Assignment: 14 Date: 22/11/2023

JAVA PROGRAMMING
CS6308

SOCKET PROGRAMMING

Name: VIJAI SURIA M

Reg No.: 2021503568

1. **TCP/UDP Socket to get the square root of the given number,**

TCP Server (*finding square root****):***

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;

public class SquareRootServer3568 {
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        try {
            ServerSocket serverSocket = new ServerSocket(9999); // Port to listen
on
            System.out.println("Server started. Waiting for a client...");

            Socket clientSocket = serverSocket.accept(); // Accept incoming
connection
            System.out.println("Client connected.");

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

            String inputLine;
```

```
            while ((inputLine = in.readLine()) != null) {
                double number = Double.parseDouble(inputLine);
                double squareRoot = Math.sqrt(number);
                out.println("Square root of " + number + " is: " + squareRoot);
            }

            clientSocket.close();
            serverSocket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

TCP Client (*finding square root****):***

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;

public class SquareRootClient3568 {
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        try {
            Socket socket = new Socket("localhost", 9999); // Connect to server
            BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in));
            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
            System.out.print("Enter a number to find its square root: ");
            String number = userInput.readLine();
            out.println(number); // Send number to server

            String serverResponse = in.readLine(); // Receive square root from
server
```

```
            System.out.println("Server response: " + serverResponse);
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT (*finding square root*):

TCP Client:

```
Run    SquareRootServer3568 ×    SquareRootClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:
Current Date: 2023-11-22
Current Time: 16:27:36.413275200
Name: Vijai Suria M
Register Number: (2021503568)

Enter a number to find its square root: 81
Server response: Square root of 81.0 is: 9.0

Process finished with exit code 0
```

TCP Server:

```
Run    SquareRootServer3568 ×    SquareRootClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:
Current Date: 2023-11-22
Current Time: 16:27:30.149811300
Server started. Waiting for a client...
Client connected.

Process finished with exit code 0
```

UDP Server (*finding square root*):

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;

public class SquareRootUDPServer3568 {
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");

        try {
            DatagramSocket serverSocket = new DatagramSocket(9999); // Port to
listen on
            System.out.println("Server started. Waiting for a client...");

            byte[] receiveData = new byte[1024];
            byte[] sendData = new byte[1024];

            while (true) {
                DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
                serverSocket.receive(receivePacket);

                String numberString = new String(receivePacket.getData(), 0,
receivePacket.getLength());
                double number = Double.parseDouble(numberString);

                double squareRoot = Math.sqrt(number);
                String response = "Square root of " + number + " is: " + squareRoot;
                sendData = response.getBytes();

                InetAddress clientIP = receivePacket.getAddress();
                int clientPort = receivePacket.getPort();
                DatagramPacket sendPacket = new DatagramPacket(sendData,
sendData.length, clientIP, clientPort);

                serverSocket.send(sendPacket);
```

```
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

UDP Client (*finding square root****):***

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;

public class SquareRootUDPClient3568 {
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        try {
            DatagramSocket clientSocket = new DatagramSocket();
            InetAddress serverIP = InetAddress.getByName("localhost");
            byte[] sendData = new byte[1024];
            byte[] receiveData = new byte[1024];
            BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in));
            System.out.print("Enter a number to find its square root: ");
            String number = userInput.readLine();
            sendData = number.getBytes();
            DatagramPacket sendPacket = new DatagramPacket(sendData,
sendData.length, serverIP, 9999);
            clientSocket.send(sendPacket);
            DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
            clientSocket.receive(receivePacket);
            String serverResponse = new String(receivePacket.getData(), 0,
receivePacket.getLength());
            System.out.println("Server response: " + serverResponse);
            clientSocket.close();
```

```
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT (*finding square root*):

UDP Client:

```
Run    SquareRootUDPServer3568 ×    SquareRootUDPClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\Pr
Current Date: 2023-11-22
Current Time: 16:31:04.439242500
Name: Vijai Suria M
Register Number: (2021503568)

Enter a number to find its square root: 64
Server response: Square root of 64.0 is: 8.0

Process finished with exit code 0
```

UDP Server:

```
Run    SquareRootUDPServer3568 ×    SquareRootUDPClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\
Current Date: 2023-11-22
Current Time: 16:30:42.214564800
Name: Vijai Suria M
Register Number: (2021503568)

Server started. Waiting for a client...
```

2. TCP/UDP socket to sort the array of inputs,

TCP Server (*sorting array of inputs****):***

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;
import java.util.Arrays;
public class TCPSortServer3568 {
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        try {
            ServerSocket serverSocket = new ServerSocket(9999); // Port to listen
on
            System.out.println("Server started. Waiting for a client...");
            Socket clientSocket = serverSocket.accept(); // Accept incoming
connection
            System.out.println("Client connected.");
            BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
            PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
            String inputLine = in.readLine();
            String[] numbersArray = inputLine.split(" ");
            int[] intArray =
Arrays.stream(numbersArray).mapToInt(Integer::parseInt).toArray();

            Arrays.sort(intArray);

            String sortedNumbers = Arrays.toString(intArray);
            out.println(sortedNumbers);

            clientSocket.close();
            serverSocket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
```

```
    }
}
```

TCP Client (*sorting array of inputs****):***

```
import java.io.*;
import java.net.*;
import java.util.Arrays;
import java.time.LocalDate;
import java.time.LocalTime;

public class TCPSortClient3568 {
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        try {
            Socket socket = new Socket("localhost", 9999); // Connect to server
            BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in));
            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

            System.out.print("Enter numbers separated by spaces to be sorted: ");
            String numbers = userInput.readLine();
            out.println(numbers); // Send numbers to server
            String sortedNumbers = in.readLine(); // Receive sorted numbers from
server
            int[] sortedArray = Arrays.stream(sortedNumbers.substring(1,
sortedNumbers.length() - 1).split(", "))
                    .mapToInt(Integer::parseInt).toArray();

            System.out.print("Sorted array received from server: ");
            for (int num : sortedArray) {
                System.out.print(num + " ");
            }
            System.out.println();
```

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

OUTPUT (*sorting array of inputs****):***

TCP Server:

```
Run    TCPSortServer3568 ×    TCPSortClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe"
Current Date: 2023-11-22
Current Time: 16:40:48.195403700
Name: Vijai Suria M
Register Number: (2021503568)

Server started. Waiting for a client...
Client connected.

Process finished with exit code 0
```

TCP Client:

```
Run    TCPSortServer3568 ×    TCPSortClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\Program Fil
Current Date: 2023-11-22
Current Time: 16:40:53.938203600
Name: Vijai Suria M
Register Number: (2021503568)

Enter numbers separated by spaces to be sorted: 9 89 99 -100 0 1 2 3
Sorted array received from server: -100 0 1 2 3 9 89 99
```

UDP Server (*sorting array of inputs****):***

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;
import java.util.Arrays;
import java.time.LocalDate;
import java.time.LocalTime;

public class UDPSortServer3568 {
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        try {
            DatagramSocket serverSocket = new DatagramSocket(9999); // Port to
listen on
            System.out.println("Server started. Waiting for a client...");

            byte[] receiveData = new byte[1024];
            byte[] sendData = new byte[1024];

            while (true) {
                DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
                serverSocket.receive(receivePacket);

                ByteArrayInputStream byteStream = new
ByteArrayInputStream(receivePacket.getData());
                ObjectInputStream objInput = new ObjectInputStream(byteStream);

                int[] receivedArray = (int[]) objInput.readObject();
                Arrays.sort(receivedArray);

                ByteArrayOutputStream byteStreamOut = new
ByteArrayOutputStream();
                ObjectOutputStream objOutput = new
ObjectOutputStream(byteStreamOut);
                objOutput.writeObject(receivedArray);
```

```
            sendData = byteStreamOut.toByteArray();

            InetAddress clientIP = receivePacket.getAddress();
            int clientPort = receivePacket.getPort();
            DatagramPacket sendPacket = new DatagramPacket(sendData,
sendData.length, clientIP, clientPort);

            serverSocket.send(sendPacket);
        }
    } catch (IOException | ClassNotFoundException e) {
        e.printStackTrace();
    }
  }
}
```

UDP Client (*sorting array of inputs****):***

```
import java.io.*;
import java.net.*;
import java.util.Arrays;
import java.time.LocalDate;
import java.time.LocalTime;

public class UDPSortClient3568 {
  public static void main(String[] args) {
    System.out.println("Current Date: " + LocalDate.now());
    System.out.println("Current Time: " + LocalTime.now());
    System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
    try {
      DatagramSocket clientSocket = new DatagramSocket();
      InetAddress serverIP = InetAddress.getByName("localhost");
      byte[] sendData = new byte[1024];
      byte[] receiveData = new byte[1024];

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

      System.out.print("Enter numbers separated by spaces to be sorted: ");
      String numbers = userInput.readLine();
```

```
            String[] numbersArray = numbers.split(" ");
            int[] intArray =
Arrays.stream(numbersArray).mapToInt(Integer::parseInt).toArray();

            ByteArrayOutputStream byteStream = new ByteArrayOutputStream();
            ObjectOutputStream objOutput = new ObjectOutputStream(byteStream);
            objOutput.writeObject(intArray);
            sendData = byteStream.toByteArray();

            DatagramPacket sendPacket = new DatagramPacket(sendData,
sendData.length, serverIP, 9999);
            clientSocket.send(sendPacket);

            DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
            clientSocket.receive(receivePacket);

            ByteArrayInputStream byteStreamIn = new
ByteArrayInputStream(receivePacket.getData());
            ObjectInputStream objInput = new ObjectInputStream(byteStreamIn);

            int[] sortedArray = (int[]) objInput.readObject();
            System.out.print("Sorted array received from server: ");
            for (int num : sortedArray) {
                System.out.print(num + " ");
            }
            System.out.println();

            clientSocket.close();
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT (**sorting array of inputs**):

UDP Server:

```
Run    UDPSortServer3568 ×    UDPSortClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaager
Current Date: 2023-11-22
Current Time: 16:34:16.405691200
Name: Vijai Suria M
Register Number: (2021503568)

Server started. Waiting for a client...
```

UDP Client:

```
Run    UDPSortServer3568 ×    UDPSortClient3568 ×

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\Progra
Current Date: 2023-11-22
Current Time: 16:34:21.752367300
Name: Vijai Suria M
Register Number: (2021503568)

Enter numbers separated by spaces to be sorted: 45 67 -1 0 9
Sorted array received from server: -1 0 9 45 67

Process finished with exit code 0
```

3. TCP/UDP chat using multiclient

TCP Multi-client chat Server:

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

```
import java.time.LocalDate;
import java.time.LocalTime;
public class TCPServer3568 {
    private static final int PORT = 8888;
    private static final Set<Socket> clientSockets = new HashSet<>();
    private static final Map<Socket, String> clientNames = new HashMap<>();
    public static void main(String[] args) {
        System.out.println("Date:" + LocalDate.now());
        System.out.println("Time:" + LocalTime.now());
        System.out.println("NAME: Vijai Suria .M \nReg No:2021503568");
        System.out.println("\n");
        try (ServerSocket serverSocket = new ServerSocket(PORT)) {
            System.out.println("Server started. Waiting for clients...");
            while (true) {
                Socket clientSocket = serverSocket.accept();
                System.out.println("New client connected: " + clientSocket);
                clientSockets.add(clientSocket);
                Thread clientThread = new Thread(new ClientHandler(clientSocket));
                clientThread.start();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    static class ClientHandler implements Runnable {
        private final Socket clientSocket;
        private PrintWriter writer;

        public ClientHandler(Socket socket) {
            this.clientSocket = socket;
        }

        @Override
        public void run() {
            try {
                BufferedReader reader = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
                writer = new PrintWriter(clientSocket.getOutputStream(), true);
                writer.println("Enter your name:");
```

```
            String name = reader.readLine();
            clientNames.put(clientSocket, name);
            broadcastMessage(name + " joined the chat");
            String clientMessage;
            while ((clientMessage = reader.readLine()) != null) {
                broadcastMessage(name + ": " + clientMessage);
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            clientNames.remove(clientSocket);
            clientSockets.remove(clientSocket);
            broadcastMessage(clientNames.getOrDefault(clientSocket,
"Anonymous") + " left the chat");
            try {
                clientSocket.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
    private void broadcastMessage(String message) {
        for (Socket socket : clientSockets) {
            if (socket != clientSocket) {
                try {
                    PrintWriter socketWriter = new
PrintWriter(socket.getOutputStream(), true);
                    socketWriter.println(message);
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }
  }
}
```

TCP Multi-client chat Client:

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.time.LocalDate;
import java.time.LocalTime;
public class TCPClient3568 {
    private static final String SERVER_IP = "127.0.0.1"; // Change this to the
server's IP address
    private static final int PORT = 8888;
    public static void main(String[] args) {
        System.out.println("NAME: Vijai Suria.M \nReg No:2021503568");
        System.out.println("Date:" + LocalDate.now());
        System.out.println("Time:" + LocalTime.now());
        System.out.println("\n");
        try (Socket socket = new Socket(SERVER_IP, PORT);
            BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
            PrintWriter writer = new PrintWriter(socket.getOutputStream(), true);
            BufferedReader serverReader = new BufferedReader(new
InputStreamReader(socket.getInputStream()))) {
            System.out.println("Connected to the chat server.");
            Thread serverListener = new Thread(() -> {
                String serverMessage;
                try {
                    while ((serverMessage = serverReader.readLine()) != null) {
                        System.out.println(serverMessage);
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            });
            serverListener.start();
            String name = serverReader.readLine();
            System.out.println(name);
            while (true) {
                String message = reader.readLine();
                writer.println(message);
            }
        } catch (IOException e) {
```

```
            e.printStackTrace();
        }
    }
}
```

OUTPUT (*Multi-client chat Server****):***

TCP Server:

```
Run    TCPServer3568 ×    TCPClient3568 ×    TCPClient3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\
Date:2023-11-22
Time:18:14:22.266814100
NAME: Vijai Suria .M
Reg No:2021503568


Server started. Waiting for clients...
New client connected: Socket[addr=/127.0.0.1,port=61839,localport=8888]
New client connected: Socket[addr=/127.0.0.1,port=61843,localport=8888]
```

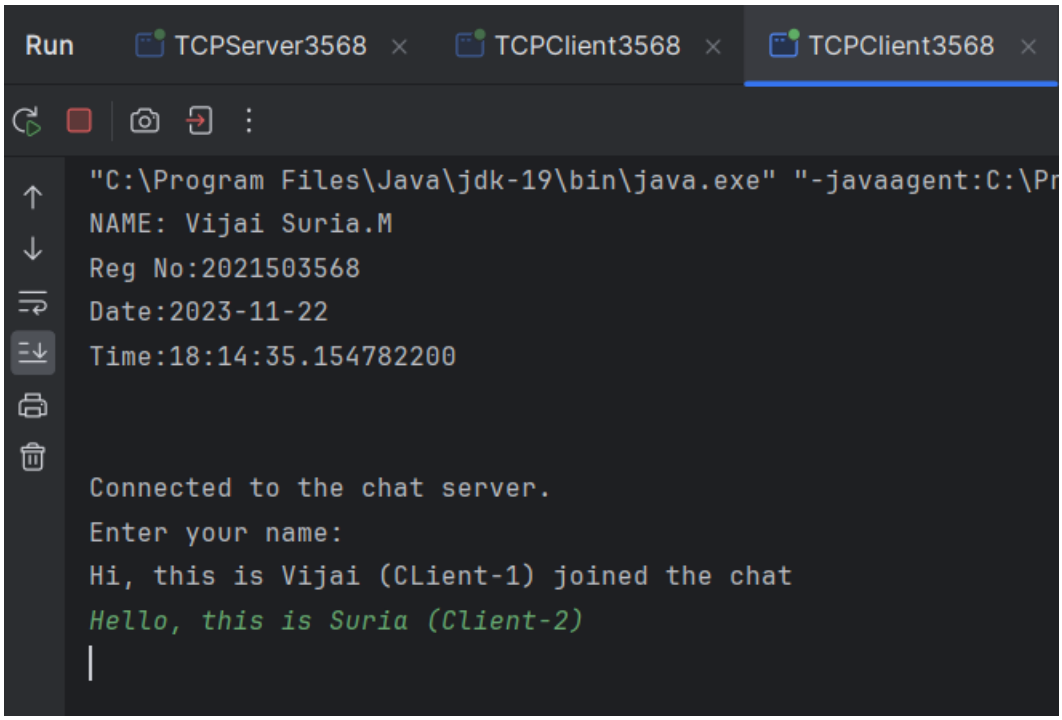
CLIENT-1

```
Run    TCPServer3568 ×    TCPClient3568 ×    TCPClient3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Pr
NAME: Vijai Suria.M
Reg No:2021503568
Date:2023-11-22
Time:18:14:30.912583700


Connected to the chat server.
Enter your name:
Hi, this is Vijai (CLient-1)
Hello, this is Suria (Client-2) joined the chat
```

CLIENT-2



UDP Multi-client chat Server:

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;
public class UDPServer3568 {
    private static final int PORT = 8888;
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        try (DatagramSocket serverSocket = new DatagramSocket(PORT)) {
            System.out.println("Server started. Waiting for clients...");
            // Receiving messages from clients
            while (true) {
                byte[] receiveData = new byte[1024];
                DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
                serverSocket.receive(receivePacket);
```

```
            InetAddress clientAddress = receivePacket.getAddress();
            int clientPort = receivePacket.getPort();
            String receivedMessage = new String(receivePacket.getData(), 0,
receivePacket.getLength());
            System.out.println("Client [" + clientAddress + ":" + clientPort + "]: " +
receivedMessage);
            // Sending received message back to the sender client
            sendData(serverSocket, clientAddress, clientPort,
receivedMessage.getBytes());
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
  }
  private static void sendData(DatagramSocket socket, InetAddress address, int
port, byte[] data) throws IOException {
    DatagramPacket sendPacket = new DatagramPacket(data, data.length,
address, port);
    socket.send(sendPacket);
  }
}
```

UDP Multi-client chat Client:

```
import java.io.*;
import java.net.*;
import java.time.LocalDate;
import java.time.LocalTime;
public class UDPClient3568 {
  private static final String SERVER_IP = "127.0.0.1"; // Change this to the
server's IP address
  private static final int PORT = 8888;

  public static void main(String[] args) {
    System.out.println("Current Date: " + LocalDate.now());
    System.out.println("Current Time: " + LocalTime.now());
    System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
```

```
        try (DatagramSocket clientSocket = new DatagramSocket();
            BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in))) {
            InetAddress serverAddress = InetAddress.getByName(SERVER_IP);
            Thread serverListener = new Thread(() -> {
                try {
                    while (true) {
                        byte[] receiveData = new byte[1024];
                        DatagramPacket receivePacket = new
DatagramPacket(receiveData, receiveData.length);
                        clientSocket.receive(receivePacket);

                        String receivedMessage = new String(receivePacket.getData(), 0,
receivePacket.getLength());
                        System.out.println(receivedMessage);
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            });
            serverListener.start();
            System.out.println("Connected to the chat server.");
            while (true) {
                String message = reader.readLine();
                byte[] sendData = message.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendData,
sendData.length, serverAddress, PORT);
                clientSocket.send(sendPacket);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

UDP Multi-chat Server:

```
Run    UDPServer3568 ×    UDPClient3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetB
Current Date: 2023-11-22
Current Time: 18:27:09.623620500
Name: Vijai Suria M
Register Number: (2021503568)

Server started. Waiting for clients...
Client [/127.0.0.1:62894]: Hi, this is vijai (Client-1) chatting with server
Client [/127.0.0.1:62894]: Bye, catch you later
Client [/127.0.0.1:51547]: Hello, Suria here (Client-2)
Client [/127.0.0.1:51547]: Have a nice nice day!!!
```

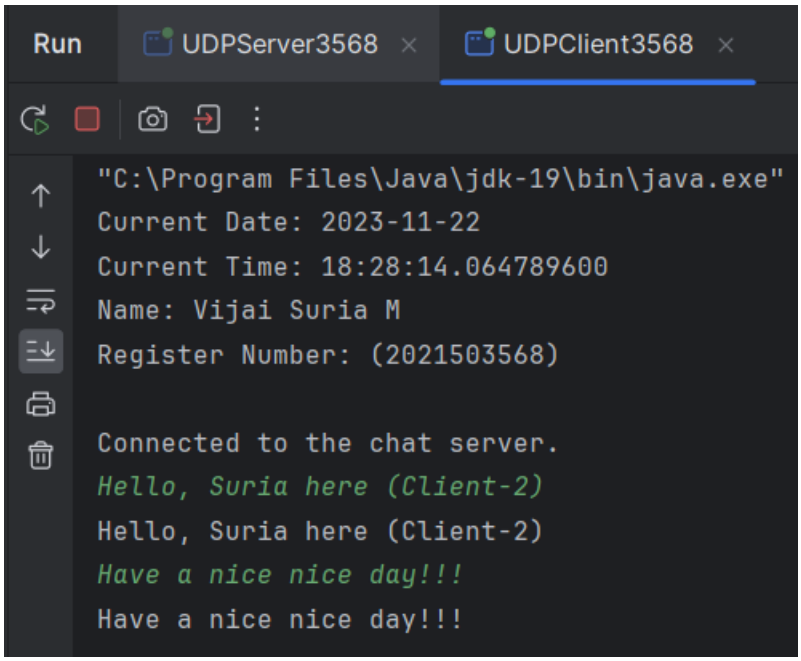
UDP Multi-chat Client-1:

```
Run    UDPServer3568 ×    UDPClient3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:
Current Date: 2023-11-22
Current Time: 18:27:17.284995
Name: Vijai Suria M
Register Number: (2021503568)

Connected to the chat server.
Hi, this is vijai (Client-1) chatting with server
Hi, this is vijai (Client-1) chatting with server
Bye, catch you later
Bye, catch you later
```

UDP Multi-chat Client-2:



4. **Write a java code for movie ticket booking system using multi-client socket programming. Implement the synchronization mechanism of ticket booking operations at server side to handle multiple clients concurrently.**

SERVER CODE (*for movie ticket booking****):***

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.time.LocalDate;
import java.time.LocalTime;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class MovieServer_3568 {
    private static final int PORT = 8888;
    private static final int MAX_AVAILABLE_TICKETS = 100;
    private static int availableTickets = MAX_AVAILABLE_TICKETS;
    private static final Object lock = new Object();

    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
```

```
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        System.out.println("\n");
        ExecutorService executor = Executors.newFixedThreadPool(10);

        try (ServerSocket serverSocket = new ServerSocket(PORT)) {
            System.out.println("Server started. Waiting for clients...");

            while (true) {
                Socket clientSocket = serverSocket.accept();
                System.out.println("New client connected: " + clientSocket);

                Runnable clientHandler = new ClientHandler(clientSocket);
                executor.execute(clientHandler);
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            executor.shutdown();
        }
    }

    static class ClientHandler implements Runnable {
        private final Socket clientSocket;

        public ClientHandler(Socket socket) {
            this.clientSocket = socket;
        }

        @Override
        public void run() {
            try {
                BufferedReader reader = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
                PrintWriter writer = new PrintWriter(clientSocket.getOutputStream(),
true);

                String clientMessage;
                while ((clientMessage = reader.readLine()) != null) {
```

```
                if (clientMessage.equalsIgnoreCase("bookTicket")) {
                    bookTicket(writer);
                } else {
                    writer.println("Invalid command.");
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private void bookTicket(PrintWriter writer) {
        synchronized (lock) {
            if (availableTickets > 0) {
                availableTickets--;
                writer.println("Ticket booked successfully. Remaining tickets: " +
availableTickets);
            } else {
                writer.println("Sorry, tickets are sold out.");
            }
        }
    }
  }
}
```

CLIENT CODE (*for movie ticket booking****):***

```
import java.io.*;
import java.net.*;
import java.io.*;
import java.net.*;
import java.util.*;
import java.time.LocalDate;
import java.time.LocalTime;

public class MovieClient1_3568 {
    private static final String SERVER_IP = "127.0.0.1"; // Change this to the
server's IP address
    private static final int PORT = 8888;
```

```
    public static void main(String[] args) {
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        System.out.println("\n");
        try (Socket socket = new Socket(SERVER_IP, PORT);
            BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
            PrintWriter writer = new PrintWriter(socket.getOutputStream(), true);
            BufferedReader serverReader = new BufferedReader(new
InputStreamReader(socket.getInputStream()))) {

            System.out.println("Connected to the Movie Ticket Booking Server.");

            while (true) {
                System.out.println("Enter 'bookTicket' to book a ticket or 'exit' to quit:");
                String userInput = reader.readLine();

                if (userInput.equalsIgnoreCase("exit")) {
                    break;
                }

                writer.println(userInput);

                String serverResponse = serverReader.readLine();
                System.out.println("Server Response: " + serverResponse);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Server (*for movie ticket booking****):***

```
Run    MovieServer_3568 ×    MovieClient1_3568 ×    MovieClient2_3568 ×    MovieClient2_3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ
Current Date: 2023-11-22
Current Time: 18:35:53.853359800
Name: Vijai Suria M
Register Number: (2021503568)



Server started. Waiting for clients...
New client connected: Socket[addr=/127.0.0.1,port=62066,localport=8888]
New client connected: Socket[addr=/127.0.0.1,port=62079,localport=8888]
New client connected: Socket[addr=/127.0.0.1,port=62083,localport=8888]
```

Client-1 (*for movie ticket booking****):-***

```
Run    MovieServer_3568 ×    MovieClient1_3568 ×    MovieClient2_3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program
Current Date: 2023-11-22
Current Time: 18:36:02.091730800
Name: Vijai Suria M
Register Number: (2021503568)



Connected to the Movie Ticket Booking Server.
Enter 'bookTicket' to book a ticket or 'exit' to quit:
bookTicket
Server Response: Ticket booked successfully. Remaining tickets: 99
Enter 'bookTicket' to book a ticket or 'exit' to quit:
```

Client-2 (*for movie ticket booking****):-***

```
Run    MovieServer_3568 ×    MovieClient1_3568 ×    MovieClient2_3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Fil
Current Date: 2023-11-22
Current Time: 18:36:21.479113700
Name: Vijai Suria M
Register Number: (2021503568)



Connected to the Movie Ticket Booking Server.
Enter 'bookTicket' to book a ticket or 'exit' to quit:
bookTicket
Server Response: Ticket booked successfully. Remaining tickets: 98
Enter 'bookTicket' to book a ticket or 'exit' to quit:
bookTicket
Server Response: Ticket booked successfully. Remaining tickets: 97
Enter 'bookTicket' to book a ticket or 'exit' to quit:
```

Client-3 (*for movie ticket booking****):-***

```
Run    MovieServer_3568 ×    MovieClient1_3568 ×    MovieClient2_3568 ×    MovieClient2_3568 ×

"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ
Current Date: 2023-11-22
Current Time: 18:37:07.832015600
Name: Vijai Suria M
Register Number: (2021503568)



Connected to the Movie Ticket Booking Server.
Enter 'bookTicket' to book a ticket or 'exit' to quit:
bookTicket
Server Response: Ticket booked successfully. Remaining tickets: 96
Enter 'bookTicket' to book a ticket or 'exit' to quit:
```

5. **Write a java JDBC code for storing student details such as name, regno, gender, current semester course mark and GPA(using CalculateGPA method):**

JDBC Code:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.Scanner;
import java.time.LocalDate;
import java.time.LocalTime;
import java.math.BigDecimal;
import java.math.RoundingMode;

public class connectionJDBC {
    public static int getCreditPoints(int mark) {
        if (mark >= 91 && mark <= 100) {
            return 10;
        } else if (mark >= 81 && mark <= 90) {
            return 9;
        } else if (mark >= 71 && mark <= 80) {
            return 8;
        } else if (mark >= 61 && mark <= 70) {
            return 7;
        } else if (mark >= 51 && mark <= 60) {
            return 6;
        } else {
            return 0;
        }
    }

    public static float calculateGPA(int[] creditPoints){
        int[] credits = {4,4,4,5,6};
        float res=0.0F;
        int total=0;
        for(int i=0;i<credits.length;i++) {
            res += (credits[i]*creditPoints[i]);
            total+=credits[i];
        }
        res = res/total;
        return res;
    }

    public static void main(String[] args) throws Exception {
```

```
        Scanner in = new Scanner(System.in);
        Connection con = null;
        System.out.println("Current Date: " + LocalDate.now());
        System.out.println("Current Time: " + LocalTime.now());
        System.out.println("Name: Vijai Suria M \nRegister Number: (2021503568)
\n");
        try {
            con =
DriverManager.getConnection("jdbc:oracle:thin:@192.168.109.28:1521:orcl",
"ct2021503568", "ct2021503568");
            if (con == null){
                System.out.println("Database not Connected");
                System.exit(0);
            }
            else{
                System.out.println("Connected to Database 2021503568");
            }
        }
        catch (Exception e) {
            System.out.println(e);
        }

        try{
            Statement stmt = con.createStatement();
            boolean flag = true;

            while(flag){
                System.out.println("Menu Board");
                System.out.println("1 --> Insert record");
                System.out.println("2 --> View records");
                System.out.println("3 --> Exit");
                System.out.print("Enter your choice: ");
                int ch = in.nextInt();

                switch (ch){
                    case 1:
                        System.out.println("Enter the Student name");
                        String name = in.next();
                        System.out.println("Enter the Student reg no.: ");
                        String regno = in.next();
                        System.out.println("Enter the Student Gender: ");
                        String gender = in.next();
                        System.out.println("Enter the course-1 marks: ");
                        int c1 = in.nextInt();
                        System.out.println("Enter the course-2 marks: ");
                        int c2 = in.nextInt();
                        System.out.println("Enter the course-3 marks: ");
```

```
                    int c3 = in.nextInt();
                    System.out.println("Enter the course-4 marks: ");
                    int c4 = in.nextInt();
                    System.out.println("Enter the course-5 marks: ");
                    int c5 = in.nextInt();

                    int[] creditPoints = new int[5]; // Assuming 5 courses
                    creditPoints[0] = getCreditPoints(c1);
                    creditPoints[1] = getCreditPoints(c2);
                    creditPoints[2] = getCreditPoints(c3);
                    creditPoints[3] = getCreditPoints(c4);
                    creditPoints[4] = getCreditPoints(c5);

                    float gpa = calculateGPA(creditPoints);
                    BigDecimal gpaBigDecimal = new BigDecimal(Float.toString(gpa));
                    gpaBigDecimal = gpaBigDecimal.setScale(2,
RoundingMode.HALF_UP);
                    int count = stmt.executeUpdate("Insert into student(name, regno,
gender, c1, c2, c3, c4, c5, gpa) VALUES ('" + name + "'," + regno + ",'" + gender +
"'," + c1 + "," + c2 + "," + c3 + "," + c4 + "," + c5 + "," + gpaBigDecimal +")");
                    System.out.println("No. of rows inserted: " + count);
                    break;

                case 2:
                    ResultSet rs = stmt.executeQuery("Select * from student");
                    System.out.println("Student Table: ");
                    System.out.println("-----------------------------------------------------------
------------");
                    while(rs.next()){
                        System.out.println(
                                rs.getString("name") + "\t|\t" + rs.getString("regno") +  "\t|\t"
+ rs.getString("gender") +  "\t|\t" + rs.getInt("c1") + "\t|\t" + rs.getInt("c2") + "\t|\t" +
rs.getInt("c3") +"\t|\t" + rs.getInt("c4") +"\t|\t" + rs.getInt("c5") + "\t|\t" +
rs.getDouble("gpa"));
                    }
                    System.out.println("-----------------------------------------------------------
------------");
                    break;
                case 3:
                    System.out.println("Thank you.......");
                    flag=false;
                    break;

                default:
                    System.out.println("Enter the valid choice....");
            }
        }
```

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

    }

}
```

OUTPUT:-

```
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe"
Current Date: 2023-11-22
Current Time: 16:16:33.756881600
Name: Vijai Suria M
Register Number: (2021503568)

Connected to Database 2021503568
Menu Board
1 --> Insert record
2 --> View records
3 --> Exit
Enter your choice: 1
Enter the Student name
Suria
Enter the Student reg no.:
68
Enter the Student Gender:
M
Enter the course-1 marks:
98
Enter the course-2 marks:
97
Enter the course-3 marks:
85
Enter the course-4 marks:
92
Enter the course-5 marks:
89
No. of rows inserted: 1
```

```
Menu Board
1 --> Insert record
2 --> View records
3 --> Exit
Enter your choice: 2
Student Table:
-------------------------------------------------------------------------
Vijai   |   68  |   M   |   98  |   78  |   88  |   95  |   94  |   9.48
Suria   |   68  |   M   |   98  |   97  |   85  |   92  |   89  |   9.57
-------------------------------------------------------------------------
Menu Board
1 --> Insert record
2 --> View records
3 --> Exit
Enter your choice: 3
Thank you.......
```