Memory Resident Unix-Like File System

19th March 2018

- K.Sai Satish Kumar Reddy (15CS10018)

- A.Surya Teja (15CS10002)

Compilation

1. Use **make all** to compile all the test case files.We get executables test1, test2, test3, test4 corresponding to each of the test cases.
2. Use ./testX to run the executables where X is 1, 2, 3,4.

Specifications:

- We have supported maximum filesystem size of 32 MB and a maximum of 126 inodes
- The first 65 blocks correspond to the super block and next 63 blocks make up the inode list
- Each block accommodates two inodes. Hence the total of 126 inodes.
- The rest of the blocks are data blocks.
- The bitmaps for inodes and data blocks are implemented using ***bitset*** data structure of C++. The bitmaps themselves are present in the super block.

Design details:

- The filesystem is a long byte array. We go to the required address, typecast the pointer to required struct and use it.
- We used the GNU - GCC **“sys/sem.h”** library for implementing semaphores to prevent race conditions.
- Three semaphores were used in the program. One in finding the free data blocks(so that no two files get the same block), another in finding the next free inode and the last in making an entry for the file/directory in its parent directory.