Ethan Miller
Philron Hozier
Paul Fretz

Lab 03 - Part 2

All the logic for this program is contained in `lib/fs.c`. To run, cd into the lib/ directory and enter `make clean` then `make`, followed by simply running the fs binary: `./fs`.

Our program defines structs as recommended in the lab: inode and super_block are listed at the top of the file, along with methods to instantiate them. Below, the required API methods are defined: create, delete, read, write, and ls. The logic for these methods is fairly straightforward.

In create, we take the name and size of the file as an argument. Then we iterate through the super_block's list of inodes and assign the new file to the first one that is not marked as being "used." Finally, we create a file object with the given name and write it to the correct location within disk0.

For delete, we find the inode with the name that matches the file to be deleted, and using that information, "zero-out" the bytes in disk0 that correspond to that file. Finally, we mark that inode's name as a blank string and assign its "used" property to 0 (indicating that it is now free).

The read method first iterates through all the inodes and finds the one with a name matching the given file name. Then, we open disk0 at the location in bytes that corresponds to the file and read that section of bytes into the char[1024] buffer. The write method behaves in almost the same way, but rather than reading the file's contents to the buffer, we write the given buffer to the file's location in disk0.

Finally, ls simply prints out the "name" property of all inodes which are currently in use.

These four API methods are called within the larger parse() method, which is responsible for reading the provided input file and calling the appropriate methods depending on the command in each line. The main() method instantiates a super_block and then calls parse() to run the program.