**Experiment no – 03**

**Aim: Port 17 is known as the ‘Quote of the day service’. When a client connects to port 17 on a server, the server responds with a quote for that day. Write a server program so that it delivers a quote of the day. The quotes should be printable ASCII characters and should contain fewer than 512 characters, although multiple lines are allowed. Since port 17 is considered well known and therefore unavailable, have your server listen to port 6017. Write the client code used to read the quotes returned by the server.**

**Theory: -**

The term "client/server" describes a type of distributed processing in which an application is divided into two parts, each possibly residing on separate operating systems, but working together to provide a service to the end user. As shown in Figure 1, one part of the application, the client, typically resides on a workstation and requests a service for the end-user. The other part of the application, the server, usually resides on a larger machine, such as a mainframe computer. The server program uses the resources of the mainframe computer to perform services requested by each client.

The client is usually the part of the application that is "seen" by the end-user. Therefore, the client half of a client/server application most often resides on a workstation, where the end-user can interact with the application through the workstation operating system's graphical user interface.

Servers, on the other hand, are usually transparent to the end-user. That is, the person who sits at the workstation only perceives the client half of the application, the part that displays the information (though it was actually retrieved by a remote server). Because there is only one server for a given set of clients, servers provide an ideal way of managing shared access to system resources, such as data sets. For this reason, servers are likely to reside on larger machines such as z/OS mainframe computers.

Usually, the same person writes both the client and server parts of a client/server application.

**Program: -**

**Server1.java**

package server1;

import java.net.\*;

import java.io.\*;

import java.util.\*;

public class Server1 {

public static void main(String[] args) {

try {

String Quote = null;

Calendar c = Calendar.getInstance();

Date date = new Date();

c.setTime(date);

int dayOfWeek = c.get(Calendar.DAY\_OF\_WEEK);

switch (dayOfWeek) {

//sunday

case 1:

Quote = "ITS SUNDAY";

break;

//monday

case 2:

Quote = "ITS MONDAY";

break;

//tuesday

case 3:

Quote = "ITS TUESDAY";

break;

//wednesday

case 4:

Quote = "ITS WEDNESDAY";

break;

//thursday

case 5:

Quote = "ITS THURSDAY";

break;

//friday

case 6:

Quote = "ITS FRIDAY";

break;

//saturday

case 7:

Quote = "ITS SATURDAY";

break;

default:

Quote = "NO WEEK DAY";

break;

}

ServerSocket sock = new ServerSocket(6017);

// now listen for connections

while (true) {

Socket client = sock.accept();

// we have a connection

PrintWriter pout = new PrintWriter(client.getOutputStream(), true);

pout.println("Quote of the day service");

pout.println("Yash Prajapati - 022");

// write the Quote to the socket

if (Quote.length() <= 512) {

pout.println(Quote);

} else {

pout.println("More than 512 characters");

}

client.close();

}

} catch (IOException ioe) {

System.err.println(ioe);

}

}

}

**Client1.java**

package client1;

import java.net.\*;

import java.io.\*;

public class Client1 {

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

InputStream in = null;

BufferedReader bin = null;

Socket sock = null;

try {

sock = new Socket("127.0.0.1", 6017);

in = sock.getInputStream();

bin = new BufferedReader(new InputStreamReader(in));

String line;

while ((line = bin.readLine()) != null) {

System.out.println(line);

}

} catch (IOException ioe) {

System.err.println(ioe);

} finally {

try {

sock.close();

} catch (Exception e) {

System.out.println(e.getMessage());

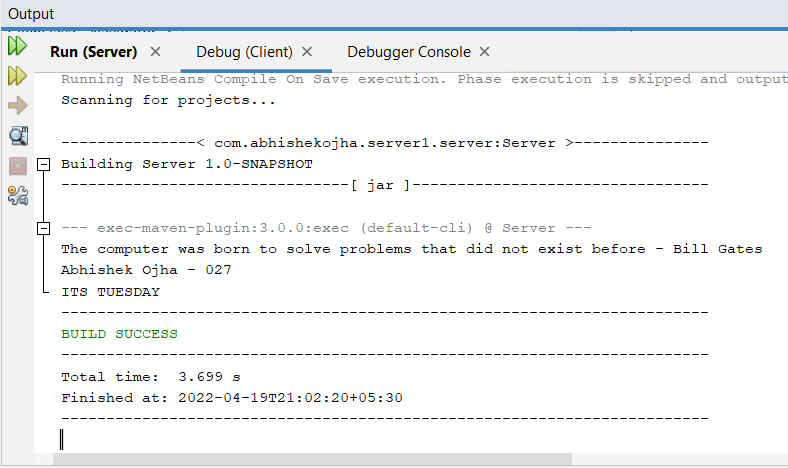
}

}

}

}

**Output:**

****