Pranav Tripuraneni

The **metadata** used for the project is a struct superblock which is always at the start of memory. The superblock contains the size of bytes remaining to create more disk blocks, pointer to the location where new file struct should go, pointer to where a new inode struct would go, a pointer to where a new disk block would go, a pointer to where the first file is, and a counter for file descriptors.

The inode struct are stored right after the superblock and combined have a partition of 1MB. The inode has a pointer to which disk block it is referencing and keeps track of how many bytes have been used by that disk block. It also has a pointer to the next inode, in the file in order.

The wo_file structs are stored after the inode partition and have 50000 bytes allocated to its partition. It has a char array for the file name, a file descriptor, a pointer to the starting inode of the file, a pointer to the next file in the file list, and ints that keep track of whether it is in read or write mode or if it is open.

The global variables of sizeof_filepartition is how much memory is partitioned to files, sizeof_inode_partition is how much is for inodes and superblock, and sizeof_disknode is the size of disks which is 1K for this. It also has a char* mem which is assigned to whatever memory is passed in wo_mount, which keeps a reference of it for the rest of the program. wo_main_file keeps track of what filename was passed in wo_mount so wo_unmount can write to the same one when unmounting.

Build Notes:

wo_open is given 3 or 4 arguments. The first is the filename, second is flags, third is optional for WO_CREAT mode, and the last argument has to be “done”, so wo_open can tell how many arguments were passed. If 4 arguments were passed then wo_create is called, since it is in create mode.

wo_mount takes in memory address, and file name and checks whether the file has content, if it doesn’t then it initializes a superblock in the memory. It returns an error if the file given fails to open. Otherwise if the file is not empty then it reads from the file according to the format given in wo_unmount and then creates the superblock, all files, inodes, and disk blocks in memory according to the file.

wo_unmount returns error if the file doesn’t open. Otherwise writes to the file with the superblock information at top followed by the first file and its inode information and disk block data in order, then it continues through all files.

printfiles just prints out the current list of files and each file inodes for testing purposes.

wo_open is given 3 or 4 arguments. The first is the filename, second is flags, third is optional for WO_CREAT mode, and the last argument has to be “done”, so wo_open can tell how many arguments were passed. If 4 arguments were passed then wo_create is called, since it is in create mode. Otherwise if 3 were passed then it checks if the filename exists in the file list,

if not it returns error. Otherwise, it finds the file in the file list and changes its flags based on which of WO_RDONLY, WO_WRONLY, WO_RDWR is passed to it. It gives an error if a flag is passed which is not one of those. It returns its file descriptor.

wo_create is called when wo_open is in create mode. It returns an error if the file already exists. If it doesn't then it creates a new file where the superblock points to, and sets the flags based on what is given. It increments the file descriptor counter and gives the file a file descriptor and puts it at the end of the files list. It returns its file descriptor.

wo_close returns an error if the file descriptor doesn't exist, otherwise just changes file given to closed.

wo_read finds if the file descriptor given is valid otherwise it returns an error. It returns an error if there is no inode in the file. Otherwise it goes through all its disk blocks and memcpy() until it is all copied.

wo_write does the same thing as read except in reverse and creates a new disk block and inode if more space is required to write.

On testing it does create over 50 files and it does store 2MB of data which was done by

```
    char file[50];
    for(int i=0; i<50; i++){
        sprintf(file, "%d F", i);
        wo_open(file, "WO_RDWR", "WO_CREAT", "done");
        char str2[3000] =
"cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccc";
        for(int j=0; j<10; j++){
            wo_write(i, str2, 3000);
        }

    }
```