SCSA2501 - Computer Network Lab 39110926
19S115591

Record - (cycle 2)

SK. JANEED SUHAIL
BE CSE-C4

Experiment - 6 : DAEMON PROGRAM

Aim :- To write a daemon program that runs continuosly and exists for purpose of handling periodic service requests that a computer system expects to receive.

Algorithm / methods used :-

1) Void setDaemon(boolean status) : Is used to mark the current thread as daemon thread or user thread.
2) On the other hand if I have Daemon thread t0 then by calling t0.setDaemon(false) would make is user thread.
3) Public final void setDaemon(boolean on)
4) If 'on' is true make thread as Daemon thread.
5) IllegalThreadStateException : If only this thread is active
6) SecurityException ; If current thread cannot modify this thread.
7) boolean isDaemon() : Checks if the current thread is daemon. Returns true if daemon else returns false.

Program :-

```
// Demonstrate usage of setDaemon() & isDaemon() thread.
Public class DemonThread extends Thread {
        Public DaemonThread (String name) {
                super(name);
        }
        Public void run(){
```

```
        if (Thread.currentThread().isDaemon()){
            System.out.println(getName() + "is Daemon thread");
        }
        else {
            System.out.println(getName() + " is User thread");
        }
    }
    Public static void main (String[] args){
        DaemonThread t1 = new DaemonThread("t1");
        DaemonThread t2 = new DaemonThread("t2");
        DaemonThread t3 = new DaemonThread ("t3");
        t1.setDaemon (true);
        t1.start();
        t2.start();
        t3.setDaemon (true);
        t3.start();
    }
}
```

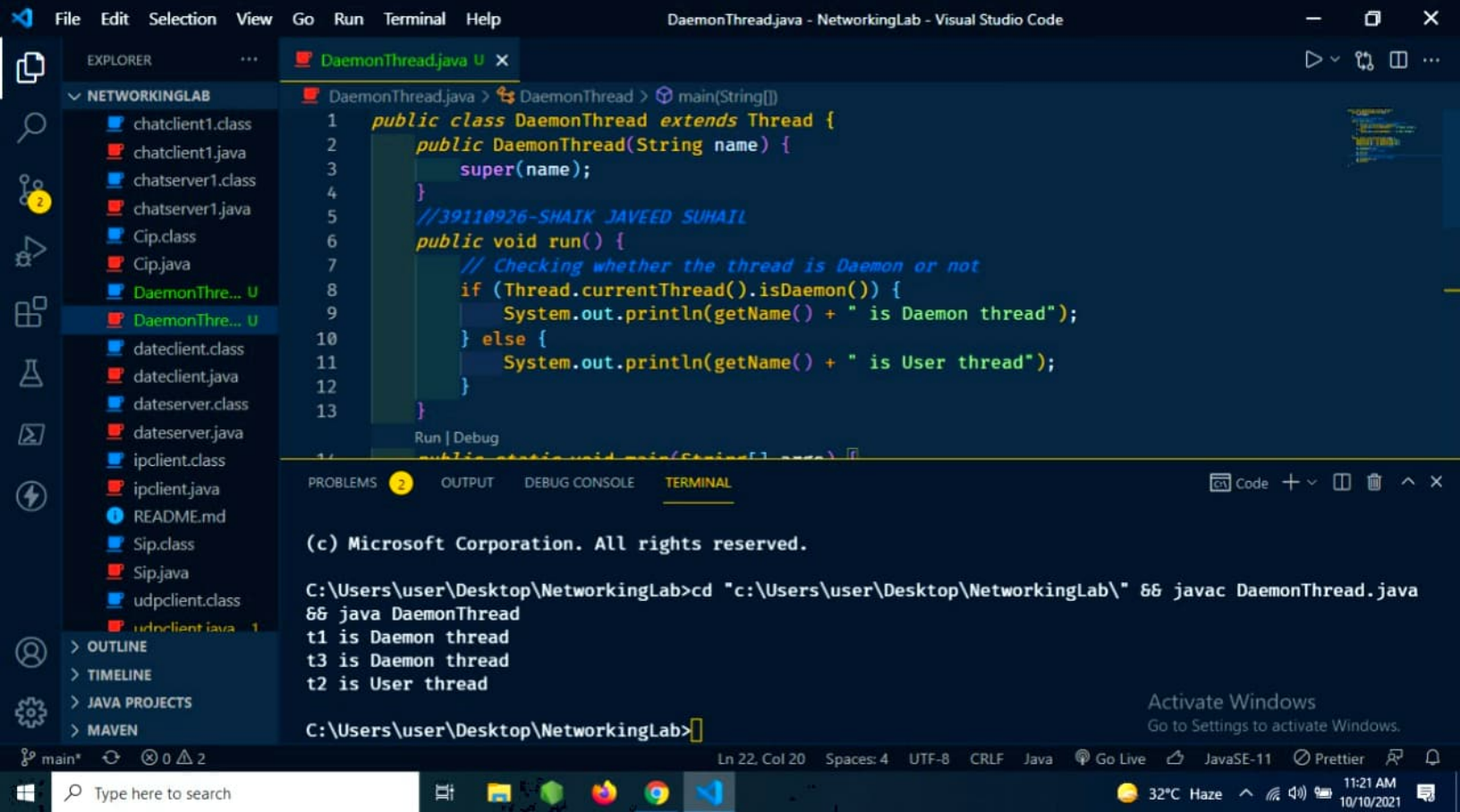
Output:-

t1 is Daemon thread.

t3 is Daemon thread.

t2 is User thread.

Result:- Daemon thread program has been executed succesfully.


File Edit Selection View Go Run Terminal Help
DaemonThread.java - NetworkingLab - Visual Studio Code
EXPLORER
NETWORKINGLAB
chatclient1.class
chatclient1.java
chatserver1.class
chatserver1.java
Cip.class
Cip.java
DaemonThre... U
DaemonThre... U
dateclient.class
dateclient.java
dateserver.class
dateserver.java
ipclient.class
ipclient.java
README.md
Sip.class
Sip.java
udpclient.class
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
DaemonThread.java U
DaemonThread.java > DaemonThread > main(String[])
1 public class DaemonThread extends Thread {
2 public DaemonThread(String name) {
3 super(name);
4 }
5 //39110926-SHAIK JAVEED SUHAIL
6 public void run() {
7 // Checking whether the thread is Daemon or not
8 if (Thread.currentThread().isDaemon()) {
9 System.out.println(getName() + " is Daemon thread");
10 } else {
11 System.out.println(getName() + " is User thread");
12 }
13 }
Run | Debug
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
(c) Microsoft Corporation. All rights reserved.
C:\Users\user\Desktop\NetworkingLab>cd "c:\Users\user\Desktop\NetworkingLab\" && javac DaemonThread.java
&& java DaemonThread
t1 is Daemon thread
t3 is Daemon thread
t2 is User thread
C:\Users\user\Desktop\NetworkingLab>
Activate Windows
Go to Settings to activate Windows.
main*
0 2
Ln 22, Col 20
Spaces: 4
UTF-8
CRLF
Java
Go Live
JavaSE-11
Prettier
Type here to search
32°C Haze
11:21 AM
10/10/2021

Exception in Daemon thread :-

Program :

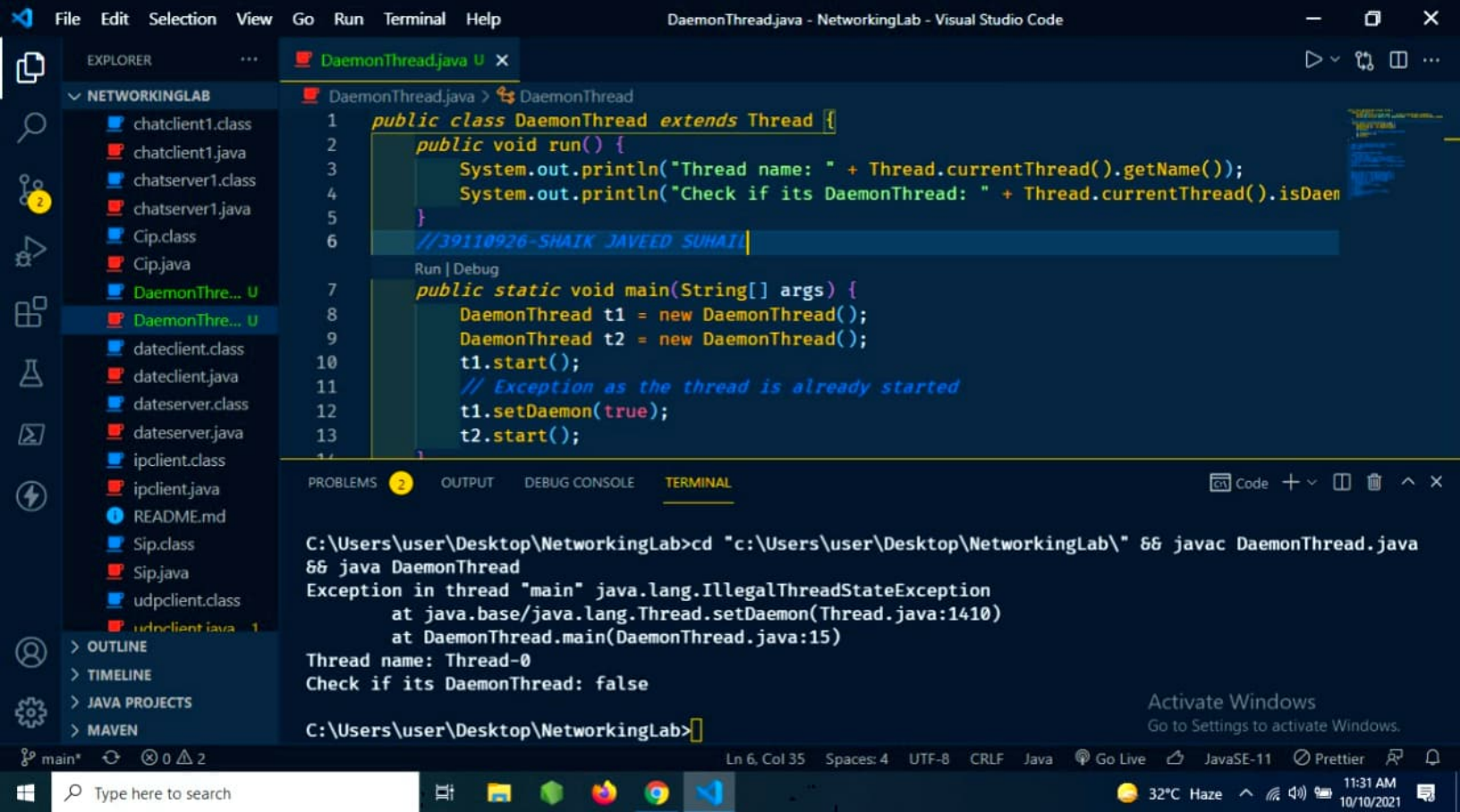
```
Public class DaemonThread extends Thread {

    public void run() {
        System.out.println("Thread name" + Thread.currentThread().getName());
        System.out.println("Check if its current thread" + Thread.currentThread().isDaemon());
    }
    Public static void main (String[] args) {
        DaemonThread t1 = new DaemonThread();
        DaemonThread t2 = new DaemonThread();
        t1.start();
        t1.setDaemon(true);
        t1.start();
    }
}
```

Output :

```
java.lang.IllegalThreadStateException
Thread name : Thread-0
Check if its DaemonThread : false
```

Result :- DaemonThread with runtime exception is also been executed.


File Edit Selection View Go Run Terminal Help
DaemonThread.java - NetworkingLab - Visual Studio Code
EXPLORER
NETWORKINGLAB
chatclient1.class
chatclient1.java
chatserver1.class
chatserver1.java
Cip.class
Cip.java
DaemonThre... U
DaemonThre... U
dateclient.class
dateclient.java
dateserver.class
dateserver.java
ipclient.class
ipclient.java
README.md
Sip.class
Sip.java
udpclient.class
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
DaemonThread.java U
DaemonThread.java > DaemonThread
1 public class DaemonThread extends Thread {
2 public void run() {
3 System.out.println("Thread name: " + Thread.currentThread().getName());
4 System.out.println("Check if its DaemonThread: " + Thread.currentThread().isDaem
5 }
6 //39110926-SHAIK JAVEED SUHAIL
Run | Debug
7 public static void main(String[] args) {
8 DaemonThread t1 = new DaemonThread();
9 DaemonThread t2 = new DaemonThread();
10 t1.start();
11 // Exception as the thread is already started
12 t1.setDaemon(true);
13 t2.start();
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
C:\Users\user\Desktop\NetworkingLab>cd "c:\Users\user\Desktop\NetworkingLab\" && javac DaemonThread.java
&& java DaemonThread
Exception in thread "main" java.lang.IllegalThreadStateException
at java.base/java.lang.Thread.setDaemon(Thread.java:1410)
at DaemonThread.main(DaemonThread.java:15)
Thread name: Thread-0
Check if its DaemonThread: false
C:\Users\user\Desktop\NetworkingLab>
Activate Windows
Go to Settings to activate Windows.
main*
0 2
Ln 6, Col 35
Spaces: 4
UTF-8
CRLF
Java
Go Live
JavaSE-11
Prettier
Type here to search
32°C Haze
11:31 AM
10/10/2021

7. HTTP protocol

39110926
19511559l

Aim :- Program to implement HTTP protocol & print URL for client.

Algorithm :-

S-①: Create URL with Http URL connections

S-②: Define HTTP protocol for client connections.

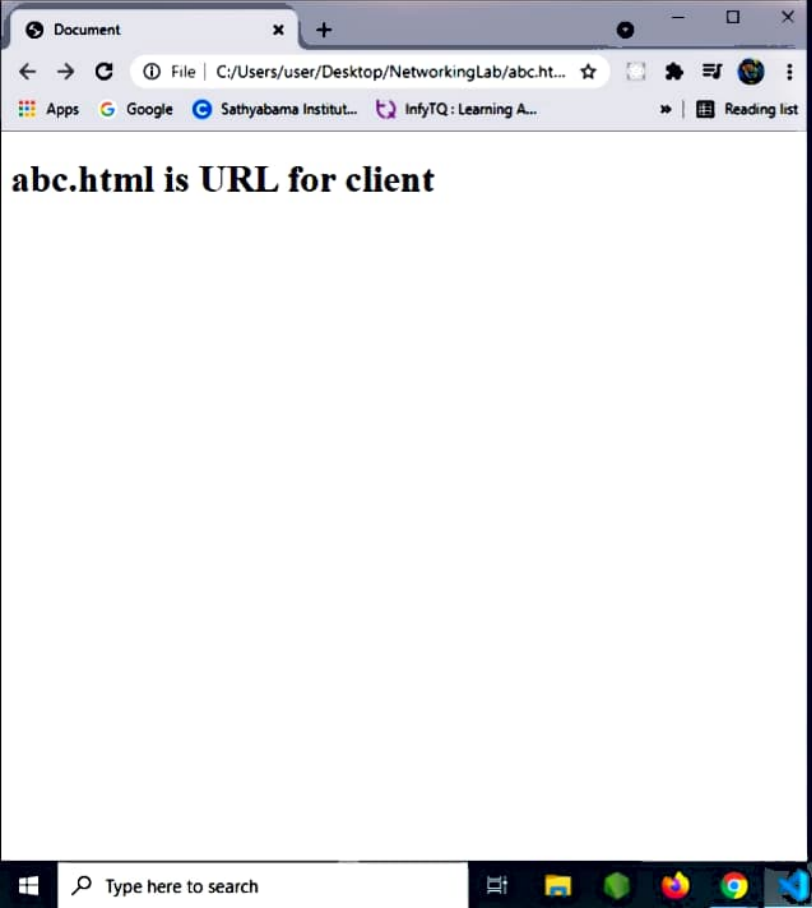
S-③: Get HTTP Connection

S-④:- 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.getInput
                                                          Stream());
        FileWriter f = new FileWriter("abc.html");
        for(int i=0; i!=-1; i=content.read()) {
            f.write((char)i);
        }
    }
}
```

Result :- Program has been executed successfully & URL for client is opening.


Document
File | C:/Users/user/Desktop/NetworkingLab/abc.ht...
Apps
Google
Sathyabama Institut...
InfyTQ : Learning A...
Reading list
abc.html is URL for client


File Edit Selection View ... myhttp.java - NetworkingLab...
myhttp.java 1, U
abc.html U
myhttp.java > ...
abc.html > ...
import java.io.*;
import java.net.*;
//39110926-SHAIK JAVEED
public class myhttp {
Run | Debug
public static void
URL url = new UI
URLConnection c
conn.connect();
InputStreamRead
FileWriter f =
for (int i = 0;
f.write((ch
}
}
<!DOCTYPE htm
<html lang="e
<head>
<meta cha
<meta htt
<meta nam
<title>Do
</head>
<body>
<h1>abc.ht
</body>
</html>
PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
C:\Users\user\Desktop\NetworkingLab>cd "c:\Users\user\Deskt
op\NetworkingLab\" && javac myhttp.java && java myhttp
C:\Users\user\Desktop\NetworkingLab>abc.html
C:\Users\user\Desktop\NetworkingLab>
Activate Windows
Go to Settings to activate Windows.
main*
CRLF
Java
Go Live
JavaSE-11
Prettier

Type here to search
33°C Light rain
12:37 PM
10/10/2021

8. FTP Protocol

39110926
19S115571

Aim :- Program to implement FTP protocol using TCP

Algorithm :-

Client side :-

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

Server side :-

1) Import io & net packages
2) Initialized socket & accept client connection
3) Initialize the bufferreader class for reading client request
4) Initialize the FileReader class for reading the file
5) Initialize the BufferReader class file contents line by line.
6) Repeatedly read content line by line & send to client through Socket s.
7) Close Socket stream, BufferedReader class.

Program :-

Client side :

```
import java.net.*;
import java.io.*;
Public class FTPClient{
    public static void main (String[] args) throws Exception {
        Socket s = new Socket (InetAddress.getLocalHost(), 4000);
        System.out.println("Enter the file name");
        BufferedReader keyRead = new BufferedReader(new InputStreamReader(
                                                    System.in));
        String fname = keyRead.readLine();
        OutputStream ostream = s.getOutputStream();
        PrintWriter pwrite = new PrintWriter (ostream, true);
        Pwrite.println(fname);
        InputStream istream = s.getInputStream();
        BufferedReader socketRead = new BufferedReader(new InputStream
                                                    Reader(istream));
        String str;
        while((str=socketRead.readLine()) != null) {
            System.out.println(str);
        }
        pwrite.close(); socketRead.close(); keyRead.close();
    }
}
```

Server side :-

```
import java.io.*;
import java.net.*;
public class FTPServer{
    public static void main (String[] args){
        Server ss = new ServerSocket (4000);
        System.out.println("Server ready for connection");
```

```
Socket s = ss.accept();
System.out.println("Connection is successfully & waiting for chatting");
InputStream istream = s.getInputStream();
BufferedReader fileRead = new BufferedReader(new InputStreamReader(istream));
String fname = fileRead.readLine();
BufferedReader contentRead = new BufferedReader(new FileReader(fname));
OutputStream ostream = s.getOutputStream();
PrintWriter pwrite = new PrintWriter(ostream, true);
String str;
while ((str = contentRead.readLine()) != null) {
    pwrite.println(str);
}
s.close(); ss.close(); pwrite.close(); fileRead.close(); contentRead.close();
}
}
```

Output :-

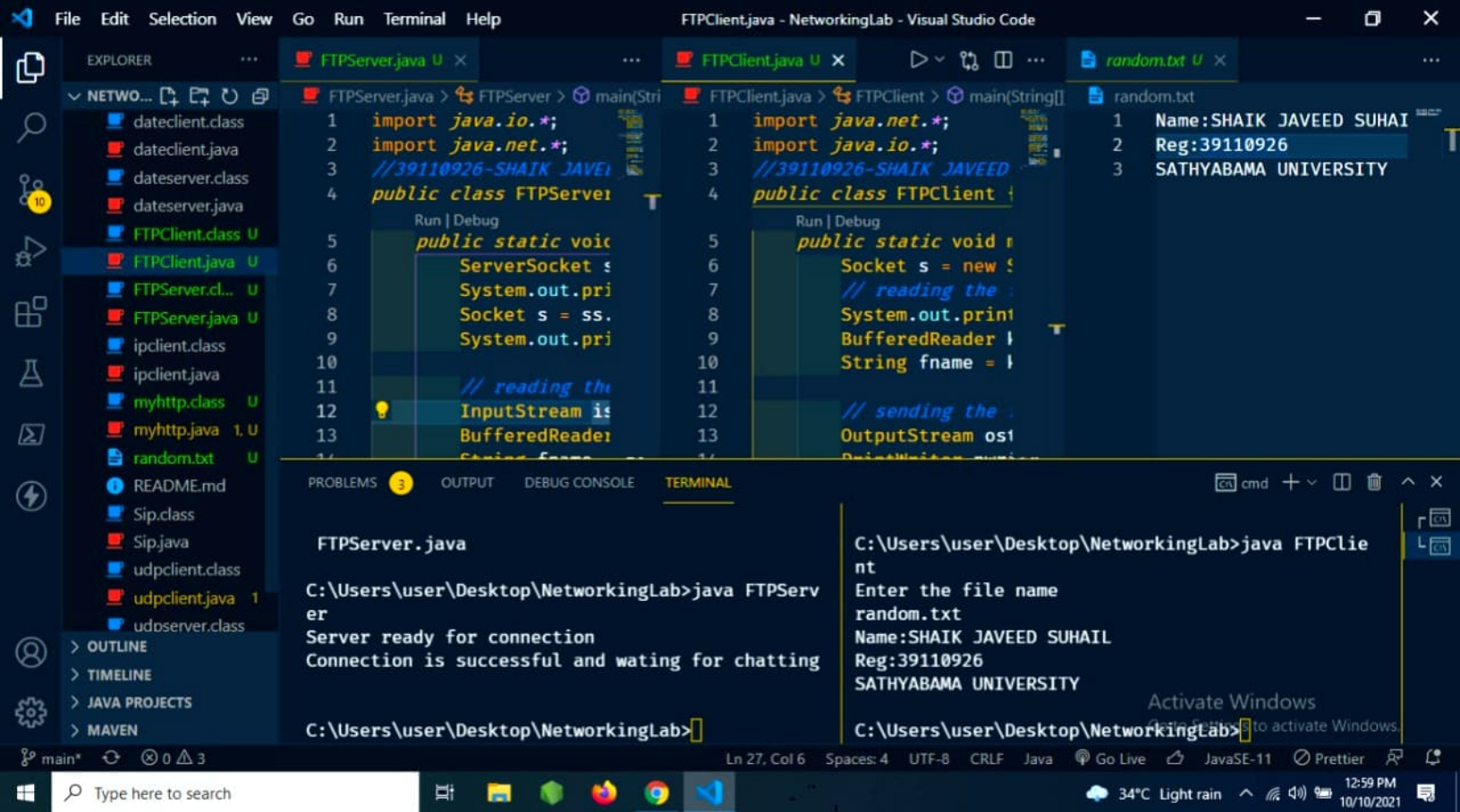
Server side :-

Server ready for connection
Connection is successfull & waiting for chatting

Client side :-

Enter the file name
random.txt
Name: SHAIK JAVEED SOHAIL
Reg: 39110926
SATYABAMA UNIVERSITY

Result :- Program to implement FTP has been executed successfully.


File Edit Selection View Go Run Terminal Help
FTPClient.java - NetworkingLab - Visual Studio Code
EXPLORER
NETWO...
dateclient.class
dateclient.java
dateserver.class
dateserver.java
FTPClient.class U
FTPClient.java U
FTPServer.cl... U
FTPServer.java U
ipclient.class
ipclient.java
myhttp.class U
myhttp.java 1, U
random.txt U
README.md
Sip.class
Sip.java
udpclient.class
udpclient.java 1
udpserver.class
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
FTPServer.java U
FTPClient.java U
random.txt U
FTPServer.java > FTPServer > main(Stri
FTPClient.java > FTPClient > main(String[]
random.txt
1 import java.io.*;
2 import java.net.*;
3 //39110926-SHAIK JAVEI
4 public class FTPServer
Run | Debug
5 public static voic
6 ServerSocket s
7 System.out.pri
8 Socket s = ss.
9 System.out.pri
10
11 // reading the
12 InputStream is
13 BufferedReader
1 import java.net.*;
2 import java.io.*;
3 //39110926-SHAIK JAVEED
4 public class FTPClient
Run | Debug
5 public static void
6 Socket s = new
7 // reading the
8 System.out.print
9 BufferedReader
10 String fname =
11
12 // sending the
13 OutputStream os
1 Name:SHAIK JAVEED SUHAI
2 Reg:39110926
3 SATHYABAMA UNIVERSITY
PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
cmd
FTPServer.java
C:\Users\user\Desktop\NetworkingLab>java FTPServ
er
Server ready for connection
Connection is successful and wating for chatting
C:\Users\user\Desktop\NetworkingLab>
C:\Users\user\Desktop\NetworkingLab>java FTPClie
nt
Enter the file name
random.txt
Name:SHAIK JAVEED SUHAIL
Reg:39110926
SATHYABAMA UNIVERSITY
C:\Users\user\Desktop\NetworkingLab>
Activate Windows
Go to Settings to activate Windows.
main*
0 3
Ln 27, Col 6
Spaces: 4
UTF-8
CRLF
Java
Go Live
JavaSE-11
Prettier
Type here to search
34°C Light rain
12:59 PM
10/10/2021

9. Traceroute command

39110926
19511591

Aim :- Progam to implement traceroute command

Algorithm :-

1) Import required BufferedReader & InputStreamReader packages
2) Initialise the Process and BufferedReader to get Runtime & Inputstream.
3) Initialize an empty string
4) If inputstream is not null print the trace route of ip.
5) Close the Buffered Reader class
6) End of program

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 InputStream
                                        (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.drranuretha.com";
        runSystemCommand("tracert "+ ip);
    }
}
```

Output:-

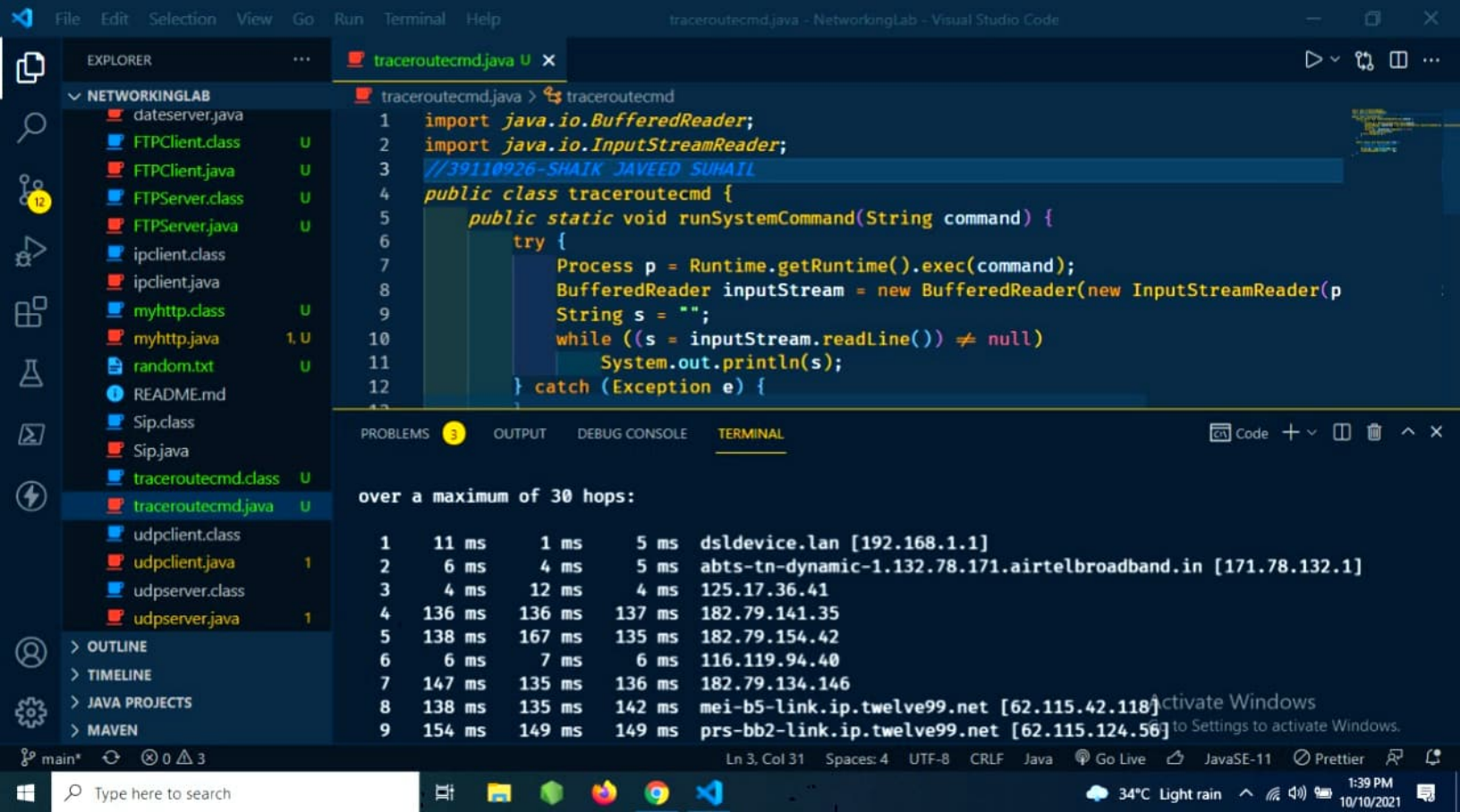
Tracing route to drranuretha.com [160.153.137.14]

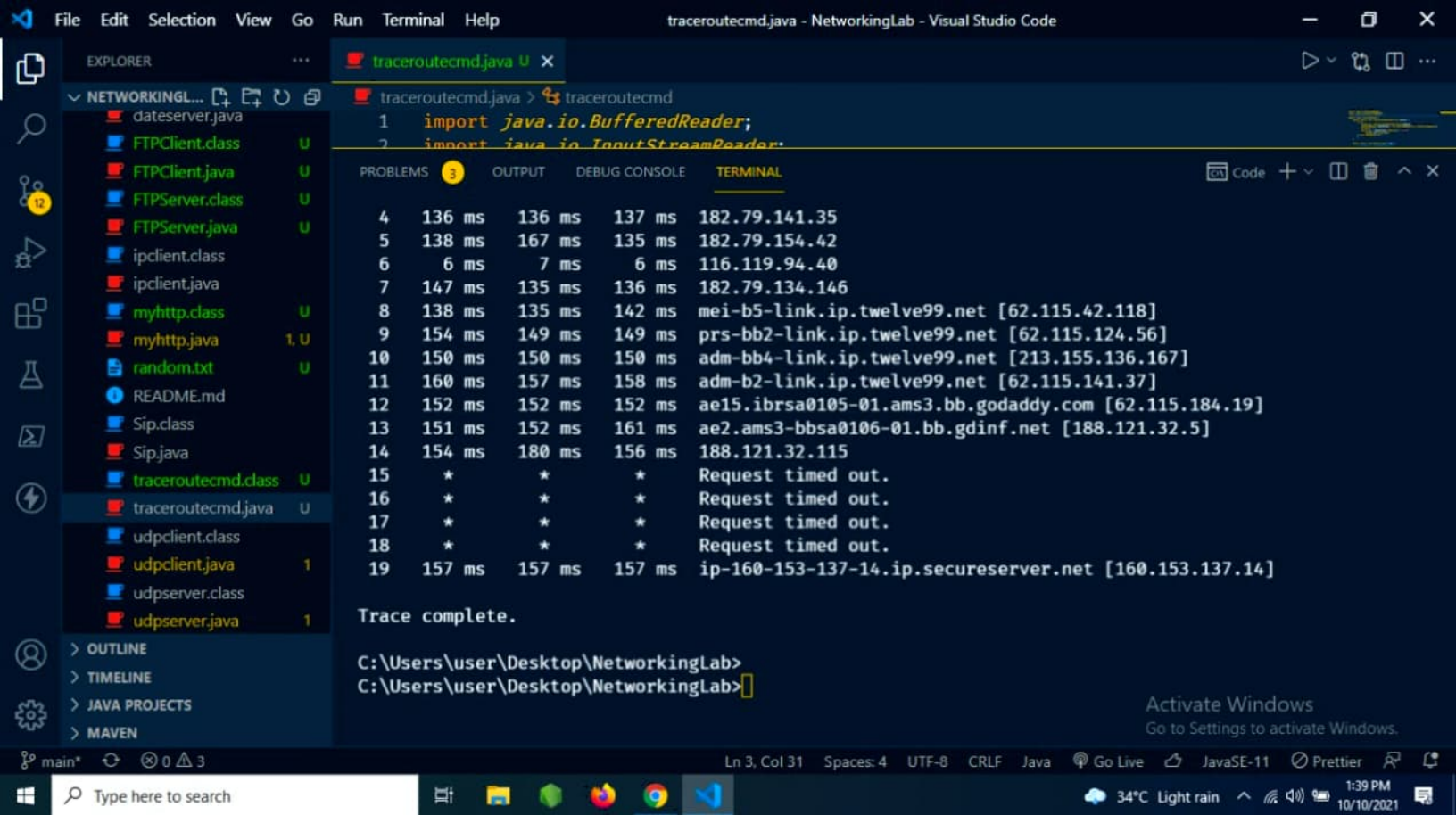
over a maximum of 30 hops:

1	11ms	1ms	5ms	dsldevice.lan [192.168.1.1]
2	6ms	4ms	5ms	a96-128-132-1.deploy.akamaitechnologies.com [171.128.132.1]
.	.	.	.	.
(	(	(	(	(
\	\	\	\	\
19	157ms	157ms	157ms	ip.---ip.secureserver.net [160.153.137.14]

Trace complete.

Result:- The program to execute traceroute cmd has been executed successfully.


File Edit Selection View Go Run Terminal Help
traceroutecmd.java - NetworkingLab - Visual Studio Code
EXPLORER
NETWORKINGLAB
dateserver.java
FTPClient.class U
FTPClient.java U
FTPServer.class U
FTPServer.java U
ipclient.class
ipclient.java
myhttp.class U
myhttp.java 1, U
random.txt U
README.md
Sip.class
Sip.java
traceroutecmd.class U
traceroutecmd.java U
udpclient.class
udpclient.java 1
udpserver.class
udpserver.java 1
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
traceroutecmd.java U
traceroutecmd.java > traceroutecmd
1 import java.io.BufferedReader;
2 import java.io.InputStreamReader;
3 //39110926-SHAIK JAVEED SUHAIL
4 public class traceroutecmd {
5 public static void runSystemCommand(String command) {
6 try {
7 Process p = Runtime.getRuntime().exec(command);
8 BufferedReader inputStream = new BufferedReader(new InputStreamReader(p
9 String s = "";
10 while ((s = inputStream.readLine()) != null)
11 System.out.println(s);
12 } catch (Exception e) {
PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
over a maximum of 30 hops:
1 11 ms 1 ms 5 ms dsldevice.lan [192.168.1.1]
2 6 ms 4 ms 5 ms abts-tn-dynamic-1.132.78.171.airtelbroadband.in [171.78.132.1]
3 4 ms 12 ms 4 ms 125.17.36.41
4 136 ms 136 ms 137 ms 182.79.141.35
5 138 ms 167 ms 135 ms 182.79.154.42
6 6 ms 7 ms 6 ms 116.119.94.40
7 147 ms 135 ms 136 ms 182.79.134.146
8 138 ms 135 ms 142 ms mei-b5-link.ip.twelve99.net [62.115.42.118]
9 154 ms 149 ms 149 ms prs-bb2-link.ip.twelve99.net [62.115.124.56]
Activate Windows
Go to Settings to activate Windows.
main*
0 3
Ln 3, Col 31
Spaces: 4
UTF-8
CRLF
Java
Go Live
JavaSE-11
Prettier
Type here to search
34°C Light rain
1:39 PM
10/10/2021


File Edit Selection View Go Run Terminal Help
traceroutecmd.java - NetworkingLab - Visual Studio Code
EXPLORER
NETWORKINGL...
dateserver.java
FTPClient.class U
FTPClient.java U
FTPServer.class U
FTPServer.java U
ipclient.class
ipclient.java
myhttp.class U
myhttp.java 1, U
random.txt U
README.md
Sip.class
Sip.java
traceroutecmd.class U
traceroutecmd.java U
udpclient.class
udpclient.java 1
udpserver.class
udpserver.java 1
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
traceroutecmd.java U
traceroutecmd.java > traceroutecmd
1 import java.io.BufferedReader;
PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
4 136 ms 136 ms 137 ms 182.79.141.35
5 138 ms 167 ms 135 ms 182.79.154.42
6 6 ms 7 ms 6 ms 116.119.94.40
7 147 ms 135 ms 136 ms 182.79.134.146
8 138 ms 135 ms 142 ms mei-b5-link.ip.twelve99.net [62.115.42.118]
9 154 ms 149 ms 149 ms prs-bb2-link.ip.twelve99.net [62.115.124.56]
10 150 ms 150 ms 150 ms adm-bb4-link.ip.twelve99.net [213.155.136.167]
11 160 ms 157 ms 158 ms adm-b2-link.ip.twelve99.net [62.115.141.37]
12 152 ms 152 ms 152 ms ae15.ibrsa0105-01.ams3.bb.godaddy.com [62.115.184.19]
13 151 ms 152 ms 161 ms ae2.ams3-bbsa0106-01.bb.gdinf.net [188.121.32.5]
14 154 ms 180 ms 156 ms 188.121.32.115
15 * * * Request timed out.
16 * * * Request timed out.
17 * * * Request timed out.
18 * * * Request timed out.
19 157 ms 157 ms 157 ms ip-160-153-137-14.ip.secureserver.net [160.153.137.14]
Trace complete.
C:\Users\user\Desktop\NetworkingLab>
C:\Users\user\Desktop\NetworkingLab>
Activate Windows
Go to Settings to activate Windows.
main*
Ln 3, Col 31 Spaces: 4 UTF-8 CRLF Java Go Live JavaSE-11 Prettier
Type here to search
34°C Light rain
1:39 PM
10/10/2021

10. Ping Command

39110926
19511559

Aim :- To write a program to execute ping command.

Algorithm :-

1. Import required io.* packages.
2. Initialize the process & buffered reader class to get runtime & input stream class.
3. Initialize a string "www"
4. If inputstream is not null print the ping command.
5. Close the Buffered reader class.
6. End of program.

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 InputStream
                                        Reader(p.getInputstream()));
            String s="www";
            while ((s=Inputstream.readline())!=null){

                System.out.println(s);
            }
        }
        catch(Exception e){

            e.printStackTrace();
    } }
```

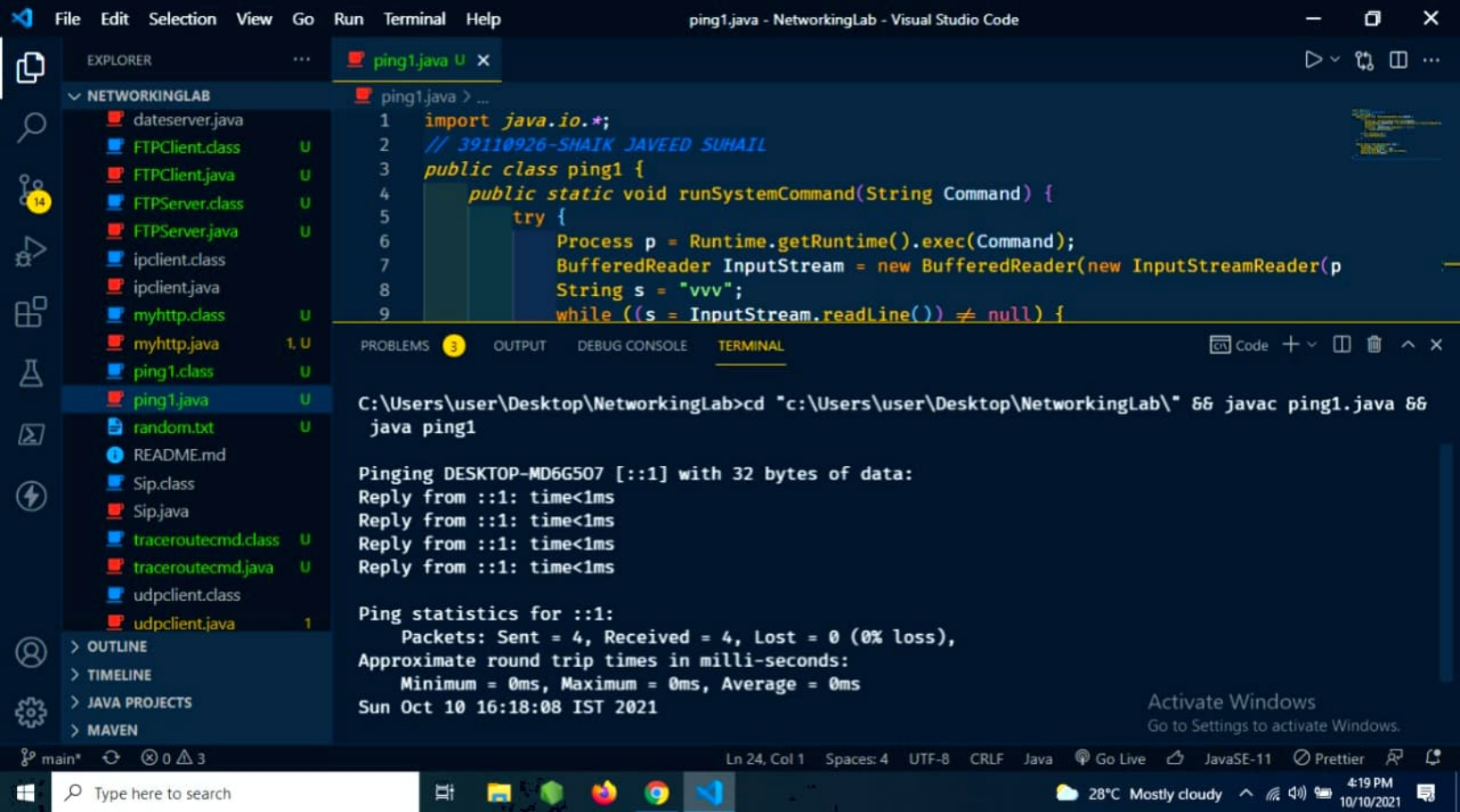
```
public static void main(String[] args) {
        String ip="localhost";
        runSystemCommand("ping"+ip);
        java.util.Date date=new java.util.Date();
        System.out.println(date);
        }
}
```

Output :-

```
Pinging DESKTOP-MD6A507 [::1] with 32 bytes of data:
Reply from ::1: time<1ms
Reply from ::1: time<1ms
Reply from ::1: time<1ms
Reply from ::1: time<1ms

Ping statistics for ::1:
        Packets: Sent=4, Received=4, Lost=0
Approximate round trip times in milliseconds:
        Minimum=0ms, Maximum=0ms, Average=0ms
SUN OCT 10 16:18:08 IST 2021
```

Result :- Program to create ping command has been executed successfully.


File Edit Selection View Go Run Terminal Help
ping1.java - NetworkingLab - Visual Studio Code
EXPLORER
NETWORKINGLAB
dateserver.java
FTPClient.class U
FTPClient.java U
FTPServer.class U
FTPServer.java U
ipclient.class
ipclient.java
myhttp.class U
myhttp.java 1, U
ping1.class U
ping1.java U
random.txt U
README.md
Sip.class
Sip.java
traceroutecmd.class U
traceroutecmd.java U
udpclient.class
udpclient.java 1
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
ping1.java U
ping1.java > ...
1 import java.io.*;
2 // 39110926-SHAIK JAVEED SUHAIL
3 public class ping1 {
4 public static void runSystemCommand(String Command) {
5 try {
6 Process p = Runtime.getRuntime().exec(Command);
7 BufferedReader InputStream = new BufferedReader(new InputStreamReader(p
8 String s = "vvv";
9 while ((s = InputStream.readLine()) != null) {
PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
C:\Users\user\Desktop\NetworkingLab>cd "c:\Users\user\Desktop\NetworkingLab\" && javac ping1.java && java ping1
Pinging DESKTOP-MD6G5O7 [::1] with 32 bytes of data:
Reply from ::1: time<1ms
Reply from ::1: time<1ms
Reply from ::1: time<1ms
Reply from ::1: time<1ms
Ping statistics for ::1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
Sun Oct 10 16:18:08 IST 2021
Activate Windows
Go to Settings to activate Windows.
main*
0 3
Ln 24, Col 1
Spaces: 4
UTF-8
CRLF
Java
Go Live
JavaSE-11
Prettier
Type here to search
28°C Mostly cloudy
4:19 PM
10/10/2021

11. Case Study

39110726
19511591

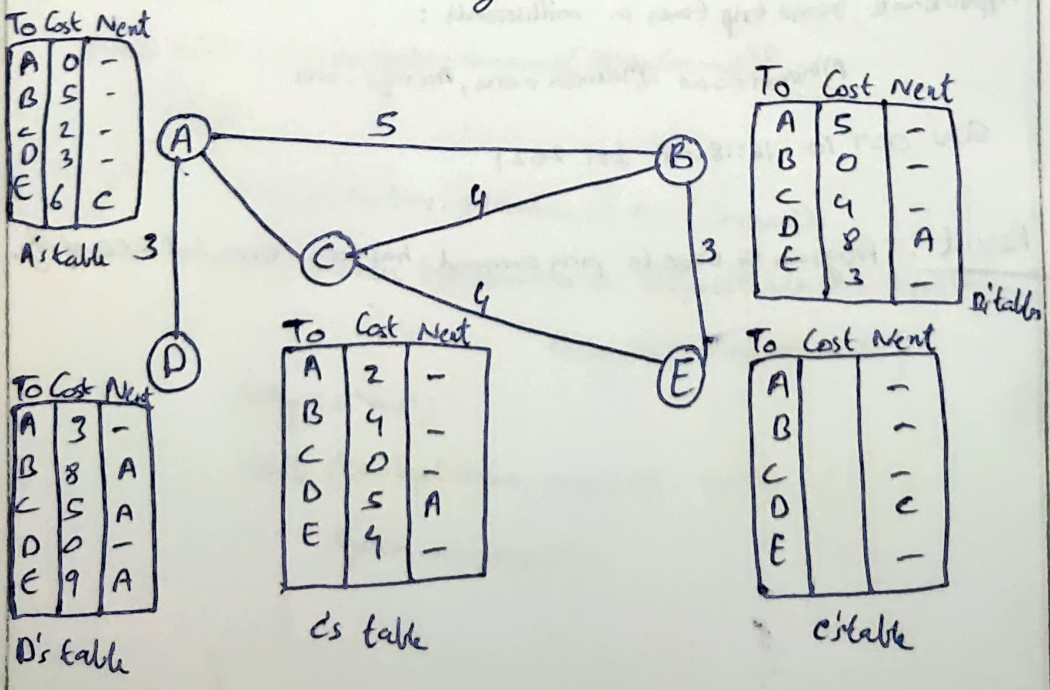
a) Distance Vector Routing

Aim:- Case study on the Distance Vector Routing (RIP)

***Process:-**

In distance vector routing the least cost route between any two nodes is the route with minimum distance. In this protocol each node maintains a vector table of minimum distance to every nodes. The table at each node also guide the packet to the desired node by showing next stop in the route.

→We can think of nodes as the cities in an area & the lines as roads connecting them.



Initialization:- Each nodes knows how to reach any other node and the cost. At the beginning however this is

not the case. Each node knows only the distance between itself & its neighbours those directly connected to it. So for the moment we assume that each node can send a message to the immediate neighbours, those directly connected to it, find the distance between itself & neighbours.

Sharing :- The whole idea of distance vector routing is the sharing of information between neighbors. Although node A does not know about node E, node C does. So if node C shares its routing table with A, node A can also know how to reach node E. On the other hand node C does not know how to reach node D, but node A does. If node A shares its routing table with node C, node C also knows how to reach node D. In other words node A & C as immediate neighbours, can improve their routing table if they help each other.

Updating :- When nodes receive a two column table from a neighbour it needs to update its routing table. It takes 3 steps

1) The receiving node needs to add the cost between itself & the sending node to each value in second column. The logic is clear

2) The receiving node needs to add the name of the sending node to each row as the third column if the receiving nodes uses information from any row. The sending node is next node in the route.

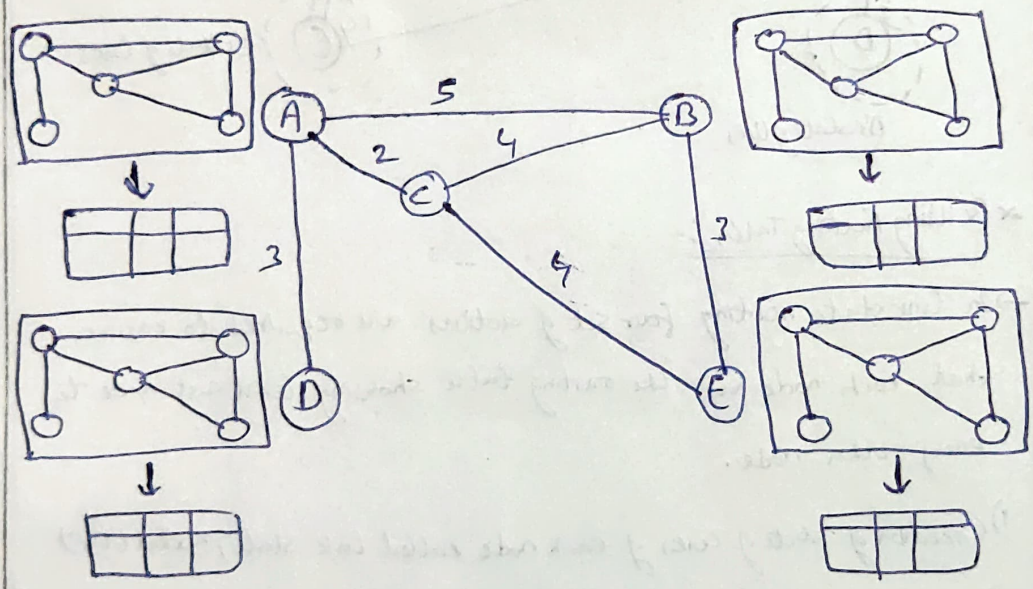
3) The receiving node needs to compare each row of its table with the corresponding row of the modified version of received table.

39110926

b) Link State Routing

Aim :- Case study on Link state Routing

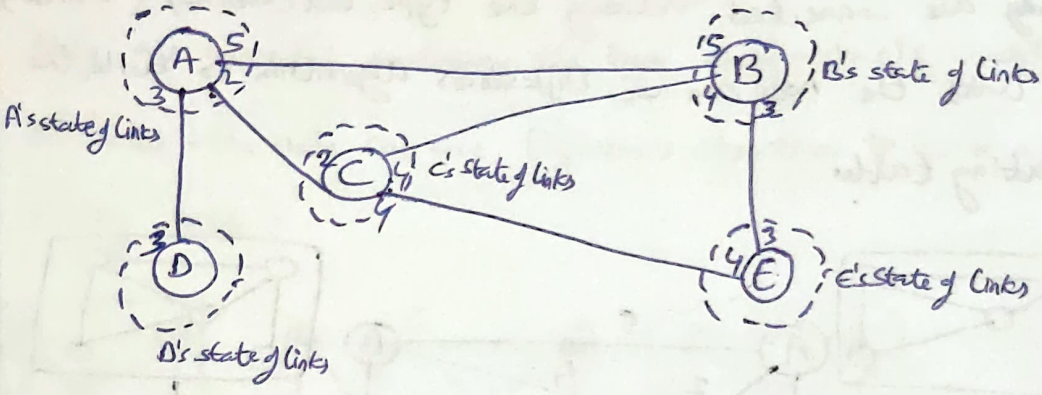
*Process :- In Link state routing it each node in domain has entire topology of domain of the list of nodes & links how they are connected including the type, cost (metric) & conditions of links the node can use Dijkstras algorithm to build the routing table



→ The figure shows a simple domain with five nodes each node can use the same topology to create a routing table, but the routing table for each node is unique because the calculations are based on different interpretation of topology. This is analogy for a city map, while each person may have same map, each needs to take a different route to reach their specific destination

→ The topology must be dynamic representing the latest state of each node & each link. If there are changes in any point of network the topology should be updated for each node.

* No node can know topology at the begining in the network. It is based on the assumption that although the global knowledge about topology is not clear each node has partial knowledge of each node.



* Building Routing Tally :-

→ In Link state routing four set of actions are required to ensure that each node has the routing table showing least cost node to every other node.

1) Creation of state of Links of each node called Link state packet (LSP)
2) Dessemination of LSP of every other router called flooding in a efficient, reliable way.
3) Formation of shortest path tree for each node.
4) Calculation of routing table based on shortest path tree.

LSP :- A Link state packet can carry a large amount of information.

Flooding of LSPs :- After a node has prepared an LSP, it must be dessemineted into all other nodes, not only to its neighbours. This process is called flooding of LSPs.

Formation of shortest Path Tree :- After receiving all LSP's each node will have a copy of whole topology. However topology is not sufficient to find shortest path to every other node ; a shortest path tree is needed.

Dijkstra's Algorithm :-