Experiment No. 1

Title: Socket Programming

1) Server

Code:

```
package lab_01_MultipleClient;


import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Scanner;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;


public class Server {


    int port;
    ServerSocket server = null;
    Socket client = null;
```

```
        ExecutorService pool = null;

        int clientCount = 0;


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

                Server serverobj = new Server(5000);

                serverobj.startServer();

        }


        Server(int port) {

                this.port = port; // Setting up the port

                pool = Executors.newFixedThreadPool(5);

        }


        public void startServer() throws IOException {



                server = new ServerSocket(5000); // Starting up a new server

                System.out.println("Server Started");

                System.out.println("To stop the server please type -1"); // Client
    count will be -1

                while (true) {

                        client = server.accept();

                        clientCount++;
```

```
            ServerThread runnable = new ServerThread(client,
clientCount, this);
            pool.execute(runnable);
        }
    }

    private static class ServerThread implements Runnable {

        Server server = null;
        Socket client = null;
        BufferedReader in;
        PrintStream out;
        Scanner sc = new Scanner(System.in);
        int id;
        String line;

        ServerThread(Socket client, int count, Server server) throws
IOException {

            this.client = client;
            this.server = server;
            this.id = count;
            System.out.println("Connection " + id + "established with
client " + client);
```

```
            in = new BufferedReader(new
InputStreamReader(client.getInputStream()));

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



        }



        @Override

        public void run() {

            int x = 1;

            try {

                while (true) {

                    line = in.readLine();

                    System.out.print("Client(" + id + ") :" + line +
"\n");

                    System.out.print("Server : ");

                    line = sc.nextLine();

                    if (line.equalsIgnoreCase("close")) {

                        out.println("closed");

                        x = 0;

                        System.out.println("Connection ended
by server!");

                        break;

                    }
                    out.println(line);
                }

                in.close();
                client.close();
                out.close();
                if (x == 0) {
                    System.out.println("Server cleaning up.");
                    System.exit(0);
                }
            } catch (IOException ex) {
                System.out.println("Error : " + ex);
            }

        }
    }

}
```

2) Client:

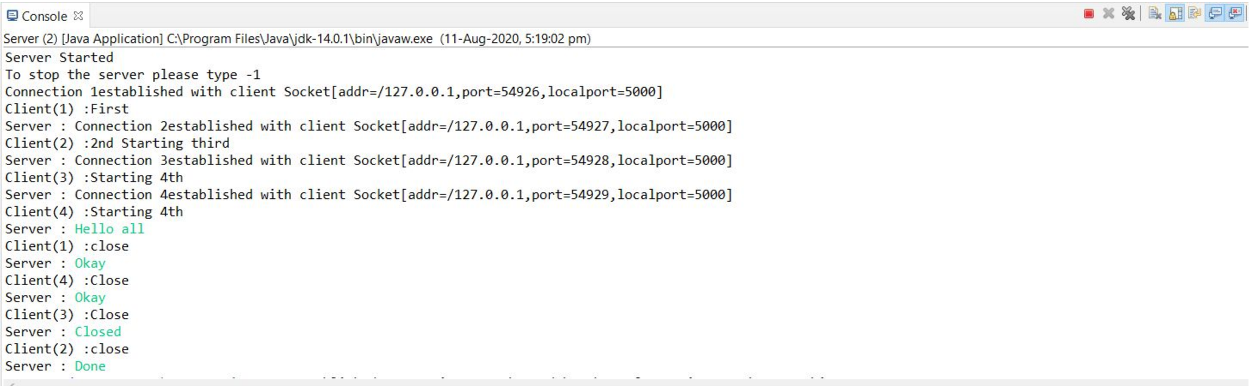
Code:

```
package lab_01_MultipleClient;
```

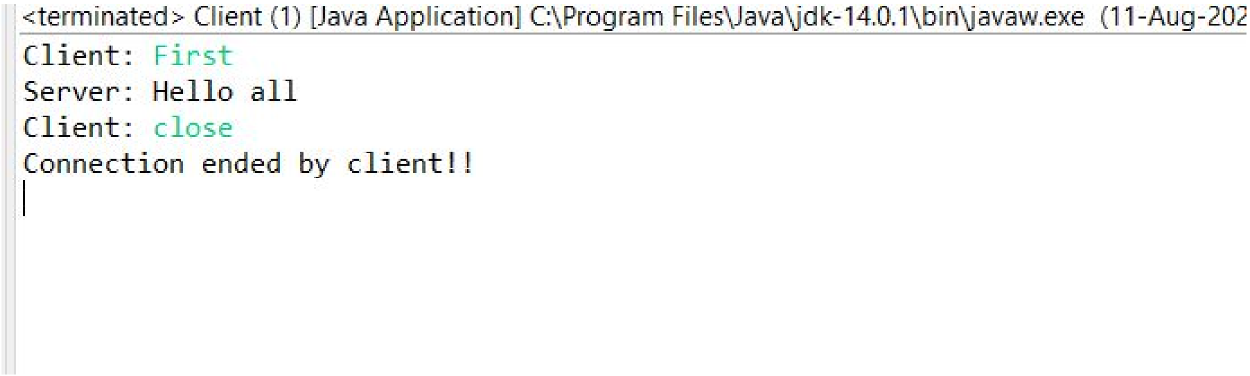
```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.net.Socket;


public class Client {


    public static void main(String args[]) throws Exception {
        Socket socket = new Socket("127.0.0.1", 5000); // New Connection
        BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream())); // Input server
        PrintStream out = new PrintStream(socket.getOutputStream()); //
Client output
        BufferedReader clientIn = new BufferedReader(new
InputStreamReader(System.in)); // Client input
        String line;
        while (true) {
            System.out.print("Client: ");
            line = clientIn.readLine();
            out.println(line);
            if (line.equalsIgnoreCase("close")) { // To close connection
                System.out.println("Connection ended by client!!");
                break;
            }
```

```
                line = in.readLine();

                System.out.print("Server: " + line + "\n");

            }

            socket.close();

            in.close();

            out.close();

            clientIn.close();

        }

    }
```

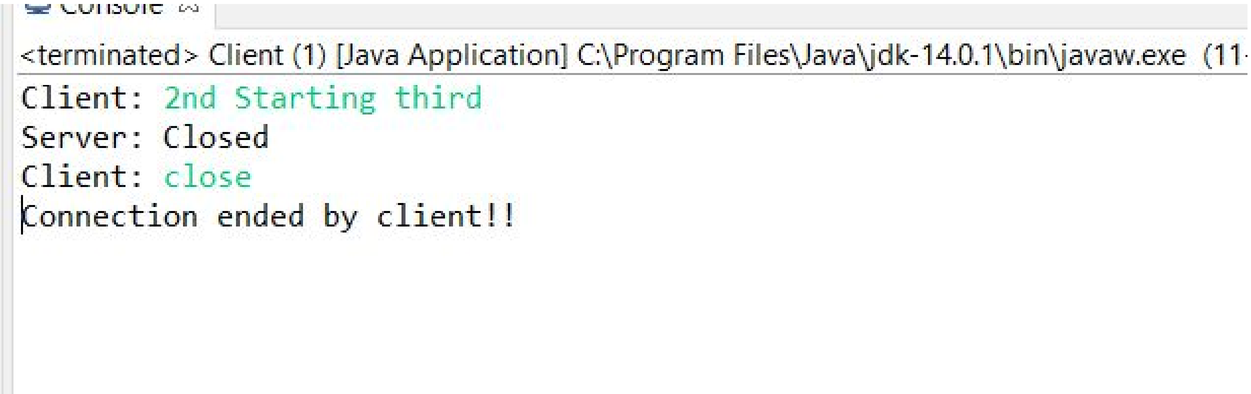
Screenshots:



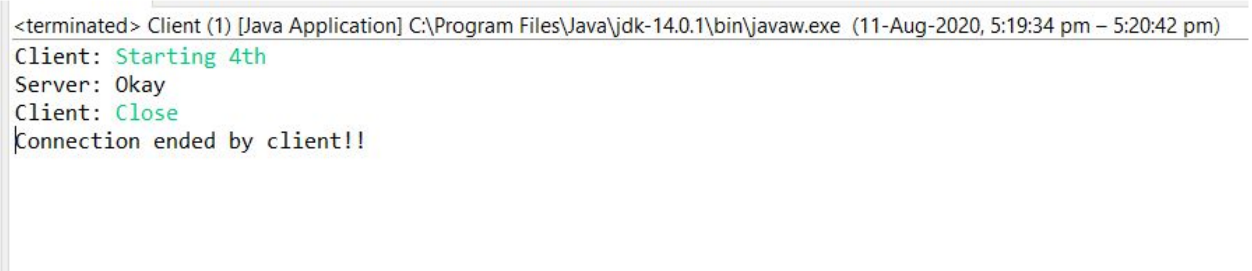
```
Server (2) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (11-Aug-2020, 5:19:02 pm)
Server Started
To stop the server please type -1
Connection 1established with client Socket[addr=/127.0.0.1,port=54926,localport=5000]
Client(1) :First
Server : Connection 2established with client Socket[addr=/127.0.0.1,port=54927,localport=5000]
Client(2) :2nd Starting third
Server : Connection 3established with client Socket[addr=/127.0.0.1,port=54928,localport=5000]
Client(3) :Starting 4th
Server : Connection 4established with client Socket[addr=/127.0.0.1,port=54929,localport=5000]
Client(4) :Starting 4th
Server : Hello all
Client(1) :close
Server : Okay
Client(4) :Close
Server : Okay
Client(3) :Close
Server : Closed
Client(2) :close
Server : Done
```



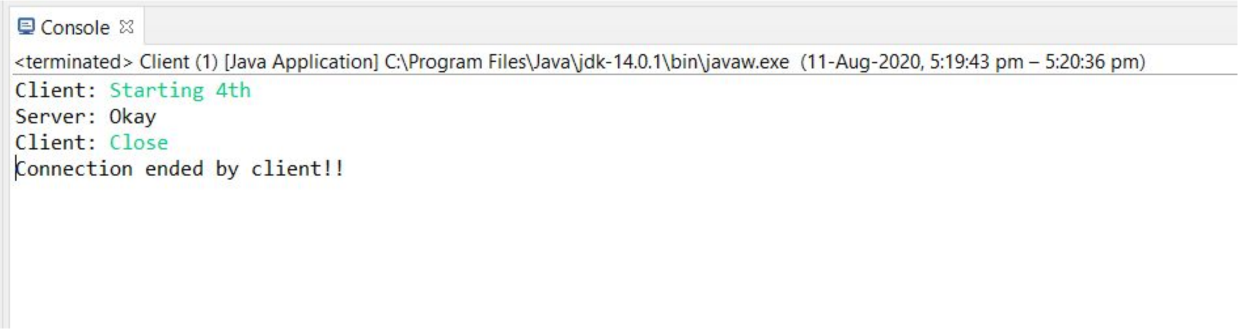
```
<terminated> Client (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (11-Aug-202
Client: First
Server: Hello all
Client: close
Connection ended by client!!
```



```
<terminated> Client (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (11-
Client: 2nd Starting third
Server: Closed
Client: close
Connection ended by client!!
```



```
<terminated> Client (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (11-Aug-2020, 5:19:34 pm – 5:20:42 pm)
Client: Starting 4th
Server: Okay
Client: Close
Connection ended by client!!
```



```
<terminated> Client (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (11-Aug-2020, 5:19:43 pm – 5:20:36 pm)
Client: Starting 4th
Server: Okay
Client: Close
Connection ended by client!!
```