CLEVELAND STATE UNIVERSITY

CIS 545 Architecture and Operating Systems

Project – 3

Submitted To

JANCHE SANG

Submitted By

PRUDHVI REDDY ARAGA (CSU ID – **2788245**)

(No access to FH128 using iMac)

DESCRIPTION OF THE CODE

DEBUG FUNCTION

- Check the inode k for being used or not, check its bitmap:

  if(block.super.block in use[k/32] & (1 << (k%32)))
- As root data is present in inode of 1, we are reading the first block and printing its size using size field of inode.
- Looping through all direct pointers and printing them.
- Looping through all direct block from root and reading the inode present in the direct block and inode block.
- Based on the result, we are looping for directory entries and checking them if they are valid.
- Loop exits only if there is a valid field of directory entry.
- Priniting name and inode number present in directory entry.
- Priniting size using inode entry.
- Looping for all the direct blocks.
- If directory entry exits, printing the value.
- If indirect block exits, reading the inode of indirect.
- Looping for direct pointers.
- Looping for pointers present in the indirect blocks.

Tfs_Debug Output

```
grail:~/TinyFS% make
/usr/bin/gcc -Wall fs.c -c -o fs.o -g
/usr/bin/gcc shell.o fs.o disk.o -o tinyfs
grail:~/TinyFS% ./tinyfs image2
TinyFS> debug
      signature is valid
      16 blocks in use
      4 inodes in use
root inode 1:
      size: 4096 bytes
      direct blocks:  2
foo inode 2:
      size: 13 bytes
      direct blocks:  3
zoo inode 7:
      size: 40967 bytes
      direct blocks:  27  28  29  30  31
      indirect block: 32
      indirect data blocks:  72  73  74  75  76  77

TinyFS>
```

DELETE FUNCTION

- Reading Block-1 to get inode data.
- Checking for each and every direct pointer and reading the block present in each direct block.
- Created a nested loop for direct entries – if the file name is matching to given input and also to check if the file is valid.
- Making the valid flag to zero and then releasing all direct and indirect blocks assigned to this inode.
- Releasing all direct and indirect blocks assigned to this inode and writing back to disk.

Tfs_delete Output

```
[TinyFS> delete foo
 file foo inode 2 deleted.
[TinyFS> debug
       signature is valid
       16 blocks in use
       4 inodes in use
 root inode 1:
       size: 4096 bytes
       direct blocks:  2
 foo inode 2:
       size: 13 bytes
       direct blocks:  3
 zoo inode 7:
       size: 40967 bytes
       direct blocks:  27  28  29  30  31
       indirect block: 32
       indirect data blocks:  72  73  74  75  76  77
```

GET INUMBER FUNCTION

- Reading block 1 for inode info.
- Looping through all direct blocks and reading the inode present in the direct block.
- Reading the block present in the inode information.
- Loop for all the directory entries and check if the file is valid and if the name matches.
- Returns inode number – Once a value is found, the loop function is terminated.

Tfs_getinumber Output

As this is an internal command and we don’t have support to execute it from shell interface.

GETSIZE FUNCTION

- Reading block 1 for inode info.
- Looping through all direct blocks and reading the inode present in the direct block.
- Reading the block present in the inode information.
- Loop for all the directory entries and check if the file is valid and if the name matches.
- Returns size – Once a value is found, the loop function is terminated.

Tfs_getsize Output

```
[TinyFS> getsize foo
 file foo has size 13
[TinyFS> debug
       signature is valid
       16 blocks in use
       4 inodes in use
 root inode 1:
       size: 4096 bytes
       direct blocks:  2
 foo inode 2:
       size: 13 bytes
       direct blocks:  3
 zoo inode 7:
       size: 40967 bytes
       direct blocks:  27  28  29  30  31
       indirect block: 32
       indirect data blocks:  72  73  74  75  76  77
```