Computer Communications and Networks
Java Lab-3

User Datagram Protocol (UDP) is one of the popular data transmission protocols for the Internet. It differs from TCP in that, TCP establishes a connection and maintains it throughout the communication session between client and server processes, whereas UDP does not establish any connection prior to transmitting data. Each packet from the sender process must contains the destination address.

The UDP`Server.java`, UDP`Client.java` programs exchange messages under UDP protocol.

1. A Client reads a line from keyboard and sends the line out its socket to the server
2. The server reads this line from its socket
3. the server converts the line to uppercase
4. the server sends the modified line from its socket to client
5. The Client reads the modified line from its socket and prints on the monitor.

Download UDP`Server.java` and UDP`Client.java` from the blackboard

```
DatagramSocket clientSocket = new DatagramSocket();
```

This line creates the object ClientSocket of type DatagramSocket but does not initiate a connection. Execution of this line opens a door but does not create a pipe between the two processes.

```
InetAddress IPAddress = InetAddress.getByName("hostname");
```

In order to send bytes to a destination process, this line calls the DNS to translate the hostname to an IP address.

```
byte[] sendData = new byte[1024];
byte[] receiveData = new byte[1024];
```

The byte arrays sendData and receiveData will hold the data the client sends and receives, respectively.

```
sendData = sentence.getBytes();
```

It takes string sentence and renames it as sendData, which is an array of bytes.

```
DatagramPacket sendPacket =
         new DatagramPacket(sendData, sendData.length, IPAddress, 9876);
```

This line constructs the packet, sendPacket, which the client will send into the network through its socket. It includes the actual data, its length, IP address of the server and port number of the application.

```
clientSocket.send(sendPacket);
```

This line sends the packet into the network through its socket.

```
DatagramPacket receivePacket =
         new DatagramPacket(receiveData, receiveData.length);
```

While waiting for the server response, client makes place to hold the message from the server.

```
clientSocket.receive(receivePacket);
```

It receives the packet and puts it in receivePacket, a place created by the client process.

a)
Compile both files. Run these two programs in two different windows.

Client enters a text and server converts it into uppercase letters and returns it back.

b) Modify the above programs so that the client and server programs terminate when client enters CLOSE string.

c) Modify Client program to transmit a packet (similar to Lab-2) and waits for an ACK. Similarly modify server program to send ACK after receiving a packet from the client.
This activity will help you to implement first part of the coursework.