

PROGRAMS :

// A Java program for a Client

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

public class Client
{
    private Socket socket = null;
    private DataInputStream input = null;
    private DataOutputStream out = null;

    public Client(String address, int port)
    {
        try
        {
            socket = new Socket(address, port);
            System.out.println("Connected");

            input = new DataInputStream(System.in);
            out = new DataOutputStream(socket.getOutputStream());

            // Start a separate thread for receiving messages from the server
            Thread receiveThread = new Thread(new ReceiveMessage());
            receiveThread.start();

            // Read messages from the terminal and send to the server
            String line = "";
            while (!line.equals("Over"))
            {
                line = input.readLine();
                out.writeUTF(line);
                if (line.equals("Over"))
                    break; // Exit loop when "Over" is sent
            }
        }
        catch (UnknownHostException u)
        {
            System.out.println(u);
        }
        catch (IOException i)
        {
            System.out.println(i);
        }
        finally
        {
            try
            {
                input.close();
                out.close();
                socket.close();
            }
            catch (IOException e)
            {
            }
        }
    }
}
```

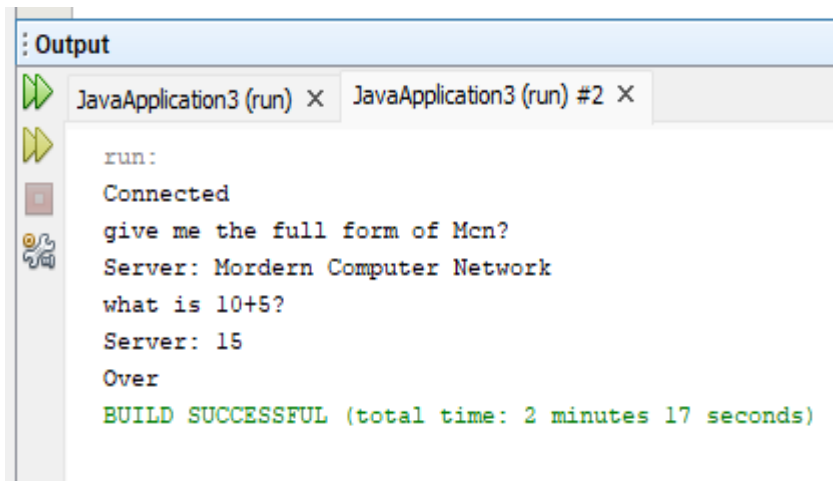
MODERN COMPUTER NETWORKS LAB EXPERIMENTS

```
        {
            //e.printStackTrace();
        }
    }
}

class ReceiveMessage implements Runnable
{
    @Override
    public void run()
    {
        try
        {
            DataInputStream in = new DataInputStream(socket.getInputStream());
            String line;
            while (true)
            {
                line = in.readUTF();
                System.out.println("Server: " + line);
                if (line.equals("Over"))
                {
                    System.out.println("Connection Terminated By Server...");
                    System.exit(0); // Terminate the program
                }
            }
        }
        catch (IOException e)
        {
            //e.printStackTrace();
        }
    }
}

public static void main(String args[])
{
    Client client = new Client("Localhost", 4999);
}
}
```

OUTPUT :



```
Output
JavaApplication3 (run) x JavaApplication3 (run) #2 x
run:
Connected
give me the full form of Mcn?
Server: Mordern Computer Network
what is 10+5?
Server: 15
Over
BUILD SUCCESSFUL (total time: 2 minutes 17 seconds)
```

// A Java program for a Server

```

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

public class Server
{
    private ServerSocket server = null;
    private Socket socket = null;

    public Server(int port)
    {
        try
        {
            server = new ServerSocket(port);
            System.out.println("Server started");

            System.out.println("Waiting for a client ...");

            socket = server.accept();
            System.out.println("Client accepted");

            // Start a separate thread for receiving messages from the client
            Thread receiveThread = new Thread(new ReceiveMessage());
            receiveThread.start();

            // Read messages from the terminal and send to the client
            DataOutputStream out = new DataOutputStream(socket.getOutputStream());
            BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
            String line;
            while (true)
            {
                line = reader.readLine();
                out.writeUTF(line);
                if (line.equals("Over"))
                    break; // Exit loop when "Over" is sent
            }
        }
        catch (IOException i)
        {
            System.out.println(i);
        }
        finally
        {
            try
            {
                socket.close();
            }
        }
    }
}

```

MODERN COMPUTER NETWORKS LAB EXPERIMENTS

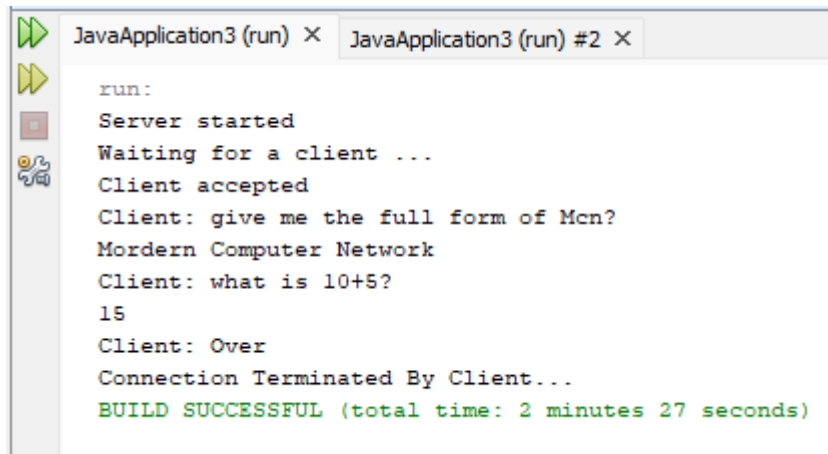
```
        server.close();
    }
    catch (IOException e)
    {
        //e.printStackTrace();
    }
}

class ReceiveMessage implements Runnable
{
    @Override
    public void run()
    {
        try
        {
            DataInputStream in = new DataInputStream(socket.getInputStream());
            String line;
            while (true)
            {
                line = in.readUTF();
                System.out.println("Client: " + line);
                if (line.equals("Over"))
                {
                    System.out.println("Connection Terminated By Client...");
                    System.exit(0); // Terminate the program
                }
            }
        }
        catch (IOException e)
        {
            //e.printStackTrace();
        }
    }
}

public static void main(String args[])
{
    Server server = new Server(4999);
}
}
```

OUTPUT :

MODERN COMPUTER NETWORKS LAB EXPERIMENTS



```
run:
Server started
Waiting for a client ...
Client accepted
Client: give me the full form of Mcn?
Mordern Computer Network
Client: what is 10+5?
15
Client: Over
Connection Terminated By Client...
BUILD SUCCESSFUL (total time: 2 minutes 27 seconds)
```

// Two machines connected

// server

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

public class Server
{
    private ServerSocket server = null;
    private Socket socket = null;

    public Server(int port)
    {
        try
        {
            server = new ServerSocket(port);
            System.out.println("Server started");

            System.out.println("Waiting for a client ...");

            socket = server.accept();
            System.out.println("Client accepted");

            // Start a separate thread for receiving messages from the client
            Thread receiveThread = new Thread(new ReceiveMessage());
            receiveThread.start();

            // Read messages from the terminal and send to the client
            DataOutputStream out = new DataOutputStream(socket.getOutputStream());
            BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
            String line;
            while (true)
```

MODERN COMPUTER NETWORKS LAB EXPERIMENTS

```
{
    line = reader.readLine();
    out.writeUTF(line);
    if (line.equals("Over"))
        break; // Exit loop when "Over" is sent
}
}
catch (IOException i)
{
    System.out.println(i);
}
finally
{
    try
    {
        socket.close();
        server.close();
    }
    catch (IOException e)
    {
        //e.printStackTrace();
    }
}
}

class ReceiveMessage implements Runnable
{
    @Override
    public void run()
    {
        try
        {
            DataInputStream in = new DataInputStream(socket.getInputStream());
            String line;
            while (true)
            {
                line = in.readUTF();
                System.out.println("Client: " + line);
                if (line.equals("Over"))
                {
                    System.out.println("Connection Terminated By Client...");
                    System.exit(0); // Terminate the program
                }
            }
        }
        catch (IOException e)
        {
            //e.printStackTrace();
        }
    }
}
```

MODERN COMPUTER NETWORKS LAB EXPERIMENTS

```
public static void main(String args[])
{
    Server server = new Server(6999);
}
}
```

// Client

// A Java program for a Client

```
import java.io.*
```

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

```
public class Client
```

```
{
```

```
    private Socket socket = null;
```

```
    private DataInputStream input = null;
```

```
    private DataOutputStream out = null;
```

```
    public Client(String address, int port)
```

```
    {
```

```
        try
```

```
        {
```

```
            socket = new Socket(address, port);
```

```
            System.out.println("Connected");
```

```
            input = new DataInputStream(System.in);
```

```
            out = new DataOutputStream(socket.getOutputStream());
```

```
            // Start a separate thread for receiving messages from the server
```

```
            Thread receiveThread = new Thread(new ReceiveMessage());
```

```
            receiveThread.start();
```

```
            // Read messages from the terminal and send to the server
```

```
            String line = "";
```

```
            while (!line.equals("Over"))
```

```
            {
```

```
                line = input.readLine();
```

```
                out.writeUTF(line);
```

```
                if (line.equals("Over"))
```

```
                    break; // Exit loop when "Over" is sent
```

```
            }
```

```
        }
```

```
    catch (UnknownHostException u)
```

```
    {
```

```
        System.out.println(u);
```

```
    }
```

```
    catch (IOException i)
```

```
    {
```

MODERN COMPUTER NETWORKS LAB EXPERIMENTS

```
        System.out.println(i);
    }
    finally
    {
        try
        {
            input.close();
            out.close();
            socket.close();
        }
        catch (IOException e)
        {
            //e.printStackTrace();
        }
    }
}

class ReceiveMessage implements Runnable
{
    @Override
    public void run()
    {
        try
        {
            DataInputStream in = new DataInputStream(socket.getInputStream());
            String line;
            while (true)
            {
                line = in.readUTF();
                System.out.println("Server: " + line);
                if (line.equals("Over"))
                {
                    System.out.println("Connection Terminated By Server...");
                    System.exit(0); // Terminate the program
                }
            }
        }
        catch (IOException e)
        {
            //e.printStackTrace();
        }
    }
}

public static void main(String args[])
{
    Client client = new Client("10.1.10.252", 6999);
}
}
```

OUTPUT :

MODERN COMPUTER NETWORKS LAB EXPERIMENTS

```
Output - JavaApplication3 (run)

run:
Server started
Waiting for a client ...
Client accepted
Client: what is your college name
agnel institute of technology and design
Client: which department
computer
over
Over
BUILD SUCCESSFUL (total time: 1 minute 26 seconds)

Output - chatApplication (run) X

run:
Connected
what is your college name
Server: agnel institute of technology and design
which department
Server: computer
Server: over
Server: Over
Connection Terminated By Server...
BUILD SUCCESSFUL (total time: 1 minute 25 seconds)
```