NAME : ASAD ULLAH KHAN

STUDENT ID : 61348

COURSE : NETWORK PROGRAMMING

SUBMITTED TO : Miss MISBAH ANWAR

Contents

OOP

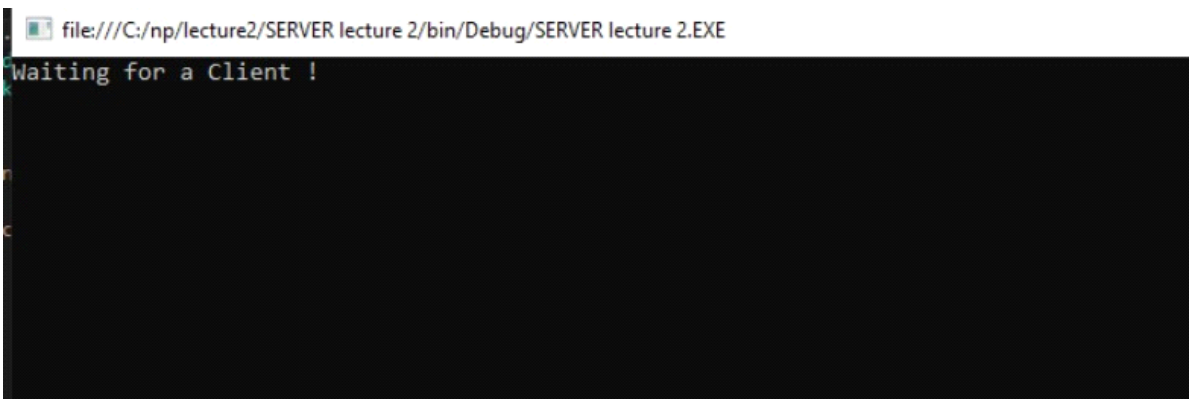
```
file:///C:/np/lecture2/lecture2/bin/Debug/lecture2.EXE
I am an animal
My name is Rohu
dog can eat
dog can sound
```

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace lecture2
{
    class Animal
    {
        public string name;

        public void display()
        {
            Console.WriteLine("I am an animal");
        }
    }
    class Dog : Animal
    {

        public void getName()
        {
            Console.WriteLine("My name is " + name);
        }
    }
    abstract class animalSound
    {
```

```
        public abstract void eat();
        public void sound()
        {
            Console.WriteLine("dog can sound");
        }
    }
    class dog : animalSound
    {
        public override void eat()
        {
            Console.WriteLine("dog can eat");
        }
    }
}
```

Create a first server



Code :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;

namespace SERVER_lecture_2
{
    class Program
    {
        static void Main(string[] args)
        {
            IPAddress ip = IPAddress.Loopback;
            IPEndPoint ep = new IPEndPoint(ip, 8000);
            Socket _socket = new Socket( AddressFamily.InterNetwork , SocketType.Stream ,
ProtocolType.Tcp );
            _socket.Bind(ep);
            _socket.Listen(10);
```

```
                Console.WriteLine("Waiting for a Client !");
                _socket.Accept();
                Console.WriteLine("Connected");
                Console.ReadLine();
            }
        }
    }
```

Lecture 2

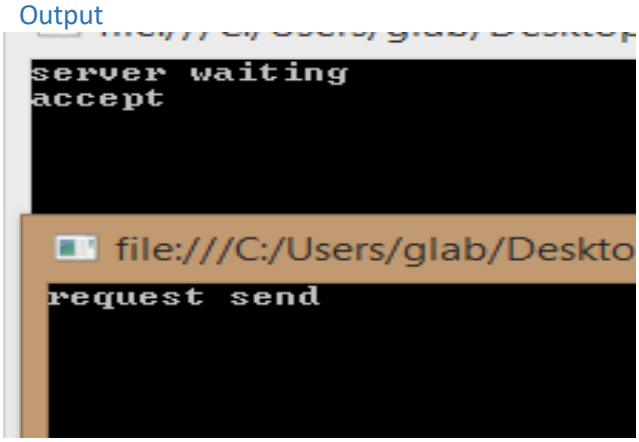
Client

```
namespace ConsoleApplication7
{
    class Program
    {
        static void Main(string[] args)
        {
            IPAddress ep = IPAddress.Loopback;
            IPEndPoint ed = new IPEndPoint(ep, 2000);
            Socket sr = new Socket(ep.AddressFamily, SocketType.Stream, ProtocolType.Tcp);
            sr.Bind(ed);
            sr.Listen(1);
            Console.WriteLine("server waiting");
            sr.Accept();
            Console.WriteLine("accept");
            Console.ReadLine();

        }
    }
```

Server

```
class Program
{
    static void Main(string[] args)
    {
        IPAddress ep = IPAddress.Loopback;
        IPEndPoint ed = new IPEndPoint(ep, 2000);
        Socket sr = new Socket(ep.AddressFamily, SocketType.Stream, ProtocolType.Tcp);
        sr.Bind(ed);
        sr.Listen(1);
        Console.WriteLine("server waiting");
       // sr.Accept();
        Console.WriteLine("accept");
        Socket cl = sr.Accept();
        byte[] arr = new byte[100];
        cl.Receive(arr);
        Console.WriteLine(Encoding.ASCII.GetString(arr));


        string str = Console.ReadLine();
        cl.Send(Encoding.ASCII.GetBytes(str));
        Console.ReadLine();

    }
}
```

Output



```
class Program
{
    static void Main(string[] args)
    {
        IPAddress ep = IPAddress.Loopback;
        IPEndPoint ed = new IPEndPoint(ep, 2000);
        Socket sr = new Socket(ep.AddressFamily, SocketType.Stream, ProtocolType.Tcp);
        sr.Connect(ed);
        Console.WriteLine("request send");
        string str="Qiuz 1 Today";
        sr.Send(Encoding.ASCII.GetBytes(str));

        byte[] arr = new byte[100];
        sr.Receive(arr);
        Console.WriteLine(Encoding.ASCII.GetString(arr));
        Console.ReadLine();
    }
}
```

```
namespace ConsoleApplication8
{
    class Program
    {
        static void Main(string[] args)
        {
            IPAddress ep = IPAddress.Loopback;
            IPEndPoint ed = new IPEndPoint(ep, 2000);
            Socket sr = new Socket(ep.AddressFamily, SocketType.Stream, ProtocolType.Tcp);
            sr.Connect(ed);
            Console.WriteLine("request send");
            Console.ReadLine();
        }
    }
```

Le

ctu

re

3

Cli

en

t

```
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        TcpClient client = new TcpClient();
        private void button1_Click(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
            IPEndPoint point = new IPEndPoint(IPAddress.Loopback, 8002);
            client = new TcpClient(point);
            client.Connect(IPAddress.Loopback, 8001);
            Thread t = new Thread(ReadMessage);
            t.Start();


        }
        public void ReadMessage()
        {
            while (true)
            {
                NetworkStream stream = client.GetStream();
                StreamReader sdr = new StreamReader(stream);
                string msg = sdr.ReadLine();
                textBox2.AppendText(Environment.NewLine);
                textBox2.AppendText("Server: " + msg);
            }
        }

        private void button2_Click(object sender, EventArgs e)
```

```
            textBox2.AppendText("Server: " + msg);
        }
    }
```

```
    private void button2_Click(object sender, EventArgs e)
    {
        NetworkStream stream = client.GetStream();
        StreamWriter sdr = new StreamWriter(stream);
        sdr.WriteLine(textBox3.Text);
        sdr.Flush();
        textBox1.AppendText(Environment.NewLine);
        textBox1.AppendText("Me: " + textBox3.Text);

    }
}
```

Server

```
}
public void AcceptClient()
{
    while (true)
    {
        TcpClient c = server.AcceptTcpClient();
        clients.Add(c);
        Thread t = new Thread(abc => ReadMessage(c));
        //lambda expression =>
        t.Start();
    }
}

public void ReadMessage(TcpClient client)
{
    while (true)
    {
        NetworkStream stream = client.GetStream();
        StreamReader sdr = new StreamReader(stream);
        string msg = sdr.ReadLine();
        textBox2.AppendText(Environment.NewLine);
        textBox2.AppendText("Client:  " + msg);
    }
}
```

```
            StreamReader sdr = new StreamReader(stream);
            string msg = sdr.ReadLine();
            textBox2.AppendText(Environment.NewLine);
            textBox2.AppendText("Client:  " + msg);
        }
    }

    private void button2_Click_1(object sender, EventArgs e)
    {
        foreach (var item in clients)
        {
            textBox1.AppendText(Environment.NewLine);
            textBox1.AppendText("Me: " + textBox3.Text);
            NetworkStream stream = item.GetStream();
            StreamWriter sdr = new StreamWriter(stream);
            sdr.WriteLine(textBox3.Text);
            sdr.Flush();
        }
    }
}
```

LECTURE 4

3.3

file:///c:/users/ashar khan/documents/visual studio 2015/Projects/Server/Server/bin/Debu

```
AddressFamily: InterNetwork
SocketType: Stream
ProtocolType: Tcp
Blocking: True
new Blocking: False
Connected: False
Local EndPoint: 127.0.0.1:8000
```

Code:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {
        static void Main(string[] args)
        {
```

test.Close();

```
IPAddress ia = IPAddress.Parse("127.0.0.1");IPEndPoint ie =
new IPEndPoint(ia, 8000);
Socket test = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
Console.WriteLine("AddressFamily: {0}", test.AddressFamily);
Console.WriteLine("SocketType: {0}",
test.SocketType);
Console.WriteLine("ProtocolType: {0}",
test.ProtocolType);
Console.WriteLine("Blocking: {0}", test.Blocking);
test.Blocking = false;
Console.WriteLine("new Blocking: {0}", test.Blocking);
Console.WriteLine("Connected: {0}", test.Connected); test.Bind(ie);
IPEndPoint iep = (IPEndPoint)test.LocalEndPoint;
Console.WriteLine("Local EndPoint: {0}", iep.ToString());


            Console.ReadLine();
        }
    }
}
```

Ex 3.4

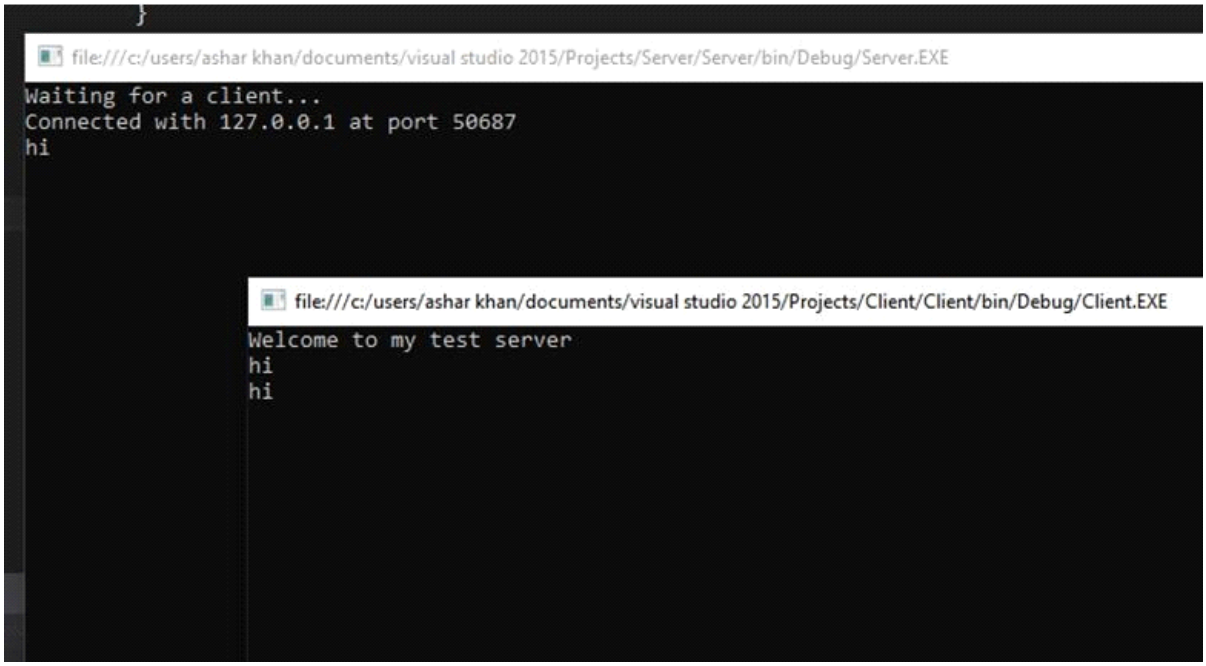
```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {
        static void Main(string[] args)
        {
            IPAddress host =
            IPAddress.Parse("192.168.1.1"); IPEndPoint
            hostep = new IPEndPoint(host, 8000); Socket
            sock = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                sock.Connect(hostep);
            }
            catch (SocketException e)
```

```
                {




                }
                try
                {

                }

Console.WriteLine("Problem connecting to host");Console.WriteLine(e.ToString());
sock.Close();return;



sock.Send(Encoding.ASCII.GetBytes("testing"));

                catch (SocketException e)
                {
                    Console.WriteLine("Problem sending
                    data");
                    Console.WriteLine(e.ToString());
                    sock.Close();
                    return;
                }
                sock.Close();

                Console.ReadL

                ine();

        }
    }
            }
```

Lecture 5

TCP

Ex 5.1 – 5.2



Server

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {
        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[1024];
            IPEndPoint ipep = new
            IPEndPoint(IPAddress.Any,9050);
            Socket newsock = new
            Socket(AddressFamily.InterNetwork,

            SocketType.Stream,
            ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a
            client...");Socket client =
            newsock.Accept();
```

```
            IPEndPoint clientep =
            (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port
            {1}",clientep.Address, clientep.Port);

            string welcome = "Welcome to my test
            server";data =
            Encoding.ASCII.GetBytes(welcome);
            client.Send(data, data.Length,
            SocketFlags.None);
            while (true)
            {
                data = new byte[1024];
                recv =
                client.Receive(data);
                if (recv == 0)
                    break;

                Console.WriteLine(
                Encoding.ASCII.GetString(data, 0,
                recv));client.Send(data, recv,
                SocketFlags.None);
            }
            Console.WriteLine("Disconnected from
            {0}",clientep.Address);
            client.Clo
            se();
            newsock.Cl
            ose();

            Console.ReadLine();
        }
    }
}
```

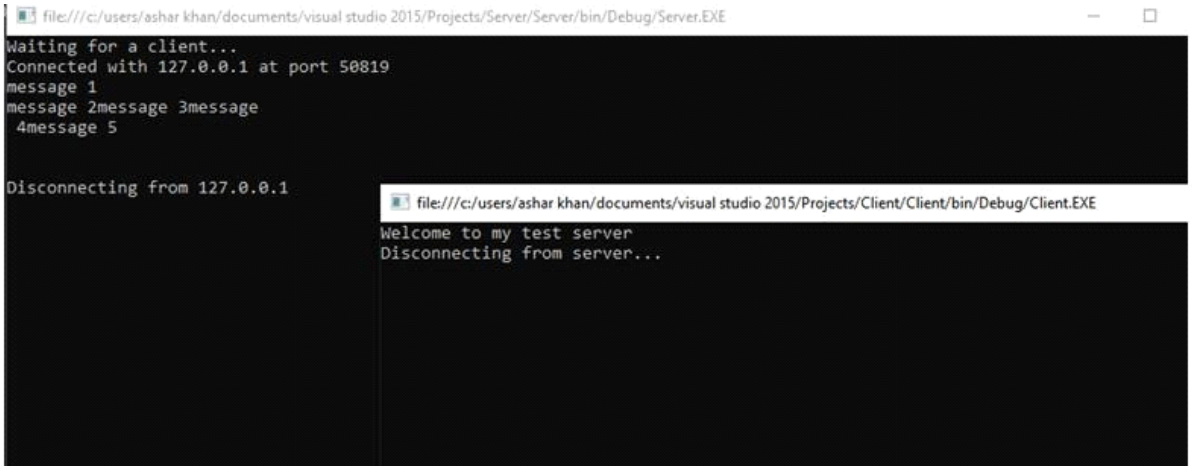
Client

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string  input, stringData;
            IPEndPoint ipep = new
```

```
            IPEndPoint(
            IPAddress.Parse("127.0.0.1"),
            9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }

            int recv = server.Receive(data);
            stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);
            while (true)
            {
                input =
                Console.ReadLine();if
                (input == "exit")
                    break;
                server.Send(Encoding.ASCII.GetBytes(input
                ));data = new byte[1024];
                recv = server.Receive(data);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Disconnecting from
            server...");
            server.Shutdown(SocketShutdown.Both);
            server.Close();
            Console.ReadLine();
        }
    }
}
```

Ex 5.3 – 5.4

TCP goes Bad



Server:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {
        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
            9050);Socket newsock = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a
            client...");Socket client =
            newsock.Accept();

            string welcome = "Welcome to my test
            server";data =
            Encoding.ASCII.GetBytes(welcome);
            client.Send(data, data.Length,
            SocketFlags.None);
            IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port {1}",
            newclient.Address, newclient.Port);
            for (int i = 0; i < 5; i++)
```

```
            {
                recv = client.Receive(data);
                Console.WriteLine(Encoding.ASCII.GetString(data, 0,
                recv));
            }
 Console.WriteLine("Disconnecting from {0}", newclient.Address);
            client.Close();
            newsock.Close();

            Console.ReadLine();
        }
    }
}
```

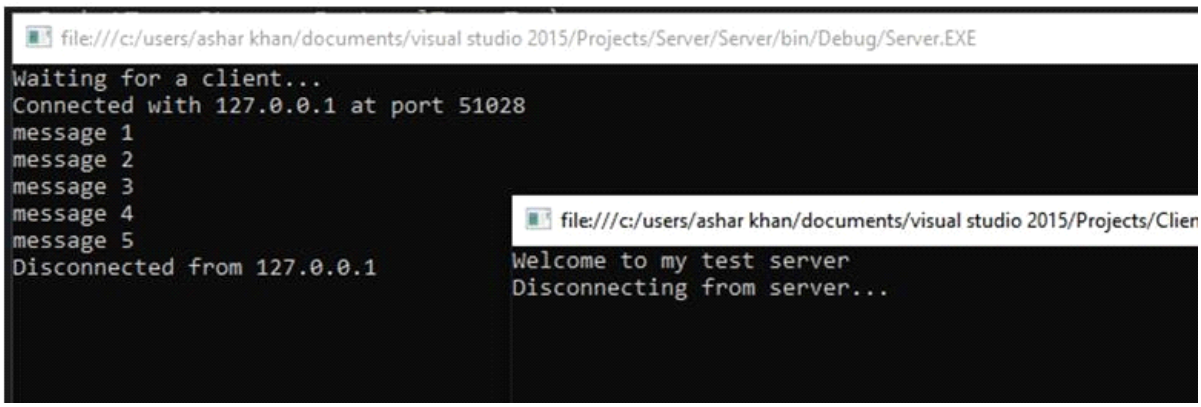
Client:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new
            byte[1024];string
            stringData;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            int recv = server.Receive(data);
            stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);
            server.Send(Encoding.ASCII.GetBytes("message 1"));
            server.Send(Encoding.ASCII.GetBytes("message 2"));
            server.Send(Encoding.ASCII.GetBytes("message 3"));
```

```
            server.Send(Encoding.ASCII.GetBytes("message 4"));
            server.Send(Encoding.ASCII.GetBytes("message
            5"));Console.WriteLine("Disconnecting from
            server...");
            server.Shutdown(SocketShutdown.Both);
            server.Close();
            Console.ReadLine();
        }

    }
}
```

Ex 5.5 – 5.6

Fixed Size Message



Server:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {
        private static int SendData(Socket s, byte[] data)
        {
            int total = 0;
            int size =
            data.Length;int
            dataleft = size;
            int sent;
            while (total < size)
            {
```

```
        sent = s.Send(data, total, dataleft,
        SocketFlags.None);total += sent;
        dataleft -= sent;
    }
    return total;
}
private static byte[] ReceiveData(Socket s, int size)
{
    int total = 0;
    int dataleft = size;
    byte[] data = new
    byte[size];int recv;
    while (total < size)
    {
        recv = s.Receive(data, total,
        dataleft, 0);if (recv == 0)
        {
            data =
            Encoding.ASCII.GetBytes("exit");
            break;
        }
        total +=
        recv;
        dataleft -=
        recv;

    }
    return data;
}

static void Main(string[] args)
{
    byte[] data = new byte[1024];
    IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
    9050);Socket newsock = new
    Socket(AddressFamily.InterNetwork,
    SocketType.Stream, ProtocolType.Tcp);
    newsock.Bind(ipep);
    newsock.Listen(10);
    Console.WriteLine("Waiting for a
    client...");Socket client =
    newsock.Accept();
    IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
    Console.WriteLine("Connected with {0} at port {1}",
    newclient.Address, newclient.Port);
    string welcome = "Welcome to my test
    server";data =
    Encoding.ASCII.GetBytes(welcome);
    int sent = SendData(client,
    data);for (int i = 0; i < 5;
    i++)
    {
        data = ReceiveData(client, 9);
        Console.WriteLine(Encoding.ASCII.GetString(data));
    }
    Console.WriteLine("Disconnected from {0}",
    newclient.Address);client.Close();
```

```
            newsock.Close();


            Console.ReadLine();
        }
    }
}
```

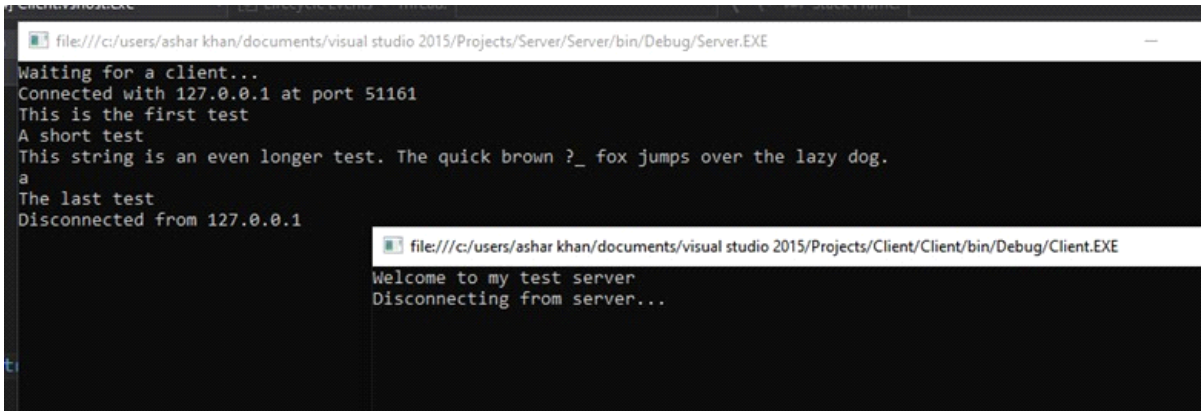
Client:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Client
{
    class Program
    {
        private static int SendData(Socket s, byte[] data)
        {
            int total = 0;
            int size =
            data.Length;int
            dataleft = size;
            int sent;
            while (total < size)
            {
                sent = s.Send(data, total, dataleft,
                SocketFlags.None);total += sent;
                dataleft -= sent;
            }
            return total;
        }
        private static byte[] ReceiveData(Socket s, int size)
        {

            int total = 0;
            int dataleft = size;
            byte[] data = new
            byte[size];int recv;
            while (total < size)
            {
recv = s.Receive(data, total, dataleft, 0);



                if (recv == 0)
                {
                    data = Encoding.ASCII.GetBytes("exit
                    ");break;
```

```
            }
            total +=
            recv;
            dataleft -=
            recv;
        }
        return data;
    }

    static void Main(string[] args)
    {
        byte[] data = new
        byte[1024];int sent;
        IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"),
        9050);Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Stream, ProtocolType.Tcp);
        try
        {
            server.Connect(ipep);
        }
        catch (SocketException e)
        {
            Console.WriteLine("Unable to connect to server.");
            Console.WriteLine(e.ToString());
            return;
        }
        int recv = server.Receive(data);
        string stringData = Encoding.ASCII.GetString(data, 0,
        recv);Console.WriteLine(stringData);
        sent = SendData(server, Encoding.ASCII.GetBytes("message
        1"));sent = SendData(server,
        Encoding.ASCII.GetBytes("message 2"));sent =
        SendData(server, Encoding.ASCII.GetBytes("message 3"));
        sent = SendData(server, Encoding.ASCII.GetBytes("message
        4"));sent = SendData(server,
        Encoding.ASCII.GetBytes("message 5"));
        Console.WriteLine("Disconnecting from server...");
        server.Shutdown(SocketShutdown.Both);
        server.Close();

        Console.ReadLine();
    }
  }
}
```

Ex 5.7 – 5.8

Fixed Variable Message



Server:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {
        private static int SendVarData(Socket s, byte[] data)
        {
            int total = 0;
            int size =
            data.Length;int
            dataleft = size;
            int sent;
            byte[] datasize = new byte[4];
            datasize =
            BitConverter.GetBytes(size);sent
            = s.Send(datasize);
            while (total < size)
            {
                sent = s.Send(data, total, dataleft,
                SocketFlags.None);total += sent;
                dataleft -= sent;
            }
            return total;
        }
        private static byte[] ReceiveVarData(Socket s)
        {
            int
            total =
            0;int
            recv;
            byte[] datasize = new byte[4];
```

```
            recv = s.Receive(datasize, 0, 4, 0);
            int size = BitConverter.ToInt32(datasize,
            0);int dataleft = size;
            byte[] data = new
            byte[size];while (total
            < size)
            {
                recv = s.Receive(data, total,
                dataleft, 0);if (recv == 0)
                {
                    data = Encoding.ASCII.GetBytes("exit
                    ");break;
                }
                total += recv;

                 dataleft -= recv;
            }
            return data;
        }

        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
            9050);Socket newsock = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a
            client...");Socket client =
            newsock.Accept();
            IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port {1}",
            newclient.Address, newclient.Port);
            string welcome = "Welcome to my test
            server";data =
            Encoding.ASCII.GetBytes(welcome);
            int sent = SendVarData(client,
            data);for (int i = 0; i < 5;
            i++)
            {
                data = ReceiveVarData(client);
                Console.WriteLine(Encoding.ASCII.GetString(data));
            }
            Console.WriteLine("Disconnected from {0}",
            newclient.Address);client.Close();
            newsock.Close();


            Console.ReadLine();
        }
    }
}
```

Client:

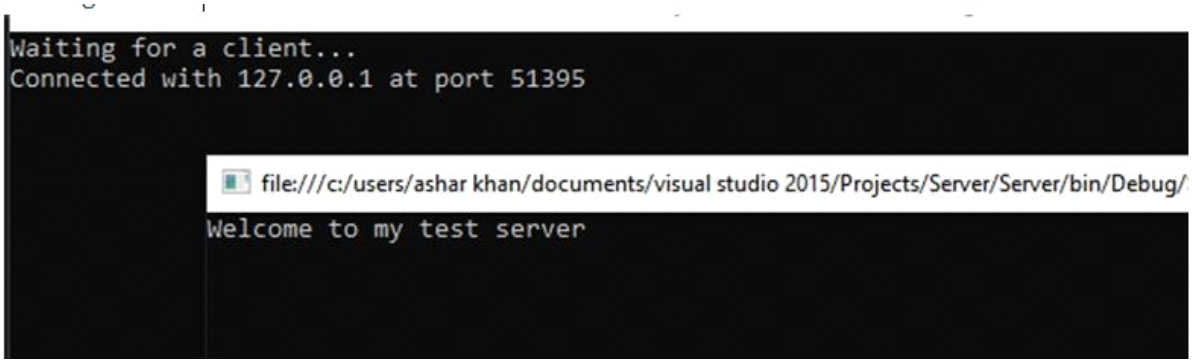
```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using

System.Threading.Tasks;

namespace Client




{
    class Program
    {
        private static int SendVarData(Socket s, byte[] data)
        {
            int total = 0;
            int size =
            data.Length;int
            dataleft = size;
            int sent;
            byte[] datasize = new byte[4];
            datasize =
            BitConverter.GetBytes(size);sent
            = s.Send(datasize);
            while (total < size)

        {
                sent = s.Send(data, total, dataleft,
                SocketFlags.None);total += sent;
                dataleft -= sent;


            }
            return total;
        }
        private static byte[] ReceiveVarData(Socket s)
        {
            int
            total =
            0;int
            recv;
            byte[] datasize = new byte[4];
            recv = s.Receive(datasize, 0, 4, 0);
            int size = BitConverter.ToInt32(datasize,
            0);int dataleft = size;
            byte[] data = new
            byte[size];while (total
            < size)
            {
```

```
                recv = s.Receive(data, total,
                dataleft, 0);if (recv == 0)
                {
                    data = Encoding.ASCII.GetBytes("exit
                    ");break;
                }
                total +=
                recv;
                dataleft -=
                recv;
            }
            return data;
        }

        static void Main(string[] args)
        {
            byte[] data = new
            byte[1024];int sent;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"),
            9050);Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            data = ReceiveVarData(server);
            string stringData =
            Encoding.ASCII.GetString(data);
            Console.WriteLine(stringData);
            string message1 = "This is the first
            test";string message2 = "A short
            test";
            string message3 = "This string is an even longer test. The quick brown Â_
foxjumps over the lazy dog.";
  string message4 = "a";
            string message5 = "The last test";
            sent = SendVarData(server,
            Encoding.ASCII.GetBytes(message1));sent =
            SendVarData(server, Encoding.ASCII.GetBytes(message2));
            sent = SendVarData(server,
            Encoding.ASCII.GetBytes(message3));sent =
            SendVarData(server, Encoding.ASCII.GetBytes(message4));
            sent = SendVarData(server,
            Encoding.ASCII.GetBytes(message5));
            Console.WriteLine("Disconnecting from server...");
            server.Shutdown(SocketShutdown.Both);
            server.Close();


            Console.ReadLine();
        }
```

```
    }
}
```

EX 5.9 – 5.10

Message Maker:



Server:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            string data;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
            9050);Socket newsock = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a
            client...");Socket client =
            newsock.Accept();
            IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port {1}",
            newclient.Address, newclient.Port);
            NetworkStream ns = new
            NetworkStream(client);StreamReader sr =
            new StreamReader(ns); StreamWriter sw =
```

```
            new StreamWriter(ns); string welcome =
            "Welcome to my test server";
            sw.WriteLine(welcome);
            sw.Flu
            sh();
            while
            (true)
            {

                try
{ data = sr.ReadLine();
```

Client:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {

        static void Main(string[] args)
        {

            byte[] data = new
            byte[1024];string
            input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            NetworkStream ns = new
            NetworkStream(server);if (ns.CanRead)
```

Client:

```
}}}};

using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {

        static void Main(string[] args)
        {

            byte[] data = new
            byte[1024];string
            input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);

            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            NetworkStream ns = new
            NetworkStream(server);if (ns.CanRead)
            {
```

Client:

```
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Net;
```

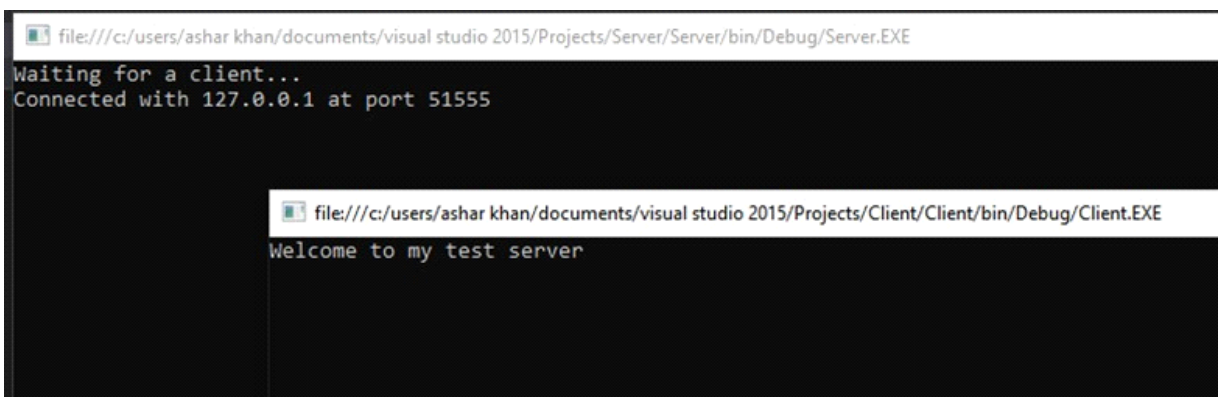
```
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {

        static void Main(string[] args)
        {

            byte[] data = new
            byte[1024];string
            input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            NetworkStream ns = new
            NetworkStream(server);if (ns.CanRead)
            {




            }
            else
            {




            }
recv = ns.Read(data, 0, data.Length);
stringData = Encoding.ASCII.GetString(data, 0, recv);Console.WriteLine(stringData);



Console.WriteLine("Error: Can't read from this socket");ns.Close();
server.Close();return;

            while (true)
            {
```

```
                input =
                Console.ReadLine();if
                (input == "exit")
                    break;
                if (ns.CanWrite)
                {
                ns.Write(Encoding.ASCII.GetBytes(input), 0,
                    input.Length);ns.Flush();

                }
                recv = ns.Read(data, 0, data.Length);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Disconnecting from
            server...");ns.Close();
            server.Shutdown(SocketShutdown.Both);
            server.Close();

            Console.ReadLine();
        }
    }
}
```

Lecture 6

Ex 5.10 – 5.11

TCP Helper Class



Server:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
```

```
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {

        static void Main(string[] args)
        {

            string data;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
            9050);Socket newsock = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);

            newsock.Bind(
            ipep);
            newsock.Liste
            n(10);
            Console.WriteLine("Waiting for a
            client...");Socket client =
            newsock.Accept();
            IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port {1}",
            newclient.Address, newclient.Port);
            NetworkStream ns = new NetworkStream(client);

            StreamReader sr = new StreamReader(ns);
            StreamWriter sw = new StreamWriter(ns);
            string welcome = "Welcome to my test
            server";sw.WriteLine(welcome);
            sw.Flu
            sh();
            while
            (true)
            {

                try
                {

                }



data = sr.ReadLine();


                catch (IOException)
                {
                    break;
                }
```

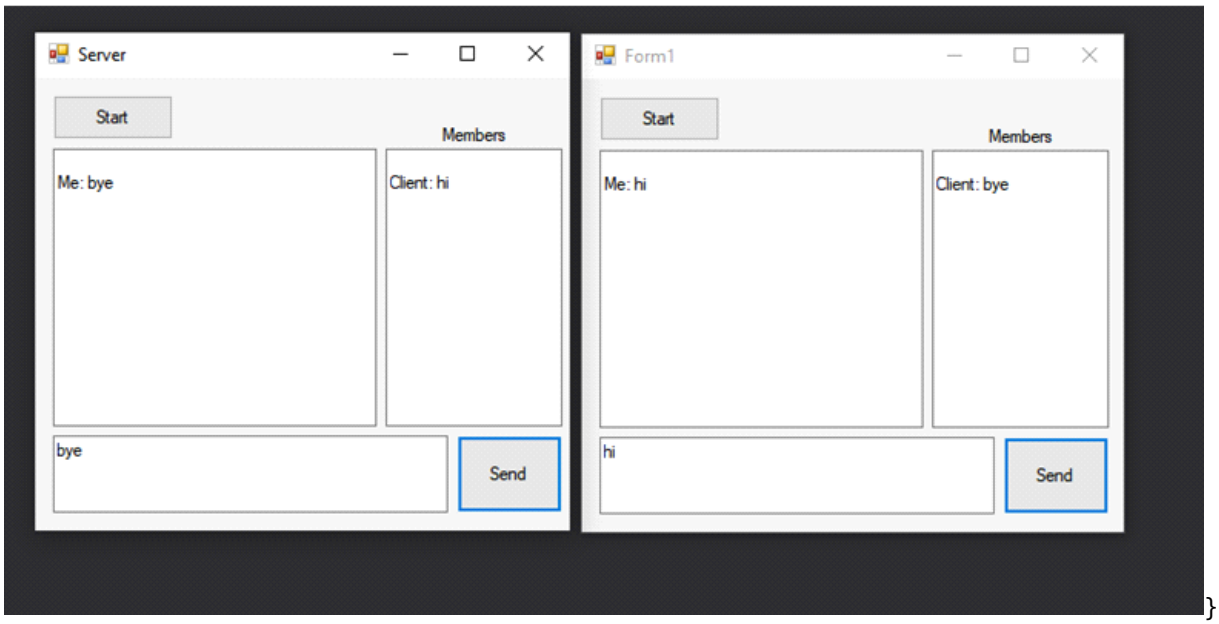
```
                Console.WriteLine(data
                );sw.WriteLine(data);
                sw.Flush();
            }
            Console.WriteLine("Disconnected from {0}",
            newclient.Address);sw.Close();
            sr.Close();
            ns.Close();

            Console.ReadLine();
        }
    }
}
```

Client:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            string
            data;
            string
            input;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);

            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
```

```
            NetworkStream ns = new
            NetworkStream(server);StreamReader sr =
            new StreamReader(ns); StreamWriter sw =
            new StreamWriter(ns);
            data = sr.ReadLine();
            Console.WriteLine(dat
            a);while (true)
            {
                input =
                Console.ReadLine();if
                (input == "exit")
                    break;
                sw.WriteLine(input);
                sw.Flush();
                data = sr.ReadLine();
                Console.WriteLine(data);
            }
            Console.WriteLine("Disconnecting from
            server...");sr.Close();
            sw.Close();
            ns.Close();
            server.Shutdown(SocketShutdown.B
            oth);server.Close();
            Console.ReadLine();
        }
    }
```

Chat Application using helper class



}

Server:

```
using System;
 using
 System.Collections.Generic;
 using
```

```
System.ComponentModel;
using System.Data;
using
System.Drawing;
using
System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using
System.Threading; using
System.Threading.Tasks;
using
System.Windows.Forms;

namespace ChatApp_Servers_
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        List<TcpClient> clients = new
        List<TcpClient>();TcpListener server;
        private void button2_Click(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
            server = new TcpListener(IPAddress.Loopback,
            8001);server.Start(10);
            Thread t2 = new
            Thread(AcceptClient);
            t2.Start();
        }
        public void AcceptClient()
        {
            while (true)
            {
                TcpClient c =
                server.AcceptTcpClient();
                clients.Add(c);
                Thread t = new Thread(asd =>
                ReadMessage(c));t.Start();
            }
        }

        public void ReadMessage(TcpClient client)
        {
            while (true)
            {

                NetworkStream stream =
                client.GetStream(); StreamReader sdr = new
                StreamReader(stream);string msg =
```

```
                sdr.ReadLine();
                textBox2.AppendText(Environment.NewLine);
                textBox2.AppendText("Client: " + msg);

            }

        }
        ///send message
        private void button1_Click(object sender, EventArgs e)
        {
            foreach (var item in clients)
            {
                textBox1.AppendText(Environment.NewLine);
                textBox1.AppendText("Me: " +
                textBox3.Text);NetworkStream stream =
                item.GetStream();

                StreamWriter sdr = new StreamWriter(stream);
                sdr.WriteLine(textBox3.Text);
                sdr.Flush();
            }

        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void textBox3_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
```

Client:

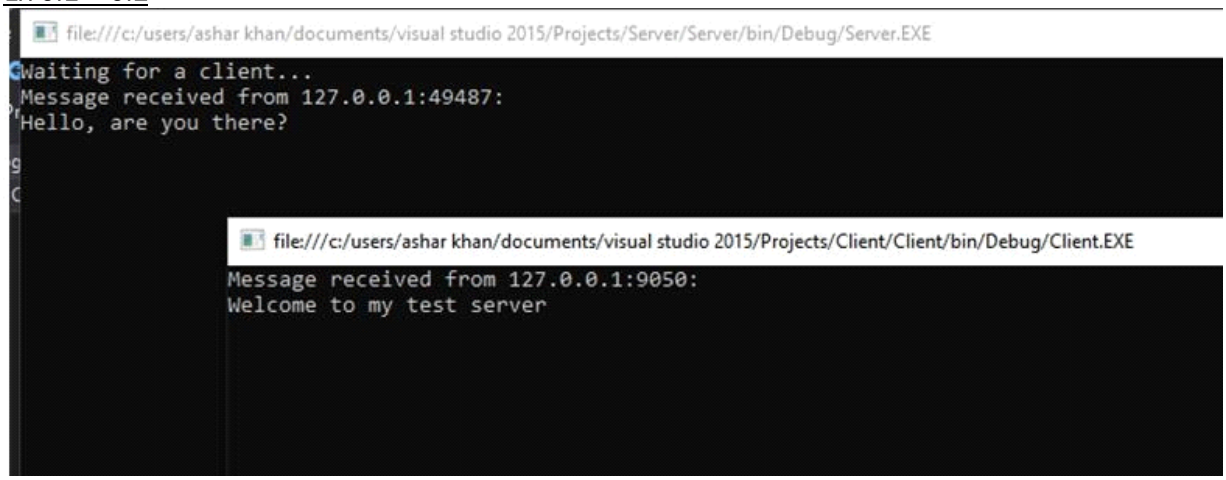
```
using System;
using
System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using
System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
```

```
using
System.Threading; using
System.Threading.Tasks;
using
System.Windows.Forms;

namespace ChatApp_Clients_
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        TcpClient client = new TcpClient();
        private void button2_Click(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;

            IPEndPoint point = new IPEndPoint(IPAddress.Loopback, 8002);

            client = new TcpClient(point);
            client.Connect(IPAddress.Loopback,
            8001);Thread t = new
            Thread(ReadMessage); t.Start();
        }
        public void ReadMessage()
        {
            while (true)
            {
                NetworkStream stream =
                client.GetStream(); StreamReader sdr = new
                StreamReader(stream);string msg =
                sdr.ReadLine();
                textBox2.AppendText(Environment.NewLine);
                textBox2.AppendText("Client: " + msg);
            }

        }

        private void button1_Click(object sender, EventArgs e)
        {
            NetworkStream stream =
            client.GetStream(); StreamWriter sdr = new
            StreamWriter(stream);
            sdr.WriteLine(textBox3.Text);
            sdr.Flush();
            textBox1.AppendText(Environment.NewLine);
            textBox1.AppendText("Me: " +
            textBox3.Text);
        }

    }
}
```

Lecture 7

UDP

Ex 6.1 – 6.2



Server:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
    {

        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
            9050);Socket newsock = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
            newsock.Bind(ipep);
            Console.WriteLine("Waiting for a client...");
            IPEndPoint sender = new IPEndPoint(IPAddress.Any,
            0);EndPoint Remote = (EndPoint)(sender);
            recv = newsock.ReceiveFrom(data, ref Remote);

            Console.WriteLine("Message received from {0}:", Remote.ToString());
            Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
```

```
            string welcome = "Welcome to my test
            server";data =
            Encoding.ASCII.GetBytes(welcome);
            newsock.SendTo(data, data.Length, SocketFlags.None,
            Remote);while (true)
            {
                data = new byte[1024];
                recv = newsock.ReceiveFrom(data, ref Remote);

                Console.WriteLine(Encoding.ASCII.GetString(data, 0,
                recv));newsock.SendTo(data, recv, SocketFlags.None,
                Remote);


                Console.ReadLine();
            }
        }
    }
}
```

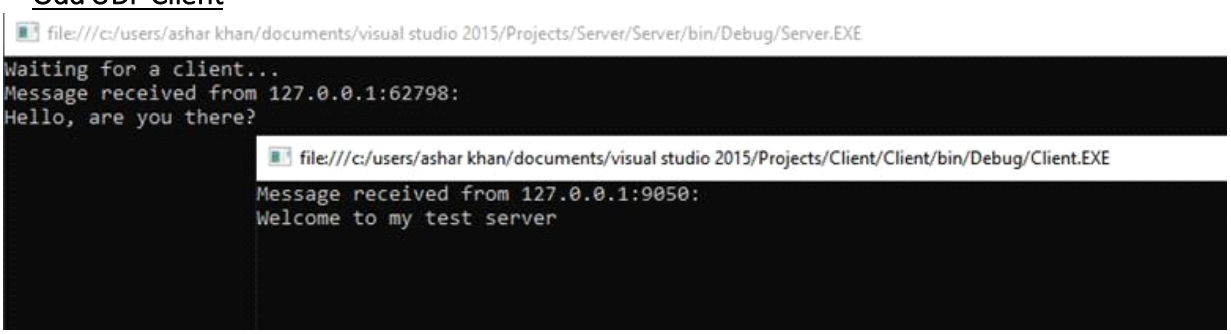
Client:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string input,  stringData;
            IPEndPoint ipep = new
            IPEndPoint(
            IPAddress.Parse("127.0.0.1"),
            9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
            string welcome = "Hello, are you
             there?";data =
             Encoding.ASCII.GetBytes(welcome);
            server.SendTo(data, data.Length, SocketFlags.None,
            ipep);IPEndPoint sender = new
            IPEndPoint(IPAddress.Any, 0); EndPoint Remote =
            (EndPoint)sender;
```

```
            data = new byte[1024];
            int recv = server.ReceiveFrom(data, ref Remote);
            Console.WriteLine("Message received from {0}:",
            Remote.ToString());
            Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
            while (true)
            {
                input =
                Console.ReadLine();if
                (input == "exit")
break; server.SendTo(Encoding.ASCII.GetBytes(input),
                Remote);data = new byte[1024];
                recv = server.ReceiveFrom(data, ref Remote);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Stopping
            client");server.Close();
            Console.ReadLine();
        }

    }
}
```

Ex 6.1 – 6.3

Odd UDP Client



Server:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
```

```
    class Program
    {

        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
            9050);Socket newsock = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
            newsock.Bind(ipep);
            Console.WriteLine("Waiting for a client...");

            IPEndPoint sender = new IPEndPoint(IPAddress.Any,
            0);EndPoint Remote = (EndPoint)(sender);
            recv = newsock.ReceiveFrom(data, ref Remote);
            Console.WriteLine("Message received from {0}:",
            Remote.ToString());
            Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
            string welcome = "Welcome to my test
            server";data =
            Encoding.ASCII.GetBytes(welcome);
            newsock.SendTo(data, data.Length, SocketFlags.None,
            Remote);while (true)
            {
                data = new byte[1024];
                recv = newsock.ReceiveFrom(data, ref Remote);

                Console.WriteLine(Encoding.ASCII.GetString(data, 0,
                recv));newsock.SendTo(data, recv, SocketFlags.None,
                Remote);


                Console.ReadLine();
            }
        }
    }
}
```

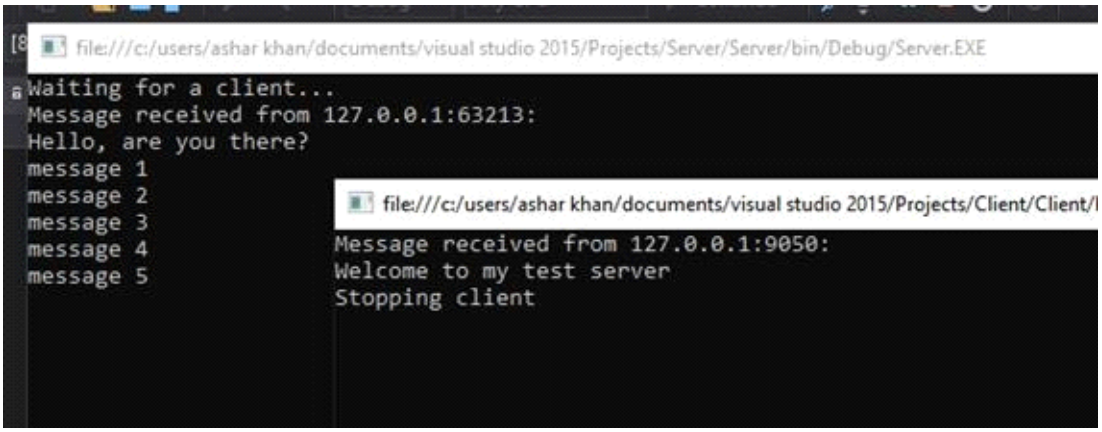
Client:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Client
```

```
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string  input, stringData;
            IPEndPoint ipep = new
            IPEndPoint(
            IPAddress.Parse("127.0.0.1"),
            9050);
            Socket server = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
            server.Connect(ipep);
            string welcome = "Hello, are you
            there?";data =
            Encoding.ASCII.GetBytes(welcome);
            server.Send(data);
            data = new byte[1024];
            int recv = server.Receive(data);
            Console.WriteLine("Message received from {0}:",
            ipep.ToString());
            Console.WriteLine(Encoding.ASCII.GetString(data, 0,
            recv)); while (true)
            {
                input =
                Console.ReadLine();if
                (input == "exit")
                    break;
                server.Send(Encoding.ASCII.GetBytes(input
                ));data = new byte[1024];
                recv = server.Receive(data);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Stopping
            client");server.Close();
            Console.ReadLine();
        }
    }
}
```

Ex 6.4 – 6.5

Distinguishing UDP Messages



Server:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;

namespace Server
{
    class Program
        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
            9050);Socket newsock = new
            Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
            newsock.Bind(ipep);
            Console.WriteLine("Waiting for a client...");
            IPEndPoint sender = new IPEndPoint(IPAddress.Any,
            0);EndPoint tmpRemote = (EndPoint)(sender);
            recv = newsock.ReceiveFrom(data, ref tmpRemote);
            Console.WriteLine("Message received from {0}:",
            tmpRemote.ToString());
            Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
            string welcome = "Welcome to my test
            server";data =
            Encoding.ASCII.GetBytes(welcome);
            newsock.SendTo(data, data.Length, SocketFlags.None,
            tmpRemote);for (int i = 0; i < 5; i++)
            {
                data = new byte[1024];
```

```
                recv = newsock.ReceiveFrom(data, ref tmpRemote);
                Console.WriteLine(Encoding.ASCII.GetString(data, 0,
                recv));
            }
            newsock.Close
            ();
            Console.ReadL
            ine();
            }
        }
    }
```

Client:

```
using System;
using
System.Collections.Generic;
using System.IO;
using
System.Linq;
using
System.Net;
using
System.Net.Sockets;
using System.Text;

using System.Threading.Tasks;

namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            IPEndPoint ipep = new
            IPEndPoint(
            IPAddress.Parse("127.0.0.1"),
            9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
            string welcome = "Hello, are you
            there?";data =
            Encoding.ASCII.GetBytes(welcome);
            server.SendTo(data, data.Length, SocketFlags.None,
            ipep);IPEndPoint sender = new
            IPEndPoint(IPAddress.Any, 0); EndPoint tmpRemote =
            (EndPoint)sender;
            data = new byte[1024];
            int recv = server.ReceiveFrom(data, ref tmpRemote);
            Console.WriteLine("Message received from {0}:", tmpRemote.ToString());
            Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
            server.SendTo(Encoding.ASCII.GetBytes("message 1"), tmpRemote);
            server.SendTo(Encoding.ASCII.GetBytes("message 2"), tmpRemote);
            server.SendTo(Encoding.ASCII.GetBytes("message 3"), tmpRemote);
            server.SendTo(Encoding.ASCII.GetBytes("message 4"), tmpRemote);
```

```
            server.SendTo(Encoding.ASCII.GetBytes("message 5"),
            tmpRemote);Console.WriteLine("Stopping client");
            server.Close(
            );
            Console.ReadL
            ine();
        }
    }
}
```

Lecture 8

Asynchronous

Server:

```
using System;
using
System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using
System.Linq;
using
System.Text;
using
System.Threading.Tasks;
using
System.Windows.Forms;
using System.Net;
using
System.Net.Sockets;
using  System.IO;
namespace
MultiClientAsync
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void button1_Click(object sender, EventArgs e)
        {

            TcpClient client = (TcpClient)lstClients[listBox1.SelectedItem.ToString()];
            NetworkStream ns = client.GetStream();
            StreamWriter sw = new StreamWriter(ns);
            string textToSend = "server says:" + textBox3.Text;
            sw.WriteLine(textToSend);
```

```
            textBox1.Text += textToSend +
            Environment.NewLine;sw.Flush();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
            TcpListener listner = new TcpListener(IPAddress.Loopback,
            11000);listner.Start();
            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnect), listner);

        }
        Dictionary<string, TcpClient> lstClients = new Dictionary<string, TcpClient>();
        byte[] b = new byte[1024];
        private void ClientConnect(IAsyncResult ar)
        {
            TcpListener listner =
            (TcpListener)ar.AsyncState;TcpClient client =
            listner.EndAcceptTcpClient(ar);NetworkStream
            ns = client.GetStream();
            object[] a = new
            object[2];a[0] = ns;
            a[1] = client;
            ns.BeginRead(b, 0, b.Length, new AsyncCallback(ReadMsg), a);
            listner.BeginAcceptTcpClient(new AsyncCallback(ClientConnect), listner);
        }
        private void ReadMsg(IAsyncResult ar)
        {
            object[] a =
            (object[])ar.AsyncState;
            NetworkStream ns =
            (NetworkStream)a[0];TcpClient
            client = (TcpClient)a[1]; int
            count = ns.EndRead(ar);
            string msg = ASCIIEncoding.ASCII.GetString(b,0,count);
            if(msg.Contains("@name@"))
            {
                string name =
                msg.Replace("@name@","");
                lstClients.Add(name,client);
                listBox1.Items.Add(name);


            }
            else
            {

            }




textBox1.Text +=msg +Environment.NewLine;


            ns.BeginRead(b, 0, b.Length, new AsyncCallback(ReadMsg), a);
```

```
        }
        private void button2_Click(object sender, EventArgs e)
        {


            foreach (object item in listBox1.Items)
            {
                var a = (item.ToString());
                TcpClient client =
                (TcpClient)lstClients[a];NetworkStream ns
                = client.GetStream(); StreamWriter sw =
                new StreamWriter(ns);
                string textToSend = "server says:" + textBox3.Text;

                sw.WriteLine(textToSend);
                textBox1.Text += textToSend + Environment.NewLine;
                sw.Flush();

            }

            //TcpClient client = (TcpClient)lstClients[listBox1];
            //NetworkStream ns = client.GetStream();
            //StreamWriter sw = new StreamWriter(ns);
            //string textToSend = "server says:" + textBox3.Text;
            //sw.WriteLine(textToSend);
            //textBox1.Text += textToSend + Environment.NewLine;
            //sw.Flush();
        }
    }
}
```

Lecture 9

Asynchronous

Client:

```
using System;
using
System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using
System.IO;
using
System.Linq;
using
System.Net;
```

```
using
System.Net.Sockets;
using System.Text;
using
System.Threading.Tasks;
using
System.Windows.Forms;

namespace asycWindowForm
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {

        }

        byte[] b = new byte[1024];
        TcpClient client = new
        TcpClient();
        private void button1_Click(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls =
            false;
            client.Connect(IPAddress.Loopback,
            11000);NetworkStream ns =
            client.GetStream(); StreamWriter sw
            = new StreamWriter(ns);
            sw.WriteLine("@name@"+_nametxt.Text)
            ; sw.Flush();
            ns.BeginRead(b,0,b.Length,ReadMsg,ns
            );


        }

        private void ReadMsg(IAsyncResult ar)
        {
            NetworkStream ns =
            (NetworkStream)ar.AsyncState;int count =
            ns.EndRead(ar);
            txtDisplay.Text += ASCIIEncoding.ASCII.GetString(b, 0, count);
            ns.BeginRead(b, 0, b.Length, ReadMsg, ns);

        }

        private void button2_Click(object sender, EventArgs e)
        {
            NetworkStream ns = client.GetStream();
            StreamWriter sw = new StreamWriter(ns);
            sw.WriteLine(_nametxt.Text + " says: " +
            textBox1);sw.Flush();
```

```
        }


    }
}
```