Assignment 2

The Trivial File Transfer Protocol (TFTP) is an Internet software utility for transferring files that is simpler to use than the File Transfer Protocol (FTP) but less capable. It is used where user authentication and directory visibility are not required. For example, it is used by Cisco routers and switches to transfer images of the operating system from/to the devices.

NOTE: TFTP's port is 69 (a system/OS port). This is below 1024 and therefore you cannot bind a Socket to it unless you have administrative access rights on the machine you are working. To avoid any complications you can use any port > 1024 to bind server sockets.

Implementation of the Trivial File Transfer Protocol (TFTP)

For this task you need to implement (in Java) the Trivial File Transfer Protocol (TFTP) as specified in **RFC 1350 (https://www.ietf.org/rfc/rfc1350.txt)**. You will submit source code for a client and server application that 'speak' the TFTP protocol. You will built your protocol on top of UDP. Compared to the specifications in the RFC, you will implement a slightly simplified version:

- **Support for octet mode only**. The files should be transferred as a raw sequence of bytes. Do not read, write or transfer files as characters.
- **Support** only for error handling when the server is unable to satisfy the request because the file cannot be found.
- **No support for error handling when data duplication occurs.**

The client and server applications should be simple Java console applications. The server should operate (i.e. read and write files) in the directory where it is started from. The server should support simultaneous file transfers to and from multiple clients. The client should just read command line arguments (or have a very simple console-based menu - e.g. "press 1 to store file, press 2 to retrieve file") and execute user commands (i.e. reading or writing a file).

Hint: the simplest way to implement timeouts is by calling the *setSoTimeout()* method on the DatagramSocket objects (assuming that you are using blocking I/O). If the timeout expires, a *java.net.SocketTimeoutException* is raised, though the DatagramSocket is still valid.

Marking Criteria

You should make sure that your code compiles. Code which does not compile will receive at most 20%.

I will assess your assignment using the following criteria:

TFTP-UDP-Server (40%)

- Is the server-side of the protocol fully and correctly impl
  the RFC)? i.e. read/write requests, acknowledgments, t
  simultaneous file transfers.

TFTP-UDP-Client (40%):

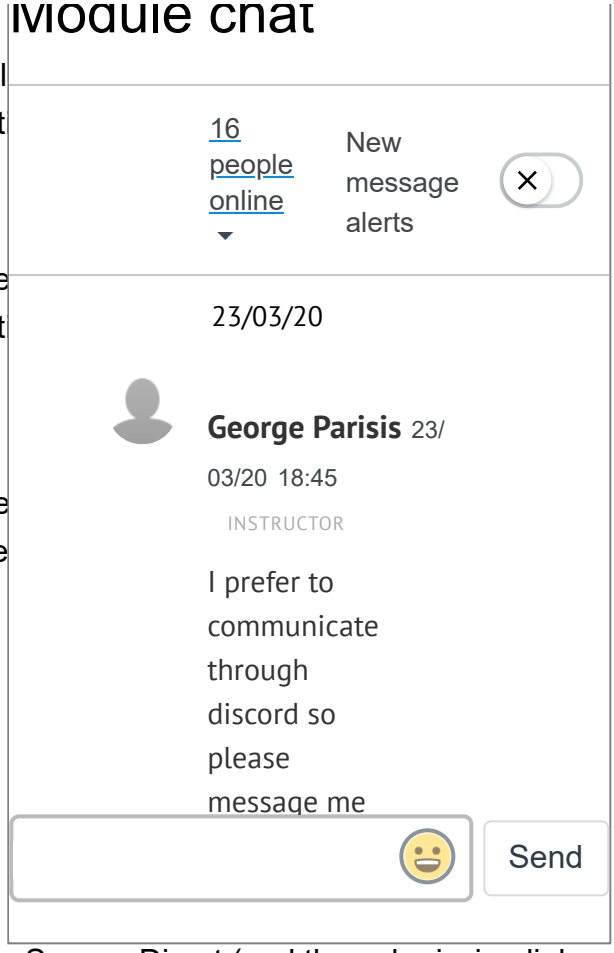
- Is the client-side of the protocol fully and correctly imple
  the RFC)? i.e. read/write requests, acknowledgments, t

Description of Protocols (15%):

- Is there description for both protocols and how/where the
- Is the description well-written with clear references to the

Quality of Code (5%):

- Is the code commented appropriately?
- Is it indented correctly?
- Is naming of variables and methods sensible?



Submission Guidelines

You should submit the coursework by the deadline posted on Sussex Direct (and the submission link on Canvas). Standard penalties for late submissions will apply. You will need to submit a .zip file containing the following:

- Well-formatted and well-documented source code (written in Java) in 2 separate Netbeans projects that can be compiled and run. The Netbeans projects, named TFTP-UDP-Server and TFTP-UDP-Client, respectively, will include all required source code to execute the file transfer protocol.
- A short report (up to 2000 words) describing your protocols, the source code and the design decisions that you made.

Please do not put your names on your submissions, but do include your candidate number. Failure to submit source code, as described in the first bullet, will result to a zero mark as I will not be able to assess your programming effort.

Plagiarism and Collusion

- The coursework you submit is supposed to have been produced by you and you alone. This means that you should not: work together with anyone else on this assignment
- give code for the assignment to other students
- request help from external sources
- do anything that means that the work you submit for assessment is not wholly your own work, but consists in part or whole of other people's work, presented as your own, for which you should in

fairness get no credit

If you need help ask your tutor The University considers it mi
than from your tutors, or to copy work from uncredited sourc
has happened, formal action will be taken. Remember that ir
each other on assignments) the student giving help is regard
the person receiving help, and is liable to potentially receive
suspicious similarities in student code are surprisingly easy t
dealing with it are stressful and unpleasant. Academic misco
do complain to us about unfairness. So please don't collude