# 1. A simple program to print the IP address of the system

**Algorithm**

1. Import all the required packages net, and io packages
2. Create a class ipclient
3. Initialize InetAddress class and create the object ia.
4. Get the Ip address of the system using getLocalHost() method
5. Print the IP address.

## PROGRAM

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

{

public static void main(String args[])

{

try

{

InetAddress ia=InetAddress.getLocalHost(); System.out.println("The client system address is: "+ia);

}

catch(IOException e)

{

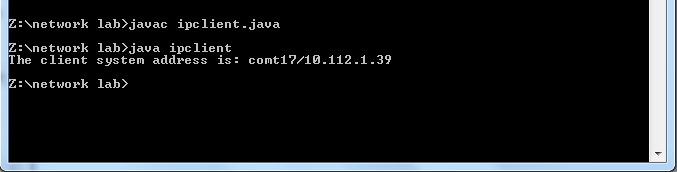
System.out.println("The exception is: "+e);

}

}

}

OUTPUT:



## Ping command

**Program:**

import java.io.\*; public class ping1

{

public static void runSystemCommand(String Command)

{

try{

Process p=Runtime.getRuntime().exec(Command);

BufferedReader InputStream=new BufferedReader(new InputStreamReader(p.getInputStream())); String s="vvv";

while((s=InputStream.readLine())!=null)

{

System.out.println(s);

}

}

catch(Exception e)

{

e.printStackTrace();

}

}

public static void main(String[]args)

{

String Ip="localhost";

runSystemCommand("ping " +Ip); java.util.Date date=new java.util.Date(); System.out.println(date);

}

}

Traceroute command

Program:

import java.io.BufferedReader; import java.io.InputStreamReader;

public class traceroutecmd

{

public static void runSystemCommand(String command)

{

try

{

}

Process p = Runtime.getRuntime().exec(command); BufferedReader inputStream = new BufferedReader(

new InputStreamReader(p.getInputStream()));

String s = "";

while ((s = inputStream.readLine()) != null) System.out.println(s);

catch (Exception e)

{

}

}

public static void main(String[] args)

{

// String ip = "[www.google.co.in](http://www.google.co.in/)";

// String ip = "127.0.0.1";

String ip = "[www.drranurekha.com](http://www.drranurekha.com/)"; runSystemCommand("tracert " + ip);

}

}

Implement FTP using TCP Algorithm

1. Create a file which has to be read by the server. Open a notepad, write the file contents and save it in the program folder sample.txt.
2. Import the required io and net packages.
3. Initialized the socket class for communicate with the server with port number 4000.
4. Client specifies the file name to the server (eg. Sample.txt)
5. Initialize the BufferReader class to read the filename from the terminal.
6. Initialize the PrintWriter class to write and send the file name to the server through the socket.
7. Initialize BufferReader class to read the file contents from the server.
8. Repeatedly read the data till end of file equals to empty.
9. Close the socket stream, filestream and BufferReader classes
10. End of the program.

Algorithm – Server

1. Import the required io and net packages.
2. Initialized the SeverSocket class and accept the client connection.
3. Initialize the BufferReader class for reading the client request (ie file name)
4. Initialize the FileReader class for reading the file.
5. Initialize Buffer Reader class for reading the file contents line by line.
6. Repeatedly read the file content line by line and send to the client through the socket S.
7. Close the socket stream, BufferedReader.

## Program – Client

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

public class FTPClient

{

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

{

Socket s = new Socket(InetAddress.getLocalHost(), 4000);

// reading the file name from keyboard. Uses input stream System.out.println("Enter the file name");

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

// sending the file name to server. Uses PrintWriter OutputStream ostream = s.getOutputStream( );

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

// receiving the contents from server. Uses input stream InputStream istream = s.getInputStream();

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

String str;

while((str = socketRead.readLine()) != null) // reading line-by-line

{

System.out.println(str);

}

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

}

}

**Program – Server** import java.io.\*; import java.net.\*;

public class FTPServer

{

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

{ // establishing the connection with the server ServerSocket ss = new ServerSocket(4000); System.out.println("Server ready for connection");

Socket s = ss.accept(); // binding with port: 4000 System.out.println("Connection is successful and wating for chatting");

// reading the file name from client InputStream istream = s.getInputStream( );

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

// reading file contents

BufferedReader contentRead = new BufferedReader(new FileReader(fname) );

// keeping output stream ready to send the contents OutputStream ostream = s.getOutputStream( );

PrintWriter pwrite = new PrintWriter(ostream, true);

String str;

while((str = contentRead.readLine()) != null) // reading line-by-line from file

{

pwrite.println(str); // sending each line to client

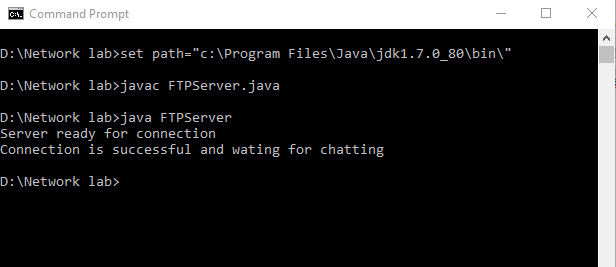
}

s.close(); ss.close(); // closing network sockets pwrite.close(); fileRead.close(); contentRead.close();

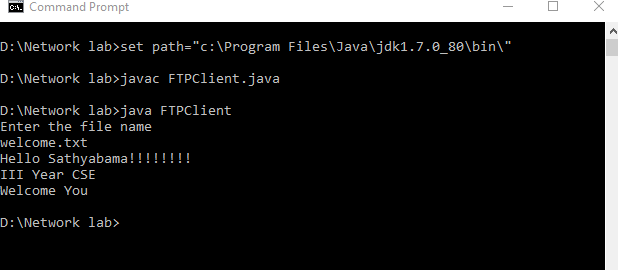
}

}

## Output Server side



**Client side**



**Program to implement HTTP protocol**

**Aim:** Program to implement HTTP protocol and to print URl for the Client.

## Algorithm:

STEP 1: Create the URL with Http URL Connections

STEP 2: Define the Http Protocol for Client Connections.

STEP3: Get the Http Connection. STEP4:Print the URL for the Client.

## Program:

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

{

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

{

URL url=new URL("<http://www.sathyabama.ac.in/>"); URLConnection conn=url.openConnection(); conn.connect();

InputStreamReader content= new InputStreamReader(conn.getInputStream()); FileWriter f=new FileWriter ("abc.html");

for(int i=0;i!=-1;i= content.read())

{

f.write((char) i);

}

}

}

# 5. Creation of a Simple Chat Program

**Client program**

import java.io.\*; import java.net.\*; public class chatclient1

{

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

{

Socket sk=new Socket(InetAddress.getLocalHost(),2000);

BufferedReader sin=new BufferedReader(new InputStreamReader(sk.getInputStream())); PrintStream sout=new PrintStream(sk.getOutputStream());

BufferedReader stdin=new BufferedReader(new InputStreamReader(System.in)); String s;

while ( true )

{

System.out.print("Client : "); s=stdin.readLine(); sout.println(s); s=sin.readLine();

System.out.print("Server : "+s+"\n"); if ( s.equalsIgnoreCase("BYE") )

break;

}

sk.close();

sin.close();

sout.close(); stdin.close();

}

}

# Server program

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

public class chatserver1

{

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

{

ServerSocket ss=new ServerSocket(2000); Socket sk=ss.accept();

BufferedReader cin=new BufferedReader(new InputStreamReader(sk.getInputStream())); PrintStream cout=new PrintStream(sk.getOutputStream());

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

String s; while ( true )

{

s=cin.readLine();

if (s.equalsIgnoreCase("END"))

{

cout.println("BYE"); break;

}

System. out.print("Client : "+s+"\n"); System.out.print("Server : "); s=stdin.readLine();

cout.println(s);

}

ss.close();

sk.close();

cin.close();

cout.close(); stdin.close();

}

}

# Creation of UDP server

**Aim:** To perform a java program for UDP client and server.

# Algorithm:

**SERVER:**

1.Create a new Datagram Socket. 2.Create a new Datagram packet. 3.Create a message to be sent.

4.Convert into bytes 5.create a packet 6.send packet

7.wait for acknowledgement from client 8.print data from client

9.stop the program

# CLIENT:

1. Create new Datagram Socket. 2.Create new Datagram packet. 3.Get the packet.

1. Print the content. 5.Create a new packet. 6.send to server 7.Stop the program.

# Program:

**Server:**

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

{

public static int client=789; public static int server=790;

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

{

String s;

InetAddress id=InetAddress.getLocalHost();

BuﬀeredReader dis=new BuﬀeredReader(new InputStreamReader(System.in)); DatagramSocket ds=new DatagramSocket(server);

byte b[]=new byte[1024]; System.out.println("Server Side.... Sending. ");

System.out.println("\n"+id); while(true)

{

s=dis.readLine(); if(s.equals("end"))

{

b=s.getBytes();

DatagramPacket dp=new DatagramPacket(b,s.length(),id,client); ds.send(dp);

break;

}

else

{

b=s.getBytes();

DatagramPacket dp=new DatagramPacket(b,s.length(),id,client); ds.send(dp);

}

}

}

}

# Client:

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

{

public static int client=789;

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

{

DatagramSocket ds=new DatagramSocket(client); byte b[]=new byte[1024]; System.out.println("client....receiving. ");

while(true)

{

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

String s=new String(dp.getData(),0,dp.getLength()); if(s.equals("end")) break;

else System.out.println(s);

}

}

}

**3. Printing Client Address at Server Side** Aim : Design a socket program to print the client address at the server side **Algorithm**

Server:

* 1. Import all the required io and net packages
  2. Create a class Sip
  3. Initialize the classes ServerSocket and Socket and their respective objects.
  4. Initialize the class DataInputStream for reading the data from socket through getInputStream() method.
  5. Establish the connection with client system using the client IP and port number 8020
  6. Read the IP address from the socket
  7. Print the IP address at the server Client:

1. Import all the required io and net packages
2. Create a class Cip
3. Initialize the classes Socket and their respective objects.
4. Initialize the class PrintStream Class for writing data into the socket through getOutputStream() method.
5. Get the IP address of the system using InetAddress class with getLocalHost() method.
6. Establish the connection with server system using the IP address and port number 8020
7. Write the IP address into the socket
8. If any exception arise, print the error.

# Program:

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

{

public static void main(String args[])

{

ServerSocket ss; Socket s; DataInputStream dis; String ip;

try

{

ss=new ServerSocket(8020); while(true)

{

s=ss.accept();

dis=new DataInputStream(s.getInputStream()); ip=dis.readLine();

System.out.println("Ip address of the client system is"+ip);

}

}

catch(IOException e)

{

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

}

}

}

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

{

public static void main(String args[])

{

Socket soc;

PrintStream ps;

try

{

InetAddress ia=InetAddress.getLocalHost(); soc=new Socket(ia,8020);

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

}

catch(IOException e)

{

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

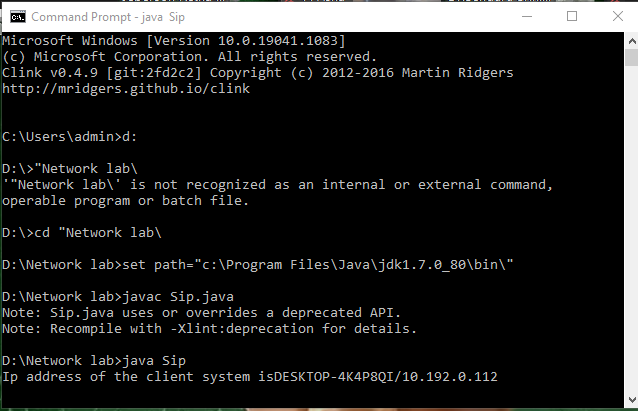
}

}

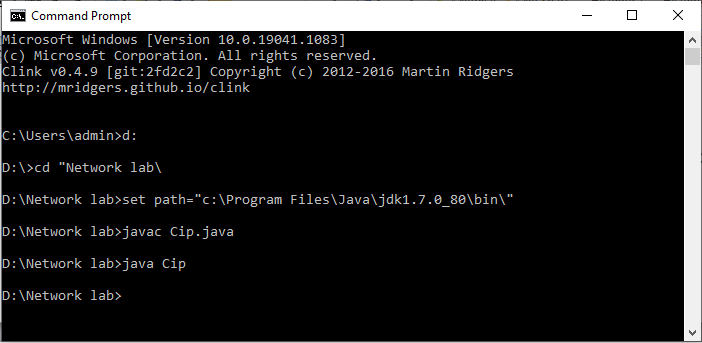
}

Output

Server side Terminal



Client Side Terminal



# 2. CREATION OF DATE AND TIME SERVER.

**SERVER PROGRAM**

import java.io.\*; import java.net.\*; import java.util.\*; class dateserver

{

public static void main(String args[])

{

ServerSocket ss; Socket s; PrintStream ps; DataInputStream dis; String inet;

try

{

ss=new ServerSocket(8020); while(true)

{

s=ss.accept();

ps=new PrintStream(s.getOutputStream()); Date d=new Date();

ps.println(d); ps.close();

}

}

catch(IOException e)

{

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

} } }

# CLIENT PROGRAM

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

{

public static void main(String args[])

{

Socket soc;

DataInputStream dis;

String sdate;

PrintStream ps;

try

{

InetAddress ia=InetAddress.getLocalHost(); soc=new Socket(ia,8020);

dis=new DataInputStream(soc.getInputStream()); sdate=dis.readLine();

System.out.println("The data in the server is: "+sdate);

}

catch(IOException e)

{

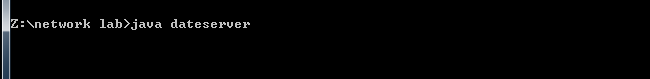
System.out.println("The exception is: "+e);

}

}

}

# OUTPUT

**Server Side**

# Client Side