TITLE: Domain Name Server	GPREC-D/CS/EXPT-DCCN-09

AIM: Implement Domain name server

```
import java.net.*;
import java.io.*;
import java.util.*;
public class DNS
{
 public static void main(String[] args)
 {
  int n;
  BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
  do
  {
   System.out.println("\n Menu: \n 1. DNS 2.Exit \n");
   System.out.println("\n Enter your choice");
   n = Integer.parseInt(System.console().readLine());
   if(n==1)
   {
    try
    {
     System.out.println("\n Enter Host Name ");
     String hname=in.readLine();
     InetAddress address;
     address = InetAddress.getByName(hname);
     System.out.println("Host Name: " + address.getHostName());
     System.out.println("IP: " + address.getHostAddress());
```

```
    }
    catch(IOException ioe)
    {
     ioe.printStackTrace();
    }
   }
  }while(!(n==2));
 }
}
```

Output:

Menu:

1. DNS 2.Exit

Enter your choice

1

Enter Host Name

www.gprec.ac.in

Host Name: www.gprec.ac.in

IP: 182.50.132.59

Menu:

1. DNS 2.Exit

Enter your choice

1

Enter Host Name

www.gmail.com

Host Name: www.gmail.com

IP: 172.253.119.19

Menu:

1. DNS 2.Exit Enter your choice 2

TITLE: Client Server Model GPREC-D/CS/EXPT-DCCN-10

Aim: Implement client server model

// **A Java program for a Client**

```
import java.net.*;
import java.io.*;
public class Client
{
   // initialize socket and input output streams
   private Socket socket          =  null;
   private DataInputStream input = null;
   private DataOutputStream out     = null;
   // constructor to put ip address and port
   public Client(String address, int port)
   {
      // establish a connection
      try
      {
         socket = new Socket(address, port);
         System.out.println("Connected");
         // takes input from terminal
         input = new DataInputStream(System.in);
         // sends output to the socket
         out    = new DataOutputStream(socket.getOutputStream());
      }
      catch(UnknownHostException u)
      {
         System.out.println(u);
      }
```

```
catch(IOException i)
{
   System.out.println(i);
}
// string to read message from input
String line = "";
// keep reading until "Over" is input
while (!line.equals("Over"))
{
   try
   {
      line = input.readLine();
      out.writeUTF(line);
   }
   catch(IOException i)
   {
      System.out.println(i);
   }
}
// close the connection
try
{
   input.close();
   out.close();
   socket.close();
}
catch(IOException i)
{
   System.out.println(i);
```

```
        }
    }
    public static void main(String args[])
    {
        Client client = new Client("127.0.0.1", 5000);
    }
}
```

// A Java program for a Server

```
import java.net.*;
import java.io.*;
public class Server
{
    //initialize socket and input stream
    private Socket          socket = null;
    private ServerSocket    server = null;
    private DataInputStream in       =  null;
    // constructor with port
    public Server(int port)
    {
        // starts server and waits for a connection
        try
        {
            server = new ServerSocket(port);
            System.out.println("Server  started");

            System.out.println("Waiting for a client ...");
            socket = server.accept();
            System.out.println("Client accepted");
```

```
        // takes input from the client socket
        in = new DataInputStream(new BufferedInputStream(socket.getInputStream()));
       String line = "";
       // reads message from client until "Over" is sent
        while (!line.equals("Over"))
        {
           try
           {
              line = in.readUTF();
              System.out.println(line);
            }
           catch(IOException i)
           {
              System.out.println(i);
           }
        }
        System.out.println("Closing connection");
      // close connection
        socket.close();
        in.close();
    }
    catch(IOException i)
    {
        System.out.println(i);
    }
}
 public static void main(String args[])
{
    Server server = new Server(5000);
```

```
    }
}
```

Output:

1. First run the Server application as,

$ java Server

Server started

Waiting for a client …

2. Then run the Client application on another terminal as,

$ java Client

It will show – Connected and the server accepts the client and shows,Client accepted

3. Then you can start typing messages in the Client window. Here is a sample input to the Client

Hello

I made my first socket connection

Over

Which the Server simultaneously receives and shows,

Hello

I made my first socket connection

Over

Closing connection