

ComputerNetworks Lab Exercise 3

AIM:

To write a socket program for implementation of echo

Algorithm:

CLIENT :

1. Start the program.
2. Create a socket which binds the Ip address of server and the port address to acquire service.
3. After establishing connection send a data to server.
4. Receive and print the same data from server.
5. Close the socket.
6. End the program

SERVER

1. Start the program.
2. Create a server socket to activate the port address.
3. Create a socket for the server socket which accepts the connection.
4. After establishing connection receive the data from client.
5. Print and send the same data to client.
6. Close the socket.
7. End the program.

Code:

EchoClient.java

```
package networkslab;

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

public class EchoClient {

    public static void main(String [] args) {

        Socket s1;

        BufferedReader r1,r2;

        PrintStream p1;

        String op="";

        try {
```

ComputerNetworks Lab Exercise 3

```
s1=new Socket(InetAddress.getLocalHost(),8080);
r1=new BufferedReader(new InputStreamReader(System.in));
r2=new BufferedReader(new InputStreamReader(s1.getInputStream()));
p1=new PrintStream(s1.getOutputStream());
while(!op.equals("end")) {
    System.out.print("\nClient :");
    op=r1.readLine();
    p1.println(op);
    System.out.print("\nServer:"+r2.readLine());
}
} catch (Exception e) {
    e.printStackTrace();
}
}
```

EchoServer.java

```
package networkslab;

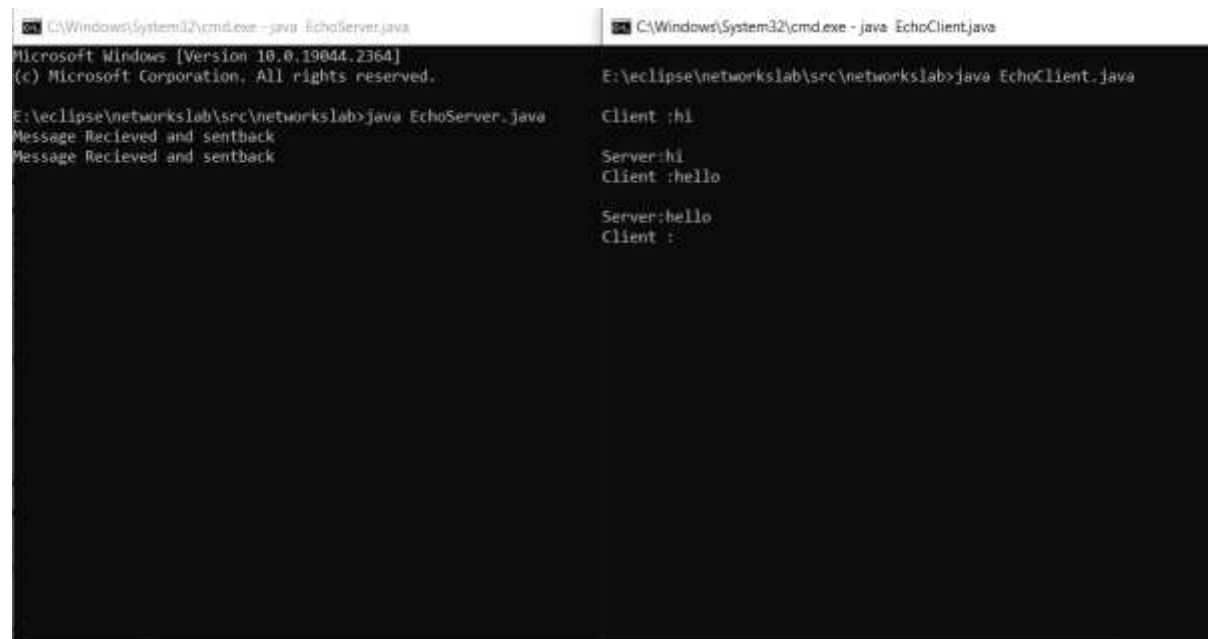
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.net.ServerSocket;
import java.net.Socket;

public class EchoServer {
    public static void main(String[] args) {
        String op1="";
        Socket s1;
```

ComputerNetworks Lab Exercise 3

```
ServerSocket serv;  
  
BufferedReader r2;  
  
PrintStream p1;  
  
try {  
  
    serv=new ServerSocket(8080);  
  
    s1=serv.accept();  
  
    r2=new BufferedReader(new InputStreamReader(s1.getInputStream()));  
  
    p1=new PrintStream(s1.getOutputStream());  
  
    while(!op1.equals("end")) {  
  
        op1=r2.readLine();  
  
        System.out.println("Message Recieved and sentback");  
  
        p1.println(op1);  
  
    }  
}catch(Exception e) {  
  
    e.printStackTrace();  
  
}  
  
}  
  
}
```

OUTPUT:



```
C:\Windows\System32\cmd.exe - java EchoServer.java  
Microsoft Windows [Version 10.0.19044.2364]  
(c) Microsoft Corporation. All rights reserved.  
  
E:\eclipse\networkslab\src\networkslab>java EchoServer.java  
Message Recieved and sentback  
Message Recieved and sentback  
  
C:\Windows\System32\cmd.exe - java EchoClient.java  
E:\eclipse\networkslab\src\networkslab>java EchoClient.java  
Client :hi  
Server:hi  
Client :hello  
Server:hello  
Client :
```

Result: Thus the echo server and client was implemented using java sockets library.

(3.2)

AIM: To write a client-server application for chat using TCP

Algorithm :

CLIENT:

1. Start the program
2. Include necessary package in java
3. To create a socket in client to server.
4. The client establishes a connection to the server.
5. The client accept the connection and to send the data from client to server.
6. The client communicates the server to send the end of the message
7. Stop the program.

Server:

1. Start the program
2. Include necessary package in java
3. To create a socket in server to client
4. The server establishes a connection to the client.
5. The server accept the connection and to send the data from server to client and
6. vice versa
7. The server communicate the client to send the end of the message.
8. Stop the program.

Code:

TCPClient.java

```
package networkslab;
```

```
import java.io.BufferedReader;
```

```
import java.io.InputStreamReader;
```

```
import java.io.PrintStream;
```

```
import java.net.InetAddress;
```

```
import java.net.Socket;
```

ComputerNetworks Lab Exercise 3

```
public class TCPClient {  
    public static void main(String [] args) {  
        Socket s1;  
        BufferedReader r1,r2;  
        PrintStream p1;  
        String op="";  
        try {  
            s1=new Socket(InetAddress.getLocalHost(),8080);  
            r1=new BufferedReader(new InputStreamReader(System.in));  
            r2=new BufferedReader(new InputStreamReader(s1.getInputStream()));  
            p1=new PrintStream(s1.getOutputStream());  
            while(!op.equals("end")) {  
                System.out.print("Client :");  
                op=r1.readLine();  
                p1.println(op);  
                System.out.println("Server: "+r2.readLine());  
            }  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```

TCPServer.java

```
package networkslab;  
  
import java.io.BufferedReader;  
import java.io.InputStreamReader;  
import java.io.PrintStream;  
import java.net.ServerSocket;
```

ComputerNetworks Lab Exercise 3

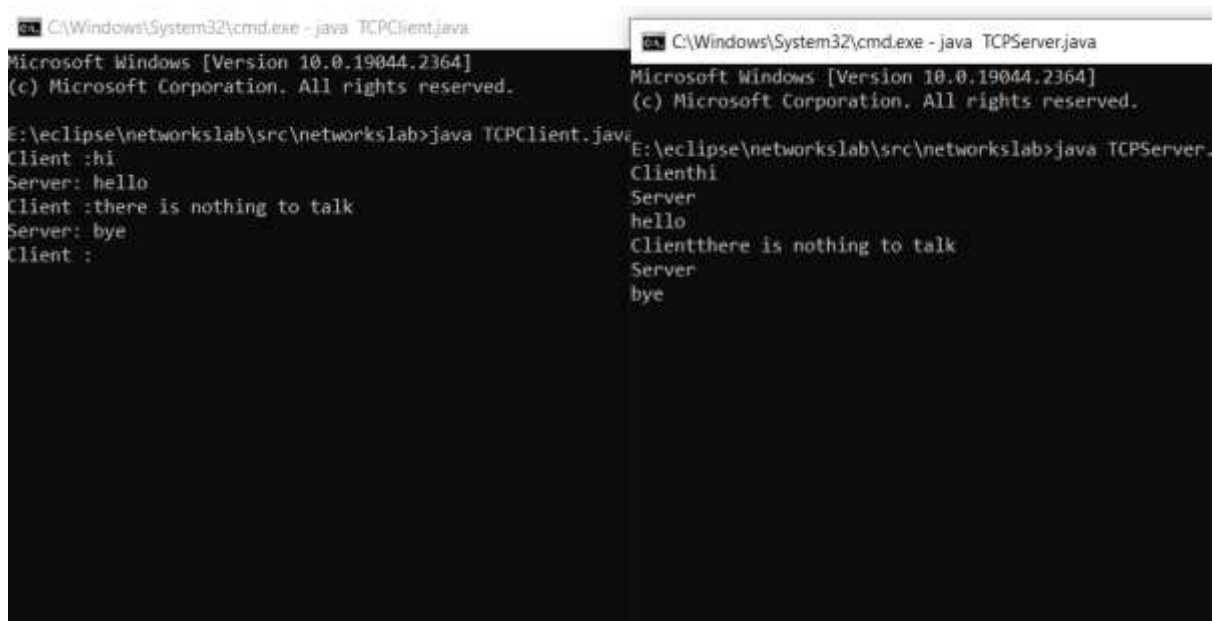
```
import java.net.Socket;

public class TCPServer {
    public static void main(String[]args) {
        String op1="";
        Socket s1;
        ServerSocket serv;
        BufferedReader r2,r1;
        PrintStream p1;
        try {
            serv=new ServerSocket(8080);
            s1=serv.accept();
            r1=new BufferedReader(new InputStreamReader(System.in));
            r2=new BufferedReader(new InputStreamReader(s1.getInputStream()));
            p1=new PrintStream(s1.getOutputStream());
            while(!op1.equals("end")) {
                System.out.println("Client"+r2.readLine());
                System.out.println("Server");
                op1=r1.readLine();

                p1.println(op1);
            }
        }catch(Exception e) {
            e.printStackTrace();
        }
    }
}
```

ComputerNetworks Lab Exercise 3

OUTPUT :



```
C:\Windows\System32\cmd.exe - java TCPClient.java
Microsoft Windows [Version 10.0.19044.2364]
(c) Microsoft Corporation. All rights reserved.

E:\eclipse\networkslab\src\networkslab>java TCPClient.java
Client :hi
Server: hello
Client :there is nothing to talk
Server: bye
Client :
```

```
C:\Windows\System32\cmd.exe - java TCPServer.java
Microsoft Windows [Version 10.0.19044.2364]
(c) Microsoft Corporation. All rights reserved.

E:\eclipse\networkslab\src\networkslab>java TCPServer.java
Clienthi
Server
hello
Clientthere is nothing to talk
Server
bye
```

Result:

Thus a simple chat server client setup was initiated using TCP Protocol using java sockets..

(3.3)

AIM:

To Perform File Transfer in Client & Server Using TCP/IP

Algorithm :

CLIENT :

1. Start.
2. Establish a connection between the Client and Server.
3. Socket ss=new Socket(InetAddress.getLocalHost(),1100);
4. Implement a client that can send two requests.
 - i) To get a file from the server.
 - ii) To put or send a file to the server.
5. After getting approval from the server ,the client either get file from the server or send
6. file to the server

SERVER:

1. Start.
2. Implement a server socket that listens to a particular port number.
3. Server reads the filename and sends the data stored in the file for the 'get' request.
4. It reads the data from the input stream and writes it to a file in the server for the 'put' instruction.
5. Exit upon client's request.
6. Stop.

Code :

FileClient.java

```
package networkslab;

import java.io.*;

import java.net.*;

import java.net.InetAddress;

public class FileClient {
```


ComputerNetworks Lab Exercise 3

```
static DataInputStream k;  
static DataOutputStream k1;  
public static void main(String [] args) {  
  
    try(Socket s=new Socket("localhost",8080)){  
        k=new DataInputStream(s.getInputStream());  
        k1=new DataOutputStream(s.getOutputStream());  
        send("E:\\oslab\\oslabex1.pdf");  
        k.close();  
        k1.close();  
    }catch(Exception e) {  
        e.printStackTrace();  
    }  
}  
  
public static void send(String path) {  
    int bytes=0;  
    File f1=new File(path);  
    try {  
        FileInputStream is=new FileInputStream(f1);  
        k1.writeLong(f1.length());  
        byte[] bfr=new byte[4*1024];  
        while((bytes=is.read(bfr))!=-1) {  
            k1.write(bfr,0,bytes);  
            k1.flush();  
        }  
        is.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

```
}
```

FileServer.java

```
package networkslab;

import java.io.*;

import java.net.*;

public class FileServer {

    private static DataInputStream in=null;

    private static DataOutputStream out=null;

    public static void main(String args[]) {

        try(ServerSocket s=new ServerSocket(8080)){

            Socket client;

            client=s.accept();

            in=new DataInputStream(client.getInputStream());

            out=new DataOutputStream(client.getOutputStream());

            recv("oslabex2.pdf");

            in.close();

            out.close();

        }catch(Exception e) {

            e.printStackTrace();

        }

    }

    public static void recv(String fname) {

        int bits=0;

        try {

            FileOutputStream f=new FileOutputStream(fname);

            long inr=in.readLong();

            byte [] buf=new byte[4*1024];

            while(inr>0 && (bits=in.read(buf, 0, (int)Math.min(inr,buf.length )))!=-1) {

                f.write(buf,0,bits);

                inr-=bits;

            }

        }

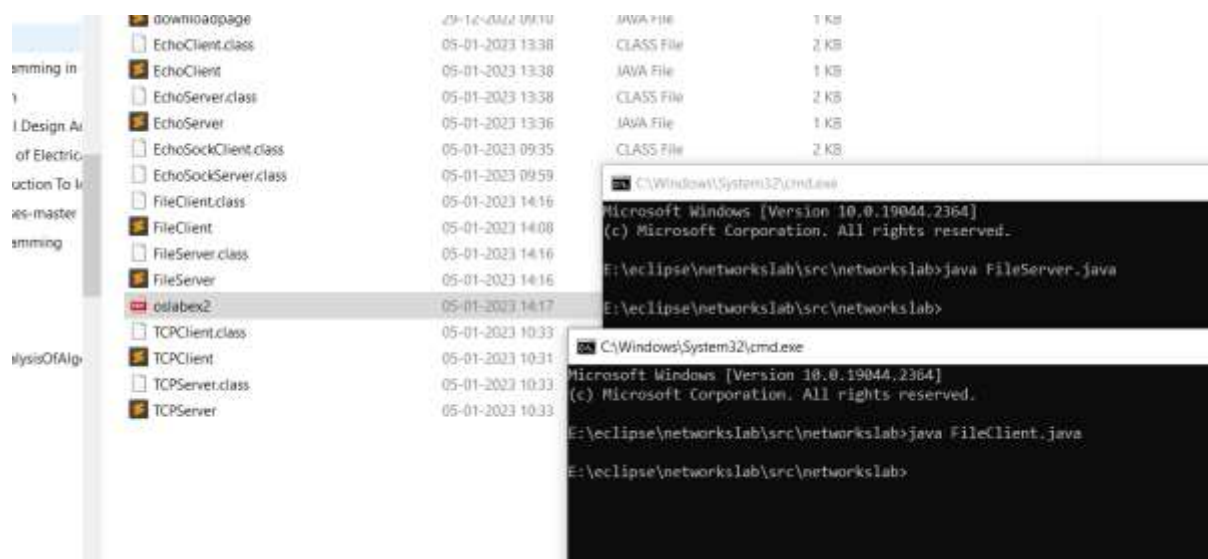
    }

}
```

ComputerNetworks Lab Exercise 3

```
f.close();  
    } catch (IOException e) {  
  
        e.printStackTrace();  
    }  
}  
}
```

OUTPUT:



Result:

Thus the File was sent by the client and received by the fileserver in the specified location.