Designs:

- When a client sends a request to open a file, the proxy will first check the server to see if the path is legal, if the file exists. If it exists, check if it is a directory and get the length of the file if it's not a directory.
- If the file F that the client requests to open exists on the server and it's not a directory, the proxy will first delete any unused old versions of F in the cache, and then download the newest version of F from server.
- Download and upload files in chunks.
- If the proxy downloads the newest version of F from the server (such as when open in READ, WRITE, and CREATE mode), it will add a timestamp suffix to the filename in the format: filename#hhmmss. E.g., "Test.java#061434".
- When a client opens a file, proxy can't evict the file (even when the file is deleted from the server) until it is closed.
- Multiple clients can read a same file.
- When client request to write a file, say Test.java#061434. The proxy will first copy it and names the copy "Test.java#061434#w". And there can be only one writer for a certain file.
- Last writer wins. When a client finished writing a file F. The proxy will deleted any outdated and unused F and then upload the file to the server. It will also cacheQueue: remove F from cacheQueue and put F at the end of the cacheQueue to implement LRU. The server will first deleted outdated F and create the new F.

Important data structure and methods in Proxy.java:

- private static final int CHUNKSIZE=1000000: Upload and download files in chunks.
- LinkedList<String> cacheQueue: Implement LRU.
- HashMap<String, Integer> reading: The number of readers of a certain file.
- HashMap<String, String> notUse: The not used filename of a certain path. E.g., <Test.java, Test.java#061434>. There can be at most one unused version (the newest) of a certain file in the proxy.
- int cacheUsed: Number of bytes that has been used in the proxy.
- private static String parsePath(String path): Simplify the path. E.g., input: A/././B, output: A/B
- private static synchronized boolean checkCache(): Check if the cache is out of space. If yes, evict files according to LRU policy.
- private static synchronized void updateQueue(String name, int op): Insert or remove elements in the queue to implement LRU.

Important data structure and methods in Server.java:

- HashMap<String, String> version: Save the timestamp for each file. <filename, timestamp> E.g., <”Test.java”, “125959”>
- public String[] checkFile(String path): Simplify the path. Return: if the file exists, if the file is a directory, if the path is legal, and the length of the file.
- public boolean deleteFile(String fileName): Delete file. Return true if success. Return false otherwise.
- public byte[] downloadFile(String fileName, int I): Download file in chunks.
- public void uploadFile(String fileName, byte[] fileData, String time, int I): Upload file in chunks.