Program execution sequence

**1. Import Statements:** The code imports necessary classes from the `java.io` and `java.net` packages.

**2. SocketClient Class:** This is the main class of the application. It has a `main` method where the execution of the program starts.

**3. Constants:**

- `SERVER\_IP`: This constant holds the IP address of the server the client will connect to.

- `SERVER\_PORT`: This constant specifies the port number on which the server is listening for connections.

**4. main Method:**

- It tries to establish a connection to the server specified by `SERVER\_IP` and `SERVER\_PORT`.

- If successful, it prints a message confirming the connection.

- Then, it starts two separate threads for sending and receiving messages.

**5. ReceiveMessages Class:**

- This is a nested class responsible for receiving messages from the server.

- It implements the `Runnable` interface, indicating that it can be executed by a separate thread.

- In the `run` method, it creates a `BufferedReader` to read messages from the server's input stream.

- It continuously reads messages from the server and prints them to the console until the connection is closed.

**6. SendMessages Class**:

- Similar to `ReceiveMessages`, this is another nested class implementing `Runnable`.

- It's responsible for sending messages to the server.

- In the `run` method, it creates a `BufferedReader` to read user input from the console and a `PrintWriter` to send messages to the server.

- It continuously reads user input and sends it to the server until the user decides to exit (by closing the input stream).

**7. Error Handling:**

- The code includes basic error handling for exceptions that may occur during socket communication, such as `UnknownHostException` and `IOException`.

- If any exception occurs, an error message is printed to the console.