HW1-Report

Name: Ra'ed Khwayreh
ID: 11822076

1-TCP

First, in the **client side**
I create a client socket connect to server, I did it in local host **so** the IP 127.0.1.1 and then i create input stream to read data from black screen to send it to server, then I create output stream attached to socket, then create input stream attached to socket to read receive data from server.

```
String sentence;
String modifiedSentence;
BufferedReader inFromUser = new BufferedReader(new InputStreamReader(System.in));
Socket clientSocket = new Socket("127.0.0.1", 6789);
DataOutputStream outToServer = new DataOutputStream(clientSocket.getOutputStream());
BufferedReader inFromServer = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
sentence = inFromUser.readLine();
outToServer.writeBytes(sentence + '\n');
modifiedSentence = inFromServer.readLine();
```

to send continuously to server, I add while loop in client class
-To end connect I make a keyword **("END")**, if I send it to server the server should send message that mean end connect successfully.

```
if(modifiedSentence.equals("End Connected Succsesfully"))
{
    System.out.println("FROM SERVER: " + modifiedSentence);
    clientSocket.close();
    break;
}
```

In the server side
First Create server socket at port 6789
I create additional class, its name course, each of object from it has an id and course name then I read from file and store all item in array, the array items should be
id course name id course name and so on.
Then I create an object array of course class, each index in this array is an object has an id and course name, I fill each object from the first array that I store its items from file.

```
File myObj = new File("Courses.txt");
String[] item=new String[10];
course c[]=new course[5];
```

```
int k=0;
for(int i=0;i<5;i++)
{
    c[i]=new course(item[k],item[k+1]);
    k+=2;
}
```

Then Wait, on server socket for contact by client, create input stream, attached to socket to read request from user, Create output stream, attached to socket to send response to client.
If the request from client is “END” the response be to end connect, else check if the request data exist in server data base, if there is an exist data the response be the course name, else the response is Error 404 Not Found.

```
if(s.equals("END"))
{
    System.out.println("End Connect");
    outToClient.writeBytes("End Connected Succsesfully"+'\n');
}
else
{
    boolean found=false;
    int i;
    for(i=0;i<c.length;i++)
    {
        if(s.equals(c[i].id))
        {
           found=true;
           break;
        }
    }
    if(found==true)
        s=c[i].courseName;
    else
        s="Error 404";
    outToClient.writeBytes(s+'\n');
}
```

Outputs:
Client Side:

```
run:
Client: 10636454
FROM SERVER: Network1
Client: 10636323
FROM SERVER: Architecture
Client:    10636455
FROM SERVER: Network2
Client: 1063575
FROM SERVER: Error 404
Client: 10636111
FROM SERVER: C programming
Client: END
FROM SERVER: End Connected Succsesfully
BUILD SUCCESSFUL (total time: 43 seconds)
```

Server Side:

```
run:
FROM Client: 10636454
FROM Client: 10636323
FROM Client:    10636455
FROM Client: 1063575
FROM Client: 10636111
FROM Client: END
End Connect
```

2-UDP

First, in the client side
I create a client socket connect to server, Translate 127.0.0.1 to IP address using DNS and then Create datagram with data-to-send,length, IP address, port.
Send datagram to server then Read datagram from server.

```
BufferedReader inFromUser = new BufferedReader(new InputStreamReader(System.in));
DatagramSocket clientSocket = new DatagramSocket();
InetAddress IPAddress = InetAddress.getByName("127.0.0.1");
byte[] sendData = new byte[65535];
byte[] receiveData = new byte[65535];
String sentence = inFromUser.readLine();
sendData = sentence.getBytes();
DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress, 9876);
clientSocket.send(sendPacket);
DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
clientSocket.receive(receivePacket);
String modifiedSentence=data(receiveData).toString();
```

Same in TCP:

- Read form file
- To send continuously to server, I add while loop in client class
- To end connect I make a keyword **(“END”)**, if I send it to server the server should send message that mean end connect successfully.

In the server side

First Create server socket at port 9876
Create space for received datagram, Receive datagram then Get IP address, port #, of sender, after that check if the request data exist In server data base the response is same in TCP, Then write out datagram to socket. wait for another datagram.

I add function in UDP to convert byte to string because toString is not enough, Byte have an array of char, so this function use String Builder and append each char in byte to it, the function return StringBuilder then convert it to string using toStirng().

```java
public static StringBuilder data(byte[] a)
{
    if (a == null)
        return null;
    StringBuilder ret = new StringBuilder();
    int i = 0;
    while (a[i] != 0)
    {
        ret.append((char) a[i]);
        i++;
    }
    return ret;
}
```

in the end of server class I define receiveData to receive new data, because it is array of Byte assume if the first receive data was 11152
if the second was 256 the receive data well be 25652 because it save the reminder digits from first receive data, so I'll clear it before receive data

Outputs:
Client Side:

```
run:
Clinet: 118264
FROM SERVER: Error 404
Clinet: 10636455
FROM SERVER: Network2
Clinet: 10636511
FROM SERVER: Security
Clinet:    10636511
FROM SERVER: Security
Clinet: END
FROM SERVER: End Connected Succsesfully
BUILD SUCCESSFUL (total time: 30 seconds)
```

Server Side:

```
run:
FROM Client: 118264
FROM Client: 10636455
FROM Client: 10636511
FROM Client:    10636511
FROM Client: END
End Connect
```