**List of Network Program (Software: C/JAVA)**

1. **TO IMPLEMENT DATE AND TIME DISPLAY FROM CLIENT TO SERVER USING TCP SOCKETS**
2. **TO IMPLEMENT A DNS SERVER AND CLIENT IN JAVA USING UDP SOCKETS**
3. **TO DEVELOP A CLIENT THAT CONTACTS A GIVEN DNS SERVER TO RESOLVE A GIVEN HOSTNAME**
4. **APPLICATIONS USING TCP ECHO SERVER AND CLIENT**
5. **APPLICATIONS USING TCP CHAT CLIENT AND CHAT SERVER**
6. **ARP PROTOCOLS**
7. **DISTANCE VECTOR ROUTING**
8. **APPLICATIONS USING TCP FILE TRANSFER**
9. **SIMULATION OF ERROR CORRECTION CODE – CRC**
10. **SLIDING WINDOW PROTOCOL**
11. Programs using TCP Sockets (Date and Time using server & client)

**AIM:** To implement date and time display from client to server using TCP Sockets

**DESCRIPTION:** TCP Server gets the system date and time and opens the server socket to read the client details. Client send its address to the server. Then client receives the date and time from server to display. TCP socket server client connection is opened for communication. After the date time is displayed the server client connection is closed with its respective streams to be closed.

**ALGORITHM:**

**Server** 1. Create a server socket and bind it to port. 2. Listen for new connection and when a connection arrives, accept it. 3. Send server‟s date and time to the client. 4. Read client‟s IP address sent by the client. 5. Display the client details. 6. Repeat steps 2-5 until the server is terminated. 7. Close all streams. 8. Close the server socket. 9. Stop.

**Client** 1. Create a client socket and connect it to the server‟s port number. 2. Retrieve its own IP address using built-in function. 3. Send its address to the server. 4. Display the date & time sent by the server. 5. Close the input and output streams. 6. Close the client socket. 7. Stop.

**PROGRAM: //TCP Date Server--tcpdateserver.java**

import java.net.\*;

import java.io.\*;

import java.util.\*;

class tcpdateserver

{

public static void main(String arg[])

{

ServerSocket ss = null;

Socket cs;

PrintStream ps;

BufferedReader dis;

String inet;

try

{

ss = new ServerSocket(4444);

System.out.println("Press Ctrl+C to quit");

while(true)

{ cs = ss.accept();

ps = new PrintStream(cs.getOutputStream());

Date d = new Date();

ps.println(d);

dis = new BufferedReader(new InputStreamReader(cs.getInputStream()));

inet = dis.readLine();

System.out.println("Client System/IP address is :"+ inet);

ps.close();

dis.close();

} }

catch(IOException e)

{ System.out.println("The exception is :" + e);

} } }

**// TCP Date Client--tcpdateclient.java**

import java.net.\*;

import java.io.\*;

class tcpdateclient

{ public static void main (String args[])

{ Socket soc;

BufferedReader dis;

String sdate;

PrintStream ps;

try { InetAddress ia = InetAddress.getLocalHost();

if (args.length == 0)

soc = new Socket(InetAddress.getLocalHost(),4444);

else soc = new Socket(InetAddress.getByName(args[0]),4444);

dis = new BufferedReader(new InputStreamReader(soc.getInputStream()));

sdate=dis.readLine();

System.out.println("The date/time on server is : " +sdate);

ps = new PrintStream(soc.getOutputStream());

ps.println(ia); ps.close();

catch(IOException e)

{

System.out.println("THE EXCEPTION is :" + e);

}

}

}

**Server:**

C:\Harini>javac htcpclient.java

C:\Harini>java htcpclient

**Client:**

C:\Harini>javac htcp.java

C:\Harini>java htcp

Date is : Sun Jun 18 18:14:30 IST 2023

2.Programs using UDP Sockets (like simple DNS)

**AIM:** To implement a DNS server and client in java using UDP sockets**.**

**DESCRIPTION** DNS stands for domain name system. unique name of the host is identified with its IP address through server client communication.

**ALGORITHM:**

**Server**

1. Create an array of hosts and its ip address in another array

2. Create a datagram socket and bind it to a port

3. Create a datagram packet to receive client request

4. Read the domain name from client to be resolved

5. Lookup the host array for the domain name

6. If found then retrieve corresponding address

7. Create a datagram packet and send ip address to client

8. Repeat steps 3-7 to resolve further requests from clients

9. Close the server socket

10. Stop

**Client**

1. Create a datagram socket

2. Get domain name from user

3. Create a datagram packet and send domain name to the server

4. Create a datagram packet to receive server message

5. Read server's response

6. If ip address then display it else display "Domain does not exist"

7. Close the client socket 8. Stop

**PROGRAM**

**// UDP DNS Server -- udpdnsserver.java**

import java.io.\*;

import java.net.\*;

public class udpdnsserver

{ private static int indexOf(String[] array, String str)

{ str = str.trim(); for (int i=0; i < array.length; i++)

{ if (array[i].equals(str)) return i;

}

return -1;

}

public static void main(String arg[])throws IOException

{ String[] hosts = {"yahoo.com", "gmail.com","cricinfo.com", "facebook.com"};

String[] ip = {"68.180.206.184", "209.85.148.19","80.168.92.140", "69.63.189.16"};

System.out.println("Press Ctrl + C to Quit");

while (true)

{ DatagramSocket serversocket=new DatagramSocket(1362);

byte[] senddata = new byte[1021];

byte[] receivedata = new byte[1021];

DatagramPacket recvpack = new DatagramPacket(receivedata, receivedata.length);

serversocket.receive(recvpack); String sen = new String(recvpack.getData());

InetAddress ipaddress = recvpack.getAddress();

int port = recvpack.getPort();

String capsent; System.out.println("Request for host " + sen);

if(indexOf (hosts, sen) != -1)

capsent = ip[indexOf (hosts, sen)];

else capsent = "Host Not Found";

senddata = capsent.getBytes();

DatagramPacket pack = new DatagramPacket(senddata, senddata.length,ipaddress,port);

serversocket.send(pack);

serversocket.close(); } } }

**//UDP DNS Client -- udpdnsclient.java**

import java.io.\*;

import java.net.\*;

public class udpdnsclient { public static void main(String args[])throws IOException

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

DatagramSocket clientsocket = new DatagramSocket(); InetAddress ipaddress;

if (args.length == 0)

ipaddress = InetAddress.getLocalHost();

else ipaddress = InetAddress.getByName(args[0]);

byte[] senddata = new byte[1024]; byte[] receivedata = new byte[1024];

int portaddr = 1362; System.out.print("Enter the hostname : ");

String sentence = br.readLine(); Senddata = sentence.getBytes(); DatagramPacket pack = new DatagramPacket(senddata,senddata.length, ipaddress,portaddr);

clientsocket.send(pack);

DatagramPacket recvpack =new DatagramPacket(receivedata,receivedata.length); clientsocket.receive(recvpack); String modified = new String(recvpack.getData());

System.out.println("IP Address: " + modified);

clientsocket.close(); }}

**OUTPUT**

**Server**

C:\Harini>java server

Press Ctrl + C to Quit

Request for host yahoo.com

**Client**

C:\Harini>java client

Enter the hostname : yahoo.com

IP Address: 68.180.206.184

3. **DNS SERVER TO RESOLVE A GIVEN HOST NAME**

**AIM:** To develop a client that contacts a given DNS server to resolve a given hostname.

**DESCRIPTION:** · Get the host name to be resolve using gethostname() · Check the host name using nslookup · Print the IP address, host name, Address length and Address type. · List the addresses stored in lookup

**ALGORITHM** Step 1. Find the host name by using gethostbyname() Step 2. The host name is followed by the list of alias names Step 3. Pointer points to the array of pointers to the individual address Step 4. For each address call the inet\_ntop() and print the returned string

**PROGRAM**

#include<stdio.h>

#include<netdb.h>

#include<arpa/inet.h>

#include<netinet/in.h>

int main(int argc,char\*\*argv)

{ char h\_name;

int h\_type;

struct hostent \*host;

struct in\_addr h\_addr;

if(argc!=2)

{ fprintf(stderr,"USAGE:nslookup\n");

}

if((host=gethostbyname(argv[1]))==NULL)

{

fprintf(stderr,"(mini)nslookup failed on %s\n",argv[1]); }

h\_addr.s\_addr=\*((unsigned long\*)host->h\_addr\_list[0]);

printf("\n IP ADDRESS=%s\n",inet\_ntoa(h\_addr));

printf("\n HOST NAME=%s\n",host->h\_name);

printf("\nADDRESS LENGTH =%d\n",host->h\_length);

printf("\nADDRESS TYPE=%d\n",host->h\_addrtype);

printf("\nLIST OF ADDRESS=%s\n",inet\_ntoa(h\_addr\_list[0]));

}

**OUTPUT**

[kali@Harini]$ vi dns.c

[kali@Harini ]$ cc dns.c

[kali@Harini]$ ./a.out

90.0.0.36

IP ADDRESS=90.0.0.36

HOST NAME=90.0.0.36

ADDRESS LENGTH =4

ADDRESS TYPE=2

LIST OF ADDRESS=90.0.0.36

1. APPLICATIONS USING TCP ECHO SERVER AND CLIENT

**PROGRAM CODING**:

**EchoClient.java** import java.io.\*; import java.net.\*; public class echoc

{

public static void main(String[] args)

{

try

{

Socket s = new Socket("127.0.0.1",9999); BufferedReader r = new BufferedReader(new

InputStreamReader(s.getInputStream()));

PrintWriter w = new PrintWriter(s.getOutputStream(),true); BufferedReader con = new BufferedReader (new

InputStreamReader(System.in));

String line; do

{

line = r.readLine(); if(line != null)

System.out.println(line); line = con.readLine(); w.println(line);

}

while(!line.trim().equals("bye"));

}catch(Exception e)

{

System.err.println(e);

}

}

}

**Echoserver.java**

import java.io.\*; import java.net.\*; public class echos

{

public echos(int portNum)

{

try

{server = new ServerSocket(portNum);} catch(Exception e)

{System.out.println(e);}

}

public void serve()

{ try

{

while(true)

{

Socket client = server.accept();

BufferedReader r = new BufferedReader(new InputStreamReader(client.getInputStream()));

PrintWriter w = new PrintWriter(client.getOutputStream(),true);

w.println("Welcome jeva Server Echo"); String line;

do

{

line = r.readLine(); if(line!=null)

w.println("Got: "+line);

}

while(!line.trim().equals("bye")); client.close();

}

}catch(Exception e)

{

System.err.println(e);

}

}

public static void main(String[] args)

{

echos s = new echos(9999); s.serve();

}

private ServerSocket server;

}

**OUTPUT:**

**SERVER:**

C:\Harini>javac server.java

C:\Harini>java server

hjii

Server Says : hjii

Client Says : hello

Im fine

Server Says : Im fine

Client Says : HOW ARE YOU

**CLIENT:**

C:\Harini>javac client.java

C:\Harini>java client

hello

Client Says :hello

Server says :hjii

HOW ARE YOU

Client Says :HOW ARE YOU

Server says :Im fine

1. **APPLICATIONS USING TCP CHAT CLIENT AND CHAT SERVER**

**PROGRAM CODING**:

**ChatClient.java**

import java.io.\*; import java.net.\*; class chatClient {

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

{

Socket sock=new Socket("127.00.1",1309);

BufferedReader keyRead=new BufferedReader(new InputStreamReader(System.in)); OutputStream ostream=sock.getOutputStream();

PrintWriter pwrite=new PrintWriter(ostream,true); InputStream istream=sock.getInputStream();

BufferedReader receiveRead =new BufferedReader(new InputStreamReader(istream)); System.out.println("Start the chitchat,type and press Enter key");

String receiveMessage,sendMessage; while(true)

{

sendMessage=keyRead.readLine(); pwrite.println(sendMessage); pwrite.flush();

if((receiveMessage=receiveRead.readLine())!=null)

{

System.out.println(receiveMessage);

}

}

}

}

**ChatServer.java**

import java.io.\*; import java.net.\*; class chatServer {

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

{

ServerSocket sersock=new ServerSocket(1309); System.out.println("server ready for chatting"); Socket sock=sersock.accept();

BufferedReader keyRead=new BufferedReader(new InputStreamReader(System.in)); OutputStream ostream=sock.getOutputStream();

PrintWriter pwrite=new PrintWriter(ostream,true); InputStream istream=sock.getInputStream();

BufferedReader receiveRead=new BufferedReader(new InputStreamReader(istream)); String receiveMessage,sendMessage;

while(true)

{

if((receiveMessage=receiveRead.readLine())!=null)

{

System.out.println(receiveMessage);

}

sendMessage=keyRead.readLine(); pwrite.println(sendMessage);

pwrite.flush();

}

}

}

**OUTPUT:**

***ChatClient***

C:\Harini>javac htcpclient.java

C:\Harini>java htcpclient

Start

hello

SERVER : hi im server

im client

***Chat Server***

C:\Harini>javac htcp.java

C:\Harini>java htcp

server ready for chatting

CLIENT : hello

hi im server

CLIENT : im client

1. **ARP PROTOCOLS**

**PROGRAM CODING**:

***arpclient.java*** import java.io.\*; import java.net.\*; import java.util.\*; public class arpc

{

public static void main(String args[])

{

try

{

BufferedReader in=new BufferedReader(new InputStreamReader(System.in)); Socket clsct=new Socket("127.0.0.1",200);

DataInputStream din=new DataInputStream(clsct.getInputStream()); DataOutputStream dout=new DataOutputStream(clsct.getOutputStream()); System.out.println("Enter the Logical address(IP):");

String str1=in.readLine(); dout.writeBytes(str1+'\n'); String str=din.readLine();

System.out.println("The Physical Address is: "+str); clsct.close();

}

catch (Exception e)

{

System.out.println(e);

}

}

}

***arpServer.java*** import java.io.\*; import java.net.\*; import java.util.\*;

public class arpServer {

public static void main(String args[])

{

try

{

ServerSocket obj=new ServerSocket(2005); Socket obj1=obj.accept();

while(true)

{

DataInputStream din=new DataInputStream(obj1.getInputStream()); DataOutputStream dout=new DataOutputStream(obj1.getOutputStream()); String str=din.readLine();

String ip[]={"165.165.80.80","165.165.79.1"};

String mac[]={"6A:08:AA:C2","8A:BC:E3:FA"};

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

{

if(str.equals(ip[i]))

{

dout.writeBytes(mac[i]+'\n'); break;

}

}obj.close();

}

}

catch(Exception e)

{

System.out.println(e);

}

}

}

**OUTPUT:**

**arpClient**

C:\Harini>javac arpc.java

C:\Harini>java arpc

Enter the Logical address(IP): 165.165.80.80

The Physical Address is:6A:08:AA:C2

1. **DISTANCE VECTOR ROUTING**

**PROGRAM CODING**:

import java.io.\*; public class DVR

{

static int graph[][]; static int via[][]; static int rt[][]; static int v;

static int e;

public static void main(String args[]) throws IOException

{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); System.out.println("Please enter the number of Vertices: ");

v = Integer.parseInt(br.readLine()); System.out.println("Please enter the number of Edges: "); e = Integer.parseInt(br.readLine());

graph = new int[v][v]; via = new int[v][v];

rt = new int[v][v]; for(int i = 0; i < v; i++) for(int j = 0; j < v; j++)

{

if(i == j) graph[i][j] = 0; else

graph[i][j] = 9999;

}

for(int i = 0; i < e; i++)

{

System.out.println("Please enter data for Edge " + (i + 1) + ":"); System.out.print("Source: ");

int s = Integer.parseInt(br.readLine()); s--;

System.out.print("Destination: ");

int d = Integer.parseInt(br.readLine()); d--;

System.out.print("Cost: ");

int c = Integer.parseInt(br.readLine()); graph[s][d] = c;

graph[d][s] = c;

}

dvr\_calc\_disp("The initial Routing Tables are: ");

System.out.print("Please enter the Source Node for the edge whose cost has changed: "); int s = Integer.parseInt(br.readLine());

s--;

System.out.print("Please enter the Destination Node for the edge whose cost has changed: ");

int d = Integer.parseInt(br.readLine()); d--;

System.out.print("Please enter the new cost: "); int c = Integer.parseInt(br.readLine()); graph[s][d] = c;

graph[d][s] = c;

dvr\_calc\_disp("The new Routing Tables are: ");

}

static void dvr\_calc\_disp(String message)

{

System.out.println(); init\_tables(); update\_tables(); System.out.println(message); print\_tables(); System.out.println();

}

static void update\_table(int source)

{

for(int i = 0; i < v; i++)

{

if(graph[source][i] != 9999)

{

int dist = graph[source][i]; for(int j = 0; j < v; j++)

{

int inter\_dist = rt[i][j]; if(via[i][j] == source) inter\_dist = 9999;

if(dist + inter\_dist < rt[source][j])

{

rt[source][j] = dist + inter\_dist; via[source][j] = i;

}

}

}

}

}

static void update\_tables()

{

int k = 0;

for(int i = 0; i < 4\*v; i++)

{

update\_table(k); k++;

if(k == v) k = 0;

}

}

static void init\_tables()

{

for(int i = 0; i < v; i++)

{

for(int j = 0; j < v; j++)

{

if(i == j)

{

rt[i][j] = 0;

45 Dept. of IT / CS8591 NetwoksLab

via[i][j] = i;

}

else

{

rt[i][j] = 9999;

via[i][j] = 100;

}

}

}

}

static void print\_tables()

{

for(int i = 0; i < v; i++)

{

for(int j = 0; j < v; j++)

{

System.out.print("Dist: " + rt[i][j] + " ");

}

System.out.println();

}

}

}

**OUTPUT:**

C:\Harini>javac dvr.java

C:\Harini>java dvr

Please enter the number of Vertices: 4 Please enter the number of Edges: 5 Please enter data for Edge 1:

Source: 1

Destination: 2

Cost: 1

Please enter data for Edge 2: Source: 1

Destination: 3

Cost: 3

Please enter data for Edge 3: Source: 2

Destination: 3

Cost: 1

Please enter data for Edge 4: Source: 2

Destination: 4

Cost: 1

Please enter data for Edge 5: Source: 3

Destination: 4

Cost: 4

|  |  |  |  |
| --- | --- | --- | --- |
| The initial Routing Tables are: Dist: 0 | Dist: 1 | Dist: 2 | Dist: 2 |
| Dist: 1 | Dist: 0 | Dist: 1 | Dist: 1 |
| Dist: 2 | Dist: 1 | Dist: 0 | Dist: 2 |
| Dist: 2 | Dist: 1 | Dist: 2 | Dist: 0 |

Please enter the Source Node for the edge whose cost has changed: 2 Please enter the Destination Node for the edge whose cost has changed: 4 Please enter the new cost: 10

The new Routing Tables are: Dist: 0 Dist: 1 Dist: 2 Dist: 6

Dist: 1 Dist: 0 Dist: 1 Dist: 5

Dist: 2 Dist: 1 Dist: 0 Dist: 4

Dist: 6 Dist: 5 Dist: 4 Dist: 0

1. **APPLICATIONS USING TCP FILE TRANSFER**

**PROGRAM CODING**:

**ftpclient.java** import java.net.\*; import java.io.\*;

public class ftpclient {

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

{

Socket sock=new Socket("127.0.0.1",3000); System.out.println("Enter the filename:");

BufferedReader keyRead=new BufferedReader (new InputStreamReader(System.in)); String fname=keyRead.readLine();

OutputStream ostream=sock.getOutputStream(); PrintWriter pwrite=new PrintWriter(ostream,true); pwrite.println(fname);

InputStream istream=sock.getInputStream();

BufferedReader socketRead=new BufferedReader(new InputStreamReader(istream)); String str;

while((str=socketRead.readLine())!=null)

{

System.out.println(str);

}

pwrite.close(); socketRead.close(); keyRead.close();

}

}

**ftpserver.java**

import java.io.\*; import java.net.\*; public class ftpserver {

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

{

ServerSocket sersock=new ServerSocket(3000); System.out.println("Server ready for connection"); Socket sock=sersock.accept();

System.out.println("connection is successful and waiting for chatting"); InputStream istream=sock.getInputStream();

BufferedReader fileRead=new BufferedReader(new InputStreamReader(istream)); String fname=fileRead.readLine();

BufferedReader contentRead=new BufferedReader (new FileReader(fname)); OutputStream ostream=sock.getOutputStream();

PrintWriter pwrite=new PrintWriter(ostream,true); String str; while((str=contentRead.readLine())!=null)

{

pwrite.println(str);

}

sock.close(); sersock.close(); pwrite.close(); fileRead.close();

contentRead.close();

}

}

**OTUPUT:**

***ftpclient***

C:\Harini>javac server.java

C:\Harini>java server

har.txt Hello FTP

***ftpserver***

C:\Harini>javac server.java

C:\Harini>java server

Server ready for connection

connection is successful and waiting for chatting

1. **SIMULATION OF ERROR CORRECTION CODE – CRC**

**PROGRAM CODING**:

import java.util.\*; class CRC {

public static void main(String args[]) {

Scanner scan = new Scanner(System.in); int n;

System.out.println("Enter the size of the data:"); n = scan.nextInt();

int data[] = new int[n]; System.out.println("Enter the data, bit by bit:"); for(int i=0 ; i < n ; i++) {

System.out.println("Enter bit number " + (n-i) + ":"); data[i] = scan.nextInt();

}

System.out.println("Enter the size of the divisor:"); n = scan.nextInt();

int divisor[] = new int[n]; System.out.println("Enter the divisor, bit by bit:"); for(int i=0 ; i < n ; i++) {

System.out.println("Enter bit number " + (n-i) + ":"); divisor[i] = scan.nextInt();

}

int remainder[] = divide(data, divisor); for(int i=0 ; i < remainder.length-1 ; i++) {

System.out.print(remainder[i]);

}

System.out.println("\nThe CRC code generated is:"); for(int i=0 ; i < data.length ; i++) {

System.out.print(data[i]);

}

for(int i=0 ; i < remainder.length-1 ; i++) { System.out.print(remainder[i]);

}

System.out.println();

int sent\_data[] = new int[data.length + remainder.length - 1]; System.out.println("Enter the data to be sent:");

for(int i=0 ; i < sent\_data.length ; i++) {

System.out.println("Enter bit number " + (sent\_data.length-i)+ ":"); sent\_data[i] = scan.nextInt();

}

receive(sent\_data, divisor);

}

static int[] divide(int old\_data[], int divisor[]) { int remainder[] , i;

int data[] = new int[old\_data.length + divisor.length]; System.arraycopy(old\_data, 0, data, 0, old\_data.length); remainder = new int[divisor.length]; System.arraycopy(data, 0, remainder, 0, divisor.length); for(i=0 ; i < old\_data.length ; i++) {

System.out.println((i+1) + ".) First data bit is : "+ remainder[0]); System.out.print("Remainder : ");

if(remainder[0] == 1) {

for(int j=1 ; j < divisor.length ; j++) {

remainder[j-1] = exor(remainder[j], divisor[j]); System.out.print(remainder[j-1]);

}

}

else {

for(int j=1 ; j < divisor.length ; j++) { remainder[j-1] = exor(remainder[j], 0); System.out.print(remainder[j-1]);

}

}

remainder[divisor.length-1] = data[i+divisor.length]; System.out.println(remainder[divisor.length-1]);

}

return remainder;

}

static int exor(int a, int b) {

if(a == b) {

return 0;

}

return 1;

}

static void receive(int data[], int divisor[]) {

int remainder[] = divide(data, divisor); for(int i=0 ; i < remainder.length ; i++) {

if(remainder[i] != 0) {

System.out.println("There is an error in received data..."); return;

}

}

System.out.println("Data was received without any error.");

}

|  |  |
| --- | --- |
| }  **OUTPUT:**  C:\Harini>javac CRC.java  C:\Harini>java CRC | |
| Enter the size of the data: | 7 |
| Enter the data, bit by bit: | |
| Enter bit number 7: | 1 |
| Enter bit number 6: | 0 |
| Enter bit number 5: | 0 |
| Enter bit number 4: | 1 |
| Enter bit number 3: | 1 |
| Enter bit number 2: | 0 |
| Enter bit number 1: | 1 |
| Enter the size of the divisor: 4 Enter the divisor, bit by bit: | |
| Enter bit number 4: | 1 |
| Enter bit number 3: | 0 |
| Enter bit number 2: | 1 |
| Enter bit number 1: | 1 |
| 1.) First data bit is : 1 | |
| Remainder : 0101 | |
| 2.) First data bit is : 0 | |
| Remainder : 1010 | |
| 3.) First data bit is : 1 | |

Remainder : 0011

4.) First data bit is : 0 Remainder : 0110

5.) First data bit is : 0 Remainder : 1100

6.) First data bit is : 1 Remainder : 1110

7.) First data bit is : 1 Remainder : 1010

101

The CRC code generated is: 1001101101 Enter the data to be sent:

Enter bit number 10: 1

Enter bit number 9: 0

Enter bit number 8: 0

Enter bit number 7: 1

Enter bit number 6: 1

Enter bit number 5: 0

Enter bit number 4: 1

Enter bit number 3: 1

Enter bit number 2: 0

Enter bit number 1: 1

First data bit is : 1 Remainder : 0101 2.) First data bit is : 0 Remainder : 1010 3.) First data bit is : 1 Remainder : 0011 4.) First data bit is : 0 Remainder : 0111 5.) First data bit is : 0 Remainder : 1110 6.) First data bit is : 1 Remainder : 1011 7.) First data bit is : 1 Remainder : 0000 8.) First data bit is : 0 Remainder : 0000 9.) First data bit is : 0 Remainder : 0000 10.) First data bit is : 0 Remainder : 0000 Data was received without any error.

1. **SLIDING WINDOW PROTOCOL**

**PROGRAM CODING**:

**// SENDER PROGRAM**

import java.net.\*; import java.io.\*; import java.rmi.\*; public class slidsender

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

{ServerSocket ser=new ServerSocket(10); Socket s=ser.accept();

DataInputStream in=new DataInputStream(System.in); DataInputStream in1=new DataInputStream(s.getInputStream()); String sbuff[]=new String[8];

PrintStream p;

int sptr=0,sws=8,nf,ano,i; String ch;

do

{p=new PrintStream(s.getOutputStream()); System.out.print("Enter the no. of frames : "); nf=Integer.parseInt(in.readLine()); p.println(nf);

if(nf<=sws-1)

{System.out.println("Enter "+nf+" Messages to be send\n"); for(i=1;i<=nf;i++)

{ sbuff[sptr]=in.readLine(); p.println(sbuff[sptr]); sptr=++sptr%8; }

sws-=nf;

System.out.print("Acknowledgment received"); ano=Integer.parseInt(in1.readLine()); System.out.println(" for "+ano+" frames"); sws+=nf; }

else

{ System.out.println("The no. of frames exceeds window size"); break; }

System.out.print("\nDo you wants to send some more frames : "); ch=in.readLine(); p.println(ch); }

while(ch.equals("yes"));

s.close(); } }

**//RECEIVER PROGRAM**

import java.net.\*; import java.io.\*; class slidreceiver

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

{ Socket s=new Socket(InetAddress.getLocalHost(),10); DataInputStream in=new DataInputStream(s.getInputStream()); PrintStream p=new PrintStream(s.getOutputStream());

int i=0,rptr=-1,nf,rws=8; String rbuf[]=new String[8]; String ch; System.out.println(); do

{ nf=Integer.parseInt(in.readLine()); if(nf<=rws-1)

{ for(i=1;i<=nf;i++)

{ rptr=++rptr%8; rbuf[rptr]=in.readLine();

System.out.println("The received Frame " +rptr+" is : "+rbuf[rptr]); } rws-=nf;

System.out.println("\nAcknowledgment sent\n"); p.println(rptr+1); rws+=nf; }

else break;

ch=in.readLine(); } while(ch.equals("yes")); } }

**OUTPUT:**

**//SENDER OUTPUT**

C:\Harini>javac send.java

C:\Harini>java send

Enter the no. of frames : 4 Enter 4 Messages to be send hiii

how r u i am fine

how is everyone

Acknowledgment received for 4 frames

Do you wants to send some more frames : no

**//RECEIVER OUTPUT**

C:\Harini>javac rec.java

C:\Harini>rec server

The received Frame 0 is : hiii The received Frame 1 is : how r u

The received Frame 2 is : i am fine

The received Frame 3 is : how is everyone Acknowledgment sent