Computer Networks
Assignment - 1

Aim :
To create an Application Level FileSharingProtocol with support for download and upload for files and indexed searching.

Features :

- The system should have 2 clients (acting as servers simultaneously) listening to the communication channel for requests and waiting to share files (avoiding collisions) using an application layer protocol (like FTP/HTTP).
- Each client has the ability to do the following :
  - **Know the files present** on each others machines in the designated shared folders.
  - **Download** files from this shared folder.
- The system should periodically check for any changes made to the shared folders.
- File transfer should incorporate MD5checksum to handle file transfer errors.

Specifications :
The system should incorporate the following commands :

- IndexGet flag (args)
  - can request the display of the shared files on the connected system.
  - the history of requests made by either clients should be maintained at each of the clients respectively.
  - the flag variable can be **shortlist, longlist or regex**.
  - shortlist :
    flag would mean that the client only wants to know the names of files between a specific set of timestamps. The sample query
    is as below.
    - $> IndexGet shortlist <starttimestamp> <endtimestamp>
    - Output : should include ‘name’, ‘size’, ‘timestamp’ and ‘type’ of the files between the start and end time stamps.
  - longlist : flag would mean that client wants to know the entire listing of the shared folder/directory including ‘name’, ‘size’, ‘timestamp’ and ‘type’ of the files.
    - $> IndexGet longlist
    - Output : similar to above, but with complete file listing.
- FileHash flag (args) :

○ this command indicates that the client wants to check if any of the files on the other end have been changed. The flag variable can take two values, verify
and checkall

■ verify : flag should check for the specific file name provided as command line argument and return its ‘checksum’ and ‘lastmodified’ timestamp.

- $> FileHash verify <filename>
- Output : checksum and lastmodified timestamp of the input file.

■ checkall : flag should check perform what ‘verify’ does for all the files in the shared folder.

(HINT : this command can be used for the periodic check of changes in the files of shared folders)

- $> FileHash checkall
- Output : filename, checksum and lastmodified timestamp of all the files in the shared directory.

● FileDownload flag (args): ○ as the name suggests, would be used to download files from the shared folder of connected user to our shared folder.

○ the flag variable can take the value TCP or UDP depending on the users request.

○ If a socket is not available, it should be created and both clients must use this socket for file transfer.

■ $> FileDownload <filename>

■ Output : should contain the filename, filesize, lastmodified timestamp and the MD5hash of the requested file.

○ HINT : the filesize parameter might be used for requesting the client side to allocate memory and use the allocated memory for downloading the file.

Instructions :

- The languages permitted for this code are C , python, C++.
- Put all your codes in a single folder named ‘FileSharingProtocol’ and compress this to .tar format and upload it on the moodle (Failing to follow the submission format would be penalised).
- Start your assignment early! It’ll take a lot of time to complete.
- All error scenarios must be gracefully handled (Programs crashing during testing will be penalised).
- Use moodle for general doubts discussion and clarification.
- You do not have to use threads for this project (You may use it for learning purposes).
- Project can be done in groups of at most 2. Only one submission is required.
- There will be manual evaluation of the assignment.
- PLAGIARISM IN ANY FORM SHALL NOT BE TOLERATED (MOSS WILL BE USED) AND A STRAIGHT ‘F’ GRADE FOR THE COURSE WILL BE GIVEN.