Practical Assignment no :1

1.Implement multi-threaded client/server Process communication using RMI in java

Server.java

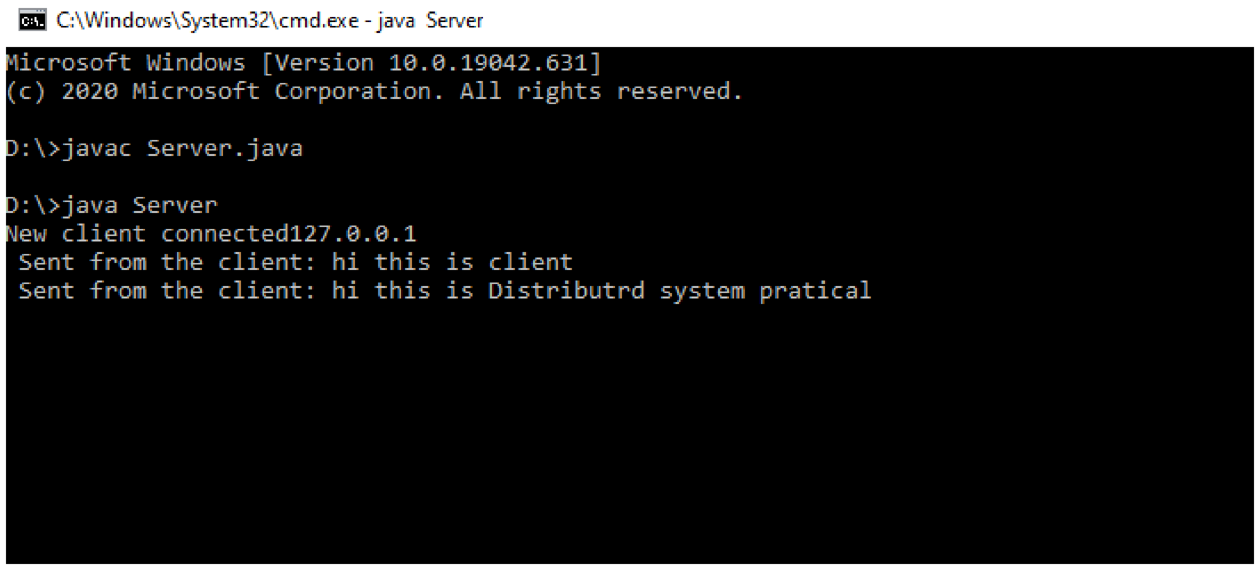
```
import java.io.*;
import java.net.*;

// Server class
class Server {
        public static void main(String[] args)
        {
                ServerSocket server = null;

                try {

                        // server is listening on port 1234
                        server = new ServerSocket(1234);
                        server.setReuseAddress(true);

                        // running infinite loop for getting
                        // client request
                        while (true) {

                                // socket object to receive incoming client
                                // requests
                                Socket client = server.accept();

                                // Displaying that new client is connected
                                // to server
                                System.out.println("New client connected"
                                                                + client.getInetAddress()
                                                                                .getHostAddress());

                                // create a new thread object
                                ClientHandler clientSock
                                        = new ClientHandler(client);

                                // This thread will handle the client
                                // separately
                                new Thread(clientSock).start();
                }
            }
            catch (IOException e) {
                e.printStackTrace();
            }
            finally {
                if (server != null) {
                    try {
                        server.close();
                    }
                    catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }
    }

    // ClientHandler class
    private static class ClientHandler implements Runnable {
            private final Socket clientSocket;

            // Constructor
            public ClientHandler(Socket socket)
            {
                this.clientSocket = socket;
            }

            public void run()
            {
                PrintWriter out = null;
                BufferedReader in = null;
                try {

                    // get the outputstream of client
                    out = new PrintWriter(
                        clientSocket.getOutputStream(), true);

                    // get the inputstream of client
                    in = new BufferedReader(
                        new InputStreamReader(
                            clientSocket.getInputStream()));

                    String line;
                    while ((line = in.readLine()) != null) {
```

```
                    // writing the received message from
                    // client
                    System.out.printf(
                        " Sent from the client: %s\n",
                        line);
                    out.println(line);
                }
            }
            catch (IOException e) {
                e.printStackTrace();
            }
            finally {
                try {
                    if (out != null) {
                        out.close();
                    }
                    if (in != null) {
                        in.close();
                        clientSocket.close();
                    }
                }
                catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }
}
```

Client.java

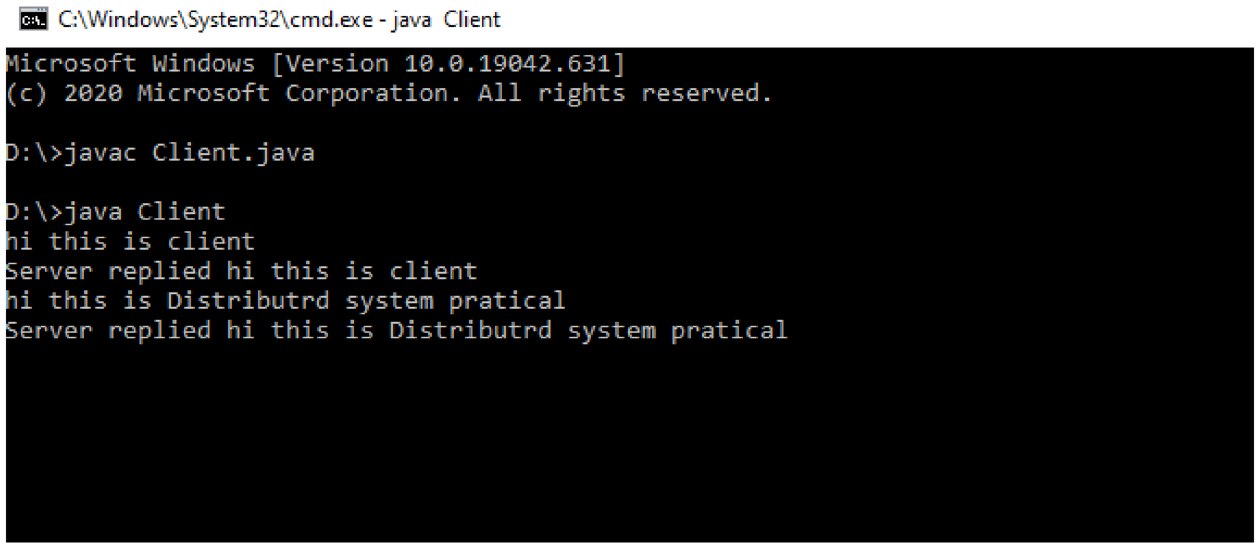
```
import java.io.*;
import java.net.*;
import java.util.*;

// Client class
class Client {

    // driver code
    public static void main(String[] args)
    {
        // establish a connection by providing host and port
        // number
        try (Socket socket = new Socket("localhost", 1234)) {

            // writing to server
            PrintWriter out = new PrintWriter(
                socket.getOutputStream(), true);

            // reading from server
            BufferedReader in
                = new BufferedReader(new InputStreamReader(
                    socket.getInputStream()));

            // object of scanner class
            Scanner sc = new Scanner(System.in);
            String line = null;

            while (!"exit".equalsIgnoreCase(line)) {

                // reading from user
                line = sc.nextLine();

                // sending the user input to server
                out.println(line);
                out.flush();

                // displaying server reply
                System.out.println("Server replied "
                                + in.readLine());
            }

            // closing the scanner object
            sc.close();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output:



C:\Windows\System32\cmd.exe - java Server

```
Microsoft Windows [Version 10.0.19042.631]
(c) 2020 Microsoft Corporation. All rights reserved.

D:\>javac Server.java

D:\>java Server
New client connected127.0.0.1
 Sent from the client: hi this is client
 Sent from the client: hi this is Distributrd system pratical
```



C:\Windows\System32\cmd.exe - java Client

```
Microsoft Windows [Version 10.0.19042.631]
(c) 2020 Microsoft Corporation. All rights reserved.

D:\>javac Client.java

D:\>java Client
hi this is client
Server replied hi this is client
hi this is Distributrd system pratical
Server replied hi this is Distributrd system pratical
```