

Lab Experiment 4

Simulate N-bit Sliding Window protocol, at data link layer.

Sender program:

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

class Dataframe implements Serializable {
    public char info;
    public int sequenceNumber;

    Dataframe() {
    }

    Dataframe(char data, int sequenceNumber) {
        this.info = data;
        this.sequenceNumber = sequenceNumber;
    }
}

class Acknowledgement implements Serializable {
    public boolean ack;
    public int sequenceNumber;

    Acknowledgement() {
    }

    Acknowledgement(boolean ack, int sequenceNumber) {
        this.ack = ack;
        this.sequenceNumber = sequenceNumber;
    }
}

public class Client {

    public static void main(String[] args) {
        try {
            System.out.println("Client Started");
            Socket socket = new Socket("localhost", 12345);

            ObjectOutputStream objectOutputStream = new
ObjectOutputStream(socket.getOutputStream());
            ObjectInputStream objectInputStream = new
ObjectInputStream(socket.getInputStream());

            int noOfBits = 16;
            int windowSize = 5;
```

```

        DataFrame[] data = new DataFrame[noOfBits];
        int left_ptr = 0;
        int right_ptr = 0;

        while (left_ptr < noOfBits) {
            while (right_ptr < noOfBits && (right_ptr - left_ptr) <
windowSize) {
                data[right_ptr] = new DataFrame((char) ('a' + right_ptr),
right_ptr);

                objectOutputStream.writeObject(data[right_ptr]);
                System.out.println("Sent: " + data[right_ptr].info);
                right_ptr++;

            }
            Acknowledgement ACK = (Acknowledgement)
objectInputStream.readObject();
            if (ACK.ack) {
                System.out.println("Acknowledgment received for Frame " +
ACK.sequenceNumber);
                left_ptr++;
            } else {
                System.out.println("NACK received for Frame " +
ACK.sequenceNumber + ". Retransmitting.");
                int resendIndex = ACK.sequenceNumber;
                objectOutputStream.writeObject(data[resendIndex]);
                System.out.println("Sent (Retransmit): " +
data[resendIndex].info);
            }

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

```

Receiver program:

```

import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.HashMap;
import java.util.Map;

class DataFrame implements Serializable {
    public char info;
}

```

```

    public int sequenceNumber;

    DataFrame() {
    }

    DataFrame(char data, int sequenceNumber) {
        this.info = data;
        this.sequenceNumber = sequenceNumber;
    }
}

class Acknowledgement implements Serializable {
    public boolean ack;
    public int sequenceNumber;

    Acknowledgement() {
    }

    Acknowledgement(boolean ack, int sequenceNumber) {
        this.ack = ack;
        this.sequenceNumber = sequenceNumber;
    }
}

public class Server {
    public static void main(String[] args) {
        try {
            System.out.println("Waiting for clients");
            ServerSocket ss = new ServerSocket(12345);
            Socket s = ss.accept();
            System.out.println("Client connected");

            ObjectInputStream objectInputStream = new
ObjectInputStream(s.getInputStream());
            ObjectOutputStream objectOutputStream = new
ObjectOutputStream(s.getOutputStream());

            Map<Integer, Boolean> explicitError = new HashMap<>();
            explicitError.put(3, true);
            explicitError.put(5, true);
            explicitError.put(15, true);
            int framesReceived = 0;
            while (framesReceived < 16) {
                DataFrame receivedFrame = (DataFrame)
objectInputStream.readObject();
                framesReceived++;
                if (explicitError.containsKey(receivedFrame.sequenceNumber)
&& explicitError.get(receivedFrame.sequenceNumber)) {

```

```

        Acknowledgement ACK = new Acknowledgement(false,
receivedFrame.sequenceNumber);
        objectOutputStream.writeObject(ACK);
        System.out.println("Sent NACK for " +
receivedFrame.sequenceNumber);
        explicitError.put(receivedFrame.sequenceNumber, false);
        framesReceived--;
    } else {
        System.out.println("Received: " + receivedFrame.info);
        Acknowledgement ACK = new Acknowledgement(true,
receivedFrame.sequenceNumber);
        objectOutputStream.writeObject(ACK);
        System.out.println("Sent Acknowledgement for " +
receivedFrame.sequenceNumber);
    }

    }
} catch (

Exception e) {
    e.printStackTrace();
}

}
}

```

Sender Output:

```
Windows PowerShell
PS D:\SRM AP\SEM 5\Computer Networks Lab\Week4> javac .\Client.java
PS D:\SRM AP\SEM 5\Computer Networks Lab\Week4> java Client
Client Started
Sent: a
Sent: b
Sent: c
Sent: d
Sent: e
Acknowledgment received for Frame 0
Sent: f
Acknowledgment received for Frame 1
Sent: g
Acknowledgment received for Frame 2
Sent: h
NACK received for Frame 3. Retransmitting.
Sent (Retransmit): d
Acknowledgment received for Frame 4
Sent: i
NACK received for Frame 5. Retransmitting.
Sent (Retransmit): f
Acknowledgment received for Frame 6
Sent: j
Acknowledgment received for Frame 7
Sent: k
Acknowledgment received for Frame 3
Sent: l
Acknowledgment received for Frame 8
Sent: m
Acknowledgment received for Frame 5
Sent: n
Acknowledgment received for Frame 9
Sent: o
Acknowledgment received for Frame 10
Sent: p
Acknowledgment received for Frame 11
Acknowledgment received for Frame 12
Acknowledgment received for Frame 13
Acknowledgment received for Frame 14
NACK received for Frame 15. Retransmitting.
Sent (Retransmit): p
Acknowledgment received for Frame 15
PS D:\SRM AP\SEM 5\Computer Networks Lab\Week4>
```

Receiver Output:

```
Windows PowerShell
Received: p
Sent Acknowledgement for 15
PS D:\SRM AP\SEM 5\Computer Networks Lab\Week4> javac .\Server.java
PS D:\SRM AP\SEM 5\Computer Networks Lab\Week4> java Server
Waiting for clients
Client connected
Received: a
Sent Acknowledgement for 0
Received: b
Sent Acknowledgement for 1
Received: c
Sent Acknowledgement for 2
Sent NACK for 3
Received: e
Sent Acknowledgement for 4
Sent NACK for 5
Received: g
Sent Acknowledgement for 6
Received: h
Sent Acknowledgement for 7
Received: d
Sent Acknowledgement for 3
Received: i
Sent Acknowledgement for 8
Received: f
Sent Acknowledgement for 5
Received: j
Sent Acknowledgement for 9
Received: k
Sent Acknowledgement for 10
Received: l
Sent Acknowledgement for 11
Received: m
Sent Acknowledgement for 12
Received: n
Sent Acknowledgement for 13
Received: o
Sent Acknowledgement for 14
Sent NACK for 15
Received: p
Sent Acknowledgement for 15
PS D:\SRM AP\SEM 5\Computer Networks Lab\Week4>
```