**CS 6111- Computer Networks**

**LAB 9 – 2nd November 2023**

**Sliding Window Protocols**

**SHIVANI SURESH**

**2021503050**

**Go-Back-N:**

**Programs:**

**Server:**

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class Go\_Back\_N\_Server {

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

ServerSocket server = new ServerSocket(55004);

System.out.println("Server established");

Socket client = server.accept();

ObjectOutputStream oos = new ObjectOutputStream(client.getOutputStream());

ObjectInputStream ois = new ObjectInputStream(client.getInputStream());

System.out.println("Client is now connected");

int x = (Integer) ois.readObject();

int k = (Integer) ois.readObject();

int j = 0;

int i = (Integer) ois.readObject();

boolean flag = true;

Random r = new Random();

int mod = r.nextInt(6);

while (mod == 1 || mod == 0) {

mod = r.nextInt(6);

}

while (true) {

int c = k;

for (int h = 0; h <= x; h++) {

System.out.print("|" + c + "|");

c = (c + 1) % x;

}

System.out.println();

System.out.println();

if (k == j) {

System.out.println("Frame " + k + " received" + "\n" + "Data:" + j);

j++;

System.out.println();

} else {

System.out.println(

"Frames received not in correct order" +

"\n" +

"Expected frame:" +

j +

"\n" +

"Received frame no :" +

k

);

System.out.println();

}

if (j % mod == 0 && flag) {

System.out.println("Error found. Acknowledgement not sent. ");

flag = !flag;

j--;

} else if (k == j - 1) {

oos.writeObject(k);

System.out.println("Acknowledgement sent");

}

System.out.println();

if (j % mod == 0) {

flag = !flag;

}

k = (Integer) ois.readObject();

if (k == -1) {

break;

}

i = (Integer) ois.readObject();

}

System.out.println("Client finished sending data. Exiting");

oos.writeObject(-1);

}

}

**Client:**

import java.io.\*;

import java.net.\*;

public class Go\_Back\_N\_Client {

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

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

System.out.print("Enter the value of m : ");

int m = Integer.parseInt(br.readLine());

int x = (int) (Math.pow(2, m) - 1);

System.out.print("Enter no. of frames to be sent: ");

int count = Integer.parseInt(br.readLine());

int data[] = new int[count];

int h = 0;

for (int i = 0; i < count; i++) {

System.out.print("Enter data for frame no " + h + " => ");

data[i] = Integer.parseInt(br.readLine());

h = (h + 1) % x;

}

Socket client = new Socket("localhost", 55004);

ObjectInputStream ois = new ObjectInputStream(client.getInputStream());

ObjectOutputStream oos = new ObjectOutputStream(client.getOutputStream());

System.out.println("Connected with the server");

boolean flag = false;

GoBackNListener listener = new GoBackNListener(ois, x);

listener.t.start();

int start = 0;

h = 0;

oos.writeObject(x);

do {

int c = h;

for (int i = h; i < count; i++) {

System.out.print("|" + c + "|");

c = (c + 1) % x;

}

System.out.println();

System.out.println();

h = start;

for (int i = start; i < count; i++) {

System.out.println("Sending frame:" + h);

h = (h + 1) % x;

System.out.println();

oos.writeObject(i);

oos.writeObject(data[i]);

Thread.sleep(100);

}

listener.t.join(4000);

if (listener.reply != count - 1) {

System.out.println(

"No reply from the server in 4 seconds. Resending data from frame no " +

(listener.reply + 1)

);

System.out.println(

"Listener.reply value: " + listener.reply + "\nx-1: " + (x - 1)

);

System.out.println();

start = listener.reply + 1;

flag = false;

} else {

System.out.println("All elements sent successfully. Exiting");

flag = true;

}

} while (!flag);

oos.writeObject(-1);

}

}

class GoBackNListener implements Runnable {

Thread t;

ObjectInputStream ois;

int reply, x;

GoBackNListener(ObjectInputStream o, int i) {

t = new Thread(this);

ois = o;

reply = -2;

x = i;

}

@Override

public void run() {

try {

int temp = 0;

while (reply != -1) {

reply = (Integer) ois.readObject();

if (reply != -1 && reply != temp + 1) reply = temp;

if (reply != -1) {

temp = reply;

System.out.println(

"Acknowledgement of frame no " + (reply % x) + " received."

);

System.out.println();

}

}

reply = temp;

} catch (Exception e) {

System.out.println("Exception => " + e);

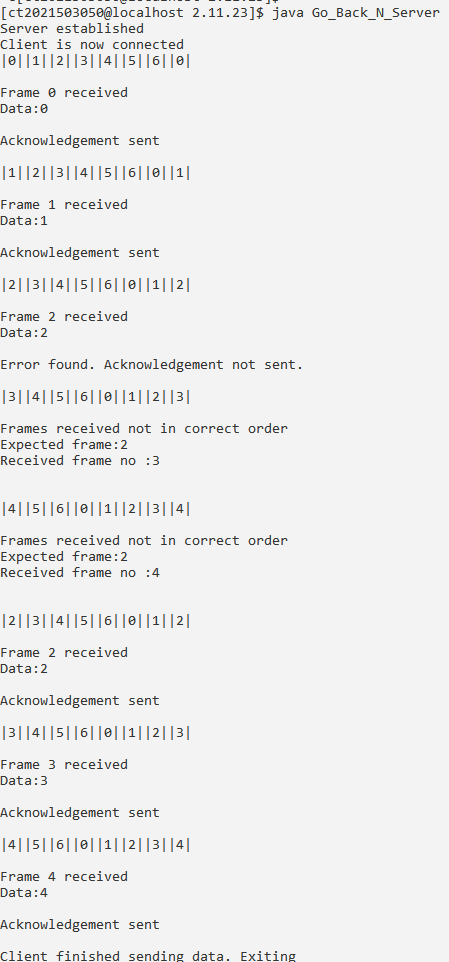
}

}

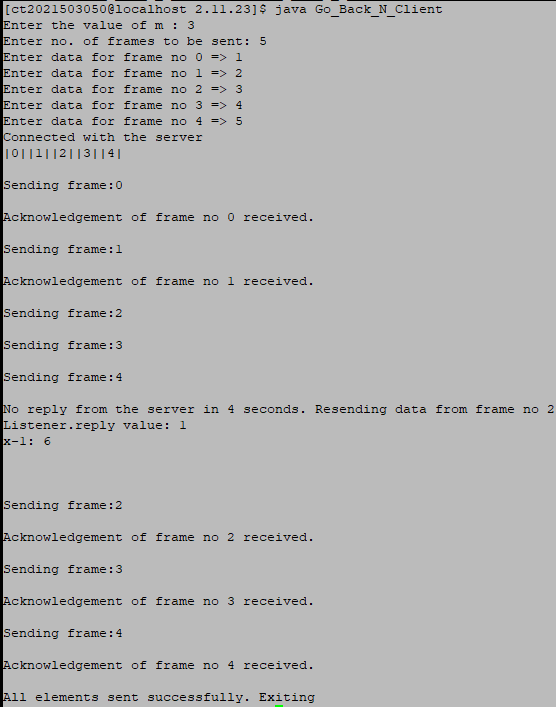
}

**Output:**

**Server:**



**Client:**



**Selective Repeat**

**Program:**

**Server:**

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

import java.net.SocketException;

public class SRs {

static ServerSocket serverSocket;

static DataInputStream dis;

static DataOutputStream dos;

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

try {

int[] a = { 30, 40, 50, 60, 70, 80, 90, 100 };

serverSocket = new ServerSocket(8011);

System.out.println("Waiting for connection");

Socket client = serverSocket.accept();

dis = new DataInputStream(client.getInputStream());

dos = new DataOutputStream(client.getOutputStream());

System.out.println("The number of packets sent is: " + a.length);

int y = a.length;

dos.write(y);

dos.flush();

for (int i = 0; i < a.length; i++) {

dos.write(a[i]);

dos.flush();

}

int k = dis.read();

dos.write(a[k]);

dos.flush();

} catch (IOException e) {

System.out.println(e);

} finally {

try {

dis.close();

dos.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

**Client:**

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

import java.net.SocketException;

public class SRs {

static ServerSocket serverSocket;

static DataInputStream dis;

static DataOutputStream dos;

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

try {

int[] a = { 30, 40, 50, 60, 70, 80, 90, 100 };

serverSocket = new ServerSocket(8011);

System.out.println("Waiting for connection");

Socket client = serverSocket.accept();

dis = new DataInputStream(client.getInputStream());

dos = new DataOutputStream(client.getOutputStream());

System.out.println("The number of packets sent is: " + a.length);

int y = a.length;

dos.write(y);

dos.flush();

for (int i = 0; i < a.length; i++) {

dos.write(a[i]);

dos.flush();

}

int k = dis.read();

dos.write(a[k]);

dos.flush();

} catch (IOException e) {

System.out.println(e);

} finally {

try {

dis.close();

dos.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

**Output:**

**Server:**

****

**Client:**

