**Source Code:**

**Client Code**

**package** trail;

**import** java.io.IOException;

**import** java.io.ObjectInputStream;

**import** java.io.ObjectOutputStream;

**import** java.net.Socket;

**import** java.net.UnknownHostException;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Client {

**private** **static** **final** String ***HOST*** = "localhost";

**private** **static** **final** **int** ***PORT*** = 5000;

**private** **static** **final** **int** ***CLIENT\_COUNT*** = 3;

**final** List<Thread> threads = **new** ArrayList<>();

**for** (**int** i = 0; i < 3; i++) {

**final** **int** finalI = i; // declare final variable

Thread thread = **new** Thread(() -> {

**try** {

Socket socket = **new** Socket("localhost", 5000);

System.***out***.println("Connected to server");

ObjectOutputStream out = **new** ObjectOutputStream(socket.getOutputStream());

ObjectInputStream in = **new** ObjectInputStream(socket.getInputStream());

// Send local clock value to server

**long** lc = System.*currentTimeMillis*();

out.writeObject(lc);

out.flush();

// Receive adjusted time from server

**long** adjustedTime = (**long**) in.readObject();

System.***out***.println("Adjusted time: " + adjustedTime);

// Close streams and socket

out.close();

in.close();

socket.close();

} **catch** (IOException | ClassNotFoundException e) {

e.printStackTrace();

}

});

threads.add(thread);

thread.start();

}

**for** (Thread thread : threads) {

**try** {

thread.join();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}

**Server Code:**

**package** trail;

**import** java.io.IOException;

**import** java.io.ObjectInputStream;

**import** java.io.ObjectOutputStream;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**import** java.time.Instant;

**import** java.time.LocalDateTime;

**import** java.time.format.DateTimeFormatter;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.time.\*;

**public** **class** Server {

**private** **static** **final** **int** ***PORT*** = 5000;

**private** List<Socket> clients;

**private** List<ObjectOutputStream> outputStreams;

**private** List<ObjectInputStream> inputStreams;

**public** Server() {

clients = **new** ArrayList<>();

outputStreams = **new** ArrayList<>();

inputStreams = **new** ArrayList<>();

}

**public** **void** start() {

ServerSocket serverSocket = **null**;

**try** {

serverSocket = **new** ServerSocket(***PORT***);

System.***out***.println("Time server started on port " + ***PORT***);

**while** (**true**) {

Socket client = serverSocket.accept();

System.***out***.println("New client connected");

clients.add(client);

ObjectOutputStream out = **new** ObjectOutputStream(client.getOutputStream());

ObjectInputStream in = **new** ObjectInputStream(client.getInputStream());

outputStreams.add(out);

inputStreams.add(in);

Thread thread = **new** Thread(() -> handleClient(client, out, in));

thread.start();

}

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**try** {

**if** (serverSocket != **null**) {

serverSocket.close();

}

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**private** **void** handleClient(Socket client, ObjectOutputStream out, ObjectInputStream in) {

**try** {

**while** (**true**) {

Thread.*sleep*(1000);

**long** currentTime = getTimeFromClient(in);

Instant instant = Instant.*ofEpochMilli*(currentTime);

LocalDateTime localDateTime = LocalDateTime.*ofInstant*(instant, ZoneId.*systemDefault*());

String formattedTime = DateTimeFormatter.***ISO\_LOCAL\_DATE\_TIME***.format(localDateTime);

System.***out***.println("Received time " + formattedTime);

broadcastTimeToClients(calculateAverageTime(), outputStreams);

}

} **catch** (IOException | ClassNotFoundException | InterruptedException e) {

e.printStackTrace();

} **finally** {

**try** {

clients.remove(client);

outputStreams.remove(out);

inputStreams.remove(in);

client.close();

out.close();

in.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**private** **long** getTimeFromClient(ObjectInputStream in) **throws** IOException, ClassNotFoundException {

**return** (**long**) in.readObject();

}

**private** **void** broadcastTimeToClients(**long** time, List<ObjectOutputStream> outputStreams) **throws** IOException {

**for** (ObjectOutputStream out : outputStreams) {

out.writeObject(time);

out.flush();

}

}

**private** **long** calculateAverageTime() {

**long** sum = 0;

**for** (ObjectInputStream in : inputStreams) {

**try** {

**long** currentTime = getTimeFromClient(in);

sum += currentTime;

} **catch** (IOException | ClassNotFoundException e) {

e.printStackTrace();

}

}

**return** sum / inputStreams.size();

}

**public** **static** **void** main(String[] args) {

Server server = **new** Server();

server.start();

}

}