# Lab 1 – TCP/IP Sockets

## Q1.

The client should

* Connect to the server
* Send a name (String) to the server
* flush() the PrintWriter [p.flush()]. This forces the data to be sent even if the TCP buffer is not full (which it won’t be).
* Receive a String back from server
* Print out that String

The server should repeatedly

* Accept a connection from the client.
* Read a String (john)
* Send back another String “Hello john”
* Close the connection with the client (p.close())

## Q1b.

Work in pairs and connect between 2 PCs. Use the IP address of the server in the client instead of localhost. [ipconfig at the command prompt will give you the IP address of your PC].

## Q2.

The client should

* Connect to the Server
* Send a name (String) to the server
* flush() the PrintWriter [p.flush()]. This forces the data to be sent even if the TCP buffer is not full (which it won’t be).
* Read a number of Strings back from server
* Print out these String

The server should

* Accept a connection
* Read a String (john)
* Send back a number of lines of text (Strings)
* Close the connection with the client (p.close())

The client should read from its Scanner in much the same way as we do in R1readFile.java.

## Q3.

Get R1ReadFile.java and R2CopyFile.java working.

## Q4.

The client should

* Connect to the Server
* Send the name of a file to the server
* flush() the PrintWriter [p.flush()]. This forces the data to be sent even if the TCP buffer is not full (which it won’t be).
* Read the contents back from server
* Print out this content

The server should repeatedly

* Accept a connection
* Read a fileName
* Read the file with that name from disk
* Send back the contents of the file to the client.
* Close the connection with the client

Note that the file should be in the project folder (not the src folder).

## Q5.

Same as above but the client should save the file in a folder called clientData.