Report for Assignment #4

# Jose Toledo and Jasmine Batten

## Design and Implementation

* The disk’s file system was divided into these sections: a “magic number” to let the OS know where the file system starts, an inode bitmap that was used to maintain which inodes in the inode table were free or full, a sector bitmap which was used to show which sectors were free or full, the inode table which stored the inodes and their metadata, and the datablock where all the file data was stored.
* For every file or directory that was created, an inode had to be added to the inode table with the appropriate type, parent inode number, and the file name. This means the inode bitmap had to be updated appropriately to find the first unused inode and the sector bitmap had to be updated by finding the appropriate sector using offsets and where the entry started. Deleting a file or directory required the opposite of creating a file or directory, meaning the inode had to be removed from the table and the sector bitmap had to be updated. Every time these files and directories were updated, they were written to the disk. Both of these operations required absolute paths and had to follow the given path/directory all the way to the last child inode that matched the required operation (create file in given directory with given name, etc).
* To read a file, you had to acquire the inode of the file and the address of where the file started in the datablock, then you needed to read each sector that contained the given file into a buffer. This required a reading position to move the “reader” along. You also needed to have previously checked that the file given was not already open.
* To write to a file, you similarly had to get the inode of the file and the address of where the file started in the datablock, first checking that the file was not already open, then copying the buffer a sector at a time into the given datablock.

## Work done by team members

* After we talked and understood the basics of the assignment (how a file system works), we both worked separately on the assignment and merged our code at the end. We discussed and worked out any bugs or issues we may have run into and why they occurred. We worked on the assignment separately at first so that we could learn how to work on the assignment independently and also come together to fix any errors and discuss it to have a better understanding of choices we made.