AIM:

To develop and execute a TCP-based Client-Server chat application in Java where both ends can exchange text messages.

Requirements:

- Java JDK installed
- Two terminals (or two systems)

Concept / Theory:

TCP is a **connection-oriented** and **reliable** transport layer protocol. In TCP chat communication, the **Server** waits for client requests using `ServerSocket`, while the **Client** uses `Socket` to connect. Once connected, both exchange information using input/output streams.

Program Code

1. Server Program (ChatServer.java)

```
import java.io.*;
import java.net.*;

public class ChatServer {
    public static void main(String[] args) {
        try {
            ServerSocket server = new ServerSocket(5000);
            System.out.println("Server is waiting for client...");

            Socket socket = server.accept();
            System.out.println("Client connected.");

            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            PrintWriter out = new PrintWriter(socket.getOutputStream(),
true);

            BufferedReader keyboard = new BufferedReader(new
InputStreamReader(System.in));

            String msg;

            while(true) {
                msg = in.readLine();
                if(msg.equalsIgnoreCase("bye")) {
                    System.out.println("Client ended chat.");
                    break;
                }
                System.out.println("Client: " + msg);

                System.out.print("Server: ");
                msg = keyboard.readLine();
```

```
                out.println(msg);

                if(msg.equalsIgnoreCase("bye")) {
                    break;
                }
            }

            socket.close();
            server.close();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }
}
```

---

2. Client Program (ChatClient.java)

```
import java.io.*;
import java.net.*;

public class ChatClient {
    public static void main(String[] args) {
        try {
            Socket socket = new Socket("localhost", 5000);
            System.out.println("Connected to Server. Type 'bye' to exit.");

            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            PrintWriter out = new PrintWriter(socket.getOutputStream(),
true);

            BufferedReader keyboard = new BufferedReader(new
InputStreamReader(System.in));

            String msg;

            while(true) {
                System.out.print("Client: ");
                msg = keyboard.readLine();
                out.println(msg);

                if(msg.equalsIgnoreCase("bye")) {
                    break;
                }

                msg = in.readLine();
                System.out.println("Server: " + msg);

                if(msg.equalsIgnoreCase("bye")) {
                    break;
                }
            }

            socket.close();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }
}
```

Procedure:

1. Save server and client programs separately.
2. Compile the programs:
3. `javac ChatServer.java`
4. `javac ChatClient.java`
5. Start server first:
6. `java ChatServer`
7. Start the client next in another terminal:
8. `java ChatClient`
9. Exchange messages.
10. Type `bye` to end the chat.

Sample Output

Server Side

```
Server is waiting for client...
Client connected.
Client: Hello Server
Server: Hello Client
Client ended chat.
```

Client Side

```
Connected to Server. Type 'bye' to exit.
Client: Hello Server
Server: Hello Client
Client: bye
```

Result:

A TCP-based client-server chat application was successfully executed in Java, demonstrating interactive two-way communication.