Files

- Hard links are effectively "mappings" between a file name and an inode. Removing all of these mappings (deleting all hard links) will effectively delete the file. Think of a "count" of the number of (hard) links associated with a file that decrements every time you call `rm`
    - If a program is currently using a file and a hard link to that file is deleted, the file will only be deleted once the program ends.
- Symbolic links can be thought of as an alias for the full directory of the file that they are linking to (i.e. when a symbolic link is expanded to /directory/file).
    - Deleting the file that the symbolic link points to can therefore lead to a "dangling" symbolic link (i.e. file doesn't exist so /directory/file is invalid). If you were to create another file with the same name as the originally deleted file, then that symbolic link will point to the new file (/directory/file exists again so the symbolic link is valid)
    - The full path that symbolic links are aliased to is relative to the directory that it was created in, so moving the symbolic link to another directory (or renaming it as you move it) can result in the symbolic link dangling or even pointing to something else.
- Files can be updated in an atomic manner by writing changes to a temporary file (FILE#) while allowing for any other programs to read the original file (FILE) during the writing process. Once the writing process is done, the temporary file can now become the main file (mv FILE# FILE).