**Implementation of mutual exclusion using the Token Ring technique concept. This technique solves the mutual exclusion existing in the process communication**

**TokenServer.java:**

import java.net.DatagramPacket;

import java.net.DatagramSocket;

public class TokenServer {

public static DatagramSocket ds;

public static DatagramPacket dp;

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

{

ds=new DatagramSocket(1000);

}

catch(Exception e)

{

e.printStackTrace();

}

while(true)

{

byte buff[]=new byte[1024]; System.out.println("Server Started...!");

ds.receive(dp=new DatagramPacket(buff, buff.length));

String str=new String(dp.getData(),0,dp.getLength());

System.out.println("Message from"+str);

}

}

}

**TokenClient2.java:**

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class TokenClient1

{

public static DatagramSocket ds;

public static DatagramPacket dp;

public static BufferedReader br;

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

{

boolean hasToken; try

{

ds= new DatagramSocket(100);

}

catch(Exception e)

{

e.printStackTrace();

}

hasToken=true; while(true)

{

if(hasToken==true)

{

System.out.println("Do you want to enter data...(yes/no:)");

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

String ans = br.readLine();

if(ans.equalsIgnoreCase("Yes"))

{

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

System.out.println("Ready to send");

System.out.println("Sending");

System.out.println("Enter the data");

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

String str="Client-1 ===>" + br.readLine();

byte buff[] = new byte[1024];

buff = str.getBytes();

ds.send(new DatagramPacket(buff,buff.length,InetAddress.getLocalHost(),1000));

System.out.println("Now Sending");

}

else if(ans.equalsIgnoreCase("No"))

{

System.out.println("I am busy state"); String msg="Token";

byte bf1[]= new byte[1024]; bf1=msg.getBytes();

ds.send(new

DatagramPacket(bf1,bf1.length,InetAddress.getLocalHost(),200));

hasToken=false;

byte bf2[]=new byte[1024];

ds.receive(dp=new DatagramPacket(bf2,bf2.length)); String clientmsg=new

String(dp.getData(),0,dp.getLength());

System.out.println("The data is"+clientmsg);

if(clientmsg.equals("Token"))

hasToken=true;

System.out.println("I am leavino busy state");

}

}

else

{

hasToken=true;

System.out.println("I am leaving busy state");

System.out.println("Entering in receiving mode...");

byte bf[]=new byte[1024];

ds.receive(dp=new DatagramPacket(bf,bf.length));

String clientmsg1=new String(dp.getData(),0,dp.getLength());

System.out.println("The data is"+clientmsg1);

if(clientmsg1.equals("Token"))

{

hasToken=true;

}

}

}

}

}

**TokenClient3.java:**

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class TokenClient2 {

static DatagramSocket ds;

static DatagramPacket dp;

static BufferedReader br;

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

{

try

{

ds= new DatagramSocket(200);

}

catch(Exception e)

{

e.printStackTrace();

}

boolean hasToken = true;

while(true)

{

if(hasToken==true)

{

System.out.println("Do you want to enter data...(yes/no:)");

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

String str= br.readLine();

if(str.equalsIgnoreCase("Yes"))

{

System.out.println("Enter the data");

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

String msg = "Client-2 ===>" + br.readLine();

byte bf1[] = new byte[1024];

bf1 = msg.getBytes(); ds.send(new

DatagramPacket(bf1,bf1.length,InetAddress.getLocalHost(),1000));

System.out.println("Data Sent");

}

else

{

String clientmsg="Token"; byte bf1[]= new byte[1024]; bf1=clientmsg.getBytes();

ds.send(new DatagramPacket(bf1,bf1.length,InetAddress.getLocalHost(),100));

hasToken=false;

}}

else

{

try

{

byte buff[]=new byte[1024]; System.out.println("Entering in receiving mode...");

ds.receive(dp=new DatagramPacket(buff,buff.length)); String clientmsg1=new

String(dp.getData(),0,dp.getLength());

System.out.println("The data is"+clientmsg1); if(clientmsg1.equals("Token"))

hasToken=true;

}

catch(Exception e)

{

e.printStackTrace();

}

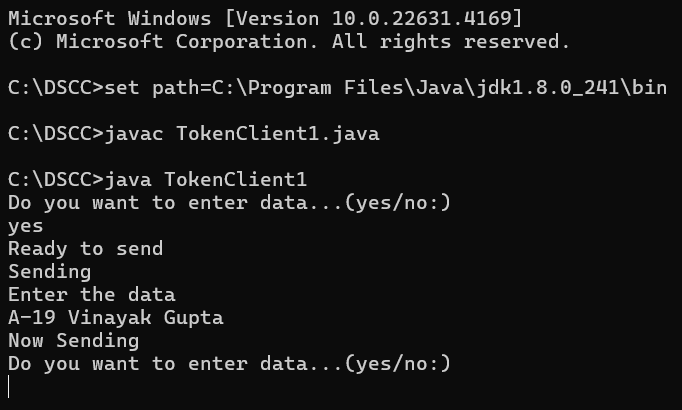
}

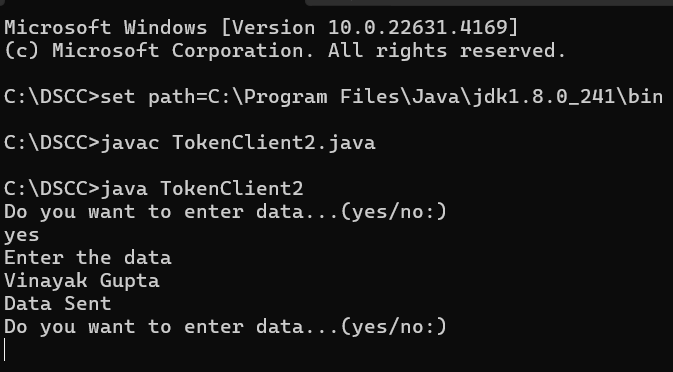
}

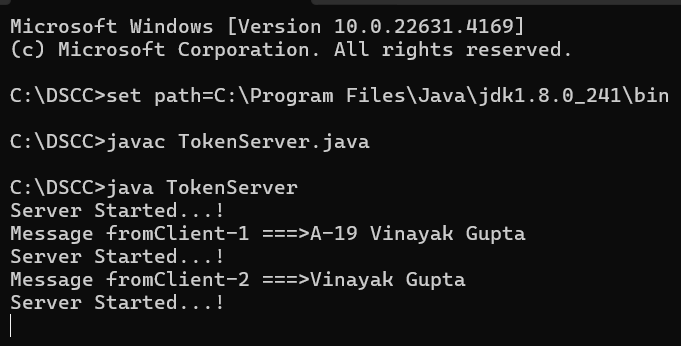
}

}

**Output:**

****

****

****