

Module – 3

Socket Programming

Experiment – 1

20181CSE0621

MODULE-3
SOCKET PROGRAMMING
EXPERIMENT-1

– QUESTION: To find website address using Socket programming.

– CODE:

```
import java.net.*;
import java.util.*;
public class IPfinder {
    public static void main (String[] args)
    {
        String host;
        Scanner inp = new Scanner(System.in);
        System.out.print("\n Enter Hosts name:");
        host = inp.next();
        try {
            InetAddress add = InetAddress.getByName(host);
            System.out.println(" IP : " + add.toString());
        }
        catch (UnknownHostException uhfx)
        {
            System.out.println("Could't find : " + host);
        }
    }
}
```

19

Output :

```
IPFinder.java
1 import java.util.*;
2 import java.net.*;
3 public class IPFinder {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         String host;
8         Scanner input = new Scanner(System.in);
9         System.out.println("Enter Host Name : ");
10        host = input.next();
11        try {
12            InetAddress address = InetAddress.getByName(host);
13            System.out.println("IP address: " + address.toString());
14        }
15        catch (UnknownHostException uhEx)
16        {
17            System.out.println("Could not find " + host);
18        }
19    }
20 }
21
22 }
23
```

```
IPFinder.java
1 import java.util.*;
2 import java.net.*;
3 public class IPFinder {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         String host;
8         Scanner input = new Scanner(System.in);
9         System.out.println("Enter Host Name : ");
10        host = input.next();
11        try {
12            InetAddress address = InetAddress.getByName(host);
13            System.out.println("IP address: " + address.toString());
14        }
15        catch (UnknownHostException uhEx)
16        {
17            System.out.println("Could not find " + host);
18        }
19    }
20 }
21
22 }
23
```

Problems | Javadoc | Declaration | Console

<terminated> IPFinder [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (15-Mar-2021, 9:42:31 am - 9:42:42 am)

Enter Host Name :
presidencyuniversity.in
IP address: presidencyuniversity.in/172.67.170.79

Experiment – 2

To find the local host IP address using socket programming

20181CSE0621

EXPERIMENT-02

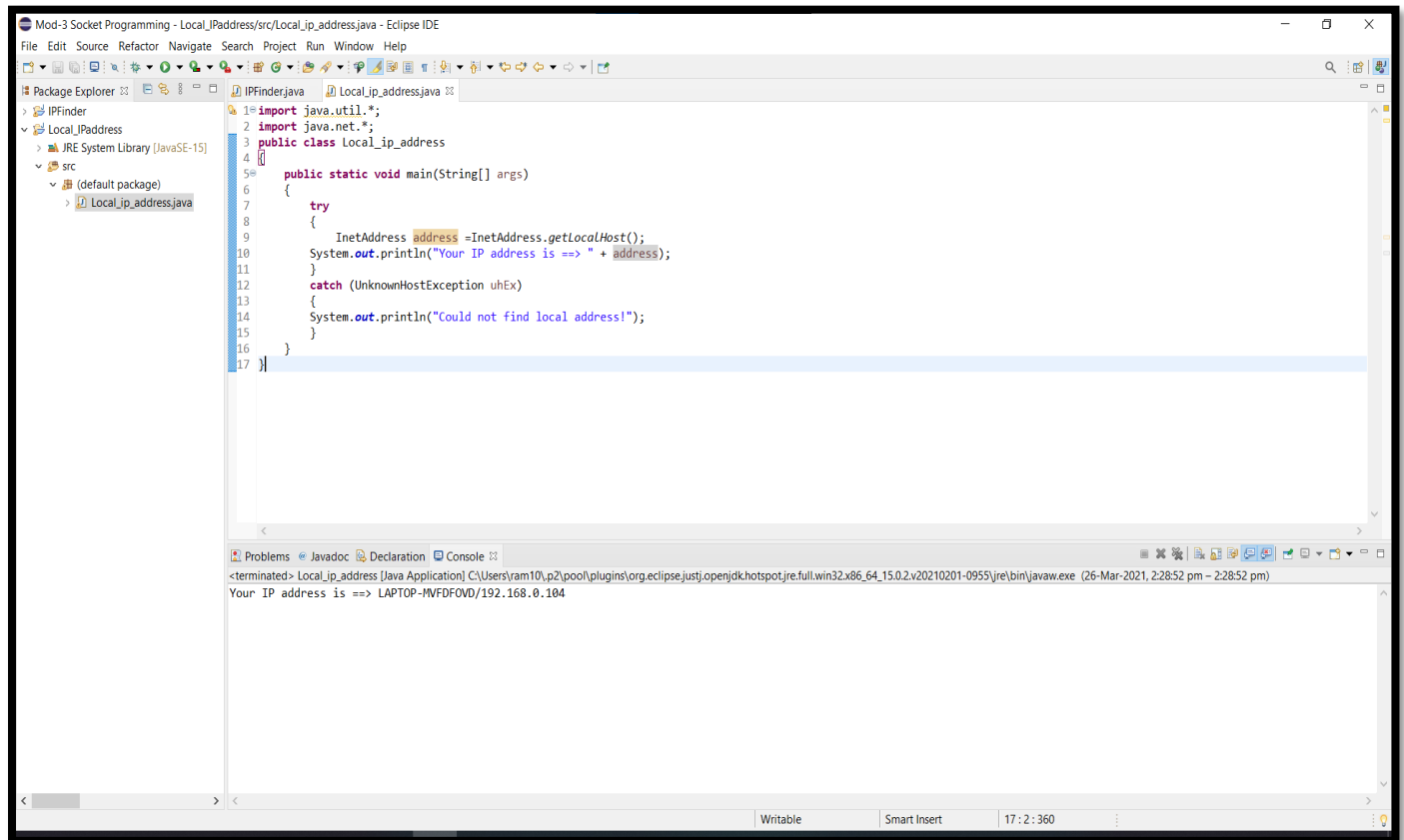
- QUESTION : To find local host IP address using Socket Programming.

- CODE:

```
import java.util.*;
import java.net.*;
public class localIP
{
    public static void main (String [] args)
    {
        try {
            // InetAddress add = InetAddress.getByName
            InetAddress add = InetAddress.getLocalHost();
            System.out.println(add);
        }
        catch (UnknownHostException uhEx)
        {
            System.out.println("Couldnt find local host");
        }
    }
}
```

20

Output :



The screenshot displays the Eclipse IDE interface. The Package Explorer on the left shows the project structure: IPFinder, Local_IPAddress, JRE System Library [JavaSE-15], src, and a default package containing Local_ip_address.java. The main editor window shows the code for Local_ip_address.java:

```
1=import java.util.*;
2=import java.net.*;
3=public class Local_ip_address
4={
5=    public static void main(String[] args)
6=    {
7=        try
8=        {
9=            InetAddress address =InetAddress.getLocalHost();
10=            System.out.println("Your IP address is ==> " + address);
11=        }
12=        catch (UnknownHostException uhEx)
13=        {
14=            System.out.println("Could not find local address!");
15=        }
16=    }
17=}
```

The Console window at the bottom shows the output of the program:

```
<terminated> Local_ip_address [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justi.openjdk hotspot\jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (26-Mar-2021, 2:28:52 pm - 2:28:52 pm)
Your IP address is ==> LAPTOP-MVDF0VD/192.168.0.104
```

The status bar at the bottom indicates the file is Writable, Smart Insert is active, and the time is 17:2:360.

Experiment – 3

Write a program to communicate between client and server using UDP.

20181CSE0621

EXPERIMENT-03

— QUESTION: Write a program to communicate between client & server using UDP protocol.

— CODE:

- Client side:

```
import java.net.*;
import java.io.*;
class client
{
    public static DatagramSocket ds;
    public static byte buffer[] = new byte[1024];
    public static int clientport = 1789, serverport = 1790;
    public static void main(String args[]) throws Exception
    {
        byte buffer[] = new byte[1024];
        ds = new DatagramSocket(clientport);
        BufferedReader bread = new BufferedReader(new InputStreamReader(
            System.in));

        System.out.print("Address");
        String msg = bread.readLine();
        buffer = msg.getBytes();
        ds.send(new DatagramPacket(buffer, msg.length(),
            InetAddress.getLocalHost(), serverport));

        System.out.println("Client is waiting for your data");
        System.out.print("Press ctrl + c to come out");
    }
}
```

21

20181CSE0621

```

while (true)
{
    DatagramPacket dp = new DatagramPacket(buffer, buffer.length);
    ds.receive(dp);
    String pdata = new String(dp.getData(), 0, dp.getLength());
    if (pdata.equals("End"))
        break;
    System.out.println(pdata);
    String str = br.readLine();
    buffer = str.getBytes();
    ds.send(new DatagramPacket(buffer, str.length(),
        InetAddress.getLocalHost(), serverport));
}
}

```

→ Server Side:

```

import java.net.*;
import java.io.*;
class Server
{
    public static DatagramSocket ds;
    public static int clientport = 1789, serverport = 1790;
    public static void main(String[] args) throws Exception
    {
        byte buffer = new byte[1024];
        ds = new DatagramSocket(serverport);
        BufferedReader br = new BufferedReader(
            new InputStreamReader(System.in));
    }
}

```

20181CSE0621

```
System.out.println("Waiting for connection");
DatagramPacket dp = new DatagramPacket(buffer, buffer.length);
ds.receive(dp);
String pdata = new String(dp.getData(), 0, dp.getLength());
System.out.println("Connected");
String st = "Hello";
buffer = st.getBytes();
ds.send(new DatagramPacket(buffer, st.length(), clientport));
break;
while(true)
{
    ds.receive(dp);
    String recv = new String(dp.getData(), 0, dp.getLength());
    System.out.println(recv);
    buffer = st.getBytes();
    if(st == NULL || st.equals("end"))
    {
        ds.send(new DatagramPacket(buffer, st.length,
            InetAddress.getLocalHost(), clientport));
        break;
    }
    ds.send(new DatagramPacket(buffer, st.length,
        InetAddress.getLocalHost(), clientport));
}
}
```

Output :

Client Request

```
server_datagram.java
6 public static DatagramSocket ds;
7 public static int clientport=3020, serverport=3021;
8
9 public static void main(String[] args) throws Exception
10 {
11     byte buffer[] = new byte[2048];
12     ds = new DatagramSocket(serverport);
13     BufferedReader breader = new BufferedReader(new InputStreamReader(System.in));
14     DatagramPacket dp = new DatagramPacket(buffer,buffer.length);
15     ds.receive(dp);
16     System.out.println("Connected ...");
17     String pdata = new String(dp.getData(),0,dp.getLength());
18     System.out.println("Message from Client : " + pdata);
19     while(true)
20     {
21         System.out.println("Enter response Message : ");
22         String str = breader.readLine();
23         buffer = str.getBytes();
24         if(str==null||str.equals("End"))
25         {
26             ds.send(new DatagramPacket(buffer,str.length(),InetAddress.getLocalHost(),clientport));
27             break;
28         }
29     }
30 }
```

```
client_datagram.java
1 import java.util.*;
2
3 public class client_datagram {
4     public static DatagramSocket ds;
5     public static int clientport=3020, serverport=3021;
6
7     public static void main(String[] args) throws Exception
8     {
9         byte buffer[] = new byte[2048];
10        ds = new DatagramSocket(clientport);
11        BufferedReader breader = new BufferedReader(new InputStreamReader(System.in));
12
13        System.out.println("Enter Message : ");
14        String msg=breader.readLine();
15        buffer=msg.getBytes();
16        ds.send(new DatagramPacket(buffer,msg.length(),InetAddress.getLocalHost(),serverport));
17
18        while(true)
19        {
20            // ...
21        }
22    }
23 }
```

Console Output:

```
server_datagram [Java Application] C:\Users\ram10\p2\pool\p
Connected ...
Message from Client : Hello This is Client
Enter response Message :
```

Server Response

```
server_datagram.java
6 public static DatagramSocket ds;
7 public static int clientport=3020, serverport=3021;
8
9 public static void main(String[] args) throws Exception
10 {
11     byte buffer[] = new byte[2048];
12     ds = new DatagramSocket(serverport);
13     BufferedReader breader = new BufferedReader(new InputStreamReader(System.in));
14     DatagramPacket dp = new DatagramPacket(buffer,buffer.length);
15     ds.receive(dp);
16     System.out.println("Connected ...");
17     String pdata = new String(dp.getData(),0,dp.getLength());
18     System.out.println("Message from Client : " + pdata);
19     while(true)
20     {
21         System.out.println("Enter response Message : ");
22         String str = breader.readLine();
23         buffer = str.getBytes();
24         if(str==null||str.equals("End"))
25         {
26             ds.send(new DatagramPacket(buffer,str.length(),InetAddress.getLocalHost(),clientport));
27             break;
28         }
29     }
30 }
```

```
client_datagram.java
1 import java.util.*;
2
3 public class client_datagram {
4     public static DatagramSocket ds;
5     public static int clientport=3020, serverport=3021;
6
7     public static void main(String[] args) throws Exception
8     {
9         byte buffer[] = new byte[2048];
10        ds = new DatagramSocket(clientport);
11        BufferedReader breader = new BufferedReader(new InputStreamReader(System.in));
12
13        System.out.println("Enter Message : ");
14        String msg=breader.readLine();
15        buffer=msg.getBytes();
16        ds.send(new DatagramPacket(buffer,msg.length(),InetAddress.getLocalHost(),serverport));
17
18        while(true)
19        {
20            // ...
21        }
22    }
23 }
```

Console Output:

```
client_datagram [Java Application] C:\Users\ram10\p2\pool\p
Enter Message :
Hello This is Client
Message from Server : Hello This is Server
```


Experiment – 4

Write a program to communicate between client and server using TCP Protocol

20181CSE0621

EXPERIMENT-04

- QUESTION:- Write a program to communicate between client & server using TCP protocol.
- CODE:
- Client side:

```
import java.net.*;
import java.io.*;

public class tcpclient
{
    public static void main(String[] args) throws IOException
    {
        Socket s = new Socket("localhost", 55);

        DataInputStream in = new DataInputStream(s.getInputStream());
        DataOutputStream out = new DataOutputStream(s.getOutputStream());
        DataInputStream sysin = new DataInputStream(System.in);

        while (true)
        {
            String str = in.readLine();
            System.out.println("Message from Server" + str);
            if (str.equals("end"))
                break;
            System.out.println("Enter reply: ");
            String line = sysin.readLine();
            out.writeBytes(line + "\n");
        }
        s.close();
    }
}
```

24

20181CSE0621

→ Server Side:

```
import java.io.*;
import java.net.*;

class TcpServer
{
    public static void main (String[] args) throws IOException
    {
        ServerSocket ss = new ServerSocket(55);
        Socket s = ss.accept();
        System.out.println("Connected");
        DataInputStream in = new DataInputStream(s.getInputStream());
        DataOutputStream out = new DataOutputStream(s.getOutputStream());
        DataInputStream sysin = new DataInputStream(System.in);
        while (true)
        {
            System.out.println("Enter a string:");
            String str = sysin.readLine();
            out.writeBytes(str + "\n");
            if (str.equals("end"))
                break;
            System.out.println("Message : " + in.readLine());
        }
        ss.close();
    }
}
```

Output :

```
1 package exp4_tcp;
2 import java.util.*;
3 import java.net.*;
4 import java.io.*;
5 public class tcpServer {
6     public static void main (String [] args) throws IOException {
7         //System.out.println("HI from server");
8         ServerSocket ss=new ServerSocket(50);
9         Socket s=ss.accept();
10        System.out.println("Server Connected....");
11        DataInputStream in=new DataInputStream(s.getInputStream());
12        DataOutputStream out=new DataOutputStream(s.getOutputStream());
13        DataInputStream sysin=new DataInputStream(System.in);
14        while(true)
15        {
16            System.out.println("Enter a string :");
17            String str=sysin.readLine();
18            out.writeBytes(str+"\n");
19            if(str.equals("End")) break;
20            System.out.println("Message from client : "+in.readLine());
21        }
22    }
23 }
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
1 //System.out.println("HI from client");
2
3 Socket s=new Socket("localhost",50);
4 DataInputStream in=new DataInputStream(s.getInputStream());
5 DataOutputStream out=new DataOutputStream(s.getOutputStream());
6 DataInputStream sysin=new DataInputStream(System.in);
7 while(true)
8 {
9     String str=in.readLine();
10    System.out.println("Message from server : "+str);
11    if(str.equals("End"))
12        break;
13    System.out.println("Enter reply message : ");
14    String lines=sysin.readLine();
15    out.writeBytes(lines+"\n");
16 }
17 s.close();
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
tcpServer (1) [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0
Server Connected....
Enter a string :
this is client
Message from client : this is server
Enter a string :
```

Experiment – 5

Host Connectivity

20181CSE0621

EXPERIMENT-05

— QUESTION: Program to check connectivity of given Hostname.

— CODE:

```
import java.net.*;
import java.io.*;

public class ping {
    public static void main (String[] args) throws IOException {
        String Host = " ";
        Scanner inp = new Scanner (System.in);
        System.out.println("Enter host name:");
        host = inp.next();
        try {
            InetAddress add = InetAddress.getByName(Host);
            System.out.println("IP: " + add.toString());
            System.out.println("Sending ping request to" + host);
            if (add.isReachable(5000))
                System.out.println(host + "is reachable");
            else
                System.out.println(host + "is not reachable");
        }
        catch (UnknownHostException uhEx)
        {
            System.out.println("Could't find" + host);
        }
    }
}
```

26

Output :



```
Console [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (09-Apr-2021, 2:11:59 pm - 2:12:07 pm)

<terminated> ping [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (09-Apr-2021, 2:11:59 pm - 2:12:07 pm)

Enter host name: google.com
IP address: google.com/172.217.31.206
Sending Ping Request to google.com
google.com is reachable.
```



```
Console [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (09-Apr-2021, 2:26:17 pm - 2:26:22 pm)

<terminated> ping [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (09-Apr-2021, 2:26:17 pm - 2:26:22 pm)

Enter host name: 123
IP address: /0.0.0.123
Sending Ping Request to 123
123 NOT reachable.
```

Experiment – 6

ARP Protocol

Client Side :

20181CSE0621

EXPERIMENT-06

— QUESTION: Write a program to implement the ARP protocol using socket programming.

— CODE:

→ Client side:

```
import java.io.*;
import java.net.*;
class arp-client
{
    public static void main (String[] args) throws IOException
    {
        Socket s = new Socket("localhost", 55);
        DataInputStream in = new DataInputStream(s.getInputStream());
        DataOutputStream out = new DataOutputStream(s.getOutputStream());
        DataInputStream sysin = new DataInputStream(System.in);
        System.out.println("Enter IP address:");
        String str = sysin.readLine();
        out.writeBytes(str + "\n");
        System.out.println("The MAC address is: " + in.readLine());
    }
}
```

27

Server side:-

20181CSE0621

→ Server side:

```

import java.net.*;
import java.io.*;
public class arpserver
{
    public static void main (String[] args)
    {
        ServerSocket ss = new ServerSocket(55);
        Socket s = s.accept();
        DataInputStream in = new DataInputStream(s.getInputStream());
        DataOutputStream out = new DataOutputStream(s.getOutputStream());
        String iparr[] = {"10.0.1.45", "172.16.5.21", "172.16.5.22"};
        String macarr[] = {"00-0c-6e-5c-3c-63", "02-11-86-F3-EF-21",
                           "03-12-83-F3-EF-18"};

        String str = in.readLine();
        System.out.println("Ip received " + str);
        int flag = 0;
        for (int i = 0; i < 3; i++)
        {
            if (str.equals(iparr[i]) == true)
            {
                flag = 1;
                String str1 = macarr[i];
                out.writeBytes(str + "\n");
                break;
            }
        }
        if (flag == 0)
        {
            System.out.println("IP not in network");
            s.close();
        }
    }
}

```

OUTPUT

```
clientARP.java
1 package exp6_ARP;
2 import java.io.*;
4 class clientARP
5 {
6     public static void main(String args[]) throws IOException
7     {
8         Socket s=new Socket("localhost",53);
9         DataInputStream in=new DataInputStream(s.getInputStream());
10        DataOutputStream out=new DataOutputStream(s.getOutputStream());
11        DataInputStream sysin=new DataInputStream(System.in);
12        System.out.println("Enter an IP Address:");
13        String str=sysin.readLine();
14        out.writeBytes(str+"\n");
15        System.out.println("The corresponding MAC address \n"+in.readLine());
16    }
17 }

serverARP.java
1 package exp6_ARP;
2 import java.net.*;
4 public class serverARP {
5
6     public static void main(String args[]) throws IOException
7     {
8         ServerSocket ss=new ServerSocket(53);
9         Socket s=ss.accept();
10        DataInputStream in=new DataInputStream(s.getInputStream());
11        DataOutputStream out=new DataOutputStream(s.getOutputStream());
12        String iparr[]={"10.0.1.45","172.16.5.21","172.16..5.22"};
13        String macarr[]={"00-0c-6e-5c-3c-63","02-11-B6-F3-EF-21","03-12-B3-F3-EF-18"};
14        String str=in.readLine();
15        System.out.println("Ip Address received from server"+str);
16        int flag=0;
17        for(int i=0;i<3;i++)
18        {
19            if(str.equals(iparr[i])==true)
20            {
21                flag=1;
22                String str1=macarr[i];
23                out.writeBytes(str1+"\n");
24                break;
25            }
26        }
27    }
28 }
```

```
Console
<terminated> clientARP [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot
Enter an IP Address:
10.0.1.45
The corresponding MAC address
00-0c-6e-5c-3c-63
```

```
Console
<terminated> clientARP [Java Application] C:\Users\ram10\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot
Enter an IP Address:
172.16.5.21
The corresponding MAC address
02-11-B6-F3-EF-21
```


Experiment – 7

FTP Protocol

Client Side: -

20181CSE0621

EXPERIMENT-07

→ QUESTION: Write a program to implement the FTP protocol using socket programming.

→ CODE:-

- Client side:

```
import java.io.*;
import java.net.*;
public class ftpclient {
    public static void main(String[] args)
    {
        Socket s = new Socket(InetAddress.getLocalHost(), 5555);
        DataInputStream s1 = new DataInputStream(s.getInputStream());
        DataInputStream inp = new DataInputStream(System.in);
        DataOutputStream so = new DataOutputStream(s.getOutputStream());
        System.out.println("Enter path: \n");
        String str = inp.readLine();
        FileOutputStream fos = new FileOutputStream("output.txt");
        int str1;
        while ((str1 = s1.read()) != -1)
            fos.write((char)str1);
        System.out.println("File received \n");
        s1.close();
        so.close();
        inp.close();
        s.close();
    }
}
```

29

Server side: -

20181CSE0621

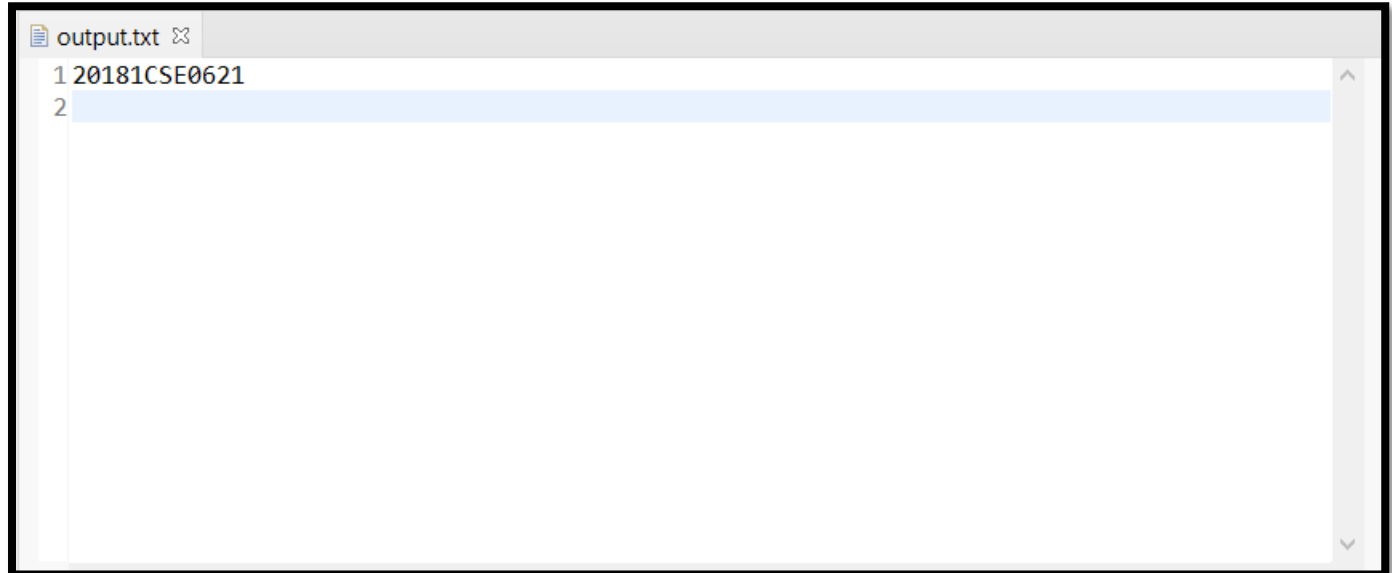
→ Server Side:

```
import java.net.*;
import java.io.*;
class Jfpserver
{
    public static void main (String[] args)
    {
        ServerSocket ss = new ServerSocket(5555);
        Socket s = ss.accept();
        DataOutputStream dos = new DataOutputStream(s.getOutputStream());
        DataInputStream din = new DataInputStream(s.getInputStream());
        String s1;
        s1 = din.readLine();
        FileInputStream fin = new FileInputStream(s1);
        int str1;
        while ((str1 = fin.read()) != -1)
            dos.writeBytes(("(char)str1");
        System.out.println("File send");
        dos.close();
        din.close();
        s.close();
    }
}
```

OUTPUT

```
clientFTP.java
1 package exp7_FTP;
2 import java.io.*;
4 public class clientFTP
5 {
6     public static void main(String a[]) throws IOException
7     {
8         Socket s=new Socket(InetAddress.getLocalHost(),5553);
9         DataInputStream s1=new DataInputStream(s.getInputStream());
10        DataInputStream inp=new DataInputStream(System.in);
11        DataOutputStream so=new DataOutputStream(s.getOutputStream());
12        System.out.println("\n enter the filename(path)");
13        String str=inp.readLine();
14        so.writeBytes(str+"\n");
15        FileOutputStream fos=new FileOutputStream("output.txt");
16        int str1;
17        while((str1=s1.read())!=-1)
18            fos.write((char)str1);
19        System.out.println("\n file received successfully");
20        s1.close();
21        so.close();
22        inp.close();
23        s.close();
24    }
25 }
```

```
serverFTP.java
1 package exp7_FTP;
2 import java.io.*;
4 public class serverFTP
5 {
6     public static void main(String a[]) throws IOException
7     {
8         ServerSocket ss=new ServerSocket(5553);
9         Socket s=ss.accept();
10        DataOutputStream dos=new DataOutputStream(s.getOutputStream());
11        DataInputStream din=new DataInputStream(s.getInputStream());
12        String s1;
13        s1=din.readLine();
14        FileInputStream fin=new FileInputStream(s1);
15        int str1;
16        while((str1=fin.read())!=-1)
17            dos.writeBytes(""+(char)str1);
18        System.out.println("\n file successfully sent");
19        dos.close();
20        din.close();
21        s.close();
22    }
23 }
```



```
output.txt
1 20181CSE0621
2
```

