Computer Networks Lab

Assignment-10

Name: S. Vishwajith

Register Number: 23BCE1145

1.

Question:

Develop a multi client UDP application in which client sends a number and server receives it and count the number of digits in it and send the same to the corresponding client.

Code:

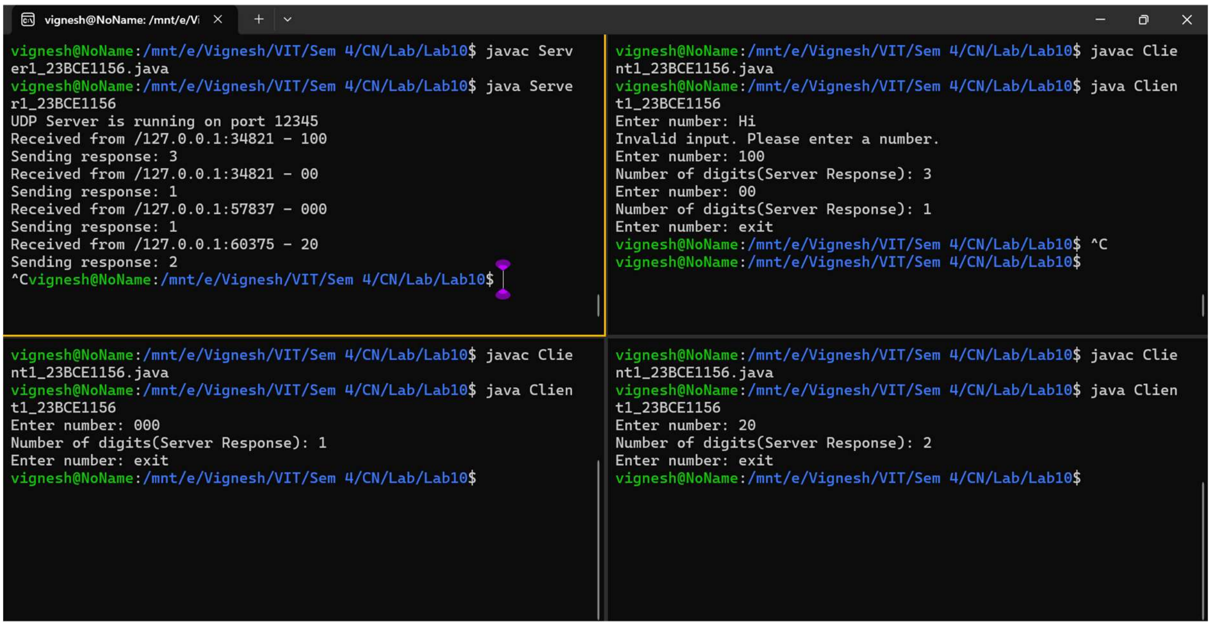
Server:

```
import java.net.*;
public class Server1_23BCE1145 {
    public static void main(String[] args) {
        final int SERVER_PORT = 12345;
        DatagramSocket serverSocket = null;
        try {
            serverSocket = new DatagramSocket(SERVER_PORT);
            System.out.println("UDP Server is running on port " + SERVER_PORT);

            byte[] receiveBuffer = new byte[1024];

            while (true) {
                DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
receiveBuffer.length);
                serverSocket.receive(receivePacket);
                String receivedMessage = new String(receivePacket.getData(), 0, receivePacket.getLength());
                System.out.println("Received from " + receivePacket.getAddress() + ":" + receivePacket.getPort() + " - " + receivedMessage);
                String responseMessage = ((Integer.parseInt(receivedMessage)+"").length()) + "";
                byte[] sendBuffer = responseMessage.getBytes();
                System.out.println("Sending response: " + responseMessage);
                DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, receivePacket.getAddress(), receivePacket.getPort());
                serverSocket.send(sendPacket);
            }
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        finally {
            serverSocket.close();
        }
    }
}
```

Client:

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

public class Client1_23BCE1145 {
    public static void main(String[] args) {
        final String SERVER_IP = "127.0.0.1";
        final int SERVER_PORT = 12345;

        try {
            DatagramSocket clientSocket = new DatagramSocket();
            InetAddress serverAddress = InetAddress.getByName(SERVER_IP);
            Scanner scanner = new Scanner(System.in);
            while (true) {
                System.out.print("Enter number: ");
                String message = scanner.nextLine();
                if (message.equalsIgnoreCase("exit")) {
                    break;
                }
                try{
                    Integer.parseInt(message);
                }
                catch (NumberFormatException e) {
                    System.out.println("Invalid input. Please enter a number.");
                    continue;
                }
                byte[] sendBuffer = message.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendBuffer,
sendBuffer.length, serverAddress, SERVER_PORT);
                clientSocket.send(sendPacket);
                byte[] receiveBuffer = new byte[1024];
                DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
receiveBuffer.length);
                clientSocket.receive(receivePacket);
                String response = new String(receivePacket.getData(), 0,
receivePacket.getLength());
                System.out.println("Number of digits(Server Response): " + response);
            }

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

Output:



```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Server1_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Server1_23BCE1156
UDP Server is running on port 12345
Received from /127.0.0.1:34821 - 100
Sending response: 3
Received from /127.0.0.1:34821 - 00
Sending response: 1
Received from /127.0.0.1:57837 - 000
Sending response: 1
Received from /127.0.0.1:60375 - 20
Sending response: 2
^Cvignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Client1_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Client1_23BCE1156
Enter number: Hi
Invalid input. Please enter a number.
Enter number: 100
Number of digits(Server Response): 3
Enter number: 00
Number of digits(Server Response): 1
Enter number: exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ ^C
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Client1_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Client1_23BCE1156
Enter number: 000
Number of digits(Server Response): 1
Enter number: exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Client1_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Client1_23BCE1156
Enter number: 20
Number of digits(Server Response): 2
Enter number: exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

2.

Question:

Develop a minimal chat application using sinlge udp client server application.

Code:

Server:

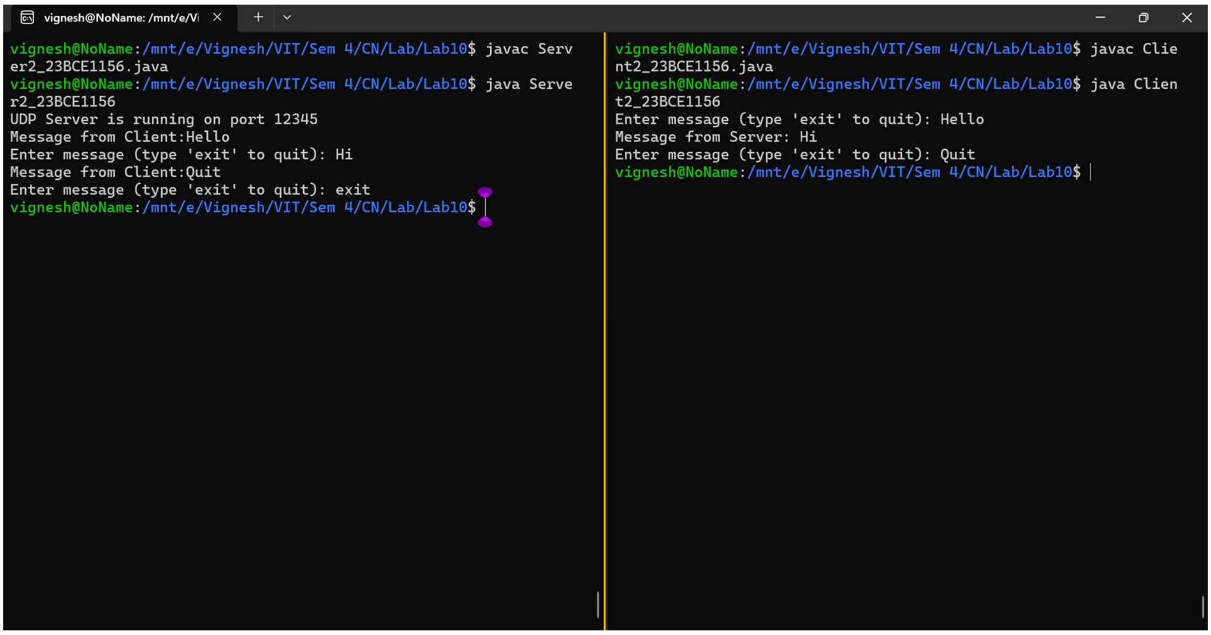
```
import java.net.*;
import java.util.*;
public class Server2_23BCE1145{
    public static void main(String[] args) {
        final int SERVER_PORT = 12345;
        DatagramSocket serverSocket = null;
        Scanner sc=new Scanner(System.in);
        try {
            serverSocket = new DatagramSocket(SERVER_PORT);
            System.out.println("UDP Server is running on port " + SERVER_PORT);
            while (true){
                byte[] receiveBuffer = new byte[1024];
                DatagramPacket receivePacket = new DatagramPacket(receiveBuffer, receiveBuffer.length);
                serverSocket.receive(receivePacket);

                String receivedMessage = new String(receivePacket.getData(), 0, receivePacket.getLength());
                if (receivedMessage.equalsIgnoreCase("exit")) {
                    System.out.println("Client has exited. Shutting down server.");
                    break;
                }
                System.out.println("Message from Client:" + receivedMessage);
                System.out.print("Enter message (type 'exit' to quit): ");
                String msg=sc.nextLine();
                byte[] sendBuffer = msg.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, receivePacket.getAddress(), receivePacket.getPort());
                serverSocket.send(sendPacket);
                if (msg.equalsIgnoreCase("exit")) {
                    break;
                }
            }

        }
        catch (Exception e) {
            e.printStackTrace();
        }
        finally{
            serverSocket.close();
            sc.close();
        }
    }
}
```

Client:

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

public class Client2_23BCE1145 {
    public static void main(String[] args) {
        final int SERVER_PORT = 12345;

        try {
            DatagramSocket clientSocket = new DatagramSocket();
            InetAddress serverAddress = InetAddress.getLocalHost();
            Scanner scanner = new Scanner(System.in);

            while (true) {
                System.out.print("Enter message (type 'exit' to quit): ");
                String message = scanner.nextLine();
                byte[] sendBuffer = message.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length,
serverAddress, SERVER_PORT);
```

```
                clientSocket.send(sendPacket);
                if (message.equalsIgnoreCase("exit")) {
                    break;
                }
                byte[] receiveBuffer = new byte[1024];
                DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
receiveBuffer.length);
                clientSocket.receive(receivePacket);
                String response = new String(receivePacket.getData(), 0,
receivePacket.getLength());
                if (response.equalsIgnoreCase("exit")) {
                    break;
                }
                System.out.println("Message from Server: " + response);
            }
            clientSocket.close();
            scanner.close();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Output:



```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Serv
er2_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Serve
r2_23BCE1156
UDP Server is running on port 12345
Message from Client:Hello
Enter message (type 'exit' to quit): Hi
Message from Client:Quit
Enter message (type 'exit' to quit): exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Clie
nt2_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Clien
t2_23BCE1156
Enter message (type 'exit' to quit): Hello
Message from Server: Hi
Enter message (type 'exit' to quit): Quit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

3.

Question:

Develop a udp data transfer between single client and server using CRC error control mechanism.

Code:

sServer:

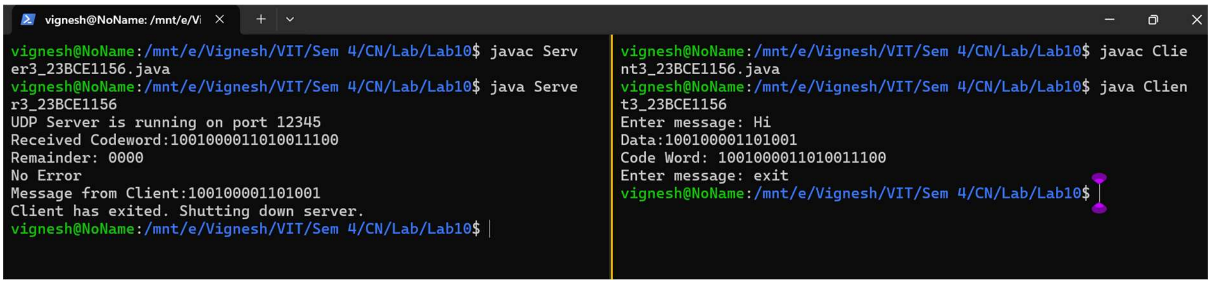
```
import java.net.*;

import java.util.*;

public class Server3_23BCE1145{

    public static void main(String[] args) {

        final int SERVER_PORT = 12345;

        DatagramSocket serverSocket = null;

        Scanner sc=new Scanner(System.in);

        String divisor="10011";

        try {

            serverSocket = new DatagramSocket(SERVER_PORT);

            System.out.println("UDP Server is running on port " + SERVER_PORT);

            while (true){

                byte[] receiveBuffer = new byte[1024];
                DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
receiveBuffer.length);
                serverSocket.receive(receivePacket);


                String receivedMessage = new String(receivePacket.getData(), 0,
receivePacket.getLength());
                if (receivedMessage.equalsIgnoreCase("exit")) {
                    System.out.println("Client has exited. Shutting down server.");
                    break;
                }
                // receivedMessage=xor(receivedMessage,"0".repeat(receivedMessage.length()-
            4)+"1010");                 //Uncomment for error
                System.out.println("Received Codeword:"+ receivedMessage);
                String r=divide(receivedMessage,divisor).substring(1);
                System.out.println("Remainder: "+ r);
                if (r.equals("0".repeat(divisor.length()-1))){
                    System.out.println("No Error");
                    System.out.println("Message from Client:" +
receivedMessage.substring(0,receivedMessage.length()-4));
                }
                else{
                    System.out.println("Error Detected");
                }


            }


        }
        catch (Exception e) {
            e.printStackTrace();
        }
        finally{
            serverSocket.close();
            sc.close();
        }
    }
    public static String divide(String dividend,String divisor){
        String q="",r="";
        int n=dividend.length()-divisor.length();
        for (int i=0;i<n+1;i++){
            if (dividend.charAt(i)=='0'){
                q+='0';
                r=xor(dividend.substring(i,i+divisor.length()),"0".repeat(divisor.length()));
            }
            else{
                q+='1';
                r=xor(dividend.substring(i,i+divisor.length()),divisor);
                dividend=dividend.substring(0,i)+r+dividend.substring(i+divisor.length());
            }
        }
        return r;
    }
    public static String xor(String a,String b){
        String c="";
        for (int i=0;i<a.length();i++){
            int a1=Integer.parseInt(a.charAt(i)+"");
            int b1=Integer.parseInt(b.charAt(i)+"");
            c+=(a1^b1)+"";
        }
        return c;
    }
}
```

Client:

```
import java.math.BigInteger;
import java.net.*;
import java.util.*;

public class Client3_23BCE1145 {
    public static void main(String[] args) {
        final int SERVER_PORT = 12345;
        String divisor="10011";
        String r="";
        try {
            DatagramSocket clientSocket = new DatagramSocket();
            InetAddress serverAddress = InetAddress.getLocalHost();
            Scanner scanner = new Scanner(System.in);
            while (true) {
                System.out.print("Enter message: ");
                String message = scanner.nextLine();
                if (message.equalsIgnoreCase("exit")) {
                    byte[] sendBuffer = message.getBytes();
                    DatagramPacket sendPacket = new DatagramPacket(sendBuffer,
sendBuffer.length, serverAddress, SERVER_PORT);
                    clientSocket.send(sendPacket);
                    break;
                }
                message=new BigInteger(message.getBytes()).toString(2);
                System.out.println("Data:"+message);
                for (int i=0;i<divisor.length()-1;i++)
                    message+='0';
                r=divide(message,divisor).substring(1);
                message=message.substring(0,message.length()-4)+r;
                System.out.println("Code Word: "+message);
                byte[] sendBuffer = message.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length,
serverAddress, SERVER_PORT);
                clientSocket.send(sendPacket);


            }
            clientSocket.close();
            scanner.close();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
    public static String divide(String dividend,String divisor){
        String q="",r="";
        int n=dividend.length()-divisor.length();
        for (int i=0;i<n+1;i++){
            if (dividend.charAt(i)=='0'){
                q+='0';
                r=xor(dividend.substring(i,i+divisor.length()),"0".repeat(divisor.length()));
            }
            else{
                q+='1';
                r=xor(dividend.substring(i,i+divisor.length()),divisor);
                dividend=dividend.substring(0,i)+r+dividend.substring(i+divisor.length());
            }
        }
        return r;
    }
    public static String xor(String a,String b){
        String c="";
        for (int i=0;i<a.length();i++){
            int a1=Integer.parseInt(a.charAt(i)+"");
            int b1=Integer.parseInt(b.charAt(i)+"");
            c+=(a1^b1)+"";
        }
        return c;
    }
}
```

Output:

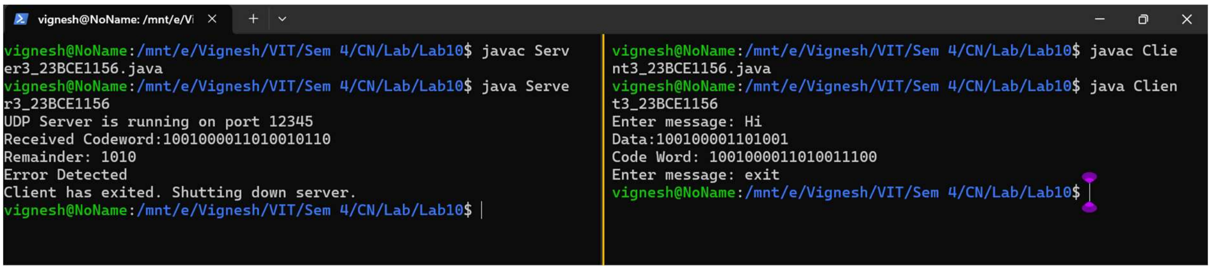
Without Error:



```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Serv
er3_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Serve
r3_23BCE1156
UDP Server is running on port 12345
Received Codeword:1001000011010011100
Remainder: 0000
No Error
Message from Client:100100001101001
Client has exited. Shutting down server.
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Clie
nt3_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Clien
t3_23BCE1156
Enter message: Hi
Data:100100001101001
Code Word: 1001000011010011100
Enter message: exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

With Error:



```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Serv
er3_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Serve
r3_23BCE1156
UDP Server is running on port 12345
Received Codeword:1001000011010010110
Remainder: 1010
Error Detected
Client has exited. Shutting down server.
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Clie
nt3_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Clien
t3_23BCE1156
Enter message: Hi
Data:100100001101001
Code Word: 1001000011010011100
Enter message: exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

4.

Question:

Develop a udp data transfer between single client and server using Select Repeat protocol and show the intermediate steps outputs in detail. Assume that client sends server "hello how are you? Welcome to CN lab. Have a great learning". Assume that m=3 and 3rd frame lost and 5th acknowledgement lost. Show the out of order delivery in receiver's side and how SR protocol takes care of order of delivery.

Code:

Server:

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

public class Server4_23BCE1145 {
   private static int m = 3;
   private static int ws = (int) Math.pow(2, m - 1);
   private static HashMap<Integer, String> receivedFrames = new HashMap<>();

   public static void main(String[] args) {
      final int SERVER_PORT = 12345;
      DatagramSocket serverSocket = null;

      try {
         serverSocket = new DatagramSocket(SERVER_PORT);
         System.out.println("UDP Server is running on port " + SERVER_PORT);

         while (true) {
            byte[] receiveBuffer = new byte[1024];
            DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
receiveBuffer.length);
            serverSocket.receive(receivePacket);
            String receivedMessage = new String(receivePacket.getData(), 0,
receivePacket.getLength());
            if (receivedMessage.equalsIgnoreCase("exit")) {
               System.out.println("Closing server.");
               break;
            }
```

```
        String[] parts = receivedMessage.split(" ");
        String frameData = parts[0];
        int seq = Integer.parseInt(parts[1]);

        if (!receivedFrames.containsKey(seq)) {
          receivedFrames.put(seq, frameData);
          System.out.println("Received Frame with Seq No. " + seq + ": " + frameData);
          byte[] sendBuffer = String.valueOf(seq).getBytes();
          DatagramPacket sendPacket = new DatagramPacket(sendBuffer,
sendBuffer.length, receivePacket.getAddress(), receivePacket.getPort());
          if (seq!=4)              // Comment for no ack loss
          serverSocket.send(sendPacket);
          System.out.println("ACK "+ seq + " sent to client.");
        }
        else {
          System.out.println("Duplicate Frame " + seq + " discarded.");
          byte[] sendBuffer = String.valueOf(seq).getBytes();
          DatagramPacket sendPacket = new DatagramPacket(sendBuffer,
sendBuffer.length, receivePacket.getAddress(), receivePacket.getPort());
          serverSocket.send(sendPacket);
        }
      }
    }
    catch (Exception e) {
      e.printStackTrace();
    }
    finally {
      if (serverSocket != null) serverSocket.close();
    }
  }
}
```

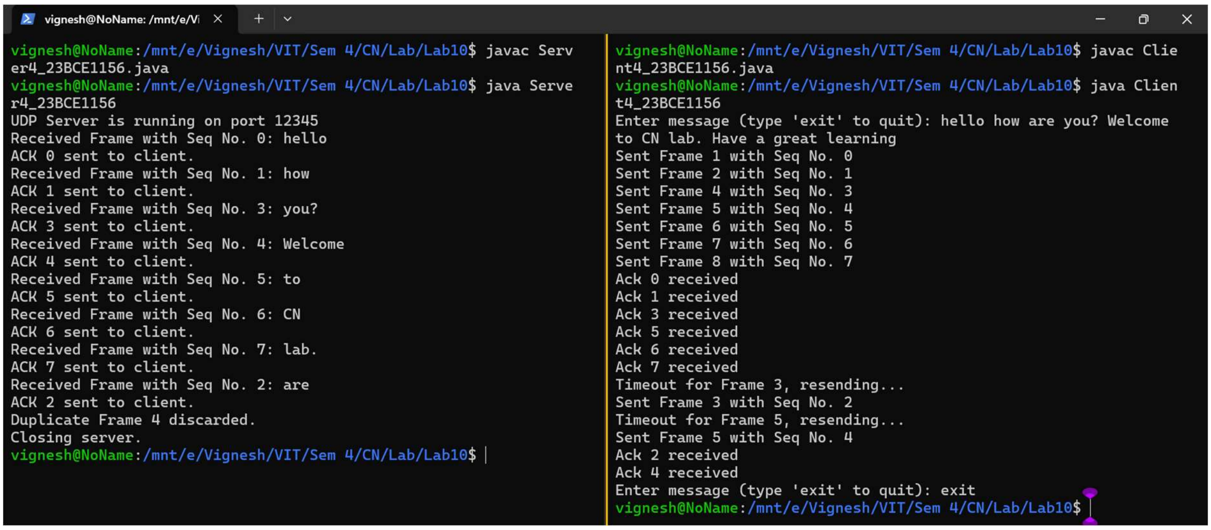
Client:

```
import java.net.*;
import java.util.*;
import java.io.*;
public class Client4_23BCE1145 {
    private static int m = 3;
    private static int ws = (int) Math.pow(2, m - 1);
    private static HashMap<Integer, String> windows = new HashMap<>();
    private static Set<Integer> acks = Collections.synchronizedSet(new HashSet<>());
    private static DatagramSocket clientSocket;
    private static InetAddress serverAddress;

    public static void main(String[] args) {
        final int SERVER_PORT = 12345;
        Scanner scanner = new Scanner(System.in);

        try {
            clientSocket = new DatagramSocket();
            serverAddress = InetAddress.getLocalHost();

            while (true) {
                System.out.print("Enter message (type 'exit' to quit): ");
                String message = scanner.nextLine();

                if (message.equalsIgnoreCase("exit")) {
                    byte[] sendBuffer = message.getBytes();
                    DatagramPacket sendPacket = new DatagramPacket(sendBuffer,
sendBuffer.length, serverAddress, SERVER_PORT);
```

```
            clientSocket.send(sendPacket);
            break;
        }

        String[] arr = message.split(" ");
        int totalFrames = arr.length;
        for (int i = 0; i < totalFrames; i++) {
            windows.put(i, arr[i]);
            acks.add(i % (ws * 2));
        }
        new Thread(() -> receiveACKs()).start();

        int fc = 0;
        while (!acks.isEmpty()) {
            for (int i = 0; i < ws*2 && fc < totalFrames; i++) {
                if (acks.contains(fc)) {
                    if (fc!=2)                  //Comment for no data loss
                    send(fc);
                }
                fc++;
            }
            Thread.sleep(3000);
            resendUnacknowledgedFrames();
        }
    }
}
catch (Exception e) {
    e.printStackTrace();
}
finally {
```

```
            clientSocket.close();
            scanner.close();
        }
    }

    public static void send(int i) {
        try {
            String msg = windows.get(i) + " " + (i % (ws * 2));
            byte[] sendBuffer = msg.getBytes();
            DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length,
serverAddress, 12345);
            clientSocket.send(sendPacket);
            System.out.println("Sent Frame " + (i + 1) + " with Seq No. " + (i % (ws * 2)));
        }
        catch (Exception e) {
            System.out.println(e);
        }
    }

    public static void receiveACKs() {
        try {
            while (!acks.isEmpty()) {
                byte[] ackBuffer = new byte[1024];
                DatagramPacket ackPacket = new DatagramPacket(ackBuffer, ackBuffer.length);
                clientSocket.receive(ackPacket);

                String ack = new String(ackPacket.getData(), 0, ackPacket.getLength());
                int ackNum = Integer.parseInt(ack);

                if (acks.contains(ackNum)) {
                    System.out.println("Ack " + ackNum + " received");
                    acks.remove(ackNum);
                }
            }
        }
        catch (IOException e) {
            System.out.println(e);
        }
    }

    public static void resendUnacknowledgedFrames() {
        for (int index : new ArrayList<>(acks)) {
            if (acks.contains(index)) {
                System.out.println("Timeout for Frame " + index + ", resending...");
                send(index);
            }
        }
    }

}
```

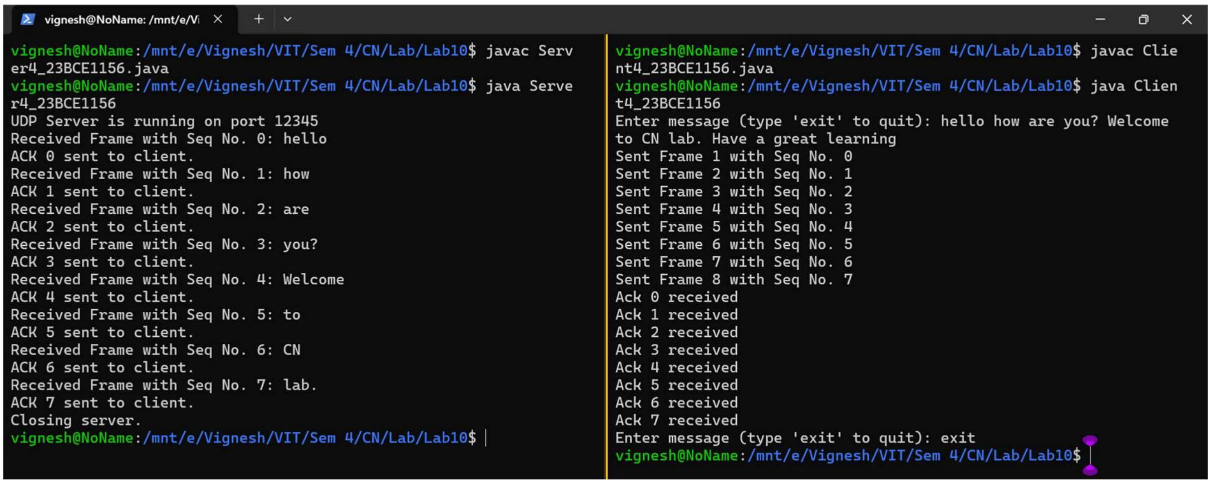
Output:



```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Serv
er4_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Serve
r4_23BCE1156
UDP Server is running on port 12345
Received Frame with Seq No. 0: hello
ACK 0 sent to client.
Received Frame with Seq No. 1: how
ACK 1 sent to client.
Received Frame with Seq No. 3: you?
ACK 3 sent to client.
Received Frame with Seq No. 4: Welcome
ACK 4 sent to client.
Received Frame with Seq No. 5: to
ACK 5 sent to client.
Received Frame with Seq No. 6: CN
ACK 6 sent to client.
Received Frame with Seq No. 7: lab.
ACK 7 sent to client.
Received Frame with Seq No. 2: are
ACK 2 sent to client.
Duplicate Frame 4 discarded.
Closing server.
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ |
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Clie
nt4_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Clien
t4_23BCE1156
Enter message (type 'exit' to quit): hello how are you? Welcome
to CN lab. Have a great learning
Sent Frame 1 with Seq No. 0
Sent Frame 2 with Seq No. 1
Sent Frame 4 with Seq No. 3
Sent Frame 5 with Seq No. 4
Sent Frame 6 with Seq No. 5
Sent Frame 7 with Seq No. 6
Sent Frame 8 with Seq No. 7
Ack 0 received
Ack 1 received
Ack 3 received
Ack 5 received
Ack 6 received
Ack 7 received
Timeout for Frame 3, resending...
Sent Frame 3 with Seq No. 2
Timeout for Frame 5, resending...
Sent Frame 5 with Seq No. 4
Ack 2 received
Ack 4 received
Enter message (type 'exit' to quit): exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

With no loss:



```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Serv
er4_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Serve
r4_23BCE1156
UDP Server is running on port 12345
Received Frame with Seq No. 0: hello
ACK 0 sent to client.
Received Frame with Seq No. 1: how
ACK 1 sent to client.
Received Frame with Seq No. 2: are
ACK 2 sent to client.
Received Frame with Seq No. 3: you?
ACK 3 sent to client.
Received Frame with Seq No. 4: Welcome
ACK 4 sent to client.
Received Frame with Seq No. 5: to
ACK 5 sent to client.
Received Frame with Seq No. 6: CN
ACK 6 sent to client.
Received Frame with Seq No. 7: lab.
ACK 7 sent to client.
Closing server.
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```

```
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ javac Clie
nt4_23BCE1156.java
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$ java Clien
t4_23BCE1156
Enter message (type 'exit' to quit): hello how are you? Welcome
to CN lab. Have a great learning
Sent Frame 1 with Seq No. 0
Sent Frame 2 with Seq No. 1
Sent Frame 3 with Seq No. 2
Sent Frame 4 with Seq No. 3
Sent Frame 5 with Seq No. 4
Sent Frame 6 with Seq No. 5
Sent Frame 7 with Seq No. 6
Sent Frame 8 with Seq No. 7
Ack 0 received
Ack 1 received
Ack 2 received
Ack 3 received
Ack 4 received
Ack 5 received
Ack 6 received
Ack 7 received
Enter message (type 'exit' to quit): exit
vignesh@NoName:/mnt/e/Vignesh/VIT/Sem 4/CN/Lab/Lab10$
```