


|                                                                                   |                                               |                |                        |
|-----------------------------------------------------------------------------------|-----------------------------------------------|----------------|------------------------|
|  | COLLEGE OF COMPUTING AND INFORMATION SCIENCES |                |                        |
|                                                                                   | Mid-Term Assessment Spring 2021 Semester      |                |                        |
| Class Id                                                                          | 106383,84,85                                  | Course Title   | Network Programming    |
| Program                                                                           | BSCS                                          | Campus / Shift | North Campus / Morning |
| Date                                                                              | 11 <sup>th</sup> – March- 2021                | Total Marks    | 30                     |
| Duration                                                                          | 03 hours                                      | Faculty Name   | Misbah Anwer           |
| Student Id                                                                        | 64397                                         | Student Name   | MUHAMMAD ARQUM         |

**Instructions:**

- Filling out Student-ID and Student-Name on exam header is mandatory.
- Do not remove or change any part of exam header or question paper.
- Write down your answers in given space or at the end of exam paper with proper title “Answer for Question# \_ \_”.
- Answers should be formatted correctly (font size, alignment and etc.)
- Handwritten text or image should be on A4 size page with clear visibility of contents.
- Only PDF format is accepted (Student are advise to install necessary software)
- In case of CHEATING, COPIED material or any unfair means would result in negative marking or ZERO.
- A mandatory recorded viva session will be conducted to ascertain the quality of answer scripts where deemed necessary.
- Caution: Duration to perform Mid-Term Assessment is 02 hours only. Extra 01 hours are given to cater all kinds of odds in submission of Answer-sheet. Therefore, if you failed to upload answer sheet on LMS (in PDF format) within 03 hours limit, you would be considered as ABSENT/FAILED.
- Mention all the source code and output screenshots.

## QUESTION NO. 1

[3+5+2 Marks]

- a. Suppose you need to send some packets to destination but due to network failure for any reason, packets do not deliver to the intended destination. You need to write a program that can determine the packet failure and the problem can be reported to the user.

### TCPCLIENT CODE :

```
static void Main(string[] args)
{
    byte[] data = new byte[1024];
    string input, stringData;
    TcpClient server;
    try
    {
        server = new TcpClient("220.0.0.1", 7050);
    }
    catch (SocketException)
    {
        Console.WriteLine("Unable to connect to server");
        return;
    }
    NetworkStream ns = server.GetStream();
    int recv = ns.Read(data, 0, data.Length);
    stringData = Encoding.ASCII.GetString(data, 0, recv);
    Console.WriteLine(stringData);
    If( stringData != null)
    {
        Console.WriteLine("data is not recieved")
    }
    while (true)
    {
        input = Console.ReadLine();
        if (input == "exit")
            break;
        ns.Write(Encoding.ASCII.GetBytes(input), 0, input.Length);
        ns.Flush();
        data = new byte[1024];
        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.Close();
}
```

### TCPLISTENER CODE:

```
int recv;
byte[] data = new byte[1024];
TcpListener newsock = new TcpListener(7500);
newsock.Start();
Console.WriteLine("Waiting for a client...");
TcpClient client = newsock.AcceptTcpClient();
NetworkStream ns = client.GetStream();
```

```

        string welcome = "it is to inform you that your test for corona have come
positive \n regards abc hospital";
        data = Encoding.ASCII.GetBytes(welcome);
        ns.Write(data, 0, data.Length);
        while (true)
        {
            data = new byte[1024];
            recv = ns.Read(data, 0, data.Length);
            if (recv == 0)
                break;

            Console.WriteLine(
                Encoding.ASCII.GetString(data, 0, recv));
            ns.Write(data, 0, recv);
        }
        ns.Close();
        client.Close();
        newsock.Stop();
    }
}

```

- b. State the need of network programming. ABC hospital need to send the corona test report (in message format) to the patient, you need to create a program that can send message using connection-oriented socket and C# Socket helper classes. Mention all the code and output

**The term network programming refers to writing programs that execute across multiple devices, in which the devices are all connected to each other using a network. In order to share resources, exchange files or allow communications and to perform this we needs network programming.**

#### **TCPCLIENT CODE :**

```

static void Main(string[] args)
{
    byte[] data = new byte[1024];
    string input, stringData;
    TcpClient server;
    try
    {
        server = new TcpClient("220.0.0.1", 7050);
    }
    catch (SocketException)
    {
        Console.WriteLine("Unable to connect to server");
        return;
    }
    NetworkStream ns = server.GetStream();
    int recv = ns.Read(data, 0, data.Length);
    stringData = Encoding.ASCII.GetString(data, 0, recv);
    Console.WriteLine(stringData);
}

```

```

while (true)
{
    input = Console.ReadLine();
    if (input == "exit")
        break;
    ns.Write(Encoding.ASCII.GetBytes(input), 0, input.Length);
    ns.Flush();
    data = new byte[1024];
    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.Close();
}

```

#### TCPLISTENER CODE:

```

int recv;
byte[] data = new byte[1024];
TcpListener newsock = new TcpListener(7500);
newsock.Start();
Console.WriteLine("Waiting for a client...");
TcpClient client = newsock.AcceptTcpClient();
NetworkStream ns = client.GetStream();
string welcome = "it is to inform you that your test for corona have come
positive \n regards abc hospital";
data = Encoding.ASCII.GetBytes(welcome);
ns.Write(data, 0, data.Length);
while (true)
{
    data = new byte[1024];
    recv = ns.Read(data, 0, data.Length);
    if (recv == 0)
        break;

    Console.WriteLine(
        Encoding.ASCII.GetString(data, 0, recv));
    ns.Write(data, 0, recv);
}
ns.Close();
client.Close();
newsock.Stop();
}

```

The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window displays the output of the TCPLISTENER code, which is the message "it is to inform you that your test for corona have come positive" followed by a new line and "regards abc hospital". The text is displayed in a monospaced font on a black background.

## CONNECTION ORIENTED :

### CLIENT:

```
{
    IPAddress ip1 = IPAddress.Parse("192.168.1.1");
    IPEndPoint ep = new IPEndPoint(ip1, 2000);
    Socket c1= new Socket(ip1.AddressFamily, SocketType.Stream,
        ProtocolType.Tcp);
    c1.Connect(ep);

    byte[] arr = new Byte[1024]; // maximum available size
    c1.Receive(arr);
    Console.WriteLine(Encoding.ASCII.GetString(arr));
    Console.WriteLine("MESSAGE RECIEVED");

}
```

### SERVER:

```
{
    IPAddress ip1 = IPAddress.Parse("192.168.1.1");
    IPEndPoint ep = new IPEndPoint(ip1, 2000);
    Socket sk = new Socket(ip1.AddressFamily , SocketType.Stream
        ,ProtocolType.Tcp );
    sk.Bind(ep);
    Console.WriteLine("server waiting");
    sk.Listen(1); // backlog tell about pending connection ,(1) denotes 1 client

    Socket cl = sk.Accept();

    Console.WriteLine("ABC HOSPITAL");
    string str = Console.ReadLine();
    cl.Send(Encoding.ASCII.GetBytes(str));
    byte[] arr = new Byte[1024]; // maximum available size
    cl.Receive(arr);
    Console.WriteLine(Encoding.ASCII.GetString(arr));

    Console.WriteLine("MESSAGE DELIVERED");

}
```

- c. Assuming you can only access the visual studio in any system and you want to find host name and IP address Write the C# code to find and display the IP address and the host name. Also discuss loopback and broadcast IP addresses.

```
IPHostEntry IHE = Dns.GetHostEntry(Dns.GetHostName());
IPAddress IP = IHE.AddressList[0];
IPEndPoint EP = new IPEndPoint(IP, 3000);
    Console.WriteLine(EP);
```

**Loop back address is a reserved ip address which is used to check network adresss. While broadcast adresss is used to broadcast the network over devices.**

## QUESTION NO. 2

[3+3+6 Marks]

- a. Discuss the limitations of Send () and Receive () methods, what happened if we will use connect method in UDP client? Explain all the pros and cons, also write the client-side code for connectionless socket.

**Send () and receive () method are dependent on connection. Client side sending request through connect () will be accepted by server though accept(). If they fail in connection then they cannot send and received data and are limited. this limitation become disadvantage of using tcp , hence we cannot communicate without connection . and one more disadvantage is that server can go in blocking mode if client doesn't repond. But its have advantage as well that after connection it provide reliability of data transfer and acknowledgement.**

**Client side code :**

```
public static void Main()
{
    byte[] data = new byte[1024];
    IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050);
    UdpClient newsock = new UdpClient(ipep);
    Console.WriteLine("Waiting for a client...");
    IPEndPoint sender = new IPEndPoint(IPAddress.Any, 0);
    data = newsock.Receive(ref sender);
    Console.WriteLine("Message received from {0}:", sender.ToString());
    Console.WriteLine(Encoding.ASCII.GetString(data, 0, data.Length));
    string welcome = "Welcome to my test server";
    data = Encoding.ASCII.GetBytes(welcome);
    newsock.Send(data, data.Length, sender);
    while(true)
    {
        data = newsock.Receive(ref sender);

        Console.WriteLine(Encoding.ASCII.GetString(data, 0, data.Length));
        newsock.Send(data, data.Length, sender);
    }
}
```

- b. Rewrite the following code for Non-blocking socket. Also write the code for Socket time out.

```
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);
}
```

**Socket time\_out :**

```
public static void Main()
{
    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.Dgram, ProtocolType.Udp);

    int sockopt = (int)server.GetSocketOption(SocketOptionLevel.Socket,
        SocketOptionName.ReceiveTimeout);

    Console.WriteLine("Default timeout: {0}", sockopt);

    server.SetSocketOption(SocketOptionLevel.Socket,
        SocketOptionName.ReceiveTimeout, 3000);

    sockopt = (int)server.GetSocketOption(SocketOptionLevel.Socket,
        SocketOptionName.ReceiveTimeout);

    Console.WriteLine("New timeout: {0}", sockopt);

    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];

    recv = server.ReceiveFrom(data, ref tmpRemote);

    Console.WriteLine("Message received from {0}:", tmpRemote.ToString());

    Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));

    while(true)
    {
        input = Console.ReadLine();
```

```

if (input == "exit")
break;

server.SendTo(Encoding.ASCII.GetBytes(input), tmpRemote);

data = new byte[1024];

recv = server.ReceiveFrom(data, ref tmpRemote);

stringData = Encoding.ASCII.GetString(data, 0, recv);

Console.WriteLine(stringData);

}

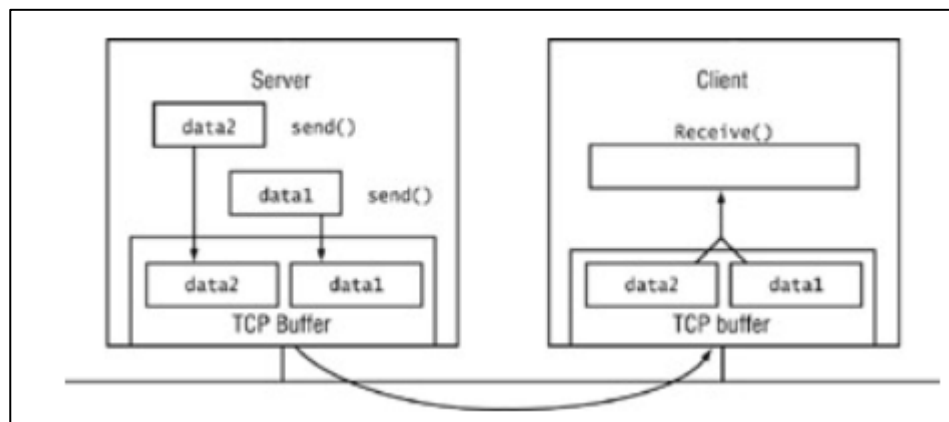
Console.WriteLine("Stopping client");

server.Close();

}

```

- c. Critique the following diagram, discuss all the issues and how to circumvent these issues, write the code for all three solutions



The two problems are ;

1. improper data buffer handling in which there can be shortage of space in the buffer because we don't know the size and type of data coming.
2. improper message handling across the network . there are no boundaries to differentiate two messages. In order to solve it we must send fixed size messages or we can size with each message or we can use marker system to separate messages.

**Variable length messages :**

**sending :**

```
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;
}

```

#### **recieving :**

```

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);
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;
}

```

**Fixed length messages :****Sending fixed messages:**

```
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;
}
```

**Receiving fixed size message :**

```
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;
}
```

QUESTION NO. 3

[6+2Marks]

- a. A device is mounted inside car compartment which has one or more cameras and a speaker. It connects to the smartphone with 4G data network. Location uses Smartphone GPS receiver only. No other computing device. - Create an application which can be used by a backend application to query car info (retrieve video feed and location), sends commands to a mobile which in turn send commands to the device, etc. - Make suitable assumptions.

Your task is to

- i. Identify which protocol should we used for such application (TCP or UDP) and justify your answer with valid arguments.

**We will use TCP protocol for this because car is connected to smartphone . here for connection smartphone would have send request for connect( ) to car which is acting as server and will accept ( ) request hence forming connection.also tcp allows crosss platform communication among hetrogenous network.**

- ii. Implement multithreaded Client server chat application for above protocol.
- b. Update the above code for group communication