NAME : Arman Makhani and Raj Kariya
IDs: 202103006 and 202103048
OS Lab-Assignment 8

Exercise
Create a complete design document for the file system and the system calls. This should include:
1. Structure of the superblock, nodes, and other needed structures for maintaining free nodes, directory structure, etc.
2. Associated data structures for these structures.
3. Justification of the choices made above and the choice of various parameters, like node size, superblock size, etc.
4. Similarly provide details of the "system call" design and justification.
5. Provide a top-level pseudocode for the implementation of these system Calls.

Description:

We defined maxNumOfFiles as 100 , maxDataBlocksPerFile as 10 and fileNameLimit as 100 , maxNumDataBlocks as 1000 , totalSizeofDisk as 4096000 and blockSizeLimit as 4096. So , 1000 datablocks are allowed and 10 data blocks are there for each file so basically 100 files are maximum number. The structure of file has four elements. isCreated represents the creation of file , fileName represents the name of file , dataBlocks represents datablocks per file and numBlocks represents represents the current block. The structure superblock has 6 elements : totalSize represents the total size of super block , blockSize represents size of block , totalNumBlocks represents total number of blocks , free represents which blocks are free or not , numEmptyBlocks represents number of empty blocks and numFullBlocks represents number of full blocks.

The structure directory contains three elements files , maxFileSize and maxNumFiles. The structure datablock has data having the size of block size limit. We made a file pointer called disk , a superblock sb and a directory dir. Then we printed the choices for different file operations to the user . user choices are stored in format. If format is 1 , then we opened the file in read-write format . If disk is zero , the file cant be opened otherwise we mounted the file. Then we defined suberblock's size . Intially we freed every block , initially evry block is empty. Then we defined file size. Then we calculated number of files. Then we initiliased every file's isCreated as 0. Then we wrote data to the file. Then if the user chooses to use the disk as it is then we again open CFS data , if disk is 0 then error is shown otherwise we would be able to mount. We read the superblock and directory block from the disk . Function listFiles lists all the created files. Function create file is used to create a file by checking the isCreate array, taking the data and the name of the file from the user . Then we will traverse the dtablocks and enter the data into

datablocks. '$' is used to show the end of data. numBlocks is increased as we are putiing data into the blocks . Then we will assign the value of numBlocks to the dir.file index. Then we will again take the pointer to the beginning. Then we updated the values of number of empty and full blocks . Then we wrote the updates back to the disk and wrote the updates directory back to the disk.

Function deletefiles is used to delete the files. We will ask the user to enter the name of the file to be deleted , then we will compare the input with file names . If we exceed the number of files then the given file is not present otherwise we found the file and we will empty all the data blocks and turn it's isCreated to 0 , then we again updates number of empty and full blocks . Function unMount is used to close the disk if the operations are not successful.

In the main function , we asked the user to enter their choice to perform different operations and called the respective functions to do certain operations. And that is how we performed different tasks on a given disk.