CSE 3320 Operating Systems
Summer 2021, © DL, UTA, 2021

Programming Assignment 3
File System Internals
Due: On Canvas

Description:
You will develop and implement a small file system, a combination of Unix and CP/M concepts, the file system (in this example) is called "FS".
Your file system will not be part of an operating system, but similar to Many modern file systems it will run in several different operating systems to provide a "portable" file system (a group of files).

Details:
Your FS will use a file (for example "disk01"), rather than directly using a physical flash or disk, to store data.
You may have several disk-like files (for example: disk01, disk02), used to store data. The data stored in disk01 may be a user's programs, text files, other data files, or any type of binary information. In addition to the data stored, your FS will need to store other, meta-information, such as free space (blocks), directory details, and possibly other information. The FS directory is flat (one level) fixed sized, has a user name associated with each file, and has fixed sized "blocks" (entries).
You should use fixed size blocks (similar to disk blocks) of size 256 bytes to store files and all meta-data in your "disk".
(Your "disk" (for example "disk01" is logically divided into a number of "sectors", which are fixed size blocks. Everything that is stored (persistent) is in these blocks)

Your program (the "FS" executable) should provide the following operations:
Createfs #ofblocks – creates a filesystem (disk) with #ofblocks size, each 256 bytes
Formatfs #filenames #DABPTentries
Savefs name– save the "disk" image in a file "name"
Openfs name- use an existing disk image
List – list files (and other meta-information) in a FS
Remove name –remove named file from fs
Rename oldname newname – rename a file stored in the FS
Put ExternalFile – put (store) Host OS file into the disk
Get ExternalFile – get disk file, copy from "disk" to host OS file system
User name – this user owns this user's files
Link/Unlink – Unix style file linking
Bonus: Set/Use file permissions for r/w/x, implement subdirectories, "check disk"
Implement in either the "Go" or "Rust" programming language (20 to 75 point bonus).

Implementation:
Your FS should have 4 (or more, if easier to implement) sections:
A FileNameTable (FNT), an directory and attribute/block pointer table (DABPT), and the data blocks.
The FNT should be of size allocated, each entry should contain a 56 char (maximum) file name and an inode pointer (index to DABPT)(blocks).
The DABPT should be allocated from disk blocks, 4 entries per block, where each entry should contain a file meta-information (FileSize, last time+date (secs), "pointers" to data blocks- (that is a pointer to the first entry in the Block Pointer Table), and a user name (maximum length 40 characters)
The Block Pointer Table has direct pointers to data blocks, and one additional pointer to another entry in the Table, if needed (for big files), these may be chained for very large files. (Similar to CP/M extents), there is an array of sets (groups) of 8 pointers, of 32 bits each, and the last (8^{th}) pointer can chain.

Since disks (and some meta-information) are fixed size, many small or one large file might not fit on the "disk". File names, file attributes and other file information stored in FS are restrictive (for example, file creation time).