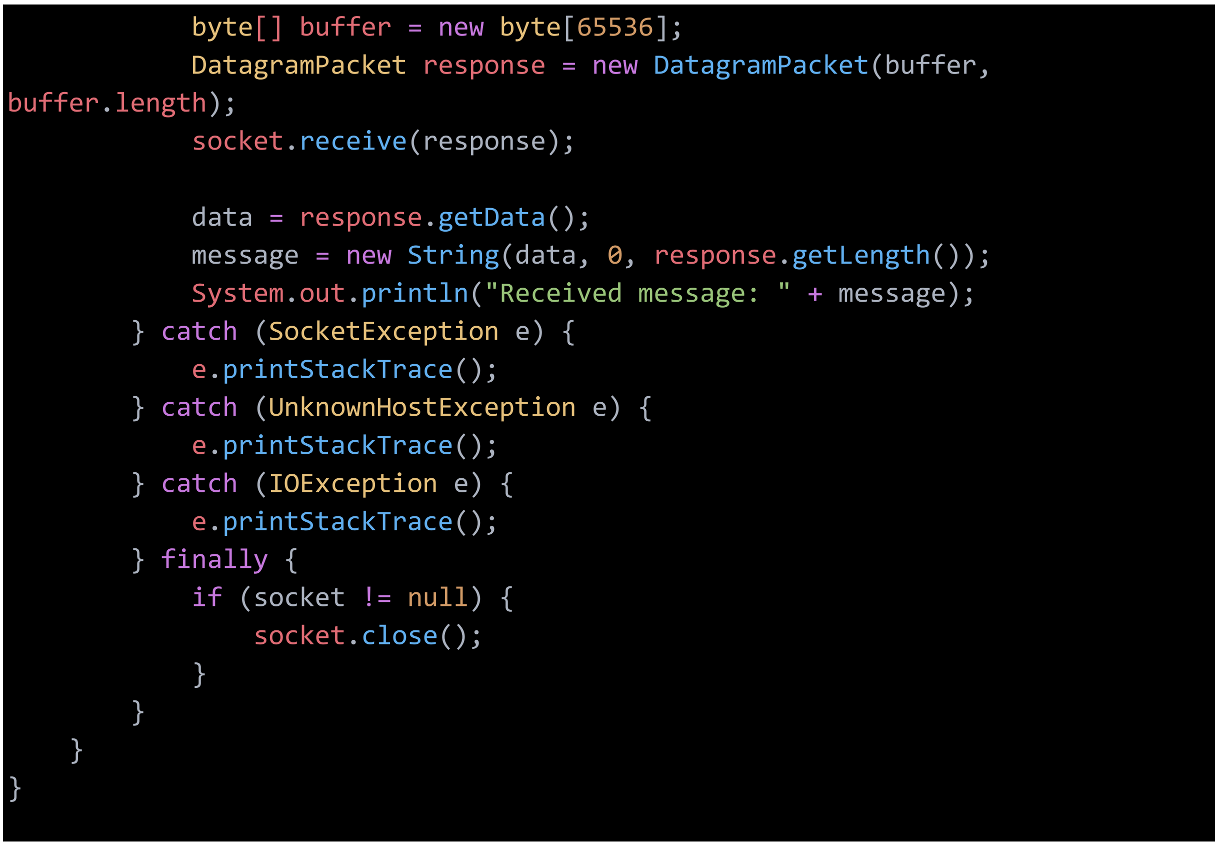
5. The server program creates a UDP socket and binds it to a specific port to listen for incoming messages.
6. The server waits to receive a UDP packet from the client.
7. When a UDP packet is received, the server extracts the message from the packet.
8. The server processes the received message as needed (e.g., displays it, performs operations, etc.).
9. The server can optionally send a response message back to the client.
10. The server and client can continue the communication loop until a termination condition is met.

CODE:

```
//UDP Server - 21MIS1146
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;

public class UDPServer {
    public static void main(String[] args) {
        DatagramSocket socket = null;
        try {
```

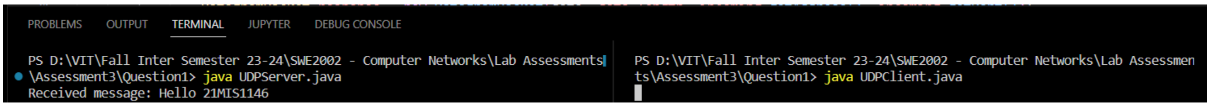


```
            socket = new DatagramSocket(8000);
            byte[] buffer = new byte[65536];
            DatagramPacket incoming = new DatagramPacket(buffer,
buffer.length);

            while (true) {
                socket.receive(incoming);
                byte[] data = incoming.getData();
                String message = new String(data, 0, incoming.getLength());
                System.out.println("Received message: " + message);

                // Send response
                DatagramPacket response = new DatagramPacket(data,
data.length, incoming.getAddress(), incoming.getPort());
                socket.send(response);
            }
        } catch (SocketException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null) {
                socket.close();
            }
        }
    }
}
//UDP Clinet - 21MIS1146
import java.io.IOException;
import java.net.*;

public class UDPClient {
    public static void main(String[] args) {
        DatagramSocket socket = null;
        try {
        -    socket = new DatagramSocket();
            InetAddress host = InetAddress.getByName("localhost");
            int port = 8000;

            String message = "Hello 21MIS1146";
            byte[] data = message.getBytes();

            // Send data
            DatagramPacket request = new DatagramPacket(data, data.length,
host, port);
            socket.send(request);

            // Receive response
```



```
            byte[] buffer = new byte[65536];
            DatagramPacket response = new DatagramPacket(buffer,
buffer.length);
            socket.receive(response);

            data = response.getData();
            message = new String(data, 0, response.getLength());
            System.out.println("Received message: " + message);
        } catch (SocketException e) {
            e.printStackTrace();
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null) {
                socket.close();
            }
        }
    }
}
```

OUTPUT:



```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessments
\Assessment3\Question1> java UDPServer.java
Received message: Hello 21MIS1146
```

```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessmen
ts\Assessment3\Question1> java UDPClient.java
```

(b) Write a UDP-based server code to get the date of birth of the client and calculate the age as of today. The client has to enter, the year, month, and day of birth.

Problem Definition:

To write a UDP-based server code to get the date of birth of the client and calculate the age as of today. The client has to enter, the year, month, and day of birth.

Method:

Server Side:

1. The server initializes and binds to a specific port to listen for incoming UDP packets.
2. The server creates a DatagramSocket to handle UDP communication.

Client Side:

3. The client prompts the user to enter their year, month, and day of birth.
4. The client constructs a UDP packet containing the date of birth information.
5. The client sends the UDP packet to the server using a DatagramSocket.

Server Side:

6. The server receives the UDP packet containing the date of birth information.
7. The server extracts the year, month, and day of birth from the received packet.
8. The server calculates the age based on the current date and the provided date of birth.
9. The server prepares a response message containing the calculated age.
10. The server sends the response message back to the client using a DatagramSocket.

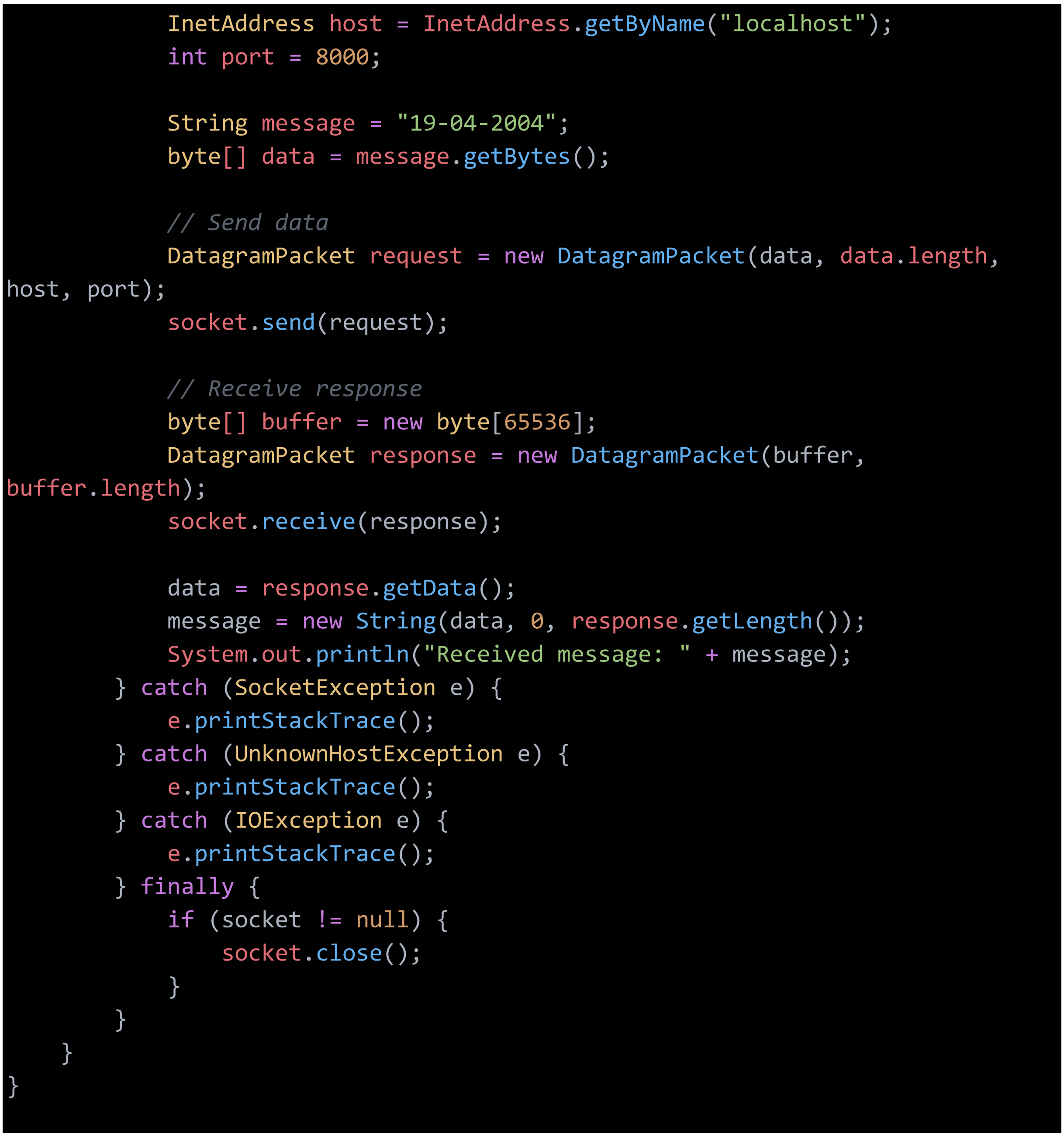
Client Side:

11. The client receives the response message from the server.
12. The client displays the received age calculated by the server.

CODE:

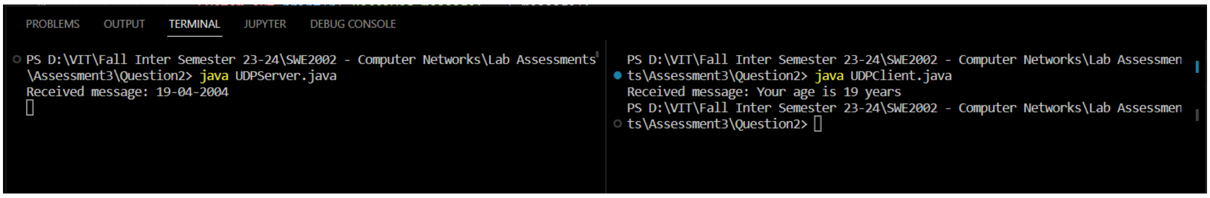
```
//UDP Server - 21MIS1146
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;
import java.time.LocalDate;
import java.time.Period;
import java.time.format.DateTimeFormatter;

public class UDPServer {
    public static void main(String[] args) {
        DatagramSocket socket = null;
        try {
            socket = new DatagramSocket(8000);
            byte[] buffer = new byte[65536];
            DatagramPacket incoming = new DatagramPacket(buffer,
buffer.length);

            while (true) {
                socket.receive(incoming);
                byte[] data = incoming.getData();
                String message = new String(data, 0, incoming.getLength());
                System.out.println("Received message: " + message);

                // Calculate age
                DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd-
MM-yyyy");
                LocalDate dob = LocalDate.parse(message, formatter);
                LocalDate now = LocalDate.now();
                Period period = Period.between(dob, now);
                int age = period.getYears();
                String responseMessage = "Your age is " + age + " years";

                // Send response
                byte[] responseData = responseMessage.getBytes();
                DatagramPacket response = new DatagramPacket(responseData,
responseData.length, incoming.getAddress(), incoming.getPort());
                socket.send(response);
            }
        } catch (SocketException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null) {
                socket.close();
            }
        }
    }
}
```

```
//UDP Client - 21MIS1146
import java.io.IOException;
import java.net.*;

public class UDPClient {
    public static void main(String[] args) {
        DatagramSocket socket = null;
        try {
            socket = new DatagramSocket();
```



```
            InetAddress host = InetAddress.getByName("localhost");
            int port = 8000;

            String message = "19-04-2004";
            byte[] data = message.getBytes();

            // Send data
            DatagramPacket request = new DatagramPacket(data, data.length,
host, port);
            socket.send(request);

            // Receive response
            byte[] buffer = new byte[65536];
            DatagramPacket response = new DatagramPacket(buffer,
buffer.length);
            socket.receive(response);

            data = response.getData();
            message = new String(data, 0, response.getLength());
            System.out.println("Received message: " + message);
        } catch (SocketException e) {
            e.printStackTrace();
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null) {
                socket.close();
            }
        }
    }
}
```

OUTPUT:



```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessments
\Assessment3\Question2> java UDPServer.java
Received message: 19-04-2004
```

```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessmen
ts\Assessment3\Question2> java UDPClient.java
Received message: Your age is 19 years
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessmen
ts\Assessment3\Question2>
```

(c) Write a UDP-based server code to broadcast a message to the nodes in a LAN

Problem Definition:

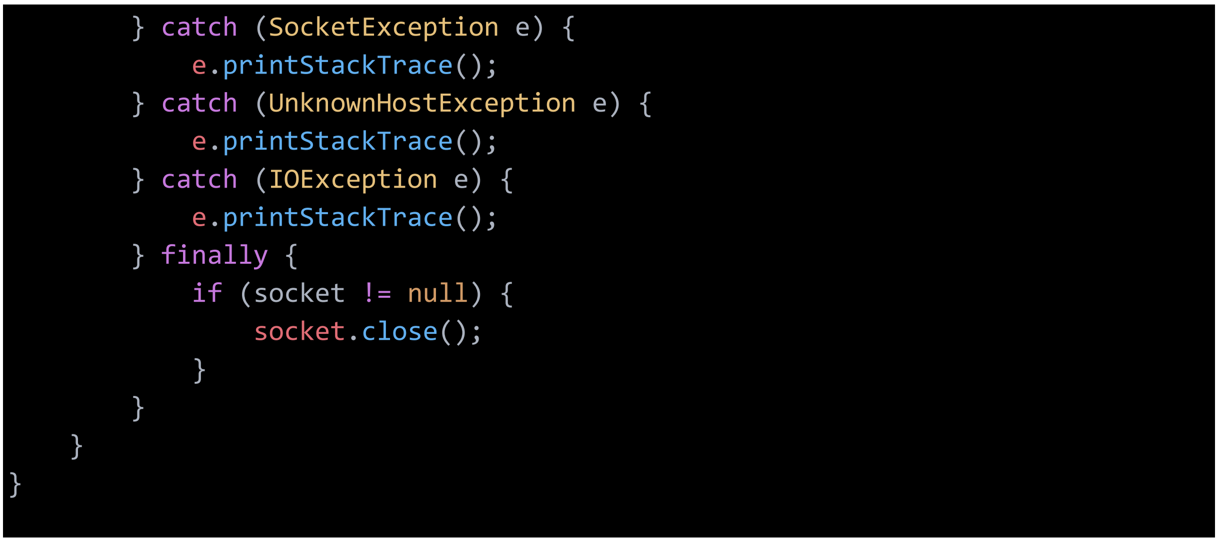
To write a Java Code for implementation of server code to broadcast a message to the nodes in a LAN.

Method:

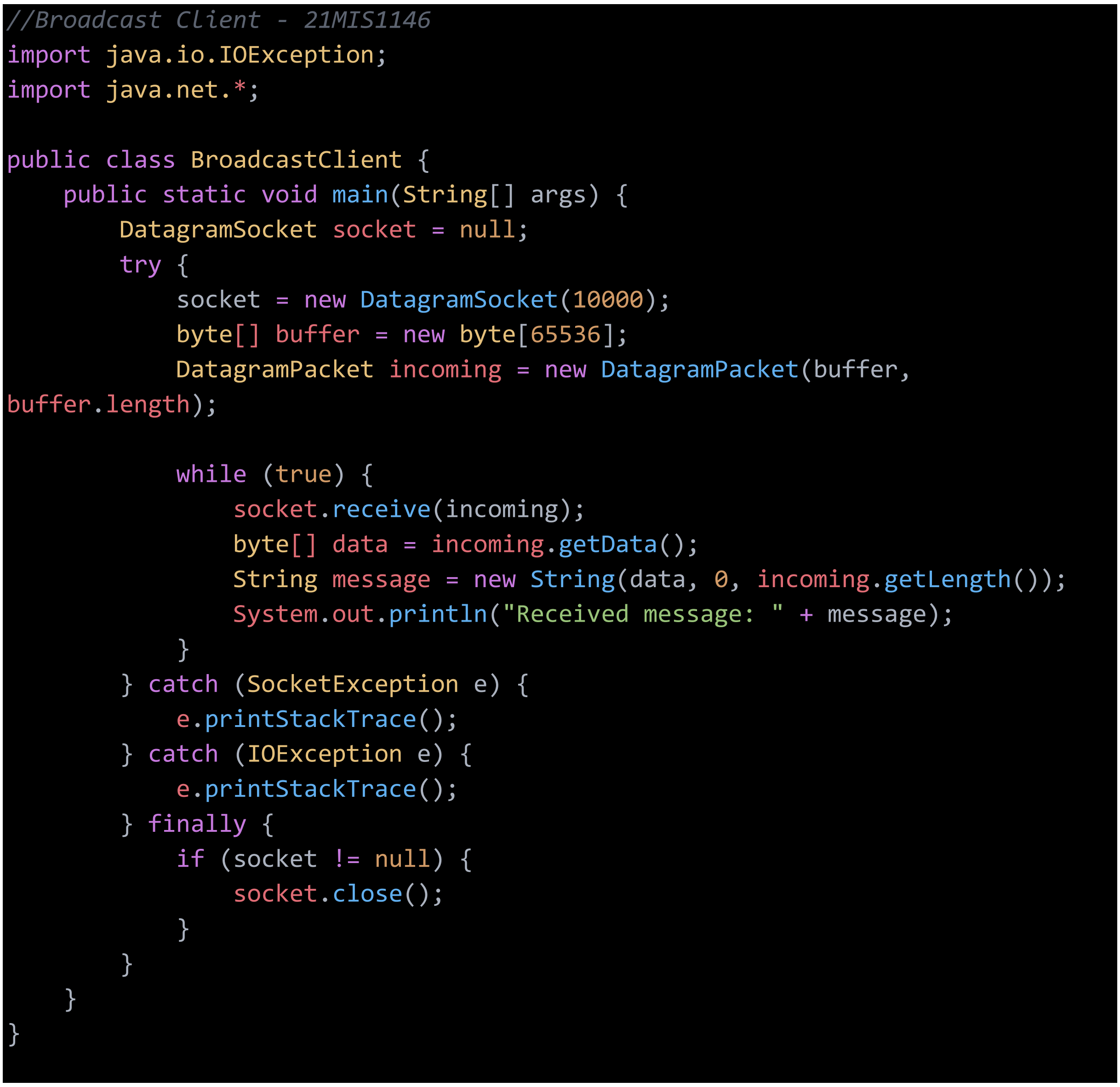
1. The server initializes and listens on a specific port to accept incoming connections.
2. Clients within the LAN establish connections to the server by specifying its IP address and port number.
3. The server keeps track of the connected clients or maintains a list of their socket connections.
4. When a client sends a message to the server, the server stores the message or the latest received message.
5. The server iterates through the list of connected clients.
6. For each client, the server sends the stored message or the new message using the client's socket connection.
7. Clients continuously listen for messages from the server using their respective socket connections.
8. When a message is received, the client can process or display it to the user.
9. This message broadcasting process continues until a termination condition is met.
10. Upon termination, the server closes its server socket, and each client closes its socket connection.

CODE:

```
//Broadcast Server - 21MIS1146
import java.io.IOException;
import java.net.*;
public class BroadcastServer {
    public static void main(String[] args) {
        DatagramSocket socket = null;
        try {
            socket = new DatagramSocket();
            socket.setBroadcast(true);

            String message = "This is a broadcast message";
            byte[] buffer = message.getBytes();

            InetAddress broadcastAddress =
InetAddress.getByName("255.255.255.255");
            DatagramPacket packet = new DatagramPacket(buffer, buffer.length,
broadcastAddress, 10000);
            socket.send(packet);
```

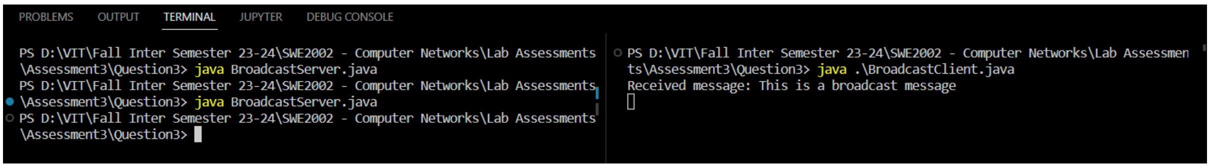


```java
        } catch (SocketException e) {
            e.printStackTrace();
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null) {
                socket.close();
            }
        }
    }
}
```



```java
//Broadcast Client - 21MIS1146
import java.io.IOException;
import java.net.*;

public class BroadcastClient {
    public static void main(String[] args) {
        DatagramSocket socket = null;
        try {
            socket = new DatagramSocket(10000);
            byte[] buffer = new byte[65536];
            DatagramPacket incoming = new DatagramPacket(buffer,
buffer.length);

            while (true) {
                socket.receive(incoming);
                byte[] data = incoming.getData();
                String message = new String(data, 0, incoming.getLength());
                System.out.println("Received message: " + message);
            }
        } catch (SocketException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (socket != null) {
                socket.close();
            }
        }
    }
}
```

OUTPUT:



```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessments
\Assessment3\Question3> java BroadcastServer.java
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessments
\Assessment3\Question3> java BroadcastServer.java
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessments
\Assessment3\Question3>
```

```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Assessmen
ts\Assessment3\Question3> java .\BroadcastClient.java
Received message: This is a broadcast message
```

(d) Write a code to implement border gateway protocol (BGP).

Problem Definition:

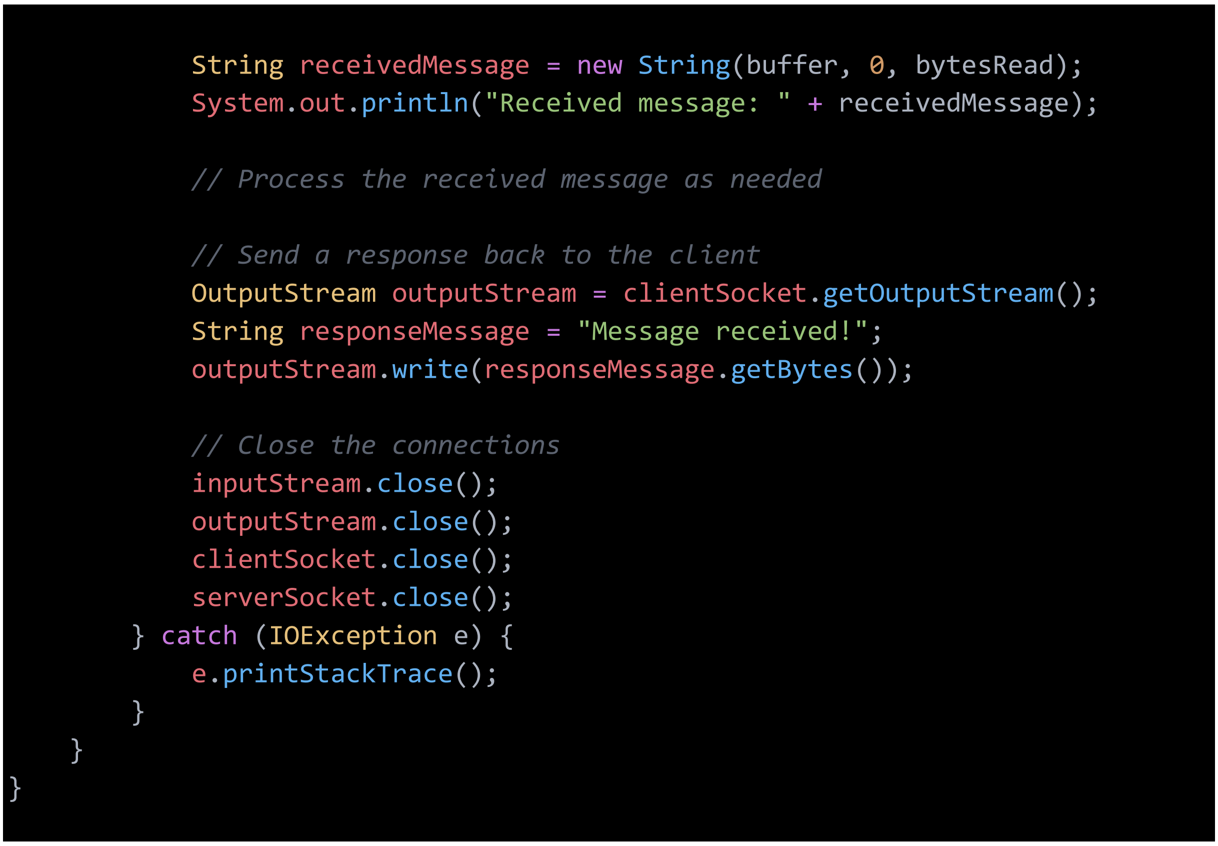
To write a Java Code for implementation of simple Message Transfer using Border Gateway Protocol (BGP).

Method:

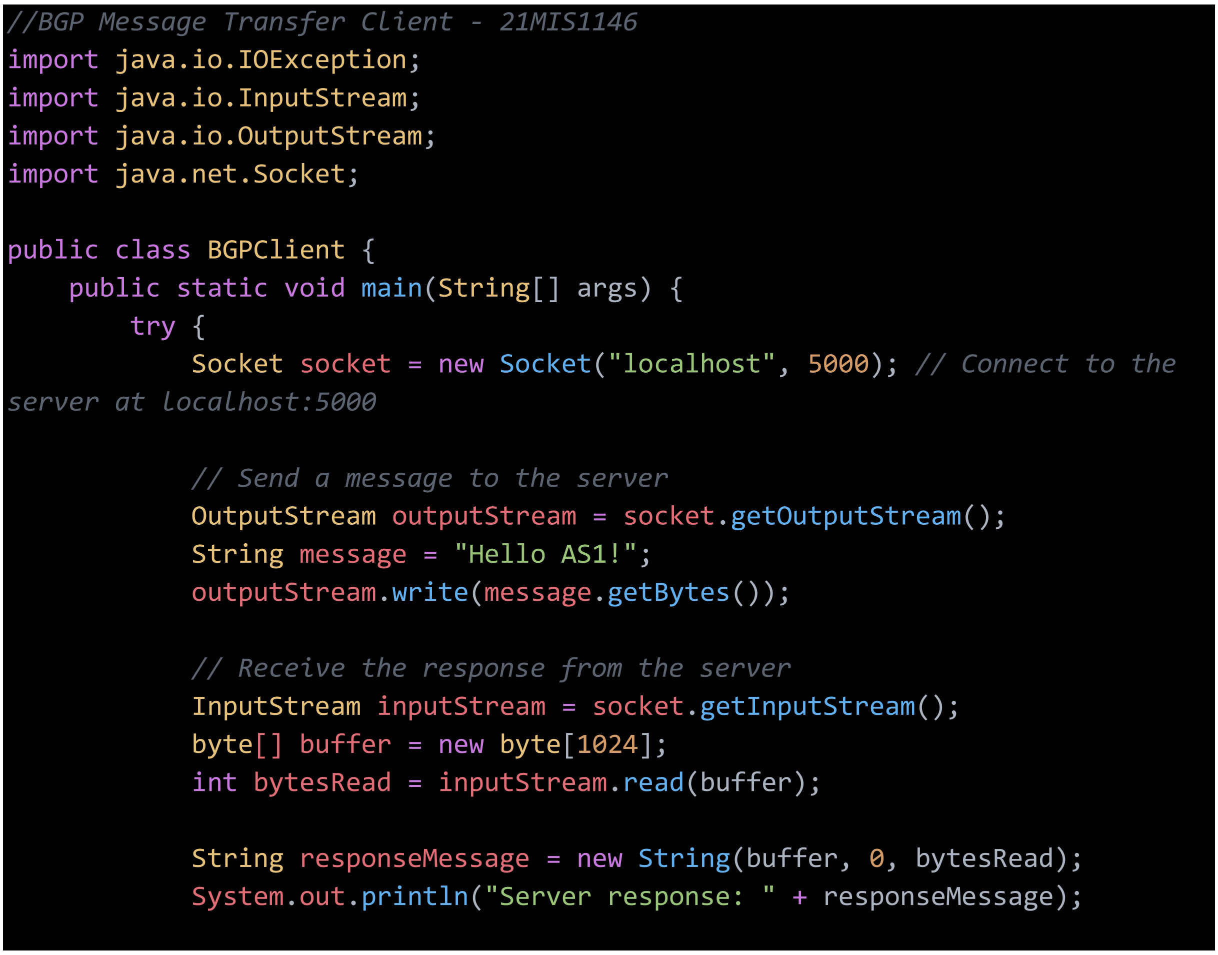
1. The server program listens for incoming connections on port 5000 using a ServerSocket.
2. When a client connects to the server, the server accepts the connection using the accept() method, which blocks until a connection is established.
3. The server retrieves the client's IP address and prints it.
4. The server sets up input and output streams to communicate with the client.
5. The server receives the message sent by the client using the input stream's read() method.
6. The server can process the received message as needed.
7. The server sends a response message back to the client using the output stream's write() method.
8. The server closes the input/output streams and the client socket to release resources.

CODE:

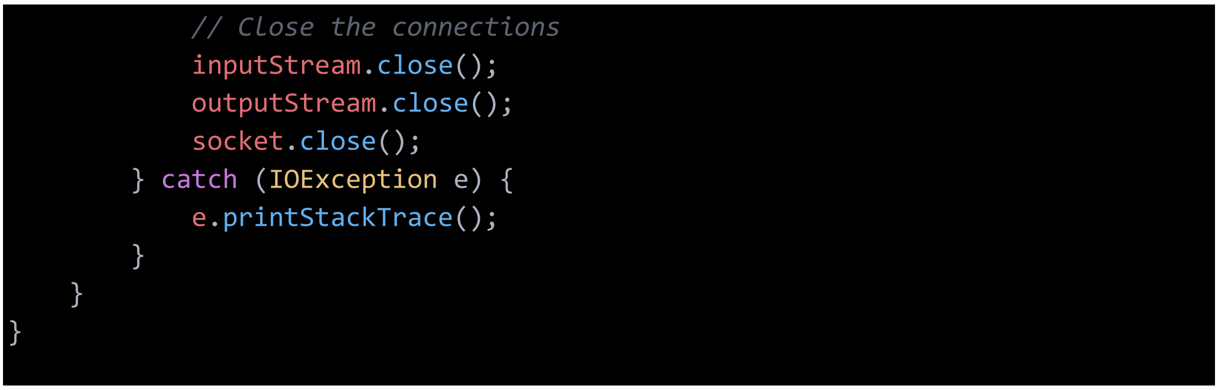
```
//BGP Message Transfer Server - 21MIS1146
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;

public class BGPServer {
    public static void main(String[] args) {
        try {
            ServerSocket serverSocket = new ServerSocket(5000); // Create a
server socket on port 5000

            System.out.println("Waiting for a client connection...");

            Socket clientSocket = serverSocket.accept(); // Wait for a client
to connect

            System.out.println("Client connected: " +
clientSocket.getInetAddress().getHostAddress());

            // Receive the message from the client
            InputStream inputStream = clientSocket.getInputStream();
            byte[] buffer = new byte[1024];
            int bytesRead = inputStream.read(buffer);
```



```
            String receivedMessage = new String(buffer, 0, bytesRead);
            System.out.println("Received message: " + receivedMessage);

            // Process the received message as needed

            // Send a response back to the client
            OutputStream outputStream = clientSocket.getOutputStream();
            String responseMessage = "Message received!";
            outputStream.write(responseMessage.getBytes());

            // Close the connections
            inputStream.close();
            outputStream.close();
            clientSocket.close();
            serverSocket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

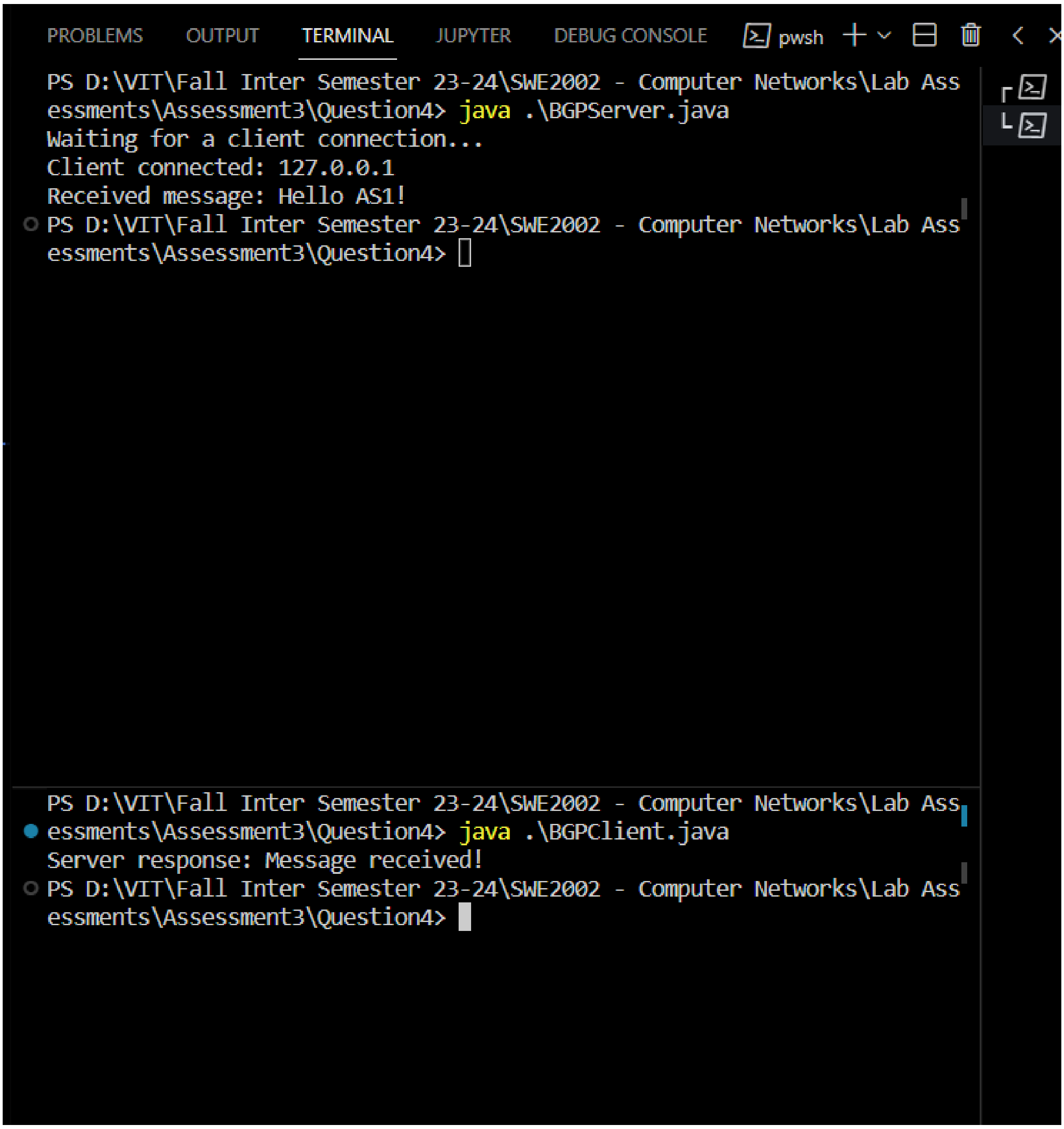


```
//BGP Message Transfer Client - 21MIS1146
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;

public class BGPClient {
    public static void main(String[] args) {
        try {
            Socket socket = new Socket("localhost", 5000); // Connect to the
server at localhost:5000

            // Send a message to the server
            OutputStream outputStream = socket.getOutputStream();
            String message = "Hello AS1!";
            outputStream.write(message.getBytes());

            // Receive the response from the server
            InputStream inputStream = socket.getInputStream();
            byte[] buffer = new byte[1024];
            int bytesRead = inputStream.read(buffer);

            String responseMessage = new String(buffer, 0, bytesRead);
            System.out.println("Server response: " + responseMessage);
```



```
            // Close the connections
            inputStream.close();
            outputStream.close();
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT:



```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Ass
essments\Assessment3\Question4> java .\BGPServer.java
Waiting for a client connection...
Client connected: 127.0.0.1
Received message: Hello AS1!
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Ass
essments\Assessment3\Question4> 
```

```
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Ass
essments\Assessment3\Question4> java .\BGPClient.java
Server response: Message received!
PS D:\VIT\Fall Inter Semester 23-24\SWE2002 - Computer Networks\Lab Ass
essments\Assessment3\Question4> 
```